Global Constraint Catalog, Volume II, Time-Series Constraints
Ekaterina Arafailova, Nicolas Beldiceanu, Rémi Douence, Mats Carlsson, Pierre Flener, Justin Pearson, María Andreína Francisco Rodríguez, Helmut Simonis

To cite this version:

HAL Id: hal-01374721
https://inria.hal.science/hal-01374721
Submitted on 20 Nov 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Global Constraint Catalog
Volume II
Time-Series Constraints

Ekaterina Arafailova, Nicolas Beldiceanu¹, Rémi Douence
IMT Atlantique (LS2N-CNRS), Nantes, France

Mats Carlsson
RISE SICS, Kista, Sweden

Pierre Flener, Justin Pearson
Uppsala University, Uppsala, Sweden

María Andreína Francisco Rodríguez,
School of Computing, National University of Singapore

Helmut Simonis
Insight Centre for Data Analytics, University College Cork, Ireland

Abstract: First this report presents a restricted set of finite transducers used to synthesise structural time-series constraints described by means of a multi-layered function composition scheme. Second it provides the corresponding synthesised catalogue of structural time-series constraints where each constraint is explicitly described in terms of automata with registers.

Keywords: constraint programming, global constraint, finite transducer, automaton with registers, reversible automaton, glue matrix, sharp bounds, parametrised invariants, meta-data, ontology, sequential pattern mining, minimum description length.

2018-09-17

¹Corresponding author, Email: nicolas.beldiceanu@imt-atlantique.fr

This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc-sa/4.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.
Contents

Preface i

1 Describing Global Constraints 1
  1.1 Basic data types 1
  1.2 Compound data types 2
  1.3 Restrictions 3
  1.4 Declaring a global constraint 8
  1.5 Keywords 8

2 Overview of Time-Series Constraints 11
  2.1 Predicate 13
  2.2 Function 14
  2.3 Feature 14
  2.4 Extreme 16

3 Patterns and Corresponding Tables 19
  3.1 Patterns, Seed Transducers and Parametrised Glue Matrices 23
    3.1.1 BUMP_ON_DECREASING_SEQUENCE 24
    3.1.2 DECREASING 28
    3.1.3 DECREASING_SEQUENCE 30
    3.1.4 DECREASING_TERRACE 33
    3.1.5 DIP_ON_INCREASING_SEQUENCE 36
    3.1.6 GORGE 40
    3.1.7 INCREASING 44
    3.1.8 INCREASING_SEQUENCE 46
    3.1.9 INCREASING_TERRACE 49
    3.1.10 INFLEXION 52
    3.1.11 PEAK 54
    3.1.12 PLAIN 57
    3.1.13 PLATEAU 60
    3.1.14 PROPER_PLAIN 63
    3.1.15 PROPER_PLATEAU 66
    3.1.16 STEADY 69
    3.1.17 STEADY_SEQUENCE 71
3.1.18 STRICTLY_DECREASING_SEQUENCE .......................... 74
3.1.19 STRICTLY_INCREASING_SEQUENCE .......................... 77
3.1.20 SUMMIT .................................................. 80
3.1.21 VALLEY .................................................. 84
3.1.22 ZIGZAG .................................................. 87
3.2 Decoration Tables ............................................. 93
3.3 Tables of Regular-Expression Characteristics .................. 121
3.3.1 Big-width .................................................. 122
3.3.2 Height ..................................................... 124
3.3.3 Interval of interest ......................................... 126
3.3.4 Maximum-value-occurrence-number ......................... 128
3.3.5 Overlap ................................................... 130
3.3.6 Range ..................................................... 132
3.3.7 Set-of-inducing-words .................................... 134
3.3.8 Shift ....................................................... 135
3.3.9 Size ......................................................... 136
3.3.10 Smallest-variation-of-maxima ............................... 137

4 Global Constraint Catalogue ..................................... 139
ALL_EQUAL_HEIGHT_DECREASING_TERRACE ...................... 158
ALL_EQUAL_HEIGHT_INCREASING_TERRACE ....................... 162
ALL_EQUAL_HEIGHT_PLAIN ........................................ 166
ALL_EQUAL_HEIGHT_PLATEAU .................................... 170
ALL_EQUAL_HEIGHT_PROPER_PLAIN ................................ 174
ALL_EQUAL_HEIGHT_PROPER_PLATEAU ............................. 178
ALL_EQUAL_HEIGHT_STEADY ....................................... 182
ALL_EQUAL_HEIGHT_STEADY_SEQUENCE .......................... 186
ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE ............... 190
ALL_EQUAL_MAX_DECREASING ...................................... 194
ALL_EQUAL_MAX_DECREASING_SEQUENCE ......................... 198
ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE .................. 202
ALL_EQUAL_MAX_INCREASING ...................................... 206
ALL_EQUAL_MAX_INCREASING_SEQUENCE ......................... 210
ALL_EQUAL_MAX_INFLEXION ....................................... 214
ALL_EQUAL_MAX_PEAK ............................................ 218
ALL_EQUAL_MAX_STRICTLY_DECREASING_SEQUENCE ............... 222
ALL_EQUAL_MAX_STRICTLY_INCREASING_SEQUENCE ............... 226
ALL_EQUAL_MAX_SUMMIT .......................................... 230
ALL_EQUAL_MAX_ZIGZAG ......................................... 234
ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE ............... 238
ALL_EQUAL_MIN_DECREASING ...................................... 242
ALL_EQUAL_MIN_DECREASING_SEQUENCE ......................... 246
ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE .................. 250
ALL_EQUAL_MIN_GORGE ........................................... 254
ALL_EQUAL_MIN_INCREASING ...................................... 258
ALL_EQUAL_MIN_INCREASING_SEQUENCE ......................... 262
## CONTENTS

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_MIN_INFLEXION</td>
<td>266</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN STRICTLY DECREASING_SEQUENCE</td>
<td>270</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN STRICTLY INCREASING_SEQUENCE</td>
<td>274</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN VALLEY</td>
<td>278</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN_ZIGZAG</td>
<td>282</td>
</tr>
<tr>
<td>ALL_EQUAL_RANGE_DECREASING</td>
<td>286</td>
</tr>
<tr>
<td>ALL_EQUAL_RANGE_DECREASING_SEQUENCE</td>
<td>290</td>
</tr>
<tr>
<td>ALL_EQUAL_RANGE_INCREASING</td>
<td>294</td>
</tr>
<tr>
<td>ALL_EQUAL_RANGE_INCREASING_SEQUENCE</td>
<td>298</td>
</tr>
<tr>
<td>ALL_EQUAL_RANGE STRICTLY DECREASING_SEQUENCE</td>
<td>302</td>
</tr>
<tr>
<td>ALL_EQUAL_RANGE STRICTLY INCREASING_SEQUENCE</td>
<td>306</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
<td>310</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_DECREASING</td>
<td>314</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_DECREASING_SEQUENCE</td>
<td>318</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_DECREASING_TERRACE</td>
<td>322</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>326</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_GORGE</td>
<td>330</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_INCREASING</td>
<td>334</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_INCREASING_SEQUENCE</td>
<td>338</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_INCREASING_TERRACE</td>
<td>342</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_INFLEXION</td>
<td>346</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_PEAK</td>
<td>350</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_PLAIN</td>
<td>354</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_PLATEAU</td>
<td>358</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_PROPER_PLAIN</td>
<td>362</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_PROPER_PLATEAU</td>
<td>366</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_STEADY</td>
<td>370</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_STEADY_SEQUENCE</td>
<td>374</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_STEADY_SEQUENCE</td>
<td>378</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_STRICTLY DECREASING_SEQUENCE</td>
<td>382</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_STRICTLY INCREASING_SEQUENCE</td>
<td>386</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_SUMMIT</td>
<td>390</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_VALLEY</td>
<td>394</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_DECREASING SEQUENCE</td>
<td>398</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_DECREASING_TERRACE</td>
<td>402</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_GORGE</td>
<td>406</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_INCREASING SEQUENCE</td>
<td>410</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_INCREASING_TERRACE</td>
<td>414</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_INFLEXION</td>
<td>418</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_PEAK</td>
<td>422</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_PLAIN</td>
<td>426</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_PLATEAU</td>
<td>430</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_PROPER_PLAIN</td>
<td>434</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_PROPER_PLATEAU</td>
<td>438</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_STEADY_SEQUENCE</td>
<td>442</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_STRICTLY DECREASING_SEQUENCE</td>
<td>446</td>
</tr>
</tbody>
</table>
## CONTENTS

<table>
<thead>
<tr>
<th>Category</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>634</td>
</tr>
<tr>
<td>DECREASING_SURF_GORGE</td>
<td>638</td>
</tr>
<tr>
<td>DECREASING_SURF_INCREASING</td>
<td>642</td>
</tr>
<tr>
<td>DECREASING_SURF_INCREASING_SEQUENCE</td>
<td>646</td>
</tr>
<tr>
<td>DECREASING_SURF_INCREASING_TERRACE</td>
<td>650</td>
</tr>
<tr>
<td>DECREASING_SURF_INFLEXION</td>
<td>654</td>
</tr>
<tr>
<td>DECREASING_SURF_PEAK</td>
<td>658</td>
</tr>
<tr>
<td>DECREASING_SURF_PLAIN</td>
<td>662</td>
</tr>
<tr>
<td>DECREASING_SURF_PLATEAU</td>
<td>666</td>
</tr>
<tr>
<td>DECREASING_SURF_PROPER_PLAIN</td>
<td>670</td>
</tr>
<tr>
<td>DECREASING_SURF_PROPER_PLATEAU</td>
<td>674</td>
</tr>
<tr>
<td>DECREASING_SURF_STEADY</td>
<td>678</td>
</tr>
<tr>
<td>DECREASING_SURF_STEADY_SEQUENCE</td>
<td>682</td>
</tr>
<tr>
<td>DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE</td>
<td>686</td>
</tr>
<tr>
<td>DECREASING_SURF_STRICTLY_INCREASING_SEQUENCE</td>
<td>690</td>
</tr>
<tr>
<td>DECREASING_SURF_SUMMIT</td>
<td>694</td>
</tr>
<tr>
<td>DECREASING_SURF_VALLEY</td>
<td>698</td>
</tr>
<tr>
<td>DECREASING_SURF_ZIGZAG</td>
<td>702</td>
</tr>
<tr>
<td>DECREASING_WIDTH_DECREASING_SEQUENCE</td>
<td>706</td>
</tr>
<tr>
<td>DECREASING_WIDTH_DECREASING_TERRACE</td>
<td>710</td>
</tr>
<tr>
<td>DECREASING_WIDTH_GORGE</td>
<td>714</td>
</tr>
<tr>
<td>DECREASING_WIDTH_INCREASING_SEQUENCE</td>
<td>718</td>
</tr>
<tr>
<td>DECREASING_WIDTH_INCREASING_TERRACE</td>
<td>722</td>
</tr>
<tr>
<td>DECREASING_WIDTH_INFLEXION</td>
<td>726</td>
</tr>
<tr>
<td>DECREASING_WIDTH_PEAK</td>
<td>730</td>
</tr>
<tr>
<td>DECREASING_WIDTH_PLAIN</td>
<td>734</td>
</tr>
<tr>
<td>DECREASING_WIDTH_PLATEAU</td>
<td>738</td>
</tr>
<tr>
<td>DECREASING_WIDTH_PROPER_PLAIN</td>
<td>742</td>
</tr>
<tr>
<td>DECREASING_WIDTH_PROPER_PLATEAU</td>
<td>746</td>
</tr>
<tr>
<td>DECREASING_WIDTH_STEADY_SEQUENCE</td>
<td>750</td>
</tr>
<tr>
<td>DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE</td>
<td>754</td>
</tr>
<tr>
<td>DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE</td>
<td>758</td>
</tr>
<tr>
<td>DECREASING_WIDTH_SUMMIT</td>
<td>762</td>
</tr>
<tr>
<td>DECREASING_WIDTH_VALLEY</td>
<td>766</td>
</tr>
<tr>
<td>DECREASING_WIDTH_ZIGZAG</td>
<td>770</td>
</tr>
<tr>
<td>HEIGHT_DECREASING_TERRACE</td>
<td>774</td>
</tr>
<tr>
<td>HEIGHT_INCREASING_TERRACE</td>
<td>778</td>
</tr>
<tr>
<td>HEIGHT.PLAIN</td>
<td>782</td>
</tr>
<tr>
<td>HEIGHT_PLATEAU</td>
<td>786</td>
</tr>
<tr>
<td>HEIGHT_PROPER_PLAIN</td>
<td>790</td>
</tr>
<tr>
<td>HEIGHT_PROPER_PLATEAU</td>
<td>794</td>
</tr>
<tr>
<td>HEIGHT_STEADY</td>
<td>798</td>
</tr>
<tr>
<td>HEIGHT_STEADY_SEQUENCE</td>
<td>802</td>
</tr>
<tr>
<td>INCREASING_HEIGHT_DECREASING_TERRACE</td>
<td>806</td>
</tr>
<tr>
<td>INCREASING_HEIGHT_INCREASING_TERRACE</td>
<td>810</td>
</tr>
<tr>
<td>INCREASING_HEIGHT_PLAIN</td>
<td>814</td>
</tr>
</tbody>
</table>
CONTENTS

INCREASING_HEIGHT_PLATEAU ............................................ 818
INCREASING_HEIGHT_PROPER_PLAIN ................................. 822
INCREASING_HEIGHT_PROPER_PLATEAU ............................. 826
INCREASING_HEIGHT_STEADY ........................................... 830
INCREASING_HEIGHT_STEADY_SEQUENCE .............................. 834
INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE ................. 838
INCREASING_MAX_DECENDING ........................................... 842
INCREASING_MAX_DECENDING_SEQUENCE ............................ 846
INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE ................. 850
INCREASING_MAX_INCREASING ........................................... 854
INCREASING_MAX_INCREASING_SEQUENCE ............................ 858
INCREASING_MAX_INFLEXION ............................................ 862
INCREASING_MAX_PEAK .................................................... 866
INCREASING_MAX_STRICTLY_DECENDING_SEQUENCE ................. 870
INCREASING_MAX_STRICTLY_INCREASING_SEQUENCE ................. 874
INCREASING_MAX_SUMMIT ................................................. 878
INCREASING_MAX_ZIGZAG ................................................. 882
INCREASING_MIN_BUMP_ON_DECENDING_SEQUENCE ................. 886
INCREASING_MIN_DECENDING ............................................. 890
INCREASING_MIN_DECENDING_SEQUENCE ............................ 894
INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE ................... 898
INCREASING_MIN_GORGE ................................................... 902
INCREASING_MIN_INCREASING ............................................ 906
INCREASING_MIN_INCREASING_SEQUENCE ............................ 910
INCREASING_MIN_INFLEXION ............................................. 914
INCREASING_MIN_STRICTLY_DECENDING_SEQUENCE ................. 918
INCREASING_MIN_STRICTLY_INCREASING_SEQUENCE .......... 922
INCREASING_MIN_VALLEY .................................................. 926
INCREASING_MIN_ZIGZAG .................................................. 930
INCREASING_RANGE_DECENDING ......................................... 934
INCREASING_RANGE_DECENDING_SEQUENCE ........................... 938
INCREASING_RANGE_INCREASING ......................................... 942
INCREASING_RANGE_INCREASING_SEQUENCE ........................ 946
INCREASING_RANGE_STRICTLY_DECENDING_SEQUENCE .......... 950
INCREASING_RANGE_STRICTLY_INCREASING_SEQUENCE ........... 954
INCREASING_SURF_BUMP_ON_DECENDING_SEQUENCE ............... 958
INCREASING_SURF_DECENDING ........................................... 962
INCREASING_SURF_DECENDING_SEQUENCE ............................ 966
INCREASING_SURF_DECENDING_TERRACE .............................. 970
INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE ................. 974
INCREASING_SURF_GORGE .................................................. 978
INCREASING_SURF_INCREASING ........................................... 982
INCREASING_SURF_INCREASING_SEQUENCE ........................... 986
INCREASING_SURF_INCREASING_TERRACE ............................. 990
INCREASING_SURF_INFLEXION ............................................. 994
INCREASING_SURF_PEAK ..................................................... 998
<table>
<thead>
<tr>
<th>CONTENTS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MAX_RANGE_INCREASING</td>
<td>1380</td>
</tr>
<tr>
<td>MAX_RANGE_INCREASING_SEQUENCE</td>
<td>1384</td>
</tr>
<tr>
<td>MAX_RANGE STRICTLY DECREASING_SEQUENCE</td>
<td>1388</td>
</tr>
<tr>
<td>MAX_RANGE STRICTLY INCREASING_SEQUENCE</td>
<td>1392</td>
</tr>
<tr>
<td>MAX STRICTLY DECREASING_SEQUENCE</td>
<td>1396</td>
</tr>
<tr>
<td>MAX STRICTLY INCREASING_SEQUENCE</td>
<td>1400</td>
</tr>
<tr>
<td>MAX SUMMIT</td>
<td>1404</td>
</tr>
<tr>
<td>MAX_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
<td>1408</td>
</tr>
<tr>
<td>MAX_SURF_DECREASING</td>
<td>1412</td>
</tr>
<tr>
<td>MAX_SURF_DECREASING_SEQUENCE</td>
<td>1416</td>
</tr>
<tr>
<td>MAX_SURF_DECREASING_TERRACE</td>
<td>1420</td>
</tr>
<tr>
<td>MAX_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>1424</td>
</tr>
<tr>
<td>MAX_SURF_GORGE</td>
<td>1428</td>
</tr>
<tr>
<td>MAX_SURF_INCREASING</td>
<td>1432</td>
</tr>
<tr>
<td>MAX_SURF_INCREASING_SEQUENCE</td>
<td>1436</td>
</tr>
<tr>
<td>MAX_SURF_INCREASING_TERRACE</td>
<td>1440</td>
</tr>
<tr>
<td>MAX_SURF_INFLEXION</td>
<td>1444</td>
</tr>
<tr>
<td>MAX_SURF_PEAK</td>
<td>1448</td>
</tr>
<tr>
<td>MAX_SURF_PLAIN</td>
<td>1452</td>
</tr>
<tr>
<td>MAX_SURF_PLATEAU</td>
<td>1456</td>
</tr>
<tr>
<td>MAX_SURF_PROPER_PLAIN</td>
<td>1460</td>
</tr>
<tr>
<td>MAX_SURF_PROPER_PLATEAU</td>
<td>1464</td>
</tr>
<tr>
<td>MAX_SURF_STEADY</td>
<td>1468</td>
</tr>
<tr>
<td>MAX_SURF_STEADY_SEQUENCE</td>
<td>1472</td>
</tr>
<tr>
<td>MAX_SURF STRICTLY DECREASING_SEQUENCE</td>
<td>1476</td>
</tr>
<tr>
<td>MAX_SURF STRICTLY INCREASING_SEQUENCE</td>
<td>1480</td>
</tr>
<tr>
<td>MAX_SURF SUMMIT</td>
<td>1484</td>
</tr>
<tr>
<td>MAX_SURF VALLEY</td>
<td>1488</td>
</tr>
<tr>
<td>MAX_SURF_ZIGZAG</td>
<td>1492</td>
</tr>
<tr>
<td>MAX WIDTH DECREASING_SEQUENCE</td>
<td>1500</td>
</tr>
<tr>
<td>MAX WIDTH DECREASING_TERRACE</td>
<td>1504</td>
</tr>
<tr>
<td>MAX WIDTH GORGE</td>
<td>1508</td>
</tr>
<tr>
<td>MAX WIDTH INCREASING_SEQUENCE</td>
<td>1514</td>
</tr>
<tr>
<td>MAX WIDTH INCREASING_TERRACE</td>
<td>1518</td>
</tr>
<tr>
<td>MAX WIDTH INFLEXION</td>
<td>1522</td>
</tr>
<tr>
<td>MAX WIDTH PEAK</td>
<td>1526</td>
</tr>
<tr>
<td>MAX WIDTH PLAIN</td>
<td>1530</td>
</tr>
<tr>
<td>MAX WIDTH PLATEAU</td>
<td>1534</td>
</tr>
<tr>
<td>MAX WIDTH PROPER_PLAIN</td>
<td>1538</td>
</tr>
<tr>
<td>MAX WIDTH PROPER_PLATEAU</td>
<td>1542</td>
</tr>
<tr>
<td>MAX WIDTH STEADY_SEQUENCE</td>
<td>1546</td>
</tr>
<tr>
<td>MAX WIDTH STRICTLY DECREASING_SEQUENCE</td>
<td>1550</td>
</tr>
<tr>
<td>MAX WIDTH STRICTLY INCREASING_SEQUENCE</td>
<td>1554</td>
</tr>
<tr>
<td>MAX WIDTH SUMMIT</td>
<td>1558</td>
</tr>
<tr>
<td>MAX WIDTH VALLEY</td>
<td>1564</td>
</tr>
<tr>
<td>MAX WIDTH ZIGZAG</td>
<td>1568</td>
</tr>
<tr>
<td>CONTENTS</td>
<td>Page</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>MIN_RANGE_STRICTLY_INCREASING_SEQUENCE</td>
<td>1770</td>
</tr>
<tr>
<td>MIN_STRICTLY_DECREASING_SEQUENCE</td>
<td>1774</td>
</tr>
<tr>
<td>MIN_STRICTLY_INCREASING_SEQUENCE</td>
<td>1778</td>
</tr>
<tr>
<td>MIN_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
<td>1782</td>
</tr>
<tr>
<td>MIN_SURF_DECREASING</td>
<td>1786</td>
</tr>
<tr>
<td>MIN_SURF_DECREASING_SEQUENCE</td>
<td>1790</td>
</tr>
<tr>
<td>MIN_SURF_DECREASING_TERRACE</td>
<td>1794</td>
</tr>
<tr>
<td>MIN_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>1798</td>
</tr>
<tr>
<td>MIN_SURF_GORGE</td>
<td>1802</td>
</tr>
<tr>
<td>MIN_SURF_INCREASING</td>
<td>1806</td>
</tr>
<tr>
<td>MIN_SURF_INCREASING_SEQUENCE</td>
<td>1810</td>
</tr>
<tr>
<td>MIN_SURF_INCREASING_TERRACE</td>
<td>1814</td>
</tr>
<tr>
<td>MIN_SURF_INPLEXION</td>
<td>1818</td>
</tr>
<tr>
<td>MIN_SURF_PEAK</td>
<td>1822</td>
</tr>
<tr>
<td>MIN_SURF_PLAIN</td>
<td>1826</td>
</tr>
<tr>
<td>MIN_SURF_PLATEAU</td>
<td>1830</td>
</tr>
<tr>
<td>MIN_SURF_PROPER_PLAIN</td>
<td>1834</td>
</tr>
<tr>
<td>MIN_SURF_PROPER_PLATEAU</td>
<td>1838</td>
</tr>
<tr>
<td>MIN_SURF_STEADY</td>
<td>1842</td>
</tr>
<tr>
<td>MIN_SURF_STEADY_SEQUENCE</td>
<td>1846</td>
</tr>
<tr>
<td>MIN_SURF_STRICTLY_DECREASING_SEQUENCE</td>
<td>1850</td>
</tr>
<tr>
<td>MIN_SURF_STRICTLY_INCREASING_SEQUENCE</td>
<td>1854</td>
</tr>
<tr>
<td>MIN_SURF_SUMMIT</td>
<td>1858</td>
</tr>
<tr>
<td>MIN_SURF_VALLEY</td>
<td>1862</td>
</tr>
<tr>
<td>MIN_SURF_ZIGZAG</td>
<td>1866</td>
</tr>
<tr>
<td>MIN_VALLEY</td>
<td>1874</td>
</tr>
<tr>
<td>MIN_WIDTH_DECREASING_SEQUENCE</td>
<td>1878</td>
</tr>
<tr>
<td>MIN_WIDTH_DECREASING_TERRACE</td>
<td>1882</td>
</tr>
<tr>
<td>MIN_WIDTH_GORGE</td>
<td>1886</td>
</tr>
<tr>
<td>MIN_WIDTH_INCREASING_SEQUENCE</td>
<td>1890</td>
</tr>
<tr>
<td>MIN_WIDTH_INCREASING_TERRACE</td>
<td>1894</td>
</tr>
<tr>
<td>MIN_WIDTH_INPLEXION</td>
<td>1898</td>
</tr>
<tr>
<td>MIN_WIDTH_PEAK</td>
<td>1902</td>
</tr>
<tr>
<td>MIN_WIDTH_PLAIN</td>
<td>1906</td>
</tr>
<tr>
<td>MIN_WIDTH_PLATEAU</td>
<td>1910</td>
</tr>
<tr>
<td>MIN_WIDTH_PROPER_PLAIN</td>
<td>1914</td>
</tr>
<tr>
<td>MIN_WIDTH_PROPER_PLATEAU</td>
<td>1918</td>
</tr>
<tr>
<td>MIN_WIDTH_STEADY_SEQUENCE</td>
<td>1922</td>
</tr>
<tr>
<td>MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE</td>
<td>1926</td>
</tr>
<tr>
<td>MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE</td>
<td>1930</td>
</tr>
<tr>
<td>MIN_WIDTH_SUMMIT</td>
<td>1934</td>
</tr>
<tr>
<td>MIN_WIDTH_VALLEY</td>
<td>1938</td>
</tr>
<tr>
<td>MIN_WIDTH_ZIGZAG</td>
<td>1942</td>
</tr>
<tr>
<td>MIN_ZIGZAG</td>
<td>1950</td>
</tr>
<tr>
<td>NB_BUMP_ON_DECREASING_SEQUENCE</td>
<td>1954</td>
</tr>
<tr>
<td>NB_DECREASING</td>
<td>1960</td>
</tr>
<tr>
<td>CONTENTS</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td></td>
</tr>
<tr>
<td>NB_DECREASING_SEQUENCE</td>
<td>1966</td>
</tr>
<tr>
<td>NB_DECREASING_TERRACE</td>
<td>1972</td>
</tr>
<tr>
<td>NB_DIP_ON_INCREASING_SEQUENCE</td>
<td>1978</td>
</tr>
<tr>
<td>NB_GORGE</td>
<td>1984</td>
</tr>
<tr>
<td>NB_INCREASING</td>
<td>1992</td>
</tr>
<tr>
<td>NB_INCREASING_SEQUENCE</td>
<td>1998</td>
</tr>
<tr>
<td>NB_INCREASING_TERRACE</td>
<td>2004</td>
</tr>
<tr>
<td>NB_INFLEXION</td>
<td>2010</td>
</tr>
<tr>
<td>NB_PEAK</td>
<td>2016</td>
</tr>
<tr>
<td>NB_PLAIN</td>
<td>2022</td>
</tr>
<tr>
<td>NB_PLATEAU</td>
<td>2028</td>
</tr>
<tr>
<td>NB_PROPER_PLAIN</td>
<td>2034</td>
</tr>
<tr>
<td>NB_PROPER_PLATEAU</td>
<td>2040</td>
</tr>
<tr>
<td>NB_STEADY</td>
<td>2046</td>
</tr>
<tr>
<td>NB_STEADY_SEQUENCE</td>
<td>2052</td>
</tr>
<tr>
<td>NB_STRICTLY_DECREASING_SEQUENCE</td>
<td>2058</td>
</tr>
<tr>
<td>NB_STRICTLY_INCREASING_SEQUENCE</td>
<td>2064</td>
</tr>
<tr>
<td>NB_SUMMIT</td>
<td>2070</td>
</tr>
<tr>
<td>NB_VALLEY</td>
<td>2078</td>
</tr>
<tr>
<td>NB_ZIGZAG</td>
<td>2084</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_DECREASING_TERRACE</td>
<td>2092</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_INCREASING_TERRACE</td>
<td>2096</td>
</tr>
<tr>
<td>POS_MAX_HEIGHTPLAIN</td>
<td>2100</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_PLATEAU</td>
<td>2104</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_PROPER_PLAIN</td>
<td>2108</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_PROPER_PLATEAU</td>
<td>2112</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_STEADY</td>
<td>2116</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_STEADY_SEQUENCE</td>
<td>2120</td>
</tr>
<tr>
<td>POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE</td>
<td>2124</td>
</tr>
<tr>
<td>POS_MAX_MAX_DECREASING</td>
<td>2128</td>
</tr>
<tr>
<td>POS_MAX_MAX_DECREASING_SEQUENCE</td>
<td>2132</td>
</tr>
<tr>
<td>POS_MAX_MAX_DECREASING_SEQUENCE</td>
<td>2136</td>
</tr>
<tr>
<td>POS_MAX_MAX_INCREASING</td>
<td>2140</td>
</tr>
<tr>
<td>POS_MAX_MAX_INCREASING_SEQUENCE</td>
<td>2144</td>
</tr>
<tr>
<td>POS_MAX_MAX_INFLEXION</td>
<td>2148</td>
</tr>
<tr>
<td>POS_MAX_MAX_PEAK</td>
<td>2152</td>
</tr>
<tr>
<td>POS_MAX_MAX_STRICTLY_DECREASING_SEQUENCE</td>
<td>2156</td>
</tr>
<tr>
<td>POS_MAX_MAX_STRICTLY_INCREASING_SEQUENCE</td>
<td>2160</td>
</tr>
<tr>
<td>POS_MAX_MAX_SUMMIT</td>
<td>2164</td>
</tr>
<tr>
<td>POS_MAX_MAX_ZIGZAG</td>
<td>2168</td>
</tr>
<tr>
<td>POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE</td>
<td>2172</td>
</tr>
<tr>
<td>POS_MAX_MIN_DECREASING</td>
<td>2176</td>
</tr>
<tr>
<td>POS_MAX_MIN_DECREASING_SEQUENCE</td>
<td>2180</td>
</tr>
<tr>
<td>POS_MAX_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td>2184</td>
</tr>
<tr>
<td>POS_MAX_MIN_GORGE</td>
<td>2188</td>
</tr>
<tr>
<td>POS_MAX_MIN_INCREASING</td>
<td>2192</td>
</tr>
</tbody>
</table>
CONTENTS

POS_MAX_MIN_INCREASING_SEQUENCE .......................... 2196
POS_MAX_MIN_INFLEXION ................................. 2200
POS_MAX_MIN_STRICTLY_DECREASING_SEQUENCE .......... 2204
POS_MAX_MIN_STRICTLY_INCREASING_SEQUENCE ........ 2208
POS_MAX_MIN_VALLEY .......................... 2212
POS_MAX_MIN_ZIGZAG .......................... 2216
POS_MAX_SURF_BUMP_ON_DECREASING_SEQUENCE ...... 2220
POS_MAX_SURF_DECREASING .......................... 2224
POS_MAX_SURF_DECREASING_SEQUENCE .......... 2228
POS_MAX_SURF_DECREASING_TERRACE .................. 2232
POS_MAX_SURF_DIP_ON_INCREASING_SEQUENCE ....... 2236
POS_MAX_SURF_GORGE ............................ 2240
POS_MAX_SURF_INCREASING .......................... 2244
POS_MAX_SURF_INCREASING_SEQUENCE .......... 2248
POS_MAX_SURF_INCREASING_TERRACE ........ 2252
POS_MAX_SURF_INFLEXION .......................... 2256
POS_MAX_SURF_PEAK ............................. 2260
POS_MAX_SURF_PLAIN ............................. 2264
POS_MAX_SURF_PLATEAU .......................... 2268
POS_MAX_SURF_PROPRIETY .......................... 2272
POS_MAX_SURF_PROPRIETY_PLATEAU .............. 2276
POS_MAX_SURF_STEADY ............................. 2280
POS_MAX_SURF_STEADY_SEQUENCE .............. 2284
POS_MAX_SURF_STRICTLY_DECREASING_SEQUENCE .... 2288
POS_MAX_SURF_STRICTLY_INCREASING_SEQUENCE .... 2292
POS_MAX_SURF_SUMMIT .............................. 2296
POS_MAX_SURF_VALLEY .............................. 2300
POS_MAX_SURF_ZIGZAG .............................. 2304
POS_MAX_WIDTH_DECREASING_SEQUENCE .......... 2308
POS_MAX_WIDTH_DECREASING_TERRACE .......... 2312
POS_MAX_WIDTH_GORGE .............................. 2316
POS_MAX_WIDTH_INCREASING_SEQUENCE .......... 2320
POS_MAX_WIDTH_INCREASING_TERRACE ........ 2324
POS_MAX_WIDTH_INFLEXION .......................... 2328
POS_MAX_WIDTH_PEAK ............................. 2332
POS_MAX_WIDTH_PLAIN ............................. 2336
POS_MAX_WIDTH_PLATEAU .......................... 2340
POS_MAX_WIDTH_PROPRIETY .......................... 2344
POS_MAX_WIDTH_PROPRIETY_PLATEAU .......... 2348
POS_MAX_WIDTH_STEADY_SEQUENCE .......... 2352
POS_MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE .... 2356
POS_MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE .... 2360
POS_MAX_WIDTH_SUMMIT .............................. 2364
POS_MAX_WIDTH_VALLEY .............................. 2368
POS_MAX_WIDTH_ZIGZAG .............................. 2372
POS_MIN_HEIGHT_DECREASING_TERRACE .......... 2376
| CONTENTS                                                                 |
|-------------------------------------------------------|---------|
| POS_MIN_HEIGHT_INCREMENTAL_TERRACE                    | 2380    |
| POS_MIN_HEIGHT_PLAIN                                  | 2384    |
| POS_MIN_HEIGHT_PLATEAU                               | 2388    |
| POS_MIN_HEIGHT_PROPER_PLAIN                          | 2392    |
| POS_MIN_HEIGHT_PROPER_PLATEAU                        | 2396    |
| POS_MIN_HEIGHT_STEADY                                 | 2400    |
| POS_MIN_HEIGHT_STEADY_SEQUENCE                       | 2404    |
| POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE               | 2408    |
| POS_MIN_MAX_DECREASING                               | 2412    |
| POS_MIN_MAX_DECREASING_SEQUENCE                      | 2416    |
| POS_MIN_MAX_DECREASING_SEQUENCE                      | 2420    |
| POS_MIN_MAX_INCREMENTAL                              | 2424    |
| POS_MIN_MAX_INCREMENTAL_SEQUENCE                     | 2428    |
| POS_MIN_MAX_INCREASE                                 | 2432    |
| POS_MIN_MAX_PEAK                                      | 2436    |
| POS_MIN_MAX.Strictly_DECREASING_SEQUENCE             | 2440    |
| POS_MIN_MAX.Strictly_INCREMENTAL_sequence             | 2444    |
| POS_MIN_MAX_SUMMIT                                   | 2448    |
| POS_MIN_MAX_ZIGZAG                                   | 2452    |
| POS_MIN_MIN_BUMP_ON_DECREASING_SEQUENCE              | 2456    |
| POS_MIN_MIN_DECREASING                               | 2460    |
| POS_MIN_MIN_DECREASING_SEQUENCE                      | 2464    |
| POS_MIN_MIN_INCREMENTAL_SEQUENCE                     | 2468    |
| POS_MIN_MIN_GORGE                                    | 2472    |
| POS_MIN_MIN_INCREMENTAL_SEQUENCE                     | 2476    |
| POS_MIN_MIN_INCREMENTAL_SEQUENCE                     | 2480    |
| POS_MIN_MIN_INCREASE                                 | 2484    |
| POS_MIN_MIN.Strictly_DECREASING_SEQUENCE             | 2488    |
| POS_MIN_MIN.Strictly_INCREMENTAL_sequence             | 2492    |
| POS_MIN_MIN_VALLEY                                   | 2496    |
| POS_MIN_MIN_ZIGZAG                                   | 2500    |
| POS_MIN_SURF_BUMP_ON_DECREASING_SEQUENCE             | 2504    |
| POS_MIN_SURF_DECREASING                              | 2508    |
| POS_MIN_SURF_DECREASING_SEQUENCE                     | 2512    |
| POS_MIN_SURF_DECREASING_TERRACE                      | 2516    |
| POS_MIN_SURF_DIP_ON_INCREMENTAL_SEQUENCE             | 2520    |
| POS_MIN_SURF_GORGE                                   | 2524    |
| POS_MIN_SURF_INCREMENTAL                             | 2528    |
| POS_MIN_SURF_INCREMENTAL_SEQUENCE                    | 2532    |
| POS_MIN_SURF_INCREMENTAL_TERRACE                     | 2536    |
| POS_MIN_SURF_INCREASE                                | 2540    |
| POS_MIN_SURF_PEAK                                    | 2544    |
| POS_MIN_SURF_PLAIN                                   | 2548    |
| POS_MIN_SURF_PLATEAU                                | 2552    |
| POS_MIN_SURF_PROPER_PLAIN                            | 2556    |
| POS_MIN_SURF_PROPER_PLATEAU                          | 2560    |
## CONTENTS

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>POS_MIN_SURF_STEADY</td>
<td>2564</td>
</tr>
<tr>
<td>POS_MIN_SURF_STEADY_SEQUENCE</td>
<td>2568</td>
</tr>
<tr>
<td>POS_MIN_SURF_STRICTLY_DECREASING_SEQUENCE</td>
<td>2572</td>
</tr>
<tr>
<td>POS_MIN_SURF_STRICTLY_INCREASING_SEQUENCE</td>
<td>2576</td>
</tr>
<tr>
<td>POS_MIN_SURF_SUMMIT</td>
<td>2580</td>
</tr>
<tr>
<td>POS_MIN_SURF_VALLEY</td>
<td>2584</td>
</tr>
<tr>
<td>POS_MIN_SURF_ZIGZAG</td>
<td>2588</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_Decreasing_sequence</td>
<td>2592</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_Decreasing_Terrace</td>
<td>2596</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_Gorge</td>
<td>2600</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_Increasing_sequence</td>
<td>2604</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_Increasing_Terrace</td>
<td>2608</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_Inflexion</td>
<td>2612</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_Peak</td>
<td>2616</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_Plateau</td>
<td>2620</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_PlateauROADCAST</td>
<td>2624</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_Proper_Plateau</td>
<td>2628</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_Proper_PlateauROADCAST</td>
<td>2632</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_STeady_sequence</td>
<td>2636</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_STRICTLY_Decreasing_sequence</td>
<td>2640</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_STRICTLY_Increasing_sequence</td>
<td>2644</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_Summit</td>
<td>2648</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_Valley</td>
<td>2652</td>
</tr>
<tr>
<td>POS_MIN_WIDTH_ZIGZAG</td>
<td>2656</td>
</tr>
<tr>
<td>SUM_HEIGHT_Decreasing_Terrace</td>
<td>2660</td>
</tr>
<tr>
<td>SUM_HEIGHT_Increasing_Terrace</td>
<td>2666</td>
</tr>
<tr>
<td>SUM_HEIGHT_Plateau</td>
<td>2672</td>
</tr>
<tr>
<td>SUM_HEIGHT_Plateau_BROADCAST</td>
<td>2676</td>
</tr>
<tr>
<td>SUM_HEIGHT_Proper_Plateau_NETWORK</td>
<td>2680</td>
</tr>
<tr>
<td>SUM_HEIGHT_Proper_PlateauROADCAST</td>
<td>2684</td>
</tr>
<tr>
<td>SUM_HEIGHT_STeady</td>
<td>2688</td>
</tr>
<tr>
<td>SUM_HEIGHT_STeady_SEQUENCE</td>
<td>2692</td>
</tr>
<tr>
<td>SUM_MAX_Bump_On_Decreasing_sequence</td>
<td>2696</td>
</tr>
<tr>
<td>SUM_MAX_Decreasing</td>
<td>2700</td>
</tr>
<tr>
<td>SUM_MAX_Decreasing_Sequence</td>
<td>2706</td>
</tr>
<tr>
<td>SUM_MAX_Dip_On_Increasing_sequence</td>
<td>2710</td>
</tr>
<tr>
<td>SUM_MAX_Increasing</td>
<td>2714</td>
</tr>
<tr>
<td>SUM_MAX_Increasing_Sequence</td>
<td>2720</td>
</tr>
<tr>
<td>SUM_MAX_Inflexion</td>
<td>2724</td>
</tr>
<tr>
<td>SUM_MAX_Peak</td>
<td>2728</td>
</tr>
<tr>
<td>SUM_MAX_STRICTLY_Decreasing_sequence</td>
<td>2732</td>
</tr>
<tr>
<td>SUM_MAX_STRICTLY_Increasing_sequence</td>
<td>2736</td>
</tr>
<tr>
<td>SUM_MAX_Summit</td>
<td>2740</td>
</tr>
<tr>
<td>SUM_MAX_ZIGZAG</td>
<td>2746</td>
</tr>
<tr>
<td>SUM_MIN_Bump_On_Decreasing_sequence</td>
<td>2754</td>
</tr>
<tr>
<td>SUM_MIN_Decreasing</td>
<td>2758</td>
</tr>
<tr>
<td>CONTENTS</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>SUM_MIN_DECENDING_SEQUENCE</td>
<td>2764</td>
</tr>
<tr>
<td>SUM_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td>2768</td>
</tr>
<tr>
<td>SUM_MIN_GORGE</td>
<td>2772</td>
</tr>
<tr>
<td>SUM_MIN_INCREASING</td>
<td>2778</td>
</tr>
<tr>
<td>SUM_MIN_INCREASING_SEQUENCE</td>
<td>2784</td>
</tr>
<tr>
<td>SUM_MIN_INFLEXION</td>
<td>2788</td>
</tr>
<tr>
<td>SUM_MIN.Strictly_decending_sequence</td>
<td>2792</td>
</tr>
<tr>
<td>SUM_MIN.Strictly_increasing_sequence</td>
<td>2796</td>
</tr>
<tr>
<td>SUM_MIN_VALLEY</td>
<td>2800</td>
</tr>
<tr>
<td>SUM_MIN_ZIGZAG</td>
<td>2804</td>
</tr>
<tr>
<td>SUM_RANGE_DECENDING</td>
<td>2812</td>
</tr>
<tr>
<td>SUM_RANGE_DECENDING_SEQUENCE</td>
<td>2816</td>
</tr>
<tr>
<td>SUM_RANGE_INCREASING</td>
<td>2820</td>
</tr>
<tr>
<td>SUM_RANGE_INCREASING_SEQUENCE</td>
<td>2824</td>
</tr>
<tr>
<td>SUM_RANGE.Strictly_decending_sequence</td>
<td>2828</td>
</tr>
<tr>
<td>SUM_RANGE.Strictly_increasing_sequence</td>
<td>2832</td>
</tr>
<tr>
<td>SUM_SURF_BUMP_ON_DECENDING_SEQUENCE</td>
<td>2836</td>
</tr>
<tr>
<td>SUM_SURF_DECENDING</td>
<td>2840</td>
</tr>
<tr>
<td>SUM_SURF_DECENDING_SEQUENCE</td>
<td>2846</td>
</tr>
<tr>
<td>SUM_SURF_DECENDING_TERRACE</td>
<td>2850</td>
</tr>
<tr>
<td>SUM_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>2854</td>
</tr>
<tr>
<td>SUM_SURF_GORGE</td>
<td>2858</td>
</tr>
<tr>
<td>SUM_SURF_INCREASING</td>
<td>2864</td>
</tr>
<tr>
<td>SUM_SURF_INCREASING_SEQUENCE</td>
<td>2870</td>
</tr>
<tr>
<td>SUM_SURF_INCREASING_TERRACE</td>
<td>2874</td>
</tr>
<tr>
<td>SUM_SURF_INFLEXION</td>
<td>2878</td>
</tr>
<tr>
<td>SUM_SURF_PEAK</td>
<td>2882</td>
</tr>
<tr>
<td>SUM_SURF.PLAIN</td>
<td>2886</td>
</tr>
<tr>
<td>SUM_SURF_PLATEAU</td>
<td>2890</td>
</tr>
<tr>
<td>SUM_SURF_PROPER.PLAIN</td>
<td>2894</td>
</tr>
<tr>
<td>SUM_SURF_PROPER.PLATEAU</td>
<td>2898</td>
</tr>
<tr>
<td>SUM_SURF_STEADY</td>
<td>2902</td>
</tr>
<tr>
<td>SUM_SURF_STEADY.SEQUENCE</td>
<td>2906</td>
</tr>
<tr>
<td>SUM_SURF.Strictly_decending_sequence</td>
<td>2910</td>
</tr>
<tr>
<td>SUM_SURF.Strictly_increasing_sequence</td>
<td>2914</td>
</tr>
<tr>
<td>SUM_SURF_SUMMIT</td>
<td>2918</td>
</tr>
<tr>
<td>SUM_SURF_VALLEY</td>
<td>2924</td>
</tr>
<tr>
<td>SUM_SURF_ZIGZAG</td>
<td>2928</td>
</tr>
<tr>
<td>SUM_WIDTH_DECENDINGSEQUENCE</td>
<td>2936</td>
</tr>
<tr>
<td>SUM_WIDTH_DECENDING_TERRACE</td>
<td>2942</td>
</tr>
<tr>
<td>SUM_WIDTH_GORGE</td>
<td>2948</td>
</tr>
<tr>
<td>SUM_WIDTH_INCREASINGSEQUENCE</td>
<td>2956</td>
</tr>
<tr>
<td>SUM_WIDTH_INCREASING_TERRACE</td>
<td>2962</td>
</tr>
<tr>
<td>SUM_WIDTH_INFLEXION</td>
<td>2968</td>
</tr>
<tr>
<td>SUM_WIDTH_PEAK</td>
<td>2974</td>
</tr>
<tr>
<td>SUM_WIDTH_PLAIN</td>
<td>2980</td>
</tr>
<tr>
<td>CONTENTS</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>SUM_WIDTH_PLATEAU .......................... 2986</td>
<td></td>
</tr>
<tr>
<td>SUM_WIDTH_PROPER_PLAIN ..................... 2992</td>
<td></td>
</tr>
<tr>
<td>SUM_WIDTH_PROPER_PLATEAU ................... 2998</td>
<td></td>
</tr>
<tr>
<td>SUM_WIDTH_STEADY_SEQUENCE .................. 3004</td>
<td></td>
</tr>
<tr>
<td>SUM_WIDTH STRICTLY_DECREASING_SEQUENCE .... 3010</td>
<td></td>
</tr>
<tr>
<td>SUM_WIDTH STRICTLY_INCREASING_SEQUENCE ..... 3016</td>
<td></td>
</tr>
<tr>
<td>SUM_WIDTH_SUMMIT ........................... 3022</td>
<td></td>
</tr>
<tr>
<td>SUM_WIDTH_VALLEY ............................ 3030</td>
<td></td>
</tr>
<tr>
<td>SUM_WIDTH_ZIGZAG ............................ 3036</td>
<td></td>
</tr>
<tr>
<td>SURF_BUMP_ON_DECREASING_SEQUENCE .......... 3046</td>
<td></td>
</tr>
<tr>
<td>SURF_DECREASING ................................ 3050</td>
<td></td>
</tr>
<tr>
<td>SURF_DECREASING_SEQUENCE ................... 3054</td>
<td></td>
</tr>
<tr>
<td>SURF_DECREASING_TERRACE .................... 3058</td>
<td></td>
</tr>
<tr>
<td>SURF_DIP_ON_INCREASING_SEQUENCE ............ 3062</td>
<td></td>
</tr>
<tr>
<td>SURF_GORGE ................................... 3066</td>
<td></td>
</tr>
<tr>
<td>SURF_INCREASING ............................. 3070</td>
<td></td>
</tr>
<tr>
<td>SURF_INCREASING_SEQUENCE .................... 3074</td>
<td></td>
</tr>
<tr>
<td>SURF_INCREASING_TERRACE ..................... 3078</td>
<td></td>
</tr>
<tr>
<td>SURF_INFLEXION ................................ 3082</td>
<td></td>
</tr>
<tr>
<td>SURF_PEAK ..................................... 3086</td>
<td></td>
</tr>
<tr>
<td>SURF_PLAIN ..................................... 3090</td>
<td></td>
</tr>
<tr>
<td>SURF_PLATEAU .................................. 3094</td>
<td></td>
</tr>
<tr>
<td>SURF_PROPER_PLAIN ............................ 3098</td>
<td></td>
</tr>
<tr>
<td>SURF_PROPER_PLATEAU ......................... 3102</td>
<td></td>
</tr>
<tr>
<td>SURF_STEADY .................................... 3106</td>
<td></td>
</tr>
<tr>
<td>SURF_STEADY_SEQUENCE ....................... 3110</td>
<td></td>
</tr>
<tr>
<td>SURF_STRICTLY_DECREASING_SEQUENCE .......... 3114</td>
<td></td>
</tr>
<tr>
<td>SURF_STRICTLY_INCREASING_SEQUENCE .......... 3118</td>
<td></td>
</tr>
<tr>
<td>SURF_SUMMIT .................................... 3122</td>
<td></td>
</tr>
<tr>
<td>SURF_VALLEY ..................................... 3126</td>
<td></td>
</tr>
<tr>
<td>SURF_ZIGZAG .................................... 3130</td>
<td></td>
</tr>
<tr>
<td>WIDTH_DECREASING_SEQUENCE .................. 3134</td>
<td></td>
</tr>
<tr>
<td>WIDTH_DECREASING_TERRACE .................... 3138</td>
<td></td>
</tr>
<tr>
<td>WIDTH_GORGE .................................... 3142</td>
<td></td>
</tr>
<tr>
<td>WIDTH_INCREASING_SEQUENCE ................... 3146</td>
<td></td>
</tr>
<tr>
<td>WIDTH_INCREASING_TERRACE ..................... 3150</td>
<td></td>
</tr>
<tr>
<td>WIDTH_INFLEXION ................................ 3154</td>
<td></td>
</tr>
<tr>
<td>WIDTH_PEAK ..................................... 3158</td>
<td></td>
</tr>
<tr>
<td>WIDTH_PLAIN .................................... 3162</td>
<td></td>
</tr>
<tr>
<td>WIDTH_PLATEAU .................................. 3166</td>
<td></td>
</tr>
<tr>
<td>WIDTH_PROPER_PLAIN ............................ 3170</td>
<td></td>
</tr>
<tr>
<td>WIDTH_PROPER_PLATEAU ......................... 3174</td>
<td></td>
</tr>
<tr>
<td>WIDTH_STEADY_SEQUENCE ....................... 3178</td>
<td></td>
</tr>
<tr>
<td>WIDTH_STRICTLY_DECREASING_SEQUENCE .......... 3182</td>
<td></td>
</tr>
<tr>
<td>WIDTH_STRICTLY_INCREASING_SEQUENCE .......... 3186</td>
<td></td>
</tr>
<tr>
<td>WIDTH_SUMMIT .................................... 3190</td>
<td></td>
</tr>
</tbody>
</table>
CONTENTS

WIDTH_VALLEY ................................................................. 3194
WIDTH_ZIGZAG ............................................................. 3198

5 A Database of Parameterised Invariants 3201

A Electronic Constraint Catalogue 3699

Bibliography 3701

Index 3703
Preface

“Efficiency can only be attained through generality.”

This second volume of the Global Constraint Catalogue [10] is devoted to time-series constraints. Within the context of Constraint Programming, time-series constraints go back to the work of Goldin and Kanellakis [14]. This volume contains 719 constraints, which are explicitly described in terms of automata with registers [7]. Checkers and propagators for all these constraints were synthesised [8] from 22 transducers [20, 12, 16, 19] given in the second chapter of this volume.

As in the first volume, the global constraints described in this second volume are not only accessible to humans, who can read the catalogue when searching for some information. It is also available to machines, which can read and interpret it. This is why there also exists an electronic version of this catalogue where one can get, for all time-series constraints, a complete description in terms of metadata used in the first volume. In fact, unlike the first volume, all the metadata of the electronic version, as well as all text and figures of this second volume, were automatically generated. While this second volume is by no means supposed to contain all possible time-series constraints, it contributes in the context of time-series constraints to the systematic reconstruction of the Global Constraint Catalogue that we have previously advocated [11]. This reconstruction is based on the following methodology:

• First reuse, adapt or come up with abstractions, which allow to concisely represent structures and properties of time series as abstract combinatorial objects. In our context these abstractions essentially correspond to:

1. Transducers where letters of the output alphabet are interpreted as phase letters indicating the different steps for recognising pattern occurrences denoted by regular expressions.¹

2. Transducers glue matrices expressing the relationship between the prefix, the suffix and the full sequence passed to a transducer.

3. Properties associated with regular expressions corresponding to fragments of the input language of our transducers.

¹Note that an approach developed by R. Alur named qualitative regular expressions is presented in [1]. This approach focuses essentially on the mining aspect: given a fixed time series, the question is how to efficiently evaluate a quantitative regular function. However, it does not consider the question of generating a time series verifying a conjunction of quantitative regular functions.
• Second, create from these abstract combinatorial objects a database of concrete combinatorial objects.

• Third, synthesise concrete code for various technologies, languages, tasks from this database. In this context, correctness and efficiency [15] of the synthesised code are essentially side product of:
  – The correctness of the formulae of our database, which is itself based on well-formed abstractions.
  – The generality behind our abstract combinatorial objects.

The time-series catalogue is done in the following way:

• All time-series constraints are now defined in a compositionally way from a few basic constituents, i.e., patterns, features, aggregators, and predicates, which completely define the meaning of a constraint, where patterns are defined using regular expressions.

• Constraint names are now constructed in a systematic way as the concatenation of pattern name, feature name, and aggregation or predicate name.

• Given a pattern \( p \), checkers and constraints are now systematically synthesised from a transducer that, given an input sequence over the input alphabet \( \{<, =, >\} \), compares two adjacent values of a time series and determines an output sequence over an output phase alphabet describing how to recognise the occurrences of \( p \).

• For each time-series constraint associated with a pattern \( p \), the generation of an automaton with registers is completely driven by the transducer associated with the pattern \( p \), as well as by decoration tables describing for each phase letter of the output alphabet of the transducers how to generate register updates. Code optimisation is ensured by using decoration tables that depend on properties of the pattern, of the feature, and of the aggregator associated with the time-series constraint.

• Lower and upper bounds of characteristics of time series that appear in the restriction slot of a time-series constraint are synthesised from a few parameterised formulae that only depend on a restricted set of characteristics of the regular expression associated with the pattern [6].

• Parametrised glue matrices are provided for each transducer that corresponds to reversible time-series constraints. A concrete glue matrix is given for each reversible time-series constraint.

• Linear invariants are systematically obtained by applying the Farkas Lemma [13] to the automata with registers that were synthesised. They consist of linear constraints typically linking consecutive register values, e.g., see the legend of the second automaton of the MAX_MAX_PEAK, MAX_RANGE_DECREASING, MAX_RANGE_INCREASING, MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE,
MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE, MIN_MAX_PEAK, MIN_WIDTH_PLAIN, MIN_WIDTH_PLATEAU, NB_BUMP_ON_DECREASING_SEQUENCE, NB_DIP_ON_INCREASING_SEQUENCE, NB_GORGE, NB_PEAK, NB_SUMMIT, NB_VALLEY, and NB_ZIGZAG constraints, which are generated even with non-linear register updates.

• Last but not least, time-series constraints were used for generating time series verifying a conjunction of constraints both in the context of Constraint Programming [3] and in the context of Linear Programming. The latter gives a linear reformulation [2].

• In the context of sequential pattern mining, time-series constraint checkers can be used to identify and extract patterns from fixed sequences. Specialised code was synthesised for SICStus, C and Java. While the time-series catalogue may need to be extended in order to capture more patterns, having a possibly large set of fixed time-series constraints is a natural safeguard to prevent overfitting when dealing with few sequences, at a price of not finding patterns that are not covered by the catalogue.

• SICStus code is synthesised and MiniZinc code is available. The latter allows using time-series constraints on many platforms such as Choco, Gecode, OR-tools, Cplex or Gurobi and is available from the Electronic Constraint Catalogue in Appendix A of this document.

The catalogue contains the following types of figures, which were all synthesised by computer programs producing TikZ [21] code:

• Figures representing the logo of each pattern.

• Figures for visualising the transducer associated with each pattern. The following scheme was consistently used over all transducers:
  
  – An arc labelled by the output symbol out, maybe, found, or in is respectively coloured in red, orange, blue, and violet.
  
  – If all input transitions of a state have the same colour, then the state uses this colour, otherwise it is coloured in black.
  
  – The name of a state is preceded by all input symbols of the transitions that enter this state. We use special characters for representing combinations of input symbols, e.g. \( \leq \) for \(<\) and \(=\).

• Figures giving time series illustrating lower and upper bounds of the VALUE argument of a time-series constraint.

• Figures illustrating each constraint on a relevant time series.

• Figures providing the synthesised automaton with registers associated with each constraint. Such automata use the same graphical scheme as the transducers from which they were generated.
• Figures displaying the linear and non-linear invariants linking the result variables of two time-series constraints.

In order to see PDF annotations and attached files, you are advised to use Adobe Acrobat Reader with the PDF version of the catalogue.\footnote{Since we are using the \LaTeX\ packages \texttt{pdfcomment} and \texttt{attachfile} and since most PDF viewers do not support PDF annotations or attachments.} If you do not see on your screen a small yellow bullet at the beginning of this paragraph, you are using a PDF viewer that does not fully support PDF annotations. To get started the reader should first consult Chapter 2 which presents a short overview of the available constraint classes of this volume, their naming conventions and their arguments, and some of their potential uses. Then to get a more precise idea of the different objects used in this volume the reader may look at Chapter 3, which defines the notions of a pattern, of a feature, of an aggregator, of a seed transducer and of a glue matrix used throughout this document, as well as the conventions used for drawing time series.

The initial work on synthesising automata with registers from transducers was done by N. Beldiceanu, R. Douence, and H. Simonis. The generation of the catalogue was done by H. Simonis except, for most examples and figures, which were handled by N. Beldiceanu. The code generation part dealing with constraint checkers was rewritten by M. Carlsson for getting optimised checkers that do not use any constraint. The creation of dedicated decoration tables that depend on properties of patterns, features, and aggregators was done by E. Arafailova under the supervision of N. Beldiceanu and R. Douence \cite{3}. The bounds based on regular expression characteristics \cite{6}, the among implied constraints \cite{4}, the linear \cite{5} and non-linear invariants linking the results variables of two time-series constraints, and the definition of constant size finite automata encoding a restriction on the result variable of a time-series constraint were done by E. Arafailova under the supervision of N. Beldiceanu; the generation of the corresponding code was done by H. Simonis. The generation of linear invariants linking consecutive register values of a register automaton \cite{17, 3}, the generation of parametrised glue matrices \cite{9, 2}, and the generation of seed transducers from their regular expressions \cite{18} were done by M. A. Francisco Rodríguez under the supervision of P. Flener and J. Pearson.

The authors have benefited from the following support:

• E. Arafailova and R. Douence were supported by the European H2020 FETPROACT-2014 project “GRACeFUL” (project number 640954).

• From 2014 to 2018 N. Beldiceanu was partially supported by the Gaspard Monge Programme for Optimisation and Operations Research (PGMO), as well as by the European H2020 FETPROACT-2014 project “GRACeFUL”. In 2018 N. Beldiceanu was also partially supported by the HYDDA project.

• M. Carlsson was supported by SICS.

• P. Flener, J. Pearson and M. A. Francisco Rodríguez were supported by grants 2011-6133 and 2012-4908 of the Swedish Research Council (VR).
During 2014 H. Simonis was partially supported by a senior researcher chair by the Région Pays de la Loire and by the EU FET grant ICON (project number 284715). From 2014 to 2016 H. Simonis was also supported by Science Foundation Ireland under Grant Number SFI/10/IN.1/I3032. The INSIGHT Centre for Data Analytics is supported by Science Foundation Ireland under Grant Number SFI/12/RC/2289.

Readers may send their suggestions via email to the corresponding author with catalogue as subject.

Cork, Ireland, Nantes, France, Uppsala, Sweden, September 2018
— EA, NB, RD, MC, PF, MAFR, JP, HS
PREFACE
1

Describing Global Constraints

Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Basic data types</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Compound data types</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Restrictions</td>
<td>3</td>
</tr>
<tr>
<td>1.4 Declaring a global constraint</td>
<td>8</td>
</tr>
<tr>
<td>1.5 Keywords</td>
<td>8</td>
</tr>
</tbody>
</table>

This chapter, taken from the first volume of the global constraint catalogue [10], recalls how to describe a global constraint and its arguments.

Since global constraints have to receive their arguments in some form, no matter whether we use the graph-based or automaton-based description, we start by describing the abstract data types that we use in order to specify the arguments of a global constraint. These abstract data types are not related to any specific programming language like Caml, C, C++, Java, or Prolog. If one wants to focus on a specific language, then one has to map these abstract data types to the data types that are available in the considered language. Second we describe all the restrictions that one can impose on the arguments of a global constraint. Finally, we show how to use these ingredients in order to declare the arguments of a global constraint.

1.1 Basic data types

We use the following basic data types:

- **atom** corresponds to an atom. Predefined atoms are MININT and MAXINT, which respectively correspond to the smallest integer and largest integer.

- **int** corresponds to an integer value.

- **dvar** corresponds to a domain variable. A domain variable \( V \) is a variable that will be assigned an integer value taken from an initial finite set of integer values.
1. DESCRIBING GLOBAL CONSTRAINTS

denoted by \( \text{dom}(V) \). \( V^- \) and \( V^+ \) respectively denote the minimum and maximum values of \( \text{dom}(V) \).

- **fdvar** corresponds to a possibly unbounded domain variable. A possibly unbounded domain variable is a variable that will be assigned an integer value from an initial finite set of integer values denoted by \( \text{dom}(V) \) or from \( \infty, +\infty \).

- **sint** corresponds to a finite set of integer values.

- **svar** corresponds to a set variable. A set variable \( V \) is a variable that will be assigned to a finite set of integer values. Its lower bound \( V^- \) denotes the set of integer values that for sure belong to \( V \), while its upper bound \( V^+ \) denotes the set of integer values that may belong to \( V \). Let \( \text{dom}(V) = \{v_1, \ldots, v_n, v_{n+1}, \ldots, v_m\} \) be a shortcut for combining the lower and upper bounds of \( V \) in a single notation:
  - **Bold** values designate those values that only belong to \( V^- \).
  - **Italic** values indicate those values that belong only to \( V^+ \).

- **mint** corresponds to a multiset of integer values.

- **mvar** corresponds to a multiset variable. A multiset variable is a variable that will be assigned to a multiset of integer values.

- **real** corresponds to a real number.

- **rvar** corresponds to a real number variable. A real number variable is a variable that will be assigned a real number taken from an initial finite set of intervals. A real number is usually represented by an interval of two floating-point numbers.

1.2 Compound data types

We use the following compound data types:

- **list(\( T \))** corresponds to a list of elements of the type \( T \), where \( T \) is a basic or compound data type.

- **collection(\( A_1, A_2, \ldots, A_n \))** corresponds to a collection of ordered items, where each item consists of \( n > 0 \) attributes \( A_1, A_2, \ldots, A_n \). Each attribute is an expression of the form \( a - T \), where \( a \) is the name of the attribute and \( T \) the type of the attribute (a basic or compound data type). All names of the attributes of a given collection should be distinct, and different from the keyword key, which corresponds to an implicit\(^1\) attribute. The value of the key attribute is the position of an item within the collection. The first item of a collection is associated with position 1.

\(^1\)This attribute is not explicitly defined.
1.3. RESTRICTIONS

The following notation is used for instantiated arguments:

- A list of elements \( e_1, e_2, \ldots, e_n \) is denoted by \([e_1, e_2, \ldots, e_n]\).
- A finite set of integers \( i_1, i_2, \ldots, i_n \) is denoted by \( \{i_1, i_2, \ldots, i_n\} \).
- A collection of \( n \) items, each item having \( m \) attributes, is denoted by \( \langle a_1 - v_{11}, a_1 - v_{12}, \ldots, a_1 - v_{1m}, a_2 - v_{21}, a_2 - v_{22}, \ldots, a_2 - v_{2m}, \ldots, a_n - v_{n1}, a_n - v_{n2}, \ldots, a_n - v_{nm} \rangle \). Each item is separated from the previous item by a comma. If the items of the collection involve a single attribute \( a_1 \), then \( \langle v_{11}, v_{21}, \ldots, v_{n1} \rangle \) can be used as a shortcut for \( \langle a_1 - v_{11}, a_1 - v_{21}, \ldots, a_1 - v_{n1} \rangle \).
- Item \( i \) of a collection \( c \) is denoted by \( c[i] \).
- The value of the attribute \( a \) of the \( i \)th item of a collection \( c \) is denoted by \( c[i].a \). Note that, within an arithmetic expression, we can use the shortcut \( c[i] \) when the collection \( c \) involves a single attribute.
- The number of items of a collection \( c \) is denoted by \( |c| \).

1.3 Restrictions

When defining the arguments of a global constraint, it is often the case that one needs to express additional conditions that refine the type declarations of its arguments. For this purpose we provide restrictions that allow the specification of these additional conditions. Currently the list of restrictions is:

- **in_list(Arg, ListAtoms)**
  - Arg is an argument of the type atom,
  - ListAtoms is a non-empty list of distinct atoms.
  This restriction enforces Arg to be one of the atoms specified in ListAtoms.

- **in_list(Arg, Attr, ListIntOrAtom)**
  - Arg is an argument of the type collection,
  - Attr is an attribute of the type int or atom of Arg.
  - If Attr is an attribute of the type int, then ListIntOrAtom is a non-empty list of distinct integers; if Attr is an attribute of the type atom, then ListIntOrAtom is a non-empty list of distinct atoms.
  This restriction enforces, for all items of Arg, the attribute Attr to take its value within ListIntOrAtom.

- **in_attr(Arg1, Attr1, Arg2, Attr2)**
  - Arg1 is an argument of the type collection,
1. DESCRIBING GLOBAL CONSTRAINTS

- Attr1 is an attribute of the type dvar or int of Arg1.
- Arg2 is an argument of the type collection.
- Attr2 is an attribute of the type int of Arg2.

Let $S_2$ denote the set of values assigned to the Attr2 attributes of the items of Arg2. This restriction enforces the following condition: for all items of Arg1, the attribute Attr1 takes its value in the set $S_2$.

- **distinct(Arg,Attrs)**
  - Arg is an argument of the type collection,
  - Attrs is an attribute of the type int or dvar, or a (possibly empty) list of distinct attributes of the type int or dvar of Arg.

For each pair of distinct items of Arg, this restriction enforces that there be at least one attribute specified by Attrs with two distinct values. If Attrs is the empty list, then all items of Arg should be distinct.

- **increasing_seq(Arg,Attrs)**
  - Arg is an argument of the type collection,
  - Attrs is an attribute of the type int or a list of distinct attributes of the type int of Arg.

Let $n$ and $m$ respectively denote the number of items of Arg, and the number of attributes of Attrs. For item $i$ of Arg let $t_i$ denote the tuple of values $(v_{i,1}, v_{i,2}, \ldots, v_{i,m})$ where $v_{i,j}$ is the value of attribute $j$ of Attrs of item $i$ of Arg. The restriction enforces a strict lexicographical ordering on the tuples $t_1, t_2, \ldots, t_n$.

- **non_increasing_size(Arg,Attr)**
  - Arg is an argument of the type collection,
  - Attr is an attribute of the type collection of Arg.

This restriction enforces for each pair of consecutive items Arg[$i$], Arg[$i + 1$] that the number of items of Arg[$i$].Attr is greater than or equal to the number of items of the collection Arg[$i + 1$].Attr.

- **required(Arg,Attrs)**
  - Arg is an argument of the type collection,
  - Attrs is an attribute or a list of distinct attributes of Arg.

This restriction enforces all attributes denoted by Attrs to be explicitly used within all items of Arg.

This restriction is usually systematically used for every attribute of a collection. It is not used when some attributes may be implicitly defined according to other...
attributes. In this context, we use the `require_at_least` restriction, which we now introduce.

- **require_at_least(Atleast, Arg, Attrs)**
  - `Atleast` is a positive integer,
  - `Arg` is an argument of the type collection,
  - `Attrs` is a non-empty list of distinct attributes of `Arg`. The length of this list should be strictly greater than `Atleast`.

This restriction enforces that at least `Atleast` attributes of the list `Attrs` be explicitly used within all items of `Arg`.

- **same_size(Arg, Attr)**
  - `Arg` is an argument of the type collection,
  - `Attr` is an attribute of the type collection of `Arg`.

This restriction enforces that all collections of `Attr` have the same number of items.

- **Term₁ Comparison Term₂**
  - `Term₁` is a term. A term is an expression that can be evaluated to one or possibly several integer values.
  - Comparison is one of the comparison operators `≤, ≥, <, >, =, ≠`.
  - `Term₂` is a term.

Let `v₁,₁, v₁,₂, ..., v₁,n₁` and `v₂,₁, v₂,₂, ..., v₂,n₂` be the values respectively associated with `Term₁` and `Term₂`. The restriction `Term₁ Comparison Term₂` enforces `v₁,i Comparison v₂,j` to hold for every `i ∈ [1, n₁]` and every `j ∈ [1, n₂]`.

A term is one of the following expressions:

- `e`, where `e` is an integer. The corresponding value is `e`.
- `|c|`, where `c` is an argument of the type collection. The value of `|c|` is the number of items of the collection `c`.
- `first(c,a)` : `first(c,a)` denotes the value assigned to the attribute `a` of the first item of the collection `c`. It is equal to 0 if the collection is empty.
- `last(c,a)` : `last(c,a)` denotes the value assigned to the attribute `a` of the last item of the collection `c`. It is equal to 0 if the collection is empty.
- `sum(c,a)` : `sum(c,a)` denotes the sum of the values assigned to the attribute `a` of the collection `c`; it is equal to 0 if the collection is empty.
- `sum(ℓ)` : `sum(ℓ)`, where `ℓ` is a list of collection attributes, each of the form `cᵢ,aᵢ` (with `i ∈ [1, n]`), is the sum of the values assigned to the attributes `aᵢ` of the collections `cᵢ` (with `i ∈ [1, n]`).
1. DESCRIBING GLOBAL CONSTRAINTS

- \text{range}(c.a) : \text{range}(c.a)\text{ denotes the difference between the maximum value and minimum value plus one of the values assigned to the attribute } a \text{ of } c; \text{ it is equal to 0 if the collection is empty.}

- \text{range}(\ell) : \text{range}(\ell), \text{ where } \ell \text{ is a list of collection attributes, each of the form } c_i.a_i \text{ (with } i \in [1,n]), \text{ is the difference between the maximum value and minimum value plus one of the values assigned to the attributes } a_i \text{ of the } c_i \text{ (with } i \in [1,n]).

- \text{minval}(c.a) : \text{minval}(c.a)\text{ denotes the minimum of the values assigned to the attribute } a \text{ of the collection denoted by } c; \text{ it is equal to 0 if the collection is empty.}

- \text{minval}(\ell) : \text{minval}(\ell), \text{ where } \ell \text{ is a list of collection attributes, each of the form } c_i.a_i \text{ (with } i \in [1,n]), \text{ is the minimum of the values assigned to the attributes } a_i \text{ of the } c_i \text{ (with } i \in [1,n]).

- \text{maxval}(c.a) : \text{maxval}(c.a)\text{ denotes the maximum of the values assigned to the attribute } a \text{ of } c; \text{ it is equal to 0 if the collection is empty.}

- \text{maxval}(\ell) : \text{maxval}(\ell), \text{ where } \ell \text{ is a list of collection attributes, each of the form } c_i.a_i \text{ (with } i \in [1,n]), \text{ is the maximum of the values assigned to the attributes } a_i \text{ of the } c_i \text{ (with } i \in [1,n]).

- \text{nval}(c.a) : \text{nval}(c.a)\text{ denotes the number of distinct values over the values assigned to the attribute } a \text{ of } c; \text{ it is equal to 0 if the collection is empty.}

- \text{nval}(\ell) : \text{nval}(\ell), \text{ where } \ell \text{ is a list of collection attributes, each of the form } c_i.a_i \text{ (with } i \in [1,n]), \text{ is the number of distinct values over the values assigned to the attributes } a_i \text{ of the } c_i \text{ (with } i \in [1,n]).

- \text{prod}(c.a) : \text{prod}(c.a)\text{ denotes the product of the values assigned to the attribute } a \text{ of } c; \text{ it is equal to 1 if the collection is empty.}

- \text{prod}(\ell) : \text{prod}(\ell), \text{ where } \ell \text{ is a list of collection attributes, each of the form } c_i.a_i \text{ (with } i \in [1,n]), \text{ is the product of the values assigned to the attributes } a_i \text{ of the } c_i \text{ (with } i \in [1,n]).

- \text{pfdc}(c.a) : \text{pfdc}(c.a), \text{ where } \text{pfdc} \text{ is a pure functional dependency constraint of the form } \text{pfdc}(v, col) \text{ that computes a value } v \text{ from a collection of variables } col, \text{ and where } c.a \text{ is a collection with attribute } a \text{ denotes the pfdc of the values assigned to the attribute } a \text{ of } c; \text{ it is equal to 0 if the collection } c \text{ is empty.}

- \text{pfdc}(\ell) : \text{pfdc}(\ell), \text{ where } \text{pfdc} \text{ is a pure functional dependency constraint of the form } \text{pfdc}(v, col) \text{ that computes a value } v \text{ from a collection of variables } col, \text{ and where } \ell \text{ is a list of collection attributes, each of the form } c_i.a_i \text{ (with } i \in [1,n]), \text{ is the pfdc of the values assigned to the attributes } a_i \text{ of the } c_i \text{ (with } i \in [1,n]).
1.3. RESTRICTIONS

- \( t \), where \( t \) is an argument of the type int. The value of \( t \) is the value of the corresponding argument.

- \( v \), where \( v \) is an argument of the type dvar. The value of \( v \) is the value assigned to variable \( v \). Note that restrictions are defined on the ground instance of a global constraint.

- \( s \), where \( s \) is an argument of the type sint or svar. The values denoted by \( s \) are all the values of the corresponding set.

- \( c.a \), where \( c \) is an argument of the type collection, and \( a \) an attribute of \( c \) of the type int or dvar. The values denoted by \( c.a \) are all the values corresponding to attribute \( a \) for the items of \( c \). When \( c.a \) designates a domain variable we consider the value assigned to that variable.

- \( c.a \), where \( c \) is an argument of the type collection, and \( a \) an attribute of \( c \) of the type sint or svar. The values denoted by \( c.a \) are all the values belonging to the sets corresponding to attribute \( a \) for the items of \( c \). When \( c.a \) designates a set variable we consider the values that belong to that set.

- \( t_1 \text{ op } t_2 \), where \( t_1 \) and \( t_2 \) are terms and \( \text{op} \) one of the operators +, −, ∗, / or mod. Let \( V_1 \) and \( V_2 \) denote the sets of values respectively associated with the terms \( t_1 \) and \( t_2 \). The set of values associated with \( t_1 \text{ op } t_2 \) is \( V_{12} = \{ v : v = v_1 \text{ op } v_2, v_1 \in V_1, v_2 \in V_2 \} \).

- \(|t|\), where \( t \) is a term. Let \( V \) denote the set of values associated with the term \( t \). The set of values associated with \(|t|\) is \( V = \{ v : v = |val|, val \in V \} \).

- \( \min_{v \in [\ell,u]}(t) \), where \( v \) is a variable that occurs in the term \( t \), and \( \ell, u \) are two terms. The value of \( \min_{v \in [\ell,u]}(t) \) is the smallest value among all possible values of the term \( t \) where \( v \) is assigned a value in the interval \([\ell,u]\).

- \( \max_{v \in [\ell,u]}(t) \), where \( v \) is a variable that occurs in the term \( t \), and \( \ell, u \) are two terms. The value of \( \max_{v \in [\ell,u]}(t) \) is the largest value among all possible values of the term \( t \) where \( v \) is assigned a value in the interval \([\ell,u]\).

- Using the \( \lor \) connector we can express a disjunction between two restrictions.

- Using the \( \Rightarrow \) connector we can express an implication between a restriction (or a conjunction of restrictions) and a restriction.

- We can also use a constraint \( C \) of the catalogue for expressing a restriction as long as that constraint is not defined according to the constraint under consideration. The constraint \( C \) should have a graph-based or automaton-based description so that its meaning is explicitly defined.

For instance the AMONG(NVAR, VARIABLES, VALUES) constraint holds if \( \text{NVAR} \) is the number of variables of the collection VARIABLES that are assigned a value from the collection of values VALUES.

\( \text{mod} \) denotes a floor division, that is a division in which the result is rounded to the nearest integer that is smaller or equal.
Within the restriction slot it is possible to define local variables to designate intermediate expressions. This is done by using the \texttt{where} statement after the set of restrictions.

### 1.4 Declaring a global constraint

Declaring a global constraint consists of providing the following information:

- A \texttt{term constraint($A_1, A_2, \ldots, A_n$)}, where \texttt{constraint} corresponds to the \texttt{name} of the global constraint and \texttt{$A_1, A_2, \ldots, A_n$} to its \texttt{arguments}.

- A possibly empty \texttt{list of type declarations}, where each declaration has the form \texttt{type:type declaration: type} is the \texttt{name} of the new type we define and \texttt{type declaration} is a basic data type, a compound data type, or a previously defined type.

- An \texttt{argument declaration $A_1:T_1, A_2:T_2, \ldots, A_n:T_n$} giving for each argument \texttt{$A_1, A_2, \ldots, A_n$} of the global constraint \texttt{constraint} its type. Each type is a basic data type, a compound data type, or a type that was declared in the list of type declarations.

- A possibly empty \texttt{list of restrictions}, where each restriction is one of the restrictions described in Section 1.3.

### 1.5 Keywords

This section lists the keywords used in the time-series catalogue.

- \texttt{added}: Denotes the fact that, even though the same constant is added to all variables of the \texttt{VARIABLES} collection, the corresponding constraint still holds.

- \texttt{functional dependency}: A constraint that allows for representing a functional dependency between possibly several domain variables and a single domain variable. A sequence of variables \texttt{$X_1, X_2, \ldots, X_n$} is said to functionally determine another variable \texttt{$Y$} if and only if each potential tuple of values of \texttt{$X_1, X_2, \ldots, X_n$} is associated with exactly one potential value of \texttt{$Y$} (i.e., \texttt{$Y$} is a function of \texttt{$X_1, X_2, \ldots, X_n$}).

- \texttt{reversed}: Denotes the fact that, even though the variables of the \texttt{VARIABLES} collection are reversed, the corresponding constraint still holds.

- \texttt{reverse of a constraint}: A constraint which has a reverse constraint, where the reverse is defined in the following way. Consider two constraints \texttt{$c(col, r_1, \ldots, r_n)$} and \texttt{$c'(col, r_1, \ldots, r_n)$} for which, in both cases, the argument \texttt{col} is a collection of items that functionally determines all the other arguments \texttt{$r_1, \ldots, r_n$}.

The constraint \texttt{$c'$} is the \texttt{reverse constraint} of constraint \texttt{$c$} if, for any collection of items \texttt{col}, we have the equivalence \texttt{$c(col, r_1, \ldots, r_n) \Leftrightarrow c'(col^{rev}, r_1, \ldots, r_n)$}.
where \( \text{col}^{rev} \) denotes the collection \( \text{col} \) where the items of the collection are reversed. When constraints \( c \) and \( c' \) are identical, we say that constraint \( c \) is its own reverse.
1. DESCRIBING GLOBAL CONSTRAINTS
2

Overview of Time-Series Constraints

Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Predicate</td>
<td>13</td>
</tr>
<tr>
<td>2.2</td>
<td>Function</td>
<td>14</td>
</tr>
<tr>
<td>2.3</td>
<td>Feature</td>
<td>14</td>
</tr>
<tr>
<td>2.4</td>
<td>Extreme</td>
<td>16</td>
</tr>
</tbody>
</table>

This chapter provides a brief overview of the available constraint classes of this volume. Each constraint class is described in the following way:

- We give the naming convention used to construct systematically the constraint names.
- We provide the common argument pattern of all constraints belonging to a class.
- We state the purpose of the constraints of a class.
- We illustrate the use of such constraints with an example where we compute some information from a fixed time series, and another example where we generate time series satisfying a given property.

No matter to which class they belong, all constraints of this volume have a finite time series \( X = (x_1, x_2, \ldots, x_n) \) as one of their arguments, where each \( x_i \) is a fixed integer or a domain variable (see Section 1.1). We derive from \( X \) the sequence of signature values \( S = (s_1, s_2, \ldots, s_{n-1}) \) via the signature constraints \( x_i < x_{i+1} \iff s_i = '<' \) \( \land (x_i = x_{i+1} \iff s_i = '=' \) \( \land (x_i > x_{i+1} \iff s_i = '>' \) for all \( i \in [1, n - 1] \).

Names of constraints are systematically constructed using the following grammar, where terminals use lowercase that are shown in quotation marks, even if all constraint names will be capitalised.
2. OVERVIEW OF TIME-SERIES CONSTRAINTS

| CONDITION | ::= | ‘all_equal’|‘decreasing’|‘increasing’ |
| EXTREMEAGG | ::= | ‘max’|‘min’ |
| AGGREGATOR | ::= | EXTREMEAGG |‘sun’ |
| NONCONSTANTFEAT | ::= | ‘max’|‘min’|‘range’|‘surf’|‘width’ |
| FEATURE | ::= | NONCONSTANTFEAT |‘one’ |
| MONOTONICPATTERN | ::= | ‘decreasing’ |
| | | ‘decreasing_sequence’ |
| | | ‘increasing’ |
| | | ‘increasing_sequence’ |
| | | ‘strictly_decreasing_sequence’ |
| | | ‘strictly_increasing_sequence’ |
| CONSTANTPATTERN | ::= | ‘decreasing_terrace’ |
| | | ‘increasing_terrace’ |
| | | ‘plain’ |
| | | ‘plateau’ |
| | | ‘proper_plain’ |
| | | ‘proper_plateau’ |
| | | ‘steady’ |
| | | ‘steady_sequence’ |
| PATTERN | ::= | MONOTONICPATTERN |
| | | CONSTANTPATTERN |
| | | ‘bump_on_decreasing_sequence’ |
| | | ‘dip_on_increasing_sequence’ |
| | | ‘gorge’ |
| | | ‘inflexion’ |
| | | ‘peak’ |
| | | ‘summit’ |
| | | ‘valley’ |
| | | ‘zigzag’ |

An extended presentation of the semantics of patterns, features, and aggregators can be found in Chapter 3. Also, to get a more precise idea of how constraint arguments are described, refer to Chapter 1.

Note that in what follows, the ‘range’ feature is only used with monotonic patterns, and the ‘min’ feature is renamed ‘height’ when used together with constant patterns.

The time-series constraint catalogue contains four constraints classes respectively called predicate, function, feature, and extreme, which we now describe. All examples of this chapter use the ‘peak’ pattern, where a peak is a maximal occurrence of the regular expression ‘< (< | =)* (> | =)* >’ in the sequence of signature values associated with a time series.
2.1 Predicate

The class of constraints $\text{PREDICATENAME(VARIABLES)}$ enforces a binary comparison operator to hold between the feature values of consecutive maximal pattern occurrences in a time series given as a collection $\text{VARIABLES}$ of variables. The name of such a constraint is constructed in the following way:

$\text{PREDICATENAME} ::= \text{CONDITION} \ '_-' \text{FEATURE} ' '_\text{PATTERN}$

where $\text{CONDITION}$ gives the binary comparison operator, i.e. ‘all_equal’, ‘decreasing’, ‘increasing’, used for comparing $\text{FEATURE}$ values of consecutive maximal occurrences of $\text{PATTERN}$.

For example, using the ‘increasing’ condition, the ‘max’ feature, and the ‘peak’ pattern, we illustrate two possible uses of the constraint

$\text{INCREASING_MAX_PEAK (VARIABLES)}$

namely:

- checking mode: are all peaks of a given time series sorted by increasing maximum value?

  For instance, $\text{INCREASING_MAX_PEAK(\langle 5,1,2,3,2,2,3,4,6,1,1,6,1 \rangle)}$ holds since the signature of the time series $\langle 5, 1, 2, 3, 2, 2, 3, 4, 6, 1, 1, 6, 1 \rangle$ contains three peaks highlighted in yellow, whose respective maximum values 3, 6, and 6 form an increasing sequence, namely [3, 6, 6].

- generation mode: generate all time series of five elements, all in the set $\{0,1,2\}$, where peaks are sorted by increasing maximum value. Searching for all solutions to the constraints

  $\{ 0 \leq x_i \leq 2 \ (i \in [1,5]) \}$

  $\text{INCREASING_MAX_PEAK(\langle x_1, x_2, x_3, x_4, x_5 \rangle)}$

  gives 241 solutions, 11 of which have two peaks, as shown in red below:

  - $\langle 0, 1, 0, 1, 0 \rangle$
  - $\langle 0, 2, 0, 2, 0 \rangle$
  - $\langle 0, 2, 1, 2, 1 \rangle$
  - $\langle 1, 2, 1, 2, 0 \rangle$
  - $\langle 0, 1, 0, 2, 0 \rangle$
  - $\langle 0, 2, 0, 2, 1 \rangle$
  - $\langle 1, 2, 0, 2, 0 \rangle$
  - $\langle 0, 1, 0, 2, 1 \rangle$
  - $\langle 0, 2, 1, 2, 0 \rangle$
  - $\langle 1, 2, 0, 2, 1 \rangle$
  - $\langle 1, 2, 1, 2, 1 \rangle$
2. OVERVIEW OF TIME-SERIES CONSTRAINTS

2.2 Function

The class of constraints `FUNCTION_NAME(VALUE, VARIABLES)` restricts `VALUE` to the aggregation of feature values of all pattern occurrences in a time series given as a collection `VARIABLES` of variables. The name of such a constraint is constructed in the following way:

```
FUNCTION_NAME ::= AGGREGATOR '·' FEATURE '·' PATTERN
```

where `AGGREGATOR` is the aggregator used to aggregate the `FEATURE` values of the maximal occurrences of `PATTERN`.

Using the ‘one’ feature, the ‘sum’ aggregator, and the ‘peak’ pattern, we illustrate two possible uses of the constraint

```
SUM_ONE_PEAK (VALUE, VARIABLES)
```

abbreviated as `NB_PEAK` where ‘nb’ is a shortcut for ‘sum_one’, namely:

- **Checking mode:** test whether `VALUE` can be restricted to the number of peaks of a given time series.
  - For instance, `NB_PEAK(3, ⟨5, 1, 2, 3, 2, 2, 3, 4, 5, 1, 1, 6, 1⟩)` holds since the signature of the time series `⟨5, 1, 2, 3, 2, 2, 3, 4, 5, 1, 1, 6, 1⟩` contains three peaks highlighted in yellow, as stated by the first argument of `NB_PEAK`.
  - Also, `VALUE ∈ [0, 9] ∧ NB_PEAK(VALUE, ⟨5, 1, 2, 3, 2, 2, 3, 4, 5, 1, 1, 6, 1⟩)` sets `VALUE` to 3 since `VALUE` is functionally determined by the second argument of `NB_PEAK`.

- **Generation mode:** generate all time series of five elements, all in the set `{0, 1, 2}`, containing two peaks. Searching for all solutions to the constraints
  ```
  \{ 0 \leq x_i \leq 2 \ (i \in [1, 5]) \\
  NB_PEAK(2, \langle x_1, x_2, x_3, x_4, x_5 \rangle) \}
  ```

  gives 13 solutions:

  - ⟨0, 1, 0, 1, 0⟩
  - ⟨0, 1, 0, 2, 0⟩
  - ⟨0, 1, 0, 2, 1⟩
  - ⟨0, 2, 0, 1, 0⟩
  - ⟨0, 2, 0, 2, 0⟩
  - ⟨0, 2, 0, 2, 1⟩
  - ⟨1, 2, 0, 1, 0⟩
  - ⟨1, 2, 0, 2, 0⟩
  - ⟨1, 2, 0, 2, 1⟩
  - ⟨1, 2, 1, 2, 0⟩
  - ⟨1, 2, 1, 2, 1⟩
  - ⟨1, 2, 1, 2, 1⟩
  - ⟨1, 2, 1, 2, 0⟩

2.3 Feature

The class of constraints `FEATURE_NAME(VARIABLES, FEATURES, DEFAULT)` constrains the feature values of all maximal pattern occurrences in a time series given as a
collection VARIABLES of variables. More precisely, if position $i$ of VARIABLES is not the first position where a maximal pattern occurrence is identified (even if this maximal pattern occurrence is not yet completed), then FEATURES[$i$] is forced to the default value DEFAULT; otherwise, FEATURES[$i$] is the feature value of the corresponding maximal pattern occurrence. The name of such a constraint is constructed in the following way:

$$\text{FeatureName ::= Feature ' _ ' Pattern}$$

where Feature and Pattern respectively are the feature and the pattern we focus on. Within this class of constraints, the feature ‘one’ is renamed ‘index’ for reasons of readability.

Using the ‘max’ feature and the ‘peak’ pattern, we illustrate two possible uses of the constraint

$$\text{MAX\_PEAK (VARIABLES, FEATURES, DEFAULT)}$$

namely:

- checking mode: test whether the non-default values that occur in the FEATURES argument correspond to the maximum values of the different peaks of a time series; in addition check that the positions of these non-default values coincide with the positions where a peak is discovered, i.e. the first decrease after a sequence of increases.

For instance, $\text{MAX\_PEAK (} \langle 5, 1, 4, 3, 2, 2, 5, 4, 4, 1, 1, 6, 6, 4, 1 \rangle, \langle 0, 0, 4, 0, 0, 0, 5, 0, 0, 0, 0, 0, 6, 0, 0 \rangle, 0) \text{ holds}$ since the signature of the time series $\langle 5, 1, 4, 3, 2, 2, 5, 4, 4, 1, 1, 6, 6, 4, 1 \rangle$ contains three peaks highlighted in yellow, whose maximum values 4, 5, and 6 where the three peaks are first identified, are positions 3, 8, and 14 as reported by the non-default values in the FEATURES argument. Note that the position where the third peak is identified corresponds to the last occurrence of value 6, namely the first decrease in the third peak.

Also, with the same time series and the same default value as first and third arguments, $\text{MAX\_PEAK (} \langle 5, 1, 4, 3, 2, 2, 5, 4, 4, 1, 1, 6, 6, 4, 1 \rangle, \text{FEATURES, 0}) \text{ sets FEATURES to } \langle 0, 0, 4, 0, 0, 0, 5, 0, 0, 0, 0, 0, 6, 0, 0 \rangle$, since FEATURES is functionally determined by the first and third arguments of $\text{MAX\_PEAK}$.

- generation mode: generate all time series of seven elements, all in the set \{0, 1, 2\}, with peaks of heights 2 and 1 at positions 2 and 6, and no peak for which the first decrease is located at position 4. Searching for all solutions to the constraints
2. OVERVIEW OF TIME-SERIES CONSTRAINTS

\[
\begin{aligned}
0 \leq x_i \leq 2 & \quad (i \in [1, 7]) \\
0 \leq f_i \leq 2 & \quad (i \in [1, 7]) \\
\text{MAX_PEAK} & = \langle x_1, x_2, x_3, x_4, x_5, x_6, x_7 \rangle, \\
f_2 = 2, f_4 = 0 & \quad (i \in [1, 7])
\end{aligned}
\]

gives 12 solutions, all of them having two peaks located at positions 2 and 6, of heights 2 and 1, as well as no peak at position 4, as shown below:

- (0, 2, 0, 0, 0, 1, 0)
- (0, 2, 0, 1, 1, 0)
- (0, 2, 0, 1, 1, 0)
- (0, 2, 1, 0, 0, 1, 0)
- (0, 2, 1, 0, 1, 1, 0)
- (1, 2, 0, 0, 0, 1, 0)
- (1, 2, 1, 0, 0, 1, 0)
- (1, 2, 1, 0, 1, 1, 0)
- (1, 2, 1, 1, 1, 1, 0)

2.4 Extreme

The class of constraints \textsc{ExtremeName}(Value, Variables, Found) constrains the maximum (or minimum) feature value \textsc{Value} of the maximal pattern occurrences in a time series given as a collection \textsc{Variables} of variables. In addition it restricts the positions of those maximal pattern occurrences achieving the corresponding minimum or maximum feature values. More precisely, if position \( i \) is not the first position where a maximal pattern occurrence with a maximum (or minimum) feature value is identified (even if this pattern occurrence is not yet completed) then \textsc{Found}[i] is 0; otherwise \textsc{Found}[i] is 1. The name of such a constraint is constructed in the following way:

\[
\textsc{ExtremeName} \ ::= \ \text{`pos_'}\textsc{ExtremeAgg} \text{`_'}\textsc{NonConstantFeat} \text{`_'}\textsc{Pattern}
\]

where \textsc{ExtremeAgg} denotes the maximum or the minimum non-constant feature \textsc{NonConstantFeat} value over the maximal pattern occurrences of pattern \textsc{Pattern}.

Using the ‘max’ feature, the ‘peak’ pattern, and the fact that we are interested in the minimum feature value, we illustrate two possible uses of the constraint

\[
\text{POS_MIN_MAX_PEAK (Value, Variables, Found)}
\]

\[
\downarrow \text{Feature} \downarrow \text{Pattern}
\]

namely:

- checking mode: test whether \textsc{Value} can be restricted to the altitude of the lowest peak of a time series; in addition check that the positions of the occurrences of 1 in \textsc{Found} coincide with the positions of all the lowest peaks.

For instance, \text{POS_MIN_MAX_PEAK} \( \langle 5, 1, 4, 3, 2, 2, 5, 4, 4, 1, 1, 4, 4, 3, 1 \rangle, \langle 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0 \rangle \)
holds since the signature of the time series \( \langle 5, 1, 4, 3, 2, 2, 5, 4, 1, 1, 4, 4, 3, 1 \rangle \) contains three peaks highlighted in yellow, where two of them are peaks that have the smallest maximum value, i.e. value 4. These two peaks are identified in positions 3 and 14 as reported by the two occurrences of 1 in the \textsc{found} argument. Note that the position where the last peak is discovered corresponds to the last occurrence of value 4 in the time series since the first decrease takes place at this position.

Also, with the same time series as second argument, \( \textsc{pos}_{\text{min}} \textsc{max}_{\text{peak}} \) \( \langle \textsc{value}, \langle 5, 1, 4, 3, 2, 2, 5, 4, 1, 1, 4, 4, 3, 1 \rangle, \textsc{found} \rangle \) sets \textsc{value} to 4 and \textsc{found} to \( \langle 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle \), since \textsc{value} and \textsc{found} are both functionally determined by the second argument of \( \textsc{pos}_{\text{min}} \textsc{max}_{\text{peak}} \).

- generation mode: generate all time series of seven elements, all in the set \{0, 1, 2\}, starting and ending with a ’1’, where the first decreases of peaks of minimum height are located in positions 2 and 6.

\[
\begin{align*}
0 & \leq v \leq 2 \\
0 & \leq x_i \leq 2 \ (i \in \{1, 7\}) \\
0 & \leq f_i \leq 1 \ (i \in \{1, 7\}) \\
\textsc{pos}_{\text{min}} \textsc{max}_{\text{peak}} & (\langle x_1, x_2, x_3, x_4, x_5, x_6, x_7 \rangle, \langle f_1, f_2, f_3, f_4, f_5, f_6, f_7 \rangle) \\
x_1 = 1, x_7 = 1 \\
f_2 = 1, f_6 = 1
\end{align*}
\]

gives 17 solutions, with peaks of minimum height in positions 2 and 6, as shown below:

- \langle 1, 2, 0, 0, 0, 2, 1 \rangle
- \langle 1, 2, 0, 0, 1, 2, 1 \rangle
- \langle 1, 2, 0, 0, 2, 2, 1 \rangle
- \langle 1, 2, 0, 1, 1, 2, 1 \rangle
- \langle 1, 2, 0, 1, 2, 2, 1 \rangle
- \langle 1, 2, 0, 2, 0, 2, 1 \rangle
- \langle 1, 2, 0, 2, 1, 2, 1 \rangle
- \langle 1, 2, 0, 2, 2, 2, 1 \rangle
- \langle 1, 2, 0, 2, 2, 0, 2, 1 \rangle
- \langle 1, 2, 0, 2, 2, 1, 2, 1 \rangle
- \langle 1, 2, 0, 2, 2, 1, 0, 2, 1 \rangle
- \langle 1, 2, 0, 2, 2, 1, 0, 2, 1 \rangle
- \langle 1, 2, 0, 2, 2, 1, 0, 2, 1 \rangle
- \langle 1, 2, 1, 1, 1, 2, 1 \rangle
- \langle 1, 2, 1, 1, 2, 2, 1 \rangle
- \langle 1, 2, 1, 2, 0, 2, 1 \rangle
- \langle 1, 2, 1, 2, 1, 2, 1 \rangle
- \langle 1, 2, 1, 2, 1, 2, 1 \rangle
- \langle 1, 2, 1, 2, 2, 2, 1 \rangle
- \langle 1, 2, 1, 2, 2, 0, 2, 1 \rangle
- \langle 1, 2, 1, 2, 2, 1, 0, 2, 1 \rangle
- \langle 1, 2, 1, 2, 2, 1, 0, 2, 1 \rangle
- \langle 1, 2, 1, 2, 2, 1, 0, 2, 1 \rangle
- \langle 1, 2, 1, 2, 2, 1, 0, 2, 1 \rangle
2. OVERVIEW OF TIME-SERIES CONSTRAINTS
3

Patterns and Corresponding Tables

Contents

3.1 Patterns, Seed Transducers and Parametrised Glue Matrices 23
  3.1.1 BUMP_ON_DECREASING_SEQUENCE .......................... 24
  3.1.2 DECREASING ........................................... 28
  3.1.3 DECREASING_SEQUENCE ................................. 30
  3.1.4 DECREASING_TERRACE .................................. 33
  3.1.5 DIP_ON_INCREASING_SEQUENCE .......................... 36
  3.1.6 GORGE .................................................. 40
  3.1.7 INCREASING ............................................ 44
  3.1.8 INCREASING_SEQUENCE ................................. 46
  3.1.9 INCREASING_TERRACE .................................. 49
  3.1.10 INFLEXION .............................................. 52
  3.1.11 PEAK .................................................... 54
  3.1.12 PLAIN .................................................. 57
  3.1.13 PLATEAU ............................................... 60
  3.1.14 PROPERPLAIN ........................................... 63
  3.1.15 PROPER_PLATEAU ....................................... 66
  3.1.16 STEADY ................................................ 69
  3.1.17 STEADY_SEQUENCE ..................................... 71
  3.1.18 STRICTLY_DECREASING_SEQUENCE ....................... 74
  3.1.19 STRICTLY_INCREASING_SEQUENCE ....................... 77
  3.1.20 SUMMIT ............................................... 80
  3.1.21 VALLEY ................................................ 84
  3.1.22 ZIGZAG ................................................ 87
  3.2 Decoration Tables ......................................... 93
  3.3 Tables of Regular-Expression Characteristics ............ 121
Patterns focus on the topological aspect of subsequences of a time series. They are defined by two components:

- First, a regular expression over the alphabet \{<, =, >\}.
- Second, two non-negative integers \(b\) and \(a\), that are intended to respectively delete a prefix and a suffix of the pattern that should be discarded for computing a characteristic of an occurrence of a pattern.

Given a pattern and a time series \(x_1, x_2, \ldots, x_n\) of integer constants, called the input values, and forming the input sequence, a single integer is computed as follows:

I. Compare each pair of adjacent input values in order to build a sequence \(s_1, s_2, \ldots, s_{n-1}\) of signature values over the alphabet \{<, =, >\}, as follows:

\[
(x_i < x_{i+1} \Leftrightarrow s_i = \text{'}<	ext{'} \land (x_i = x_{i+1} \Leftrightarrow s_i = \text{'}=\text{'} \land (x_i > x_{i+1} \Leftrightarrow s_i = \text{'}>\text{'}). The signature values form the signature sequence.

II. Within the signature sequence, find all maximal words matching the regular expression associated with the pattern.

III. For each found pattern occurrence, discard its prefix and suffix of length \(b\) and \(a\) to obtain an integer sequence \(e\) for which we compute an integer feature value, so that we obtain a feature sequence. The features we currently consider are one, width, surf, min, max, and range, and correspond respectively to the value 1, to the number of elements of \(e\), to \(\sum_{i \in e} x_i\), to \(\min_{i \in e} x_i\), to \(\max_{i \in e} x_i\), and to \(\max_{i \in e} x_i - \min_{i \in e} x_i\).

IV. Aggregate the values of the feature sequence into a single integer value. The aggregators we currently consider are taking the sum (\(\text{Sum}\)), taking the minimum (\(\text{Min}\)), and taking the maximum (\(\text{Max}\)). The feature one only makes sense with the \(\text{Sum}\) aggregator.

**Definition 1** (s-occurrence, i-occurrence, e-occurrence). Given an input sequence \(x_1, x_2, \ldots, x_n\), its signature sequence \(S = s_1, s_2, \ldots, s_{n-1}\), a pattern \(\langle r, b, a \rangle\), and a non-empty signature subsequence \(s_i, s_{i+1}, \ldots, s_j\), with \(1 \leq i \leq j \leq n - 1\), forming a maximal word that matches \(r\), the s-occurrence \((i..j)\) is the index sequence \(i, i + 1, \ldots, j\); the i-occurrence \([i + b]..j\) is the index sequence \(i + b, \ldots, j\); and the e-occurrence \(\{[i + b]..(j + 1 - a)]\) is the index sequence \(i + b, \ldots, j + 1 - a\).
Figures representing time series use the following convention for denoting i-occurrences, s-occurrences and e-occurrences:

- Positions belonging to an i-occurrence are represented by a red circle •.
- Positions belonging to a s-occurrence, but not to an i-occurrence, are represented by a circle ○.
- Positions of an occurrence of pattern corresponding to an e-occurrence have a background coloured in yellow.

When an i-occurrence and an e-occurrence of two consecutive occurrences of a pattern coincide a red circle • is used.

Figures representing transducers and automata use the following conventions:

- An arrow coming from nowhere denotes an initial state.
- A double circle indicates an accepting state.
- The acceptance function of a register automaton is depicted by a box connected by dotted lines to each state of the register automaton.
- On the transitions of a transducer, a colon separates the consumed input letter(s) from the produced output letter(s).
- A transition labelled by the input letter ≤ is a shortcut for two transitions respectively labelled by the letters < and =. The same convention is used for the letters ≥, ≳, ≻, which respectively represent a set of transitions, each of them labelled by a letter of {>, =}, {>, <}, {>, =, <}.

The name of a constraint is defined by the concatenation of the aggregator function, the feature name and the pattern. For each pattern, a so-called seed transducer recognises all pattern occurrences.

**Definition 2 (seed transducer).** A seed transducer is a transducer for which all states are accepting with input alphabet {<, =, >}, output alphabet, also-called phase alphabet, {out, maybe_b, out_r, found, maybe_b, in, out_a, found_a}.

**Definition 3 (t-occurrence).** Given an input sequence s over the input alphabet {<, =, >} the t-occurrence of s wrt a given seed transducer corresponds to the indices of the phase letters of a maximum word in the generated output sequence that matches ‘maybe_b found_e | maybe_b found(maybe_b in^?)*’.

**Definition 4 (well-formedness).** A seed transducer is well formed if (1) all output sequences it produces are recognised by the finite deterministic automaton depicted in Figure 3.1, and if (2) the t-occurrence for any input sequence s coincides with the i-occurrence of the pattern for s.

Well-formedness of seed transducers guarantees that the automata with registers obtained by applying the decoration tables for replacing phase letters of the output alphabet by register updates compute the expected result [8].
3. PATTERNS AND CORRESPONDING TABLES

Figure 3.1: Automaton describing the output language of a well-formed transducer; states $\notin$, $\in$, $\notin$ respectively mean that (1) we are outside an occurrence of pattern, (2) we are potentially inside an occurrence of pattern, (3) we are inside an occurrence of pattern that is not yet finished.

Following the presentation in [9, 2], we introduce the notion of glue matrix. A glue matrix for a reversible time-series constraint $c$ specified by a seed transducer $\sigma$, an aggregator $g$ and a feature $f$ is a matrix indexed by the states of $\sigma$ and the states of $\sigma^{rev}$, and parametrised by $g$ and $f$, where $\sigma^{rev}$ is a seed transducer of the reverse of constraint $c$. Consider the automaton $A_{\sigma}$ (respectively $A_{\sigma}^{rev}$) with registers $R$, $C$ and $D$ obtained by applying the decoration table 3.37 to the seed transducer $\sigma$ (respectively $\sigma^{rev}$). Let $A_{\sigma}$ reach state $\rightarrow Q$ and register values $\left\langle C, D, R \right\rangle$ on a prefix of a word $w$. Similarly let $A_{\sigma}^{rev}$ reach state $\left\langle Q \right\rangle$ and register values $\left\langle C, D, R \right\rangle$ on the reverse of the corresponding suffix of $w$. The value returned by $A_{\sigma}$ on the entire word $w$ is $\phi_{g}(R, R, \Gamma)$, where $\Gamma$ is the entry of the glue matrix corresponding to row $\rightarrow Q$ and column $\leftarrow Q$. That entry contains an expression parametrised by $g$ and $f$ involving a subset of the registers $\left\langle C, D, C, D \right\rangle$. 
3.1 Patterns, Seed Transducers and Parametrised Glue Matrices

<table>
<thead>
<tr>
<th>Feature</th>
<th>( f )</th>
<th>( \text{id}_f )</th>
<th>( \min_f )</th>
<th>( \max_f )</th>
<th>( \phi_f )</th>
<th>( \delta^i_f )</th>
<th>( \hat{\delta}^i_f )</th>
</tr>
</thead>
<tbody>
<tr>
<td>one</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>( \max )</td>
<td>1</td>
<td>(-1)</td>
<td></td>
</tr>
<tr>
<td>width</td>
<td>0</td>
<td>0</td>
<td>( n+1 )</td>
<td>(+)</td>
<td>(+)</td>
<td>(-x_i)</td>
<td></td>
</tr>
<tr>
<td>surf</td>
<td>0</td>
<td>(-\infty)</td>
<td>(+\infty)</td>
<td>(+)</td>
<td>(+)</td>
<td>(-x_i)</td>
<td>(n/a)</td>
</tr>
<tr>
<td>max</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
<td>(+\infty)</td>
<td>(\max)</td>
<td>(x_i)</td>
<td>(n/a)</td>
<td></td>
</tr>
<tr>
<td>min</td>
<td>(+\infty)</td>
<td>(-\infty)</td>
<td>(+\infty)</td>
<td>(\min)</td>
<td>(x_i)</td>
<td>(n/a)</td>
<td></td>
</tr>
<tr>
<td>range</td>
<td>0</td>
<td>0</td>
<td>(+\infty)</td>
<td>(n/a)</td>
<td>(x_i)</td>
<td>(n/a)</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.1: Features: identity, minimum, and maximum values; the operators \( \phi_f \) and \( \delta^i_f \) recursively define the feature value \( v_u \) of a time series \( x_\ell,\ldots,x_u \) by \( v_\ell = \phi_f(\text{id}_f,\delta^\ell_f) \) and \( v_i = \phi_f(v_{i-1},\delta^{i+1}_f) \) for \( i > \ell \), where \( \delta^\ell_f \) is the contribution of \( x_\ell \) to \( v_u \); \( \hat{\delta}^i_f \) cancels out the contribution of \( x_i \) within a feature value so that this contribution is not counted twice; \( n \) and \( n/a \) respectively stand for the length of the time series and for not available.

<table>
<thead>
<tr>
<th>Aggregator ( g )</th>
<th>( \phi_g )</th>
<th>( \text{default}_{gf} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max</td>
<td>( \max )</td>
<td>( \min_f )</td>
</tr>
<tr>
<td>Min</td>
<td>( \min )</td>
<td>( \max_f )</td>
</tr>
<tr>
<td>Sum</td>
<td>(+)</td>
<td>(0)</td>
</tr>
</tbody>
</table>

Table 3.2: Aggregators: operators and identity values relative a feature \( f \).

<table>
<thead>
<tr>
<th>Predicate ( \theta )</th>
<th>( \text{default}_{\theta} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \leq )</td>
<td>(-\infty)</td>
</tr>
<tr>
<td>( \equiv )</td>
<td>(x) (free variable)</td>
</tr>
<tr>
<td>( \geq )</td>
<td>(+\infty)</td>
</tr>
</tbody>
</table>

Table 3.3: Default value for a predicate \( \theta \) used in the decoration table 3.38.
3. PATTERNS AND CORRESPONDING TABLES

3.1.1 BUMP_ON_DECREASING_SEQUENCE

Given a sequence $S$ over the alphabet \{‘<’, ‘=’, ‘>’\}, an occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence of $S$ which matches the regular expression ‘>>><’.

Part (A) and parts (B,C) of Figure 3.2 respectively depict the seed transducer associated with the BUMP_ON_DECREASING_SEQUENCE pattern as well as one example of its execution on a time series.

The pattern DIP_ON_INCREASING_SEQUENCE = ‘<<><’ is not the reverse of the pattern BUMP_ON_DECREASING_SEQUENCE even though ‘>>><’ is the mirror word of ‘<<><’.

This is because the constants $b = 2$ and $a = 1$ (see Figure 3.2) used for trimming the regular expression ‘>>><’ are different, which makes the two corresponding $e$-occurrences different. As a consequence we do not have any glue matrix for the BUMP_ON_DECREASING_SEQUENCE pattern.

---

1. To get the mirror of a word $w$ over the alphabet \{‘<’, ‘=’, ‘>’\}, first reverse the word $w$, second swap the ‘>’ with the ‘<’. 
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.2: (A) Seed transducer for the BUMP_ON_DECREASING_SEQUENCE pattern: ‘>>><’ with $b = 2$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o, o, m_b, f_e$ are shortcut for out, out, maybe_b, found_e); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
Figure 3.3: Parametrised automaton for any functional time-series constraints of the BUMP\_ON\_DECREASING\_SEQUENCE pattern obtained by applying the decoration table 3.37 to the corresponding transducer
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.4: Parametrised automaton for any predicate time-series constraints of the BUMP_ON_DECENDING_SEQUENCE pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
3. PATTERNS AND CORRESPONDING TABLES

3.1.2 DECREASING

Given a sequence $S$ over the alphabet \{‘$<$’, ‘$=$’, ‘$>$’\}, an occurrence of the pattern **DECREASING** is the subsequence of $S$ which matches the regular expression ‘$>$’.

Part (A) and parts (B,C) of Figure 3.5 respectively depict the seed transducer associated with the **DECREASING** pattern as well as one example of its execution on a time series. 

![Seed transducer for the **DECREASING** pattern](image)

Figure 3.5: (A) Seed transducer for the **DECREASING** pattern: ‘$>$’ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $f$ are shortcut for out, found); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.

$$s$$

| $\phi_g(C, \overline{C})$ |

Table 3.4: Parametrised glue matrix for any $g\_f\_DECREASING$ constraint; the state associated with each row corresponds to a state of the **DECREASING** transducer – see Figure 3.5, while the state associated with each column corresponds to a state of the reverse of the **DECREASING** transducer, i.e. the **INCREASING** transducer – see Figure 3.20.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.6: Parametrised automaton for any functional time-series constraints of the DECREASING pattern obtained by applying the decoration table 3.37 to the corresponding transducer.

Figure 3.7: Parametrised automaton for any predicate time-series constraints of the DECREASING pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where \( \theta \) is the predicate (i.e. \( \leq, =, \geq \)) used for comparing two consecutive feature values.
3. PATTERNS AND CORRESPONDING TABLES

3.1.3 DECREASING_SEQUENCE

Given a sequence $S$ over the alphabet $\{', '=', '>'\}$, an occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence of $S$ which matches the regular expression $'>' (> | =)^* > | >'$. Part (A) and parts (B,C) of Figure 3.8 respectively depict the seed transducer associated with the DECREASING_SEQUENCE pattern as well as one example of its execution on a time series.

![Figure 3.8](image)

**Figure 3.8:** (A) Seed transducer for the DECREASING_SEQUENCE pattern: $'>' (> | =)^* > | >'$ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $f$, $m_a$, $o_a$ are shortcut for out, found maybe_a, out_a); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.9: Parametrised automaton for any functional time-series constraints of the DECREASING_SEQUENCE pattern obtained by applying the decoration table 3.37 to the corresponding transducer

Figure 3.10: Parametrised automaton for any predicate time-series constraints of the DECREASING_SEQUENCE pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
Table 3.5: Parametrised glue matrix for any $g_f$ _DECREASING_SEQUENCE_ constraint, where the cell annotation $\text{M}$ stands for merging two existing decreasing sequences sharing an element; the state associated with each row corresponds to a state of the _DECREASING_SEQUENCE_ transducer – see Figure 3.8, while the state associated with each column corresponds to a state of the reverse of the _DECREASING_SEQUENCE_ transducer, i.e. the _INCREASING_SEQUENCE_ transducer – see Figure 3.23.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

3.1.4 DECREASING_TERRACE

Given a sequence $S$ over the alphabet $\{<, \geq, >\}$, an occurrence of the pattern DECREASING_TERRACE is the maximal subsequence of $S$ which matches the regular expression $'>=+>$. Part (A) and parts (B,C) of Figure 3.11 respectively depict the seed transducer associated with the DECREASING_TERRACE pattern as well as one example of its execution on a time series.

Figure 3.11: (A) Seed transducer for the DECREASING_TERRACE pattern: $'>=+>'$ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $m_b$, $o_r$, $f_c$ are shortcut for out, maybe_b, out_r, found_r); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
Figure 3.12: Parametrised automaton for any functional time-series constraints of the \textsc{decreasing\_terrace} pattern obtained by applying the decoration table 3.37 to the corresponding transducer.

Figure 3.13: Parametrised automaton for any predicate time-series constraints of the \textsc{decreasing\_terrace} pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=\,$, $\geq$) used for comparing two consecutive feature values.
### 3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Table 3.6: Parametrised glue matrix for any $g_f$<sub>DECREASING_TERRACE</sub> constraint, where the cell annotation $C$ stands for creating a new decreasing terrace occurrence from no existing decreasing terrace; the state associated with each row corresponds to a state of the DECREASING_TERRACE transducer – see Figure 3.11, while the state associated with each column corresponds to a state of the reverse of the DECREASING_TERRACE transducer, i.e. the INCREASING_TERRACE transducer – see Figure 3.26.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_g(\overline{C}, \overline{C})$</td>
<td>$\phi_g(\overline{C}, \overline{C})$</td>
<td>$\phi_g(\overline{C}, \overline{C})$</td>
</tr>
</tbody>
</table>
| $r$ | $\phi_g(\overline{C}, \overline{C})$ | $\phi_g(\overline{C}, \overline{C})$ | $\phi_f(\overline{D}, \overline{D}, \delta_f^{i+1})$ |}


3. PATTERNS AND CORRESPONDING TABLES

3.1.5 DIP_ON_INCREASING_SEQUENCE

Given a sequence $S$ over the alphabet \{‘<’, ‘=’, ‘>’\}, an occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence of $S$ which matches the regular expression ‘<<><<’. Part (A) and parts (B,C) of Figure 3.14 respectively depict the seed transducer associated with the DIP_ON_INCREASING_SEQUENCE pattern as well as one example of its execution on a time series.

The pattern BUMP_ON_DECREASING_SEQUENCE = ‘>>><>’ is not the reverse of the pattern DIP_ON_INCREASING_SEQUENCE even though ‘<<><<’ is the mirror word of ‘>>><>’\(^2\). This is because the constants $b = 2$ and $a = 1$ (see Figure 3.14) used for trimming the regular expression ‘<<><<’ are different, which makes the two corresponding $e$-occurrences different. As a consequence we do not have any glue matrix for the DIP_ON_INCREASING_SEQUENCE pattern.

\(^2\)To get the mirror of a word $w$ over the alphabet \{‘<’, ‘=’, ‘>’\}, first reverse the word $w$, second swap the ‘>’ with the ‘<’.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.14: (A) Seed transducer for the DIP_ON_INCREASING_SEQUENCE pattern: '<<><<>' with $b = 2$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output o, o, m_b, f are shortcut for out, out, maybe_b, found); (C) Illustrating the relation between the s-occurrence, the i-occurrence and the e-occurrence.
Figure 3.15: Parametrised automaton for any functional time-series constraints of the DIP\_ON\_INCREASING\_SEQUENCE pattern obtained by applying the decoration table 3.37 to the corresponding transducer.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.16: Parametrised automaton for any predicate time-series constraints of the DIP\textsubscript{ON}\_INCREASING\_SEQUENCE pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
3. PATTERNS AND CORRESPONDING TABLES

3.1.6 GORGE

Given a sequence $S$ over the alphabet $\{\ '<', '\=', '\>'\}$, an occurrence of the pattern GORGE is the maximal subsequence of $S$ which matches the regular expression $'(\ > | (\ (= | \ >) \ >)) (\ < | (\ (= | \ <) \ <))'$. Part (A) and parts (B,C) of Figure 3.17 respectively depict the seed transducer associated with the GORGE pattern as well as one example of its execution on a time series.

Figure 3.17: (A) Seed transducer for the GORGE pattern: $'(\ > | (\ (= | \ >) \ >)) (\ < | (\ (= | \ <) \ <))'$ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $f$, $m_a$, $o_a$, $m_b$, $o_t$ are shortcut for out, found, $\text{maybe}_a$, out$_a$, $\text{maybe}_b$, out$_t$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.18: Parametrised automaton for any functional time-series constraints of the GORGE pattern obtained by applying the decoration table 3.37 to the corresponding transducer (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
Figure 3.19: Parametrised automaton for any predicate time-series constraints of the GORGE pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where \( \theta \) is the predicate (i.e. \( \leq, =, \geq \)) used for comparing two consecutive feature values; transition \( u \rightarrow r \) has the same register update as transition \( r \rightarrow u \).
Table 3.7: Parametrised glue matrix for any \( g_f \) GORGE constraint, where cell annotations have the following meaning: \( C \) stands for creating a new gorge occurrence from no existing gorge, \( R \) stands for extending to the right an existing gorge, \( L \) stands for extending to the left an existing gorge; the state associated with each row corresponds to a state of the GORGE transducer – see Figure 3.17, while the state associated with each column corresponds to a state of the reverse of the GORGE transducer, i.e. the GORGE transducer – see Figure 3.17.
3. PATTERNS AND CORRESPONDING TABLES

3.1.7 INCREASING

Given a sequence $S$ over the alphabet $\{’<’, ’=’, ’>’\}$, an occurrence of the pattern INCREASING is the subsequence of $S$ which matches the regular expression ’<’. Part (A) and parts (B,C) of Figure 3.20 respectively depict the seed transducer associated with the INCREASING pattern as well as one example of its execution on a time series.

![Diagram](image_url)

Figure 3.20: (A) Seed transducer for the INCREASING pattern: ’<’ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o, f_e$ are shortcut for out, found$_e$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.

Table 3.8: Parametrised glue matrix for any $g\cdot f$ INCREASING constraint; the state associated with each row corresponds to a state of the INCREASING transducer – see Figure 3.20, while the state associated with each column corresponds to a state of the reverse of the INCREASING transducer, i.e. the DECREASING transducer – see Figure 3.5.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.21: Parametrised automaton for any functional time-series constraints of the INCREASING pattern obtained by applying the decoration table 3.37 to the corresponding transducer

\[
\begin{align*}
C &\leftarrow \text{default}_f \\
D &\leftarrow \text{id}_f \\
R &\leftarrow \text{default}_f \\
\end{align*}
\]

\[
\begin{align*}
D &\leftarrow \text{id}_f \\
R &\leftarrow \phi_f(R, \phi_f(D, \delta_f^{i+1})) \\
\end{align*}
\]

Figure 3.22: Parametrised automaton for any predicate time-series constraints of the INCREASING pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where \(\theta\) is the predicate (i.e. \(\leq, =, \geq\)) used for comparing two consecutive feature values

\[
\begin{align*}
C &\leftarrow \text{default}_\theta \\
D &\leftarrow \text{id}_f \\
F &\leftarrow \text{default}_\theta \\
R &\leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow \phi_f(\phi_f(D, \delta_f^i), \delta_f^{i+1}) \\
D &\leftarrow \text{id}_f \\
F &\leftarrow \phi_f(\phi_f(D, \delta_f^i), \delta_f^{i+1}) \\
R &\leftarrow R \land \theta(F, \phi_f(D, \delta_f^i), \delta_f^{i+1}) \\
\end{align*}
\]
Given a sequence $S$ over the alphabet $\{ '<', '=' , '>' \}$, an occurrence of the pattern `INCREASING_SEQUENCE` is the maximal subsequence of $S$ which matches the regular expression $'(< | =)^* < | <$'. Part (A) and parts (B,C) of Figure 3.23 respectively depict the seed transducer associated with the `INCREASING_SEQUENCE` pattern as well as one example of its execution on a time series.

![Diagram](image)

Figure 3.23: (A) Seed transducer for the `INCREASING_SEQUENCE` pattern: `$'(< | =)^* < | <$' with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $f$, $m_i$, $o_a$ are shortcut for $\text{out}$, $\text{found}$, $\text{maybe}_a$, $\text{out}_a$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
Figure 3.24: Parametrised automaton for any functional time-series constraints of the INCREASING_SEQUENCE pattern obtained by applying the decoration table 3.37 to the corresponding transducer.

Figure 3.25: Parametrised automaton for any predicate time-series constraints of the INCREASING_SEQUENCE pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
3. PATTERNS AND CORRESPONDING TABLES

Table 3.9: Parametrised glue matrix for any $g_f$ INCREASING_SEQUENCE constraint, where the cell annotation $\text{M}$ stands for merging two existing increasing sequences sharing an element; the state associated with each row corresponds to a state of the INCREASING_SEQUENCE transducer – see Figure 3.23, while the state associated with each column corresponds to a state of the reverse of the INCREASING_SEQUENCE transducer, i.e. the DECREASING_SEQUENCE transducer – see Figure 3.8.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_f(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \delta_f^i)^{\text{M}}$</td>
</tr>
</tbody>
</table>
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

3.1.9 INCREASING_TERRACE

Given a sequence $S$ over the alphabet $\{<, \leq, >\}$, an occurrence of the pattern INCREASING_TERRACE is the maximal subsequence of $S$ which matches the regular expression $<\leq t<$. Part (A) and parts (B,C) of Figure 3.26 respectively depict the seed transducer associated with the INCREASING_TERRACE pattern as well as one example of its execution on a time series.

Figure 3.26: (A) Seed transducer for the INCREASING_TERRACE pattern: ‘$<\leq+t<$’ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $mb$, $on$, $fe$ are shortcut for $out$, $maybe$, $outf$, $found$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
Figure 3.27: Parametrised automaton for any functional time-series constraints of the INCREASING_TERRACE pattern obtained by applying the decoration table 3.37 to the corresponding transducer.

Figure 3.28: Parametrised automaton for any predicate time-series constraints of the INCREASING_TERRACE pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where \( \theta \) is the predicate (i.e. \( \leq, =, \geq \)) used for comparing two consecutive feature values.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Table 3.10: Parametrised glue matrix for any \texttt{g.f\_INCREASING\_TERRACE} constraint, where the cell annotation \texttt{C} stands for creating a new increasing terrace occurrence from no existing increasing terrace; the state associated with each row corresponds to a state of the \texttt{INCREASING\_TERRACE} transducer – see Figure 3.26, while the state associated with each column corresponds to a state of the reverse of the \texttt{INCREASING\_TERRACE} transducer, i.e. the \texttt{DECREASING\_TERRACE} transducer – see Figure 3.11.

\begin{table}[h]
\centering
\begin{tabular}{ccc}
  \textbf{s} & \textbf{r} & \textbf{t} \\
  \hline
  s & \phi_g(\overline{C}, \overline{C}) & \phi_g(\overline{C}, \overline{C}) & \phi_g(\overline{C}, \overline{C}) \\
  r & \phi_g(\overline{C}, \overline{C}) & \phi_g(\overline{C}, \overline{C}) & \phi_f(\overline{D}, \overline{D}, \delta_f^{i+1}) \\
  t & \phi_g(\overline{C}, \overline{C}) & \phi_f(\overline{D}, \overline{D}, \delta_f^{i+1}) & \phi_f(\overline{D}, \overline{D}, \delta_f^{i+1}) \\
\end{tabular}
\end{table}
3.1.10 INFLEXION

Given a sequence $S$ over the alphabet \{‘<’, ‘=’, ‘>’\}, an occurrence of the pattern INFLEXION is the maximal subsequence of $S$ which matches the regular expression \texttt{<( < | = )* > | > ( > | = )* <}. Part (A) and parts (B,C) of Figure 3.29 respectively depict the seed transducer associated with the INFLEXION pattern as well as one example of its execution on a time series.

The pattern INFLEXION is not its own reverse as illustrated by the following example: when scanned from left to right, the regular expression \texttt{<<<>>} contains the inflexion \texttt{<<<>}, which does not coincide with the inflexion \texttt{<>} obtained by performing a scan from right to left. As a consequence we do not have any glue matrix for the INFLEXION pattern.

![Figure 3.29: (A) Seed transducer for the INFLEXION pattern: \texttt{<( < | = )* > | > ( > | = )* <} with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output o, mb, f are shortcut for out, maybe, found); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.]}
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICESS53

Figure 3.30: Parametrised automaton for any functional time-series constraints of the INFLEXION pattern obtained by applying the decoration table 3.37 to the corresponding transducer (transition \( r \rightarrow t \) has the same register updates as transition \( t \rightarrow r \)).

Figure 3.31: Parametrised automaton for any predicate time-series constraints of the INFLEXION pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where \( \theta \) is the predicate (i.e. \( \leq, =, \geq \)) used for comparing two consecutive feature values; transition \( r \rightarrow t \) has the same register updates as transition \( t \rightarrow r \).
3. PATTERNS AND CORRESPONDING TABLES

3.1.11 PEAK

Given a sequence \( S \) over the alphabet \{‘<’, ‘=’, ‘>’\}, an occurrence of the pattern \text{PEAK} is the maximal subsequence of \( S \) which matches the regular expression ‘\(< (= | <)^* (> | =)^* >\)’. Part (A) and parts (B,C) of Figure 3.32 respectively depict the seed transducer associated with the \text{PEAK} pattern as well as one example of its execution on a time series.

Figure 3.32: (A) Seed transducer for the \text{PEAK} pattern: ‘\(< (= | <)^* (> | =)^* >\)’ with \( b = 1 \) and \( a = 1 \); (B) Illustrating the execution of the seed transducer on a time series (within the output \( o, \text{maybe}_b, f, \text{found}, \text{maybe}_a, \text{out} \); (C) Illustrating the relation between the \( s \)-occurrence, the \( i \)-occurrence and the \( e \)-occurrence.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.33: Parametrised automaton for any functional time-series constraints of the \textsc{peak} pattern obtained by applying the decoration table 3.37 to the corresponding transducer.

Figure 3.34: Parametrised automaton for any predicate time-series constraints of the \textsc{peak} pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
Table 3.11: Parametrised glue matrix for any \( g_f \_PEAK \) constraint, where cell annotations have the following meaning: \( C \) stands for creating a new peak occurrence from no existing peak, \( R \) stands for extending to the right an existing peak, \( L \) stands for extending to the left an existing peak; the state associated with each row corresponds to a state of the \( \text{PEAK} \) transducer – see Figure 3.32, while the state associated with each column corresponds to a state of the reverse of the \( \text{PEAK} \) transducer, i.e. the \( \text{PEAK} \) transducer – see Figure 3.32.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

3.1.12 PLAIN

Given a sequence $S$ over the alphabet \{‘<’, ‘=’, ‘>’\}, an occurrence of the pattern PLAIN is the maximal subsequence of $S$ which matches the regular expression ‘>=*<’. Part (A) and parts (B,C) of Figure 3.35 respectively depict the seed transducer associated with the PLAIN pattern as well as one example of its execution on a time series.

![Diagram of the seed transducer for the PLAIN pattern and its execution on a time series.]

Figure 3.35: (A) Seed transducer for the PLAIN pattern: ‘>=*<’ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $f_e$, $m_b$, $o_t$ are shortcut for out, found_e, maybe_b, out_r); (C) Illustrating the relation between the s-occurrence, the i-occurrence and the e-occurrence.
Figure 3.36: Parametrised automaton for any functional time-series constraints of the PLAIN pattern obtained by applying the decoration table 3.37 to the corresponding transducer.

Figure 3.37: Parametrised automaton for any predicate time-series constraints of the PLAIN pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
### Table 3.12: Parametrised glue matrix for any \( g \_f \_PLAIN \) constraint, where the cell annotation \( \text{C} \) stands for creating a new plain occurrence from no existing plain; the state associated with each row corresponds to a state of the PLAIN transducer – see Figure 3.35, while the state associated with each column corresponds to a state of the reverse of the PLAIN transducer, i.e. the PLAIN transducer – see Figure 3.35.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
<th>( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \phi_g(\overline{C}, \overline{C}) )</td>
<td>( \phi_g(\overline{C}, \overline{C}) )</td>
<td>( \phi_g(\overline{C}, \overline{C}) )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \phi_g(\overline{C}, \overline{C}) )</td>
<td>( \phi_f(\overline{D}, \overline{D}, \delta_f^{i+1}) )</td>
<td>( \phi_g(\overline{D}, \overline{D}, \delta_f^{i+1}) )</td>
</tr>
<tr>
<td>( t )</td>
<td>( \phi_g(\overline{C}, \overline{C}) )</td>
<td>( \phi_f(\overline{D}, \overline{D}, \delta_f^{i+1}) )</td>
<td>( \phi_f(\overline{D}, \overline{D}, \delta_f^{i+1}) )</td>
</tr>
</tbody>
</table>
3.1.13 **PLATEAU**

Given a sequence $S$ over the alphabet \{‘<’, '=' , ‘>’\}, an occurrence of the pattern PLATEAU is the maximal subsequence of $S$ which matches the regular expression ‘<=*>=’. Part (A) and parts (B,C) of Figure 3.38 respectively depict the seed transducer associated with the PLATEAU pattern as well as one example of its execution on a time series.

Figure 3.38: (A) Seed transducer for the PLATEAU pattern: ‘<=*>=’ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $f_c$, $m_b$, $o_s$ are shortcut for out, found, maybe, out); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.39: Parametrised automaton for any functional time-series constraints of the PLATEAU pattern obtained by applying the decoration table 3.37 to the corresponding transducer

Figure 3.40: Parametrised automaton for any predicate time-series constraints of the PLATEAU pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where \( \theta \) is the predicate (i.e. \( \leq, =, \geq \)) used for comparing two consecutive feature values.
Table 3.13: Parametrised glue matrix for any \( g_f \_PLATEAU \) constraint, where the cell annotation \( C \) stands for creating a new plateau occurrence from no existing plateau; the state associated with each row corresponds to a state of the \( PLATEAU \) transducer – see Figure 3.38, while the state associated with each column corresponds to a state of the reverse of the \( PLATEAU \) transducer, i.e. the \( PLATEAU \) transducer – see Figure 3.38.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

3.1.14 PROPER.PLAIN

Given a sequence $S$ over the alphabet \{‘<’, ‘=’, ‘>’\}, an occurrence of the pattern PROPER.PLAIN is the maximal subsequence of $S$ which matches the regular expression ‘$> = + <$’. Part (A) and parts (B,C) of Figure 3.41 respectively depict the seed transducer associated with the PROPER.PLAIN pattern as well as one example of its execution on a time series.

Figure 3.41: (A) Seed transducer for the PROPER.PLAIN pattern: ‘$> = + <$’ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output o, m_b, f_e are shortcut for out, maybe_b, found_e); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
3. PATTERNS AND CORRESPONDING TABLES

Figure 3.42: Parametrised automaton for any functional time-series constraints of the \texttt{PROPER\_PLAIN} pattern obtained by applying the decoration table 3.37 to the corresponding transducer.

Figure 3.43: Parametrised automaton for any predicate time-series constraints of the \texttt{PROPER\_PLAIN} pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
Table 3.14: Parametrised glue matrix for any $g_f$ _PROPER.PLAIN_ constraint, where the cell annotation $\mathbf{C}$ stands for creating a new proper plain occurrence from no existing proper plain; the state associated with each row corresponds to a state of the _PROPER.PLAIN_ transducer – see Figure 3.41, while the state associated with each column corresponds to a state of the reverse of the _PROPER.PLAIN_ transducer, i.e. the _PROPER.PLAIN_ transducer – see Figure 3.41.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\phi_g(\widetilde{C}, \widetilde{C})$</td>
<td>$\phi_g(\widetilde{C}, \widetilde{C})$</td>
<td>$\phi_g(\widetilde{C}, \widetilde{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\phi_g(\widetilde{C}, \widetilde{C})$</td>
<td>$\phi_g(\widetilde{C}, \widetilde{C})$</td>
<td>$\phi_f(\widetilde{D}, \widetilde{D}, \delta_f^{i+1})$</td>
</tr>
<tr>
<td>t</td>
<td>$\phi_g(\widetilde{C}, \widetilde{C})$</td>
<td>$\phi_f(\widetilde{D}, \widetilde{D}, \delta_f^{i+1})$</td>
<td>$\phi_f(\widetilde{D}, \widetilde{D}, \delta_f^{i+1})$</td>
</tr>
</tbody>
</table>
3.1.15 PROPER_PLATEAU

Given a sequence $S$ over the alphabet \{‘$<$’, ‘$=$’, ‘$>$’\}, an occurrence of the pattern PROPER_PLATEAU is the maximal subsequence of $S$ which matches the regular expression ‘$<=$+$>$’. Part (A) and parts (B,C) of Figure 3.44 respectively depict the seed transducer associated with the PROPER_PLATEAU pattern as well as one example of its execution on a time series.

Figure 3.44: (A) Seed transducer for the PROPER_PLATEAU pattern: ‘$<=$+$>$’ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $m_b$, $f_e$ are shortcut for out, maybe$_b$, found$_e$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
Figure 3.45: Parametrised automaton for any functional time-series constraints of the \textit{PROPER\_PLATEAU} pattern obtained by applying the decoration table \ref{table:3.37} to the corresponding transducer.

Figure 3.46: Parametrised automaton for any predicate time-series constraints of the \textit{PROPER\_PLATEAU} pattern obtained by applying the decoration table \ref{table:3.38} to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
Table 3.15: Parametrised glue matrix for any $g_f$ _PROPER_PLATEAU_ constraint, where the cell annotation $C$ stands for creating a new proper plateau occurrence from no existing proper plateau; the state associated with each row corresponds to a state of the _PROPER_PLATEAU_ transducer – see Figure 3.44, while the state associated with each column corresponds to a state of the reverse of the _PROPER_PLATEAU_ transducer, i.e. the _PROPER_PLATEAU_ transducer – see Figure 3.44.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

3.1.16 STEADY

Given a sequence $S$ over the alphabet \{'\<', '\=', '\>'\}, an occurrence of the pattern \texttt{STEADY} is the subsequence of $S$ which matches the regular expression ‘\texttt{=}’. Part (A) and parts (B,C) of Figure 3.47 respectively depict the seed transducer associated with the \texttt{STEADY} pattern as well as one example of its execution on a time series.

![Diagram of seed transducer for STEADY pattern](image)

Figure 3.47: (A) Seed transducer for the \texttt{STEADY} pattern: ‘\texttt{=}’ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output \texttt{f}, \texttt{o} are shortcut for \texttt{found}, \texttt{out}); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.

Table 3.16: Parametrised glue matrix for any $q_{\_f\_STEADY}$ constraint; the state associated with each row corresponds to a state of the \texttt{STEADY} transducer – see Figure 3.47, while the state associated with each column corresponds to a state of the reverse of the \texttt{STEADY} transducer, i.e. the \texttt{STEADY} transducer – see Figure 3.47.
3. PATTERNS AND CORRESPONDING TABLES

Figure 3.48: Parametrised automaton for any functional time-series constraints of the \textsc{steady} pattern obtained by applying the decoration table 3.37 to the corresponding transducer.

Figure 3.49: Parametrised automaton for any predicate time-series constraints of the \textsc{steady} pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=\geq$) used for comparing two consecutive feature values.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

3.1.17 STEADYSEQUENCE

Given a sequence $S$ over the alphabet \{‘<’, ‘=’, ‘>’\}, an occurrence of the pattern \textsc{steady\_sequence} is the maximal subsequence of $S$ which matches the regular expression ‘\textsc{=}\textsc{+}’. Part (A) and parts (B,C) of Figure 3.50 respectively depict the seed transducer associated with the \textsc{steady\_sequence} pattern as well as one example of its execution on a time series.

![Diagram of the seed transducer for the \textsc{steady\_sequence} pattern]

\begin{itemize}
  \item \textbf{Part (A)}: The seed transducer for the \textsc{steady\_sequence} pattern: ‘\textsc{=}\textsc{+}’ with $b = 0$ and $a = 0$.
  \item \textbf{Part (B)}: Illustrating the execution of the seed transducer on a time series (within the output $o$, $f$, $\circ$ are shortcut for \textbf{out, found, out}_). The relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
\end{itemize}

\textbf{Figure 3.50:} (A) Seed transducer for the \textsc{steady\_sequence} pattern: ‘\textsc{=}\textsc{+}’ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $f$, $\circ$ are shortcut for \textbf{out, found, out}_); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
Figure 3.51: Parametrised automaton for any functional time-series constraints of the \textsc{steady\_sequence} pattern obtained by applying the decoration table 3.37 to the corresponding transducer.

Figure 3.52: Parametrised automaton for any predicate time-series constraints of the \textsc{steady\_sequence} pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=\equiv$, $\geq$) used for comparing two consecutive feature values.
### Table 3.17: Parametrised glue matrix for any $g_f$ _STEADYSEQUENCE_ constraint, where the cell annotation $M$ stands for merging two existing steady sequences sharing an element; the state associated with each row corresponds to a state of the _STEADYSEQUENCE_ transducer – see Figure 3.50, while the state associated with each column corresponds to a state of the reverse of the _STEADYSEQUENCE_ transducer, i.e. the _STEADYSEQUENCE_ transducer – see Figure 3.50.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_g(\overline{C}, \overline{C})$</td>
<td>$\phi_g(\overline{C}, \overline{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\phi_g(\overline{C}, \overline{C})$</td>
<td>$\phi_f(\overline{C}, \overline{C}, \overline{D}, \overline{D}, \delta^{i+1}_f)^M$</td>
</tr>
</tbody>
</table>
3.1.18 STRICTLY_DECENDING_SEQUENCE

Given a sequence $S$ over the alphabet $\{<, =, >\}$, an occurrence of the pattern STRICTLY_DECENDING_SEQUENCE is the maximal subsequence of $S$ which matches the regular expression $'>+\,'$. Part (A) and parts (B,C) of Figure 3.53 respectively depict the seed transducer associated with the STRICTLY_DECENDINGSEQUENCE pattern as well as one example of its execution on a time series.

Figure 3.53: (A) Seed transducer for the STRICTLY_DECENDINGSEQUENCE pattern: $'>+\,'$ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $f$, $o_a$ are shortcut for out, found, out$_a$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.54: Parametrised automaton for any functional time-series constraints of the STRICTLY_DECREASING_SEQUENCE pattern obtained by applying the decoration table 3.37 to the corresponding transducer.

Figure 3.55: Parametrised automaton for any predicate time-series constraints of the STRICTLY_DECREASING_SEQUENCE pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where θ is the predicate (i.e. ≤, =, ≥) used for comparing two consecutive feature values.
Table 3.18: Parametrised glue matrix for any g_f.Strictly Decreasing.Sequence constraint, where the cell annotation \( M \) stands for merging two existing strictly decreasing sequences sharing an element; the state associated with each row corresponds to a state of the Strictly Decreasing.Sequence transducer – see Figure 3.53, while the state associated with each column corresponds to a state of the reverse of the Strictly Decreasing.Sequence transducer, i.e. the Strictly Increasing.Sequence transducer – see Figure 3.56.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

3.1.19 STRICTLY_INCREASING_SEQUENCE

Given a sequence $S$ over the alphabet \{‘<’, ‘=’, ‘>’\}, an occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the maximal subsequence of $S$ which matches the regular expression ‘<+’. Part (A) and parts (B,C) of Figure 3.56 respectively depict the seed transducer associated with the `STRICTLY_INCREASING_SEQUENCE` pattern as well as one example of its execution on a time series.

Figure 3.56: (A) Seed transducer for the `STRICTLY_INCREASING_SEQUENCE` pattern: ‘<+’ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output o, f, o, are shortcut for out, found, out); (C) Illustrating the relation between the s-occurrence, the i-occurrence and the e-occurrence.
3. PATTERNS AND CORRESPONDING TABLES

Figure 3.57: Parametrised automaton for any functional time-series constraints of the **STRICTLY-INCREASING SEQUENCE** pattern obtained by applying the decoration table 3.37 to the corresponding transducer.

Figure 3.58: Parametrised automaton for any predicate time-series constraints of the **STRICTLY-INCREASING SEQUENCE** pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where \( \theta \) is the predicate (i.e. \( \leq, =, \geq \)) used for comparing two consecutive feature values.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Table 3.19: Parametrised glue matrix for any $g.f$ _STRICTLY_INCREASING_SEQUENCE_ constraint, where the cell annotation $M$ stands for merging two existing strictly increasing sequences sharing an element; the state associated with each row corresponds to a state of the _STRICTLY_INCREASING_SEQUENCE_ transducer – see Figure 3.56, while the state associated with each column corresponds to a state of the reverse of the _STRICTLY_INCREASING_SEQUENCE_ transducer, i.e. the _STRICTLY_DECREASING_SEQUENCE_ transducer – see Figure 3.53.
3.1.20 SUMMIT

Given a sequence $S$ over the alphabet $\{', '=' , '>'\}$, an occurrence of the pattern SUMMIT is the maximal subsequence of $S$ which matches the regular expression

$'(< [(< (= | <)) (< | (> (= | >))*) (> | (> (= | >))*)])'$.

Part (A) and parts (B,C) of Figure 3.59 respectively depict the seed transducer associated with the SUMMIT pattern as well as one example of its execution on a time series.

Figure 3.59: (A) Seed transducer for the SUMMIT pattern: $'(< [(< (= | <)) (< | (> (= | >))*) (> | (> (= | >))*)])'$ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output out, found, maybea, outa, maybeb, outb); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
3.1. PATTERNS, SEED TRANSUDCERS AND PARAMETRISED GLUE MATRICES

Figure 3.60: Parametrised automaton for any functional time-series constraints of the SUMMIT pattern obtained by applying the decoration table 3.37 to the corresponding transducer (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
Figure 3.61: Parametrised automaton for any predicate time-series constraints of the SUMMIT pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=$, $\geq$) used for comparing two consecutive feature values; transition $u \to r$ has the same register update as transition $r \to u$. 
### Table 3.20: Parametrised glue matrix for any $g_f$ _SUMMIT_ constraint, where cell annotations have the following meaning:

- $\text{C}$ stands for creating a new summit occurrence from no existing summit,
- $\text{R}$ stands for extending to the right an existing summit,
- $\text{L}$ stands for extending to the left an existing summit; the state associated with each row corresponds to a state of the _SUMMIT_ transducer – see Figure 3.59, while the state associated with each column corresponds to a state of the reverse of the _SUMMIT_ transducer, i.e. the _SUMMIT_ transducer – see Figure 3.59.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
<th>$u$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_g(\overline{C}, \overline{C})$</td>
<td>$\phi_q(\overline{C}, \overline{C})$</td>
<td>$\phi_q(\overline{C}, \overline{C})$</td>
<td>$\phi_q(\overline{C}, \overline{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\phi_q(\overline{C}, \overline{C})$</td>
<td>$\phi_f(\overline{D}, \overline{D}, \delta_t^{i+1})$</td>
<td>$\phi_q(\overline{C}, \overline{C})$</td>
<td>$\phi_q(\overline{C}, \overline{C})$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\phi_q(\overline{C}, \overline{C})$</td>
<td>$\phi_f(\overline{C}, \overline{D}, \delta_r^{j+1})$</td>
<td>$\phi_q(\overline{C}, \overline{C})$</td>
<td>$\phi_f(\overline{C}, \overline{D}, \delta_t^{i+1})$</td>
</tr>
<tr>
<td>$u$</td>
<td>$\phi_q(\overline{C}, \overline{C})$</td>
<td>$\phi_q(\overline{C}, \overline{C})$</td>
<td>$\phi_f(\overline{C}, \overline{D}, \delta_r^{j+1})$</td>
<td>$\phi_q(\overline{C}, \overline{C})$</td>
</tr>
</tbody>
</table>
3. PATTERNS AND CORRESPONDING TABLES

3.1.21 VALLEY

Given a sequence $S$ over the alphabet \{‘$<$’, ‘$=$’, ‘$>$’\}, an occurrence of the pattern VALLEY is the maximal subsequence of $S$ which matches the regular expression ‘$>$ ($=$ | $>$)* ($<$ | $=$)*’ '$<$. Part (A) and parts (B,C) Figure 3.62 respectively depict the seed transducer associated with the VALLEY pattern as well as one example of its execution on a time series.

Figure 3.62: (A) Seed transducer for the VALLEY pattern: ‘$>$ ($=$ | $>$)* ($<$ | $=$)*’ '$<’ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output o, m_b, f, m_a, o_a are shortcut for out, maybe_b, found, maybe_a, out_a); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
Figure 3.63: Parametrised automaton for any functional time-series constraints of the VALLEY pattern obtained by applying the decoration table 3.37 to the corresponding transducer.

Figure 3.64: Parametrised automaton for any predicate time-series constraints of the VALLEY pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=\geq$) used for comparing two consecutive feature values.
3. PATTERNS AND CORRESPONDING TABLES

Table 3.21: Parametrised glue matrix for any $g_f$ _VALLEY_ constraint, where cell annotations have the following meaning: $C$ stands for creating a new valley occurrence from no existing valley, $R$ stands for extending to the right an existing valley, $L$ stands for extending to the left an existing valley; the state associated with each row corresponds to a state of the _VALLEY_ transducer – see Figure 3.62, while the state associated with each column corresponds to a state of the reverse of the _VALLEY_ transducer, i.e. the _VALLEY_ transducer – see Figure 3.62.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$r$</td>
<td>$t$</td>
</tr>
<tr>
<td>$s$</td>
<td>$\phi_g(\overline{C}, \overline{C})$</td>
<td>$\phi_g(\overline{C}, \overline{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\phi_g(\overline{C}, \overline{C})$</td>
<td>$\phi_f(\overline{D}, \overline{D}, \delta_f^{i+1})$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\phi_g(\overline{C}, \overline{C})$</td>
<td>$\phi_f(\overline{D}, \overline{D}, \delta_f^{i+1})$</td>
</tr>
</tbody>
</table>
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

3.1.22 ZIGZAG

Given a sequence $S$ over the alphabet \{$'<'$, '='$, '>'$\}$, an occurrence of the pattern ZIGZAG is the maximal subsequence of $S$ which matches the regular expression

$'(<>)^+ (< | <>)^+ | (><)^+ (> | >_<)^+$.

Figures 3.65 and 3.66 respectively depict the seed transducer associated with the ZIGZAG pattern as well as one example of its execution on a time series.
Figure 3.65: Seed transducer for the ZIGZAG pattern: ‘(<>)^+ (< | <>) | (<>)^+ (> | ><>’ with b = 1 and a = 1; missing transitions from a, d to s are labelled by = : out, missing transitions from b, e to s are labelled by = : out_r, and missing transitions from c, f to s are labelled by = : out_a.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.66: (A) Illustrating the execution of the seed transducer associated with the ZIGZAG pattern on a time series (within the output \( o \), \( m_b \), \( f \), \( o_a \) are shortcut for \( \text{out} \), \( \text{maybe} \), \( \text{found} \), \( \text{out}_a \)); (B) Illustrating the relation between the \( s \)-occurrence, the \( i \)-occurrence and the \( e \)-occurrence.
### 3. PATTERNS AND CORRESPONDING TABLES

Table 3.22: Parametrised glue matrix for any \( g_f \)\_ZIGZAG constraint, where cell annotations have the following meaning: \( C \) stands for creating a new zigzag occurrence from no existing zigzag, \( R \) stands for extending to the right an existing zigzag, \( L \) stands for extending to the left an existing zigzag, \( M \) stands for merging two existing zigzags; the state associated with each row corresponds to a state of the ZIGZAG transducer – see Figure 3.65, while the state associated with each column corresponds to a state of the reverse of the ZIGZAG transducer, i.e. the ZIGZAG transducer – see Figure 3.65.
3.1. PATTERNS, SEED TRANSDUCERS AND PARAMETRISED GLUE MATRICES

Figure 3.67: Parametrised automaton for any functional time-series constraints of the ZIGZAG pattern obtained by applying the decoration table 3.37 to the corresponding transducer; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.
Figure 3.68: Parametrised automaton for any predicate time-series constraints of the ZIGZAG pattern obtained by applying the decoration table 3.38 to the corresponding transducer, where $\theta$ is the predicate (i.e. $\leq$, $=$, $\geq$) used for comparing two consecutive feature values; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the register $F$ is reset to $C$, and the register $R$ is updated wrt $C$ and $F$
3.2 Decoration Tables

This section provides different decoration tables used for synthesising the automata with registers of this catalogue. While no automaton of this catalogue relies on tables 3.34 and 3.36, these two decoration tables may be used for generating distance constraints between consecutive occurrences of a pattern.
Table 3.23: Table for Type: cDR\textup{FoundEOut}

<table>
<thead>
<tr>
<th>Initialization</th>
<th>C ← default_0</th>
<th>(D \leftarrow \text{id})</th>
<th>(R \leftarrow \text{default}_0)</th>
<th>(R \leftarrow \phi_g(R, C))</th>
</tr>
</thead>
<tbody>
<tr>
<td>out_0</td>
<td>C ← default_0</td>
<td>(D \leftarrow \text{id})</td>
<td>(D \leftarrow \phi_g(R, C))</td>
<td>(R \leftarrow \phi_g(R, C))</td>
</tr>
<tr>
<td>out_1</td>
<td>0</td>
<td>(D \leftarrow \text{id})</td>
<td>(D \leftarrow \phi_g(R, C))</td>
<td>(R \leftarrow \phi_g(R, C))</td>
</tr>
<tr>
<td>maybe_0</td>
<td>0</td>
<td>(D \leftarrow \text{id})</td>
<td>(D \leftarrow \phi_g(R, C))</td>
<td>(R \leftarrow \phi_g(R, C))</td>
</tr>
<tr>
<td>maybe_1</td>
<td>0</td>
<td>(D \leftarrow \text{id})</td>
<td>(D \leftarrow \phi_g(R, C))</td>
<td>(R \leftarrow \phi_g(R, C))</td>
</tr>
<tr>
<td>found_0</td>
<td>C ← default_0</td>
<td>(D \leftarrow \text{id})</td>
<td>(D \leftarrow \phi_g(R, C))</td>
<td>(R \leftarrow \phi_g(R, C))</td>
</tr>
<tr>
<td>found_1</td>
<td>0</td>
<td>(D \leftarrow \text{id})</td>
<td>(D \leftarrow \phi_g(R, C))</td>
<td>(R \leftarrow \phi_g(R, C))</td>
</tr>
<tr>
<td>in_0</td>
<td>1</td>
<td>(D \leftarrow \text{id})</td>
<td>(D \leftarrow \phi_g(R, C))</td>
<td>(R \leftarrow \phi_g(R, C))</td>
</tr>
<tr>
<td>in_1</td>
<td>1</td>
<td>(D \leftarrow \text{id})</td>
<td>(D \leftarrow \phi_g(R, C))</td>
<td>(R \leftarrow \phi_g(R, C))</td>
</tr>
</tbody>
</table>

Semantic Letter Decoration

- After Update of C: \(C \leftarrow \text{default}\_0\) → \(C \leftarrow \phi_g(R, C)\)
- Update of D:
  - out\_0: \(D \leftarrow \text{id}\)
  - out\_1: \(D \leftarrow \phi_g(R, C)\)
  - maybe\_0: \(D \leftarrow \text{id}\)
  - maybe\_1: \(D \leftarrow \phi_g(R, C)\)
  - found\_0: \(D \leftarrow \text{id}\)
  - found\_1: \(D \leftarrow \phi_g(R, C)\)
  - in\_0: \(D \leftarrow \text{id}\)
  - in\_1: \(D \leftarrow \phi_g(R, C)\)
- Update of R:
  - \(R \leftarrow \phi_g(R, C)\)
### 3.2. DECORATION TABLES

Table 3.24: Table for Type: cDRFoundEOutNoMA

<table>
<thead>
<tr>
<th>Initialization</th>
<th>( C \leftarrow \text{default} )</th>
<th>( D \leftarrow \text{id} )</th>
<th>Update of ( D )</th>
<th>Decr ( R )</th>
<th>Update of ( R )</th>
</tr>
</thead>
<tbody>
<tr>
<td>out ( _ _ _ _ )</td>
<td>( \phi(f(R, C)) )</td>
<td>( \phi(f(D, \delta_i f)) )</td>
<td>( \phi(f(R, \delta_i f)) )</td>
<td>( \phi(f(D, \delta_i f)) )</td>
<td>( \phi(f(R, \delta_i f)) )</td>
</tr>
<tr>
<td>found _ _ _ _</td>
<td>( \phi(f(R, C)) )</td>
<td>( \phi(f(D, \delta_i f)) )</td>
<td>( \phi(f(R, \delta_i f)) )</td>
<td>( \phi(f(D, \delta_i f)) )</td>
<td>( \phi(f(R, \delta_i f)) )</td>
</tr>
<tr>
<td>found _ _ _ _</td>
<td>( \phi(f(R, C)) )</td>
<td>( \phi(f(D, \delta_i f)) )</td>
<td>( \phi(f(R, \delta_i f)) )</td>
<td>( \phi(f(D, \delta_i f)) )</td>
<td>( \phi(f(R, \delta_i f)) )</td>
</tr>
<tr>
<td>found _ _ _ _</td>
<td>( \phi(f(R, C)) )</td>
<td>( \phi(f(D, \delta_i f)) )</td>
<td>( \phi(f(R, \delta_i f)) )</td>
<td>( \phi(f(D, \delta_i f)) )</td>
<td>( \phi(f(R, \delta_i f)) )</td>
</tr>
<tr>
<td>found _ _ _ _</td>
<td>( \phi(f(R, C)) )</td>
<td>( \phi(f(D, \delta_i f)) )</td>
<td>( \phi(f(R, \delta_i f)) )</td>
<td>( \phi(f(D, \delta_i f)) )</td>
<td>( \phi(f(R, \delta_i f)) )</td>
</tr>
<tr>
<td>found _ _ _ _</td>
<td>( \phi(f(R, C)) )</td>
<td>( \phi(f(D, \delta_i f)) )</td>
<td>( \phi(f(R, \delta_i f)) )</td>
<td>( \phi(f(D, \delta_i f)) )</td>
<td>( \phi(f(R, \delta_i f)) )</td>
</tr>
</tbody>
</table>
Table 3.25: Table for Type: cR Found E Out A

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>After Update of C</th>
<th>Update of R</th>
</tr>
</thead>
<tbody>
<tr>
<td>initialization</td>
<td>$C ← \text{default}_{sf}$</td>
<td>$R ← \text{default}_{sf}$</td>
</tr>
<tr>
<td>return</td>
<td>$φ_g(R,C)$</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>out$_r$</td>
<td>$C ← \text{default}_{sf}$</td>
<td>$R ← \phi_g(R,C)$</td>
</tr>
<tr>
<td>maybe$_a$</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>maybe$_a$</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>found</td>
<td>0</td>
<td>$C ← φ_f(δ_j, δ_{j+1})$</td>
</tr>
<tr>
<td>found</td>
<td>1</td>
<td>$C ← δ_j^f$</td>
</tr>
<tr>
<td>in</td>
<td>0</td>
<td>$C ← φ_f(C, δ_{j+1})$</td>
</tr>
<tr>
<td>in</td>
<td>1</td>
<td>$C ← φ_f(C, δ_j^f)$</td>
</tr>
<tr>
<td>found$_a$</td>
<td>0</td>
<td>$C ← \text{default}_{sf}$</td>
</tr>
<tr>
<td>found$_a$</td>
<td>1</td>
<td>$C ← \text{default}_{sf}$</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3. PATTERNS AND CORRESPONDING TABLES
### Table 3.26: Table for Type: dRFoundFoundEInA

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$D \leftarrow \text{id}_f$</th>
<th>$R \leftarrow \text{default}_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Return</strong></td>
<td>$R$</td>
<td>$R$</td>
</tr>
<tr>
<td><strong>Semantic Letter</strong></td>
<td><strong>After Update of $D$</strong></td>
<td><strong>Update of $R$</strong></td>
</tr>
<tr>
<td><strong>Out</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_\delta(R, \phi_\delta(D, \delta^i_f))$</td>
</tr>
<tr>
<td><strong>out_r</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_\delta(R, \phi_\delta(D, \delta^i_f))$</td>
</tr>
<tr>
<td><strong>out_a</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_\delta(R, \phi_\delta(D, \delta^i_f))$</td>
</tr>
<tr>
<td><strong>maybe_b</strong></td>
<td>$D \leftarrow \phi_\delta(D, \delta^i_f)$</td>
<td>$R \leftarrow \phi_\delta(R, \phi_\delta(D, \delta^i_f))$</td>
</tr>
<tr>
<td><strong>maybe_a</strong></td>
<td>$D \leftarrow \phi_\delta(D, \delta^i_f)$</td>
<td>$R \leftarrow \phi_\delta(R, \phi_\delta(D, \delta^i_f))$</td>
</tr>
<tr>
<td><strong>Found</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_\delta(R, \phi_\delta(D, \delta^i_f))$</td>
</tr>
<tr>
<td><strong>found_e</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_\delta(R, \phi_\delta(D, \delta^i_f))$</td>
</tr>
<tr>
<td><strong>In</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_\delta(R, \phi_\delta(D, \delta^i_f))$</td>
</tr>
<tr>
<td><strong>in_a</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_\delta(R, \phi_\delta(D, \delta^i_f))$</td>
</tr>
<tr>
<td><strong>Found_e</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_\delta(R, \phi_\delta(D, \delta^i_f))$</td>
</tr>
<tr>
<td><strong>found_a</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_\delta(R, \phi_\delta(D, \delta^i_f))$</td>
</tr>
</tbody>
</table>
### Table 3.27: Table for Type: dRFoundFoundEinB

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>After</th>
<th>Update of $D$</th>
<th>Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>out</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \text{default}_f$</td>
<td></td>
</tr>
<tr>
<td><strong>out</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \text{default}_f$</td>
<td></td>
</tr>
<tr>
<td><strong>maybe</strong></td>
<td>$D \leftarrow \phi_f(D, \delta_f')$</td>
<td>$R \leftarrow \text{default}_f$</td>
<td></td>
</tr>
<tr>
<td><strong>maybe</strong></td>
<td>$D \leftarrow \phi_f(D, \delta_f^{i+1})$</td>
<td>$R \leftarrow \text{default}_f$</td>
<td></td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$D \leftarrow \phi_f(D, \delta_f')$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f'))$</td>
<td></td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$D \leftarrow \phi_f(D, \delta_f^{i+1})$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^{i+1}))$</td>
<td></td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>$D \leftarrow \phi_f(D, \delta_f')$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f'))$</td>
<td></td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>$D \leftarrow \phi_f(D, \delta_f^{i+1})$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^{i+1}))$</td>
<td></td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f'))$</td>
<td></td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^{i+1}))$</td>
<td></td>
</tr>
</tbody>
</table>
### Table 3.28: Table for Type: dRFoundFoundEInMonMB

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$D \leftarrow \text{id}_f$</th>
<th>$R \leftarrow \text{default}_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return</td>
<td>$R$</td>
<td>$R$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Decoration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update of $D$</td>
<td>Update of $R$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>out</th>
<th>$D \leftarrow \delta_j$</th>
</tr>
</thead>
<tbody>
<tr>
<td>out$_r$</td>
<td></td>
</tr>
<tr>
<td>out$_a$</td>
<td></td>
</tr>
<tr>
<td>maybe$_b$</td>
<td></td>
</tr>
<tr>
<td>maybe$_a$</td>
<td></td>
</tr>
<tr>
<td>found$_f$</td>
<td></td>
</tr>
<tr>
<td>in</td>
<td></td>
</tr>
<tr>
<td>found$_a$</td>
<td>$R \leftarrow \phi_a(R, \phi_f(D, \delta_j))$</td>
</tr>
</tbody>
</table>
### Table 3.29: Table for Type: dRFoundFoundEInNoMA

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$D \leftarrow \mathbf{id}_f$</th>
<th>$R \leftarrow \mathbf{default}_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return</td>
<td>$R$</td>
<td></td>
</tr>
<tr>
<td>Semantic Letter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Decoration</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>After</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Update of $D$</strong></td>
<td>$D \leftarrow \mathbf{id}_f$</td>
<td></td>
</tr>
<tr>
<td><strong>Update of $R$</strong></td>
<td></td>
<td>$R \leftarrow \mathbf{default}_f$</td>
</tr>
<tr>
<td><strong>out</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>out_a</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>maybe_a</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>maybe_a</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>found</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>in</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>found_a</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>found_a</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **out**: $D \leftarrow \mathbf{id}_f$
- **out_a**: $D \leftarrow \mathbf{id}_f$
- **maybe_a**: $D \leftarrow \phi_f(D, \delta_f)$
- **maybe_a**: $D \leftarrow \phi_f(D, \delta_f)$
- **found**: $R \leftarrow \phi_u(R, \phi_f(D, \delta_f), \delta_f)$
- **found**: $R \leftarrow \phi_u(R, \phi_f(D, \delta_f), \delta_f)$
- **in**: $D \leftarrow \mathbf{id}_f$
- **in**: $R \leftarrow \phi_u(R, \phi_f(D, \delta_f))$
- **found_a**: $D \leftarrow \mathbf{id}_f$
- **found_a**: $R \leftarrow \phi_u(R, \phi_f(D, \delta_f), \delta_f)$
- **found_a**: $D \leftarrow \mathbf{id}_f$
- **found_a**: $R \leftarrow \phi_u(R, \phi_f(D, \delta_f))$
### Table 3.30: Table for Type: dRFoundFoundEInNoMBA

<table>
<thead>
<tr>
<th>Initialization</th>
<th>After</th>
<th>Update of D</th>
<th>Update of R</th>
</tr>
</thead>
<tbody>
<tr>
<td>return</td>
<td>$D \leftarrow id_f$</td>
<td>$R \leftarrow \text{default}_{g_f}$</td>
<td></td>
</tr>
<tr>
<td><strong>Semantic Letter</strong></td>
<td><strong>Update of D</strong></td>
<td><strong>Decoration</strong></td>
<td></td>
</tr>
<tr>
<td><strong>out</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>out&lt;sub&gt;_f&lt;/sub&gt;</td>
<td>$D \leftarrow id_f$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>out&lt;sub&gt;_a&lt;/sub&gt;</td>
<td>$D \leftarrow id_f$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>maybe&lt;sub&gt;_a&lt;/sub&gt;</td>
<td>0</td>
<td>$D \leftarrow \phi_f(D, \delta_f^{i+1})$</td>
<td></td>
</tr>
<tr>
<td>maybe&lt;sub&gt;_a&lt;/sub&gt;</td>
<td>1</td>
<td>$D \leftarrow \phi_f(D, \delta_f^i)$</td>
<td></td>
</tr>
<tr>
<td>found&lt;sub&gt;_a&lt;/sub&gt;</td>
<td>0</td>
<td>$D \leftarrow id_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^{i+1}))$</td>
</tr>
<tr>
<td>found&lt;sub&gt;_a&lt;/sub&gt;</td>
<td>1</td>
<td>$D \leftarrow id_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^i))$</td>
</tr>
<tr>
<td>found&lt;sub&gt;_a&lt;/sub&gt;</td>
<td>0</td>
<td>$D \leftarrow id_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^{i+1}))$</td>
</tr>
<tr>
<td>found&lt;sub&gt;_a&lt;/sub&gt;</td>
<td>1</td>
<td>$D \leftarrow id_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^i))$</td>
</tr>
</tbody>
</table>
Table 3.31: Table for Type: dRFoundEInNoMBB

<table>
<thead>
<tr>
<th>Initializer</th>
<th>After Update of D</th>
<th>Update of R</th>
</tr>
</thead>
<tbody>
<tr>
<td>out</td>
<td>( D \leftarrow id_f )</td>
<td>( R \leftarrow \text{default}_f )</td>
</tr>
<tr>
<td>out(_f)</td>
<td>( D \leftarrow id_f )</td>
<td>( R \leftarrow \text{default}_f )</td>
</tr>
<tr>
<td>maybe(_a)</td>
<td>( D \leftarrow \phi_f(D, \delta_f) )</td>
<td>( R \leftarrow \text{default}_f )</td>
</tr>
<tr>
<td>found(_a)</td>
<td>( D \leftarrow \phi_f(\delta_f, \delta_f) )</td>
<td>( R \leftarrow \text{default}_f )</td>
</tr>
<tr>
<td>found(_e)</td>
<td>( D \leftarrow id_f )</td>
<td>( R \leftarrow \text{default}_f )</td>
</tr>
<tr>
<td>found(_e)</td>
<td>( D \leftarrow id_f )</td>
<td>( R \leftarrow \text{default}_f )</td>
</tr>
</tbody>
</table>
3.2. DECORATION TABLES

Table 3.32: Table for Type: features

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>After</th>
<th>Guard</th>
<th>Update of C</th>
<th>Update of D</th>
</tr>
</thead>
<tbody>
<tr>
<td>initialization</td>
<td>$f_n = \text{default}$</td>
<td>$e_n = C$</td>
<td>$C \leftarrow \text{id}_f$</td>
<td>$D \leftarrow \text{id}_d$</td>
</tr>
</tbody>
</table>

- **out**: $f_i = \text{default}, e_i = \text{default}$
- **out**: $f_i = \text{default}, e_i = e_{i+1}$
- **maybe**: $f_i = \text{default}, e_i = e_{i+1}$
- **maybe**: $0$ $f_i = \text{default}, e_i = e_{i+1}$ $C \leftarrow \text{default}_{g_f}$ $D \leftarrow \text{id}_d$
- **maybe**: $1$ $f_i = \text{default}, e_i = e_{i+1}$ $D \leftarrow \phi_f(D, \delta_{g_f}^{i+1})$ $D \leftarrow \phi_f(D, \delta_d^{i+1})$
- **found**: $0$ $f_i = e_i, e_i = e_{i+1}$ $C \leftarrow \phi_f(D, \delta_f^{i+1})$ $D \leftarrow \text{id}_d$
- **found**: $1$ $f_i = e_i, e_i = e_{i+1}$ $C \leftarrow \phi_f(D, \delta_f^{i+1})$ $D \leftarrow \text{id}_d$
- **found**: $0$ $f_i = \phi_f(D, \delta_f^{i+1}), e_i = \text{default}$ $C \leftarrow \phi_f(D, \delta_d^{i+1})$ $D \leftarrow \text{id}_d$
- **found**: $1$ $f_i = \phi_f(D, \delta_f^{i+1}), e_i = \text{default}$ $D \leftarrow \text{id}_d$
Table 3.33: Table for Type: footprint

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Guard</th>
<th>Decoration</th>
<th>Update of $C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial</td>
<td></td>
<td>$p_n = 0$</td>
<td>$C \leftarrow 0$</td>
</tr>
<tr>
<td>return</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>out</td>
<td>$p_1 = 0$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>out</td>
<td>$p_1 = 0$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>out</td>
<td>$p_1 = 0$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>maybe</td>
<td>$p_i = p_{i+1}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>maybe</td>
<td>$p_i = p_{i+1}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>found</td>
<td>$p_i = C + 1$</td>
<td>$C \leftarrow C + 1$</td>
<td>$C \leftarrow C + 1$</td>
</tr>
<tr>
<td>found</td>
<td>$p_i = C + 1$</td>
<td>$C \leftarrow C + 1$</td>
<td>$C \leftarrow C + 1$</td>
</tr>
<tr>
<td>in</td>
<td>$p_i = C$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 3.34: Table for Type: footprint_distance

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Guard</th>
<th>Update of $C$</th>
<th>Update of $D$</th>
<th>Update of $F$</th>
<th>Update of $M$</th>
</tr>
</thead>
<tbody>
<tr>
<td>out</td>
<td></td>
<td>$D \leftarrow D + 1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>out_r</td>
<td></td>
<td>$D \leftarrow D + M$</td>
<td>$M \leftarrow 0$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>out_i</td>
<td></td>
<td>$D \leftarrow D + 1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>maybe_b</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$M \leftarrow M + 1$</td>
</tr>
<tr>
<td>found_e</td>
<td>$F = 1$</td>
<td>$C \leftarrow \phi_g(C, D)$</td>
<td>$D \leftarrow 0$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>found_e</td>
<td>$F = 0$</td>
<td>$D \leftarrow 0$</td>
<td>$F \leftarrow 1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>found_w</td>
<td>$F = 1$</td>
<td>$C \leftarrow \phi_g(C, D)$</td>
<td>$D \leftarrow 0$</td>
<td></td>
<td>$F \leftarrow 1$</td>
</tr>
<tr>
<td>found_w</td>
<td>$F = 0$</td>
<td>$D \leftarrow 0$</td>
<td>$F \leftarrow 1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>in</td>
<td></td>
<td></td>
<td>$D \leftarrow 0$</td>
<td></td>
<td>$F \leftarrow 1$</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Pattern</th>
<th>Effect on $C$</th>
<th>Effect on $D$</th>
<th>Effect on $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_n = 0$</td>
<td>$C \leftarrow \text{default}_f$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \text{default}_f$</td>
</tr>
<tr>
<td>$C &lt; \gamma R \Rightarrow c_{tn} = 1, a_{tn} = 0$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$C = R, R \neq \text{default}<em>f \Rightarrow c</em>{tn} = 0, a_{tn} = 0$</td>
<td>$C \leftarrow \text{default}_f$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \text{default}_f$</td>
</tr>
<tr>
<td>$R &lt; \gamma C \Rightarrow c_{tn} = 0, a_{tn} = 1$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Semantic Letter Decoration**

<table>
<thead>
<tr>
<th>Initial</th>
<th>Guard Update</th>
<th>Update of $C$</th>
<th>Update of $D$</th>
<th>Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_i = 0, c_{i+1} = c_i, a_{i+1} = a_i$</td>
<td>$C \leftarrow \text{default}_f$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \text{default}_f$</td>
<td></td>
</tr>
<tr>
<td>$f_i = 0, c_{i+1} = c_i, a_{i+1} = a_i$</td>
<td>$C \leftarrow \text{default}_f$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \text{default}_f$</td>
<td></td>
</tr>
<tr>
<td>$c &lt; \gamma r, f_i = 0, a_{i+1} = a_i$</td>
<td>$C \leftarrow \text{default}_f$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \text{default}_f$</td>
<td></td>
</tr>
<tr>
<td>$r &lt; \gamma c, f_i = 0, c_{i+1} = c_i$</td>
<td>$C \leftarrow \text{default}_f$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \text{default}_f$</td>
<td></td>
</tr>
<tr>
<td>$f_i = 0, c_{i+1} = c_i, a_{i+1} = a_i$</td>
<td>$C \leftarrow \text{default}_f$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \text{default}_f$</td>
<td></td>
</tr>
<tr>
<td>$f_i = 0, c_{i+1} = c_i, a_{i+1} = a_i$</td>
<td>$C \leftarrow \text{default}_f$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \text{default}_f$</td>
<td></td>
</tr>
<tr>
<td>$C \leftarrow \text{default}_f$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \text{default}_f$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table 3.15:** Table for Type: found ($\phi$ stands for $\min$ and $\phi$ when $\gamma = \max$)
3.2. **DECORATION TABLES**

Table 3.36: Table for Type: found_distance

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Guard</th>
<th>Update of C</th>
<th>Update of F</th>
</tr>
</thead>
<tbody>
<tr>
<td>out</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>out₀</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>outₐ</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>maybe</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>found₀</td>
<td>(F &gt; 0)</td>
<td>(C \leftarrow \phi_{y}(C, i - F))</td>
<td>(F \leftarrow i)</td>
</tr>
<tr>
<td>foundₐ</td>
<td>(F = 0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>found</td>
<td>(F &gt; 0)</td>
<td>(C \leftarrow \phi_{y}(C, i - F))</td>
<td>(F \leftarrow i)</td>
</tr>
<tr>
<td>found</td>
<td>(F = 0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Initialization</td>
<td>C ← default(R)</td>
<td>D ← id</td>
<td>R ← default(R)</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------</td>
<td>--------</td>
<td>---------------</td>
</tr>
<tr>
<td>Semantic Letter</td>
<td>After Update of C</td>
<td>Update of D</td>
<td>Decroration</td>
</tr>
<tr>
<td>out</td>
<td>C ← default(R)</td>
<td>D ← id</td>
<td>R ← φ(R, C)</td>
</tr>
<tr>
<td>out</td>
<td>C ← default(R)</td>
<td>D ← φ(D, δ_1)</td>
<td>D ← id</td>
</tr>
<tr>
<td>maybe_b</td>
<td>C ← φ(C, φ(D, δ_{i+1}))</td>
<td>D ← id</td>
<td></td>
</tr>
<tr>
<td>maybe_b</td>
<td>C ← φ(C, φ(D, δ_{i+1}))</td>
<td>D ← id</td>
<td></td>
</tr>
<tr>
<td>found</td>
<td>C ← φ(C, φ(D, δ_{i+1}))</td>
<td>D ← id</td>
<td></td>
</tr>
<tr>
<td>in</td>
<td>C ← φ(C, φ(D, δ_{i+1}))</td>
<td>D ← id</td>
<td></td>
</tr>
<tr>
<td>found</td>
<td>C ← φ(C, φ(D, δ_{i+1}))</td>
<td>D ← id</td>
<td></td>
</tr>
<tr>
<td>found</td>
<td>C ← φ(C, φ(D, δ_{i+1}))</td>
<td>D ← id</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.37: Table for Type: function
Table 3.38: Table for Type: predicate

<table>
<thead>
<tr>
<th>Update of ( C )</th>
<th>Update of ( D )</th>
<th>Update of ( F )</th>
<th>Update of ( R )</th>
</tr>
</thead>
<tbody>
<tr>
<td>out</td>
<td>( D \to id_f )</td>
<td>( F \to default _\theta )</td>
<td>( R \to 1 )</td>
</tr>
<tr>
<td>maybe</td>
<td>( D \to id_f )</td>
<td>( F \to default _\theta )</td>
<td>( R \to 1 )</td>
</tr>
<tr>
<td>maybe</td>
<td>( D \to id_f )</td>
<td>( F \to C )</td>
<td>( R \to 1 )</td>
</tr>
<tr>
<td>found</td>
<td>( D \to id_f )</td>
<td>( F \to R \wedge \theta(F, C) )</td>
<td>( R \to 1 )</td>
</tr>
<tr>
<td>found</td>
<td>( D \to id_f )</td>
<td>( F \to R \wedge \theta(F, C) )</td>
<td>( R \to 1 )</td>
</tr>
<tr>
<td>maybe</td>
<td>( D \to id_f )</td>
<td>( F \to \phi_f(D, \delta_{f+1}) )</td>
<td>( R \to 1 )</td>
</tr>
<tr>
<td>maybe</td>
<td>( D \to id_f )</td>
<td>( F \to \phi_f(D, \delta_{f+1}) )</td>
<td>( R \to 1 )</td>
</tr>
<tr>
<td>found</td>
<td>( D \to id_f )</td>
<td>( F \to \phi_f(D, \delta_{f+1}) )</td>
<td>( R \to 1 )</td>
</tr>
<tr>
<td>found</td>
<td>( D \to id_f )</td>
<td>( F \to \phi_f(D, \delta_{f+1}) )</td>
<td>( R \to 1 )</td>
</tr>
<tr>
<td>maybe</td>
<td>( D \to id_f )</td>
<td>( F \to \phi_f(D, \delta_{f+1}) )</td>
<td>( R \to 1 )</td>
</tr>
<tr>
<td>maybe</td>
<td>( D \to id_f )</td>
<td>( F \to \phi_f(D, \delta_{f+1}) )</td>
<td>( R \to 1 )</td>
</tr>
<tr>
<td>found</td>
<td>( D \to id_f )</td>
<td>( F \to \phi_f(D, \delta_{f+1}) )</td>
<td>( R \to 1 )</td>
</tr>
<tr>
<td>found</td>
<td>( D \to id_f )</td>
<td>( F \to \phi_f(D, \delta_{f+1}) )</td>
<td>( R \to 1 )</td>
</tr>
</tbody>
</table>
Table 3.39: Table for Type: rFoundFoundE

<table>
<thead>
<tr>
<th>Initialization</th>
<th>r ← defaultγ_φ</th>
</tr>
</thead>
<tbody>
<tr>
<td>return</td>
<td>R</td>
</tr>
<tr>
<td>Semantic Letter</td>
<td>Decoration</td>
</tr>
<tr>
<td>out</td>
<td>out_R</td>
</tr>
<tr>
<td>out_a</td>
<td>maybe_a</td>
</tr>
<tr>
<td>maybe_a</td>
<td>found</td>
</tr>
<tr>
<td>found</td>
<td>in</td>
</tr>
<tr>
<td>found_a</td>
<td></td>
</tr>
</tbody>
</table>
Table 3.40: Table for Type: `rFoundFoundEin`

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$R \leftarrow \text{default}_{g/f}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>return</td>
<td>$R$</td>
</tr>
<tr>
<td>Semantic Letter</td>
<td>Decoration</td>
</tr>
<tr>
<td>out</td>
<td>Update of $R$</td>
</tr>
<tr>
<td>out$_r$</td>
<td></td>
</tr>
<tr>
<td>out$_v$</td>
<td></td>
</tr>
<tr>
<td>maybe$_b$</td>
<td></td>
</tr>
<tr>
<td>maybe$_u$</td>
<td></td>
</tr>
<tr>
<td>found</td>
<td>$R \leftarrow \phi_g(R, \phi_f(\delta^<em>_i, \delta^</em>_i+1))$</td>
</tr>
<tr>
<td>in</td>
<td>$R \leftarrow \phi_g(R, \delta^*_i+1)$</td>
</tr>
<tr>
<td>found$_a$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(\delta^<em>_i, \delta^</em>_i+1))$</td>
</tr>
</tbody>
</table>
Table 3.41: Table for Type: rFoundInSWZigzag

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Decoration</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initialization</strong></td>
<td>$R \leftarrow \text{default}_{gf}$</td>
</tr>
<tr>
<td><strong>return</strong></td>
<td>$R$</td>
</tr>
<tr>
<td><strong>Update of $R$</strong></td>
<td>$R$</td>
</tr>
<tr>
<td><strong>out</strong></td>
<td>found</td>
</tr>
<tr>
<td><strong>out_{i}</strong></td>
<td>$\phi_g(R, \delta^+_i)$, $\delta^+_i$</td>
</tr>
<tr>
<td><strong>maybe_{i}</strong></td>
<td></td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$\phi_g(R, \delta^+_i)$</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td></td>
</tr>
<tr>
<td><strong>found_{i}</strong></td>
<td></td>
</tr>
</tbody>
</table>
Table 3.42: Table for Type: rangeCHRFoundEOutA

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Update of $C$</th>
<th>Update of $H$</th>
<th>Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>initialization</strong></td>
<td>$C \leftarrow \text{default}_{gf}$</td>
<td>$H \leftarrow \text{VAR}_1$</td>
<td>$R \leftarrow \text{default}_{gf}$</td>
</tr>
<tr>
<td><strong>return</strong></td>
<td>$\phi_g(R, C)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>out</strong></td>
<td>$H \leftarrow \delta_j^{i+1}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>out&lt;sub&gt;r&lt;/sub&gt;</strong></td>
<td>$H \leftarrow \delta_j^{i+1}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>out&lt;sub&gt;a&lt;/sub&gt;</strong></td>
<td>$C \leftarrow \text{default}_{gf}$</td>
<td>$H \leftarrow \delta_j^{i+1}$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
</tr>
<tr>
<td><strong>maybe&lt;sub&gt;b&lt;/sub&gt;</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>maybe&lt;sub&gt;a&lt;/sub&gt;</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$C \leftarrow H - \delta_j^{i+1}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>$C \leftarrow H - \delta_j^{i+1}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>found&lt;sub&gt;e&lt;/sub&gt;</strong></td>
<td>$H \leftarrow \delta_j^{i+1}$</td>
<td>$R \leftarrow \phi_g(R, H - \delta_j^{i+1})$</td>
<td></td>
</tr>
</tbody>
</table>
### 3. PATTERNS AND CORRESPONDING TABLES

Table 3.43: Table for Type: rangeCHRFoundEOutB

<table>
<thead>
<tr>
<th>Initialization</th>
<th>Return</th>
<th>Semantic Letter</th>
<th>Update of C</th>
<th>Update of H</th>
<th>Update of R</th>
</tr>
</thead>
<tbody>
<tr>
<td>C ← default_g_f</td>
<td>H ← VAR_i</td>
<td>( \phi_g(R, C) )</td>
<td>H ← ( \delta_{f+1}^i )</td>
<td>H ← ( \delta_{f+1}^i )</td>
<td>R ← ( \phi_g(R, C) )</td>
</tr>
<tr>
<td>maybe_b</td>
<td>maybe_a</td>
<td>found</td>
<td>C ← default_g_f</td>
<td>H ← ( \delta_{f+1}^i )</td>
<td>R ← ( \phi_g(R, C) )</td>
</tr>
<tr>
<td>in</td>
<td>found_a</td>
<td>H ← ( \delta_{f+1}^i )</td>
<td>R ← ( \phi_g(R, \delta_{f+1}^i - H) )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 3.44: Table for Type: rangeHRFoundFoundEInA

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$H \leftarrow \forall k_i$</th>
<th>$R \leftarrow \text{default}_y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic Letter</td>
<td>Update of $H$</td>
<td>Update of $R$</td>
</tr>
<tr>
<td>out</td>
<td>$H \leftarrow \delta_I^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>out$_r$</td>
<td>$H \leftarrow \delta_F^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>out$_a$</td>
<td>$H \leftarrow \delta_F^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>maybe$_b$</td>
<td>$H \leftarrow \delta_F^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>maybe$_a$</td>
<td>$H \leftarrow \delta_F^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>found</td>
<td>$R \leftarrow \phi_y(R, H - \delta_I^{i+1})$</td>
<td></td>
</tr>
<tr>
<td>in</td>
<td>$R \leftarrow \phi_y(R, H - \delta_F^{i+1})$</td>
<td></td>
</tr>
<tr>
<td>found$_a$</td>
<td>$H \leftarrow \delta_F^{i+1}$</td>
<td>$R \leftarrow \phi_y(R, H - \delta_F^{i+1})$</td>
</tr>
</tbody>
</table>
### Table 3.45: Table for Type: rangeHRFoundFoundEInB

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$H \leftarrow \text{VAR}_i$</th>
<th>$R \leftarrow \text{default}_{gf}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic Letter</td>
<td>$H \leftarrow ^\delta_i^{i+1}$</td>
<td>$R \leftarrow ^\delta_i^{i+1}$</td>
</tr>
<tr>
<td>out</td>
<td>$H \leftarrow ^\delta_i^{i+1}$</td>
<td>$R \leftarrow ^\delta_i^{i+1}$</td>
</tr>
<tr>
<td>out$_r$</td>
<td>$H \leftarrow ^\delta_i^{i+1}$</td>
<td>$R \leftarrow ^\delta_i^{i+1}$</td>
</tr>
<tr>
<td>out$_a$</td>
<td>$H \leftarrow ^\delta_i^{i+1}$</td>
<td>$R \leftarrow ^\delta_i^{i+1}$</td>
</tr>
<tr>
<td>maybe$_b$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>maybe$_a$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>found</td>
<td></td>
<td></td>
</tr>
<tr>
<td>in</td>
<td></td>
<td></td>
</tr>
<tr>
<td>found$_e$</td>
<td>$H \leftarrow ^\delta_i^{i+1}$</td>
<td>$R \leftarrow ^\phi_i(R, ^\delta_i^{i+1} - H)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 3.46: Table for Type: rangeR\text{Found}FoundEInA

<table>
<thead>
<tr>
<th>Initialization</th>
<th>( R \leftarrow \text{default}_{/f} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>return</td>
<td>( R )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Update of ( R )</th>
</tr>
</thead>
<tbody>
<tr>
<td>out</td>
<td></td>
</tr>
<tr>
<td>out(_e)</td>
<td></td>
</tr>
<tr>
<td>out(_a)</td>
<td></td>
</tr>
<tr>
<td>maybe(_a)</td>
<td></td>
</tr>
<tr>
<td>found(_a)</td>
<td></td>
</tr>
<tr>
<td>found(_e)</td>
<td></td>
</tr>
<tr>
<td>in</td>
<td></td>
</tr>
<tr>
<td>found(_a)</td>
<td></td>
</tr>
</tbody>
</table>

\[
R \leftarrow \phi(g(R, \max(0, \delta^j_i - \delta^{j+1}_i)))
\]

\[
R \leftarrow \phi(g(R, \max(0, \delta^j_i - \delta^{j+1}_i)))
\]
### 3. PATTERNS AND CORRESPONDING TABLES

Table 3.47: Table for Type: rangeRFoundFoundEInB

<table>
<thead>
<tr>
<th>Initialization</th>
<th>return</th>
<th>Semantic Letter</th>
<th>Decoration</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R \leftarrow \text{default}_{gf}$</td>
<td>$R$</td>
<td>$R$</td>
<td>$R$</td>
</tr>
</tbody>
</table>

- **out**
- **out<sub>r</sub>**
- **out<sub>c</sub>**
- **maybe<sub>B</sub>**
- **found**

Update of $R$

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$R \leftarrow \phi_g(R, \max(0, \delta_i^{j+1} - \delta_i^j))$</td>
<td>$R \leftarrow \phi_g(R, \max(0, \delta_i^{j+1} - \delta_i^j))$</td>
<td>$R \leftarrow \phi_g(R, \max(0, \delta_i^{j+1} - \delta_i^j))$</td>
</tr>
</tbody>
</table>
### 3.2. DECORATION TABLES

Table 3.48: Table for Type: range_function

<table>
<thead>
<tr>
<th>Initialization</th>
<th>return $\phi_g(R, C)$</th>
<th>$C \leftarrow$ default$_{gf}$</th>
<th>$H \leftarrow$ VAR$_1$</th>
<th>$R \leftarrow$ default$_{gf}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic Letter</td>
<td>Update of $C$</td>
<td>Update of $H$</td>
<td>Update of $R$</td>
<td></td>
</tr>
<tr>
<td>out</td>
<td>$H \leftarrow \delta^{i+1}_f$</td>
<td>$H \leftarrow \delta^{i+1}_f$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
<td></td>
</tr>
<tr>
<td>out$_a$</td>
<td>$C \leftarrow$ default$_{gf}$</td>
<td>$H \leftarrow \delta^{i+1}_f$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
<td></td>
</tr>
<tr>
<td>maybe$_b$</td>
<td>$H \leftarrow \delta^{i+1}_f$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>found$_a$</td>
<td>$C \leftarrow</td>
<td>H - \delta^{i+1}_f</td>
<td>$</td>
<td>$C \leftarrow</td>
</tr>
<tr>
<td>found</td>
<td>$C \leftarrow</td>
<td>H - \delta^{i+1}_f</td>
<td>$</td>
<td>$C \leftarrow</td>
</tr>
</tbody>
</table>
Table 3.49: Table for Type: range_predicate

<table>
<thead>
<tr>
<th>Initialization</th>
<th>Update of $C$</th>
<th>Update of $F$</th>
<th>Update of $H$</th>
<th>Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C \leftarrow \text{default}_0$</td>
<td>$F \leftarrow \text{default}_0$</td>
<td>$H \leftarrow \text{VAR}$</td>
<td>$R \leftarrow 1$</td>
<td></td>
</tr>
<tr>
<td>$R \land \theta(F, C)$</td>
<td>$H \leftarrow \delta_f^{i+1}$</td>
<td>$H \leftarrow \delta_f^{i+1}$</td>
<td>$R \leftarrow R \land \theta(F, C)$</td>
<td></td>
</tr>
</tbody>
</table>

**Semantic Letter Decoration**

- **out**
  - $H \leftarrow \delta_f^{i+1}$
  - $H \leftarrow \delta_f^{i+1}$
  - $F \leftarrow |H - \delta_f^i|$ | $H \leftarrow \delta_f^{i+1}$ | $R \leftarrow R \land \theta(F, C)$ |

- **out_a**
  - $F \leftarrow |H - \delta_f^i|$ | $H \leftarrow \delta_f^{i+1}$ | $R \leftarrow R \land \theta(F, C)$ |

- **maybe_a**
  - $F \leftarrow |H - \delta_f^i|$ | $H \leftarrow \delta_f^{i+1}$ | $R \leftarrow R \land \theta(F, C)$ |

- **found_a**
  - $C \leftarrow |H - \delta_f^{i+1}|$ | $F \leftarrow |H - \delta_f^{i+1}|$ | $H \leftarrow \delta_f^{i+1}$ | $R \leftarrow R \land \theta(F, |H - \delta_f^{i+1}|)$ |

- **found**
  - $C \leftarrow |H - \delta_f^{i+1}|$ | $F \leftarrow |H - \delta_f^{i+1}|$ | $H \leftarrow \delta_f^{i+1}$ | $R \leftarrow R \land \theta(F, |H - \delta_f^{i+1}|)$ |

- **in**
  - $C \leftarrow |H - \delta_f^{i+1}|$ | $F \leftarrow |H - \delta_f^{i+1}|$ | $H \leftarrow \delta_f^{i+1}$ | $R \leftarrow R \land \theta(F, |H - \delta_f^{i+1}|)$ |
3.3 Tables of Regular-Expression Characteristics

This section provides the different tables of regular-expression characteristics defined in [6, 4] used to come up with concrete bounds and concrete AMONG implied constraints. The notation system used for these characteristics was described in [6]. Within these tables Bump, Dec, DecSeq, DecTer, Dip, Inc, IncSeq, IncTer, PropPlain, PropPlateau, SteadySeq, SDecSeq, SIncSeq are respectively shortcuts for BumpOnDecreasingSequence, Decreasing, DecreasingSequence, DecreasingTerrace, DipOnIncreasingSequence, Increasing, IncreasingSequence, IncreasingTerrace, ProperPlain, ProperPlateau, SteadySequence, StrictlyDecreasingSequence, StrictlyIncreasingSequence.
3. PATTERNS AND CORRESPONDING TABLES

3.3.1 Big-width

**Definition 5.** Consider a regular expression $\sigma$, and a time series $X$ of length $n$ over an integer interval domain $[\ell, u]$. The big width of $\sigma$ wrt $\langle \ell, u, n \rangle$, denoted by $\beta^{\langle \ell, u, n \rangle}_\sigma$, is a function that maps an element of $R_\Sigma \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}^+$ to $\mathbb{N}$. It equals the maximum width of a $\sigma$-pattern in $X$. If $X$ cannot have any $\sigma$-patterns, then $\beta^{\langle \ell, u, n \rangle}_\sigma$ is $0$.

<table>
<thead>
<tr>
<th>name $\sigma$</th>
<th>illustration</th>
<th>$\beta^{\langle \ell, u \rangle}_\sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bump</td>
<td><img src="image" alt="Bump Illustration" /></td>
<td>$\begin{cases} 3, &amp; \text{if } u - \ell \geq \eta_\sigma \ 0, &amp; \text{otherwise} \end{cases}$</td>
</tr>
<tr>
<td>Dec</td>
<td><img src="image" alt="Dec Illustration" /></td>
<td>$\begin{cases} 2, &amp; \text{if } u - \ell \geq \eta_\sigma \ 0, &amp; \text{otherwise} \end{cases}$</td>
</tr>
<tr>
<td>DecSeq</td>
<td><img src="image" alt="DecSeq Illustration" /></td>
<td>$\begin{cases} 2, &amp; \text{if } u - \ell = \eta_\sigma \ n, &amp; \text{if } u - \ell &gt; \eta_\sigma \ 0, &amp; \text{otherwise} \end{cases}$</td>
</tr>
<tr>
<td>DecTer</td>
<td><img src="image" alt="DecTer Illustration" /></td>
<td>$\begin{cases} n - 2, &amp; \text{if } u - \ell \geq \eta_\sigma \ 0, &amp; \text{otherwise} \end{cases}$</td>
</tr>
<tr>
<td>Dip</td>
<td><img src="image" alt="Dip Illustration" /></td>
<td>$\begin{cases} 3, &amp; \text{if } u - \ell \geq \eta_\sigma \ 0, &amp; \text{otherwise} \end{cases}$</td>
</tr>
<tr>
<td>Gorge</td>
<td><img src="image" alt="Gorge Illustration" /></td>
<td>$\begin{cases} 1, &amp; \text{if } u - \ell = \eta_\sigma \ n - 2, &amp; \text{if } u - \ell &gt; \eta_\sigma \ 0, &amp; \text{otherwise} \end{cases}$</td>
</tr>
<tr>
<td>Inc</td>
<td><img src="image" alt="Inc Illustration" /></td>
<td>$\begin{cases} 2, &amp; \text{if } u - \ell \geq \eta_\sigma \ 0, &amp; \text{otherwise} \end{cases}$</td>
</tr>
<tr>
<td>IncSeq</td>
<td><img src="image" alt="IncSeq Illustration" /></td>
<td>$\begin{cases} 2, &amp; \text{if } u - \ell = \eta_\sigma \ n, &amp; \text{if } u - \ell &gt; \eta_\sigma \ 0, &amp; \text{otherwise} \end{cases}$</td>
</tr>
<tr>
<td>IncTer</td>
<td><img src="image" alt="IncTer Illustration" /></td>
<td>$\begin{cases} n - 2, &amp; \text{if } u - \ell \geq \eta_\sigma \ 0, &amp; \text{otherwise} \end{cases}$</td>
</tr>
<tr>
<td>Inflexion</td>
<td><img src="image" alt="Inflexion Illustration" /></td>
<td>$\begin{cases} n - 2, &amp; \text{if } u - \ell \geq \eta_\sigma \ 0, &amp; \text{otherwise} \end{cases}$</td>
</tr>
<tr>
<td>Peak</td>
<td><img src="image" alt="Peak Illustration" /></td>
<td>$\begin{cases} n - 2, &amp; \text{if } u - \ell \geq \eta_\sigma \ 0, &amp; \text{otherwise} \end{cases}$</td>
</tr>
</tbody>
</table>
### Table 3.50: Regular-expression short names $\sigma$ and corresponding big-width (see Definition 5) shown as thick orange horizontal line segments, where $\eta_\sigma$ stands for the height characteristics of the corresponding $\sigma$ (see Definition 8)

<table>
<thead>
<tr>
<th>Short Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain</td>
<td>$n - 2$, if $u - \ell \geq \eta_\sigma$; 0, otherwise</td>
</tr>
<tr>
<td>Plateau</td>
<td>$n - 2$, if $u - \ell \geq \eta_\sigma$; 0, otherwise</td>
</tr>
<tr>
<td>PropPlain</td>
<td>$n - 2$, if $u - \ell \geq \eta_\sigma$; 0, otherwise</td>
</tr>
<tr>
<td>PropPlateau</td>
<td>$n - 2$, if $u - \ell \geq \eta_\sigma$; 0, otherwise</td>
</tr>
<tr>
<td>Steady</td>
<td>2</td>
</tr>
<tr>
<td>SteadySeq</td>
<td>$n$</td>
</tr>
<tr>
<td>SDecSeq</td>
<td>$u - \ell + 1$, if $u - \ell \geq \eta_\sigma$; 0, otherwise</td>
</tr>
<tr>
<td>SIncSeq</td>
<td>$u - \ell + 1$, if $u - \ell \geq \eta_\sigma$; 0, otherwise</td>
</tr>
<tr>
<td>Summit</td>
<td>1, if $u - \ell = \eta_\sigma$; $n - 2$, if $u - \ell &gt; \eta_\sigma$; 0, otherwise</td>
</tr>
<tr>
<td>Valley</td>
<td>$n - 2$, if $u - \ell \geq \eta_\sigma$; 0, otherwise</td>
</tr>
<tr>
<td>Zigzag</td>
<td>$n - 2$, if $u - \ell \geq \eta_\sigma$; 0, otherwise</td>
</tr>
</tbody>
</table>
3.3.2 Height

Definition 6. Consider a regular expression $\sigma$ and an integer interval domain $[\ell, u]$. The set of supporting time series of a word $v$ in $L_\sigma$ wrt $[\ell, u]$, denoted by $\Omega^{(\ell,u)}(v)$, is a function that maps an element of $R_{\Sigma} \times \mathbb{Z} \times \mathbb{Z} \times \Sigma^*$ to $\mathbb{Z}^*$. Each element of $\Omega^{(\ell,u)}(v)$ is a time series over $[\ell, u]$ whose signature is $v$, and is called a supporting time series of $v$ wrt $[\ell, u]$.

Definition 7. Consider a regular expression $\sigma$ and an integer interval domain $[\ell, u]$. The height of a word $v$ in $L_\sigma$, denoted by $\eta_\sigma(v)$, is a function that maps an element of $R_{\Sigma} \times \Sigma^*$ to $\mathbb{N}$. It is defined by $\eta_\sigma(v) = \min_{\Omega^{(\ell,u)}(v) \neq \emptyset} (u - \ell)$.

Definition 8. Consider a regular expression $\sigma$ and an integer interval domain $[\ell, u]$. The height of $\sigma$, denoted by $\eta_\sigma$, is a function that maps an element of $R_{\Sigma}$ to $\mathbb{N}$. It is defined by $\eta_\sigma = \min_{v \in L_\sigma} \eta_\sigma(v)$. 
<table>
<thead>
<tr>
<th>name $\sigma$</th>
<th>illustration</th>
<th>$\eta_{\sigma}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bump</td>
<td><img src="image1" alt="Bump Illustration" /></td>
<td>2</td>
</tr>
<tr>
<td>Dec</td>
<td><img src="image2" alt="Dec Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>DecSeq</td>
<td><img src="image3" alt="DecSeq Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>DecTer</td>
<td><img src="image4" alt="DecTer Illustration" /></td>
<td>2</td>
</tr>
<tr>
<td>Dip</td>
<td><img src="image5" alt="Dip Illustration" /></td>
<td>2</td>
</tr>
<tr>
<td>Gorge</td>
<td><img src="image6" alt="Gorge Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>Inc</td>
<td><img src="image7" alt="Inc Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>IncSeq</td>
<td><img src="image8" alt="IncSeq Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>IncTer</td>
<td><img src="image9" alt="IncTer Illustration" /></td>
<td>2</td>
</tr>
<tr>
<td>Inflexion</td>
<td><img src="image10" alt="Inflexion Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>Peak</td>
<td><img src="image11" alt="Peak Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>Plain</td>
<td><img src="image12" alt="Plain Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>Plateau</td>
<td><img src="image13" alt="Plateau Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>PropPlain</td>
<td><img src="image14" alt="PropPlain Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>PropPlateau</td>
<td><img src="image15" alt="PropPlateau Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>Steady</td>
<td><img src="image16" alt="Steady Illustration" /></td>
<td>0</td>
</tr>
<tr>
<td>SteadySeq</td>
<td><img src="image17" alt="SteadySeq Illustration" /></td>
<td>0</td>
</tr>
<tr>
<td>SDecSeq</td>
<td><img src="image18" alt="SDecSeq Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>SIncSeq</td>
<td><img src="image19" alt="SIncSeq Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>Summit</td>
<td><img src="image20" alt="Summit Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>Valley</td>
<td><img src="image21" alt="Valley Illustration" /></td>
<td>1</td>
</tr>
<tr>
<td>Zigzag</td>
<td><img src="image22" alt="Zigzag Illustration" /></td>
<td>1</td>
</tr>
</tbody>
</table>

Table 3.51: Regular-expression short names $\sigma$ and corresponding height shown as thick orange vertical line segments (see Definition 8)
3.3.3 Interval of interest

**Definition 9.** Given a $g \_f \_\sigma(X, N)$ time-series constraint with $X$ being a time series over an integer interval domain $[\ell, u]$. The interval of interest of $(g, f, \sigma)$ wrt $(\ell, u)$, denoted by $I_{(g,f,\sigma)}^{(\ell,u)}$, is a function that maps an element of $T \times \mathbb{Z} \times \mathbb{Z}$ to $\mathbb{Z} \times \mathbb{Z}$, where $T$ denotes the set of all time-series constraints, and the result pair of integers is considered as an interval. If $u - \ell < \eta_\sigma$, then the interval of interest of $(g, f, \sigma)$ wrt $(\ell, u)$ is undefined. If $u - \ell \geq \eta_\sigma$, then

- The upper limit of $I_{(g,f,\sigma)}^{(\ell,u)}$, denoted by $I^u_{(g,f,\sigma)}$, is the largest value in $[\ell, u]$ that can occur in a $\sigma$-pattern of a time series over $[\ell, u]$.

- The lower limit of $I_{(g,f,\sigma)}^{(\ell,u)}$, denoted by $I^l_{(g,f,\sigma)}$, is the smallest value $v$ in $[\max(\ell, u - \eta_\sigma - 1), u]$ such that for any $n \in \mathbb{N}$, the number of occurrences of $v$ in the union of the $\sigma$-patterns of any maximal time series for $g \_f \_\sigma$ of length $n$ over $[\ell, u]$, is a non-constant function of $n$. If such $v$ does not exist, then $I^u_{(g,f,\sigma)}$ equals $I^l_{(g,f,\sigma)} - \eta_\sigma$.

<table>
<thead>
<tr>
<th>Name</th>
<th>Illustration</th>
<th>$I^u_{(g,f,\sigma)}$</th>
<th>$I^l_{(g,f,\sigma)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bump</td>
<td><img src="image" alt="Bump Illustration" /></td>
<td>$[u - 2, u]$, if $u - \ell \geq \eta_\sigma$</td>
<td>undefined, otherwise</td>
</tr>
<tr>
<td>Dec</td>
<td><img src="image" alt="Dec Illustration" /></td>
<td>$[u - 1, u]$, if $u - \ell \geq \eta_\sigma$</td>
<td>undefined, otherwise</td>
</tr>
<tr>
<td>DecSeq</td>
<td><img src="image" alt="DecSeq Illustration" /></td>
<td>$[u - 1, u - 1]$, if $u - \ell \geq \eta_\sigma$</td>
<td>undefined, otherwise</td>
</tr>
<tr>
<td>DecTer</td>
<td><img src="image" alt="DecTer Illustration" /></td>
<td>$[u - 1, u - 1]$, if $u - \ell \geq \eta_\sigma$</td>
<td>undefined, otherwise</td>
</tr>
<tr>
<td>Dip</td>
<td><img src="image" alt="Dip Illustration" /></td>
<td>$[u - 2, u]$, if $u - \ell \geq \eta_\sigma$</td>
<td>undefined, otherwise</td>
</tr>
<tr>
<td>Gorge</td>
<td><img src="image" alt="Gorge Illustration" /></td>
<td>$[u - 1, u - 1]$, if $u - \ell \geq \eta_\sigma$</td>
<td>undefined, otherwise</td>
</tr>
<tr>
<td>Inc</td>
<td><img src="image" alt="Inc Illustration" /></td>
<td>$[u - 1, u]$, if $u - \ell \geq \eta_\sigma$</td>
<td>undefined, otherwise</td>
</tr>
<tr>
<td>IncSeq</td>
<td><img src="image" alt="IncSeq Illustration" /></td>
<td>$[u - 1, u - 1]$, if $u - \ell \geq \eta_\sigma$</td>
<td>undefined, otherwise</td>
</tr>
<tr>
<td>IncTer</td>
<td><img src="image" alt="IncTer Illustration" /></td>
<td>$[u - 1, u - 1]$, if $u - \ell \geq \eta_\sigma$</td>
<td>undefined, otherwise</td>
</tr>
</tbody>
</table>
3.3. TABLES OF REGULAR-EXPRESSION CHARACTERISTICS

Table 3.52: Regular-expression short names \( \sigma \) and corresponding interval of interests (see Definition 9) of \( \langle g, f, \sigma \rangle \) wrt \( \langle \ell, u \rangle \), where \( \eta_\sigma \) stands for the height characteristics of the corresponding \( \sigma \) (see Definition 8); \( \ell \) and \( u \) resp. stands for the minimum and maximum value of the variables of the time series.

- **Inflexion**: \( \eta_\sigma \left\{ \begin{array}{ll} [u, u], & \text{if } u - \ell \geq \eta_\sigma \\ \text{undefined, otherwise} \end{array} \right. 
- **Peak**: \( \eta_\sigma \left\{ \begin{array}{ll} [u, u], & \text{if } u - \ell \geq \eta_\sigma \\ \text{undefined, otherwise} \end{array} \right. 
- **Plain**: \( \eta_\sigma \left\{ \begin{array}{ll} [u - 1, u - 1], & \text{if } u - \ell \geq \eta_\sigma \\ \text{undefined, otherwise} \end{array} \right. 
- **Plateau**: \( \eta_\sigma \left\{ \begin{array}{ll} [u, u], & \text{if } u - \ell \geq \eta_\sigma \\ \text{undefined, otherwise} \end{array} \right. 
- **PropPlain**: \( \eta_\sigma \left\{ \begin{array}{ll} [u - 1, u - 1], & \text{if } u - \ell \geq \eta_\sigma \\ \text{undefined, otherwise} \end{array} \right. 
- **PropPlateau**: \( \eta_\sigma \left\{ \begin{array}{ll} [u, u], & \text{if } u - \ell \geq \eta_\sigma \\ \text{undefined, otherwise} \end{array} \right. 
- **Steady**: \( \eta_\sigma \left\{ \begin{array}{ll} [u, u], & \text{if } u - \ell \geq \eta_\sigma \\ \text{undefined, otherwise} \end{array} \right. 
- **SteadySeq**: \( \eta_\sigma \left\{ \begin{array}{ll} [u - 1, u], & \text{if } u - \ell \geq \eta_\sigma \land g = \text{Sum} \\ [u, u], & \text{if } u - \ell \geq \eta_\sigma \land (g = \text{Max} \lor g = \text{Min}) \\ \text{undefined, otherwise} \end{array} \right. 
- **SDecSeq**: \( \eta_\sigma \left\{ \begin{array}{ll} [u - 1, u], & \text{if } u - \ell \geq \eta_\sigma \\ \text{undefined, otherwise} \end{array} \right. 
- **SIncSeq**: \( \eta_\sigma \left\{ \begin{array}{ll} [u - 1, u], & \text{if } u - \ell \geq \eta_\sigma \\ \text{undefined, otherwise} \end{array} \right. 
- **Summit**: \( \eta_\sigma \left\{ \begin{array}{ll} [u - 1, u], & \text{if } u - \ell \geq \eta_\sigma \\ \text{undefined, otherwise} \end{array} \right. 
- **Valley**: \( \eta_\sigma \left\{ \begin{array}{ll} [u - 1, u - 1], & \text{if } u - \ell \geq \eta_\sigma \\ \text{undefined, otherwise} \end{array} \right. 
- **Zigzag**: \( \eta_\sigma \left\{ \begin{array}{ll} [u - 1, u], & \text{if } u - \ell \geq \eta_\sigma \\ \text{undefined, otherwise} \end{array} \right. 

3. PATTERNS AND CORRESPONDING TABLES

3.3.4 Maximum-value-occurrence-number

Definition 10. Consider a regular expression $\sigma$, and a time series $X$ of length $n$ over an integer interval domain $[\ell, u]$. The maximum value occurrence number of $v$ in $Z$ wrt $\sigma \langle \ell, u, n \rangle$, denoted by $\mu_{\sigma}^{\langle \ell, u, n \rangle}(v)$, is a function that maps an element of $\mathbb{R}_{\Sigma} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{N}^+ \times \mathbb{Z}$ to $\mathbb{N}$. It equals the maximum number of occurrences of the value $v$ in one $\sigma$-pattern of $X$.

<table>
<thead>
<tr>
<th>name $\sigma$</th>
<th>illustration</th>
<th>$\mu_{\sigma}^{\langle \ell, u, n \rangle}(v)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bump</td>
<td>![Bump Illustration]</td>
<td>$\begin{cases} 1, &amp; \text{if } u - \ell = \eta_{\sigma} \ 1, &amp; \text{if } u - \ell &gt; \eta_{\sigma} \land v \in {\ell, u - 1, u} \ 2, &amp; \text{if } u - \ell &gt; \eta_{\sigma} \land v \in [\ell + 1, u - 2] \end{cases}$</td>
</tr>
<tr>
<td>Dec</td>
<td>![Dec Illustration]</td>
<td>$1, \forall v \in [\ell, u]$</td>
</tr>
<tr>
<td>DecSeq</td>
<td>![DecSeq Illustration]</td>
<td>$\begin{cases} 1, &amp; \text{if } v \in {\ell, u} \ n - 2, &amp; \text{if } v \in [\ell + 1, u - 1] \end{cases}$</td>
</tr>
<tr>
<td>DecTer</td>
<td>![DecTer Illustration]</td>
<td>$\begin{cases} 0, &amp; \text{if } v \in {\ell, u} \ n - 2, &amp; \text{if } v \in [\ell + 1, u - 1] \end{cases}$</td>
</tr>
<tr>
<td>Dip</td>
<td>![Dip Illustration]</td>
<td>$\begin{cases} 1, &amp; \text{if } u - \ell = \eta_{\sigma} \ 1, &amp; \text{if } u - \ell &gt; \eta_{\sigma} \land v \in {\ell, \ell + 1, u} \ 2, &amp; \text{if } u - \ell &gt; \eta_{\sigma} \land v \in [\ell + 2, u - 1] \end{cases}$</td>
</tr>
<tr>
<td>Gorge</td>
<td>![Gorge Illustration]</td>
<td>$\begin{cases} 0, &amp; \text{if } v = u \ n - 3, &amp; \text{if } v \in [\ell + 1, u - 1] \ 1, &amp; \text{if } v = \ell \end{cases}$</td>
</tr>
<tr>
<td>Inc</td>
<td>![Inc Illustration]</td>
<td>$1, \forall v \in [\ell, u]$</td>
</tr>
<tr>
<td>IncSeq</td>
<td>![IncSeq Illustration]</td>
<td>$\begin{cases} 1, &amp; \text{if } v \in {\ell, u} \ n - 2, &amp; \text{if } v \in [\ell + 1, u - 1] \end{cases}$</td>
</tr>
<tr>
<td>IncTer</td>
<td>![IncTer Illustration]</td>
<td>$\begin{cases} 0, &amp; \text{if } v \in {\ell, u} \ n - 2, &amp; \text{if } v \in [\ell + 1, u - 1] \end{cases}$</td>
</tr>
<tr>
<td>Inflexion</td>
<td>![Inflexion Illustration]</td>
<td>$n - 2, \forall v \in [\ell, u]$</td>
</tr>
<tr>
<td>Peak</td>
<td>![Peak Illustration]</td>
<td>$\begin{cases} 0, &amp; \text{if } v = \ell \ n - 2, &amp; \text{if } v \in [\ell + 1, u] \end{cases}$</td>
</tr>
<tr>
<td>Plain</td>
<td>![Plain Illustration]</td>
<td>$\begin{cases} 0, &amp; \text{if } v = u \ n - 2, &amp; \text{if } v \in [\ell, u - 1] \end{cases}$</td>
</tr>
</tbody>
</table>
### 3.3. TABLES OF REGULAR-EXPRESSION CHARACTERISTICS

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Description</th>
<th>Formula</th>
</tr>
</thead>
</table>
| **Plateau** | $\ell_n - 2$ \{ \begin{align*} 0, & \text{ if } v = \ell \\ n - 2, & \text{ if } v \in [\ell + 1, u] \end{align*} \} | |}
| **PropPlain** | $\ell_n - 2$ \{ \begin{align*} 0, & \text{ if } v = u \\ n - 2, & \text{ if } v \in [\ell, u - 1] \end{align*} \} | |}
| **PropPlateau** | $\ell_n - 2$ \{ \begin{align*} 0, & \text{ if } v = \ell \\ n - 2, & \forall v \in [\ell + 1, u] \end{align*} \} | |}
| **Steady** | $\ell_n - 2$ \{ \begin{align*} 0, & \text{ if } v = \ell \\ n - 2, & \forall v \in [\ell, u] \end{align*} \} | |}
| **SteadySeq** | $\ell_n - 2$ \{ \begin{align*} 0, & \text{ if } v = \ell \\ n, & \forall v \in [\ell, u] \end{align*} \} | |}
| **SDecSeq** | $\ell_n - 2$ \{ \begin{align*} 0, & \text{ if } v = \ell \\ n - 2, & \forall v \in [\ell, u] \end{align*} \} | |}
| **SIncSeq** | $\ell_n - 2$ \{ \begin{align*} 0, & \text{ if } v = \ell \\ n - 2, & \forall v \in [\ell, u] \end{align*} \} | |}

Table 3.53: Regular-expression short names $\sigma$ and corresponding **maximum value occurrence number** of a value $v$ (see Definition 10); for each pattern $\sigma$ it assumes that the range of possible values is big enough to have at least one occurrence of pattern, i.e. $u - \ell \geq \eta_\sigma$ where $u$ and $\ell$ are the largest and smallest value that can be used.
3. PATTERNS AND CORRESPONDING TABLES

3.3.5 Overlap

Definition 11. Consider a regular expression $\sigma$ and an integer interval domain $[\ell, u]$. The set of superpositions of two words, $v$ and $w$ in $L_\sigma$, wrt $[\ell, u]$, denoted by $\Gamma^{(\ell, u)}_\sigma(v, w)$, is a function that maps an element of $R_{\Sigma} \times \mathbb{Z} \times \mathbb{Z} \times \Sigma^\ast \times \Sigma^\ast$ to $\mathcal{P}(\Sigma^\ast)$. Each element $z$ in $\Gamma^{(\ell, u)}_\sigma(v, w)$ is a word over $\Sigma$, called a superposition of $v$ and $w$ wrt $[\ell, u]$ and satisfying all the following conditions

(i) $z \notin L_\sigma$, (ii) $\Omega^{(\ell, u)}_\sigma(z) \neq \emptyset$, (iii) $v$ is a prefix of $z$, (iv) $w$ is a suffix of $z$, (v) $|z| \leq |vw|$.

Definition 12. Consider a regular expression $\sigma$ and an integer interval domain $[\ell, u]$. The overlap of two words, $v$ and $w$ in $L_\sigma$, wrt $[\ell, u]$, denoted by $o^{(\ell, u)}_\sigma(v, w)$, is a function that maps an element of $R_{\Sigma} \times \mathbb{Z} \times \mathbb{Z} \times \Sigma^\ast \times \Sigma^\ast$ to $\mathbb{N}$. It is defined by

$$ o^{(\ell, u)}_\sigma(v, w) = \begin{cases} |vw| - \min_{z \in \Gamma^{(\ell, u)}_\sigma(v, w)} |z| + 1 & \text{if } \Gamma^{(\ell, u)}_\sigma(v, w) \neq \emptyset \\ 0, & \text{otherwise.} \end{cases} $$

Definition 13. Consider a regular expression $\sigma$ and an integer interval domain $[\ell, u]$. The overlap of $\sigma$ wrt $[\ell, u]$, denoted by $o^{(\ell, u)}_\sigma$, is a function that maps an element of $R_{\Sigma} \times \mathbb{Z} \times \mathbb{Z}$ to $\mathbb{N}$. If there exists a constant $c$ in $\mathbb{N}$ such that for any pair words $v, w$ in $L_\sigma$, the value of $o^{(\ell, u)}_\sigma(v, w)$ is bounded by $c$, then the overlap of $\sigma$ wrt $[\ell, u]$ is defined by $o^{(\ell, u)}_\sigma = \max_{v, w \in L_\sigma} o^{(\ell, u)}_\sigma(v, w)$. Otherwise, $o^{(\ell, u)}_\sigma$ is not defined.
3.3. TABLES OF REGULAR-EXPRESSION CHARACTERISTICS

<table>
<thead>
<tr>
<th>Inflexion</th>
<th>( u - \ell )</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peak</td>
<td>( u - \ell )</td>
<td>1</td>
</tr>
<tr>
<td>Plain</td>
<td>( u - \ell )</td>
<td>1</td>
</tr>
<tr>
<td>Plateau</td>
<td>( u - \ell )</td>
<td>1</td>
</tr>
<tr>
<td>PropPlain</td>
<td>( u - \ell )</td>
<td>1</td>
</tr>
<tr>
<td>PropPlateau</td>
<td>( u - \ell )</td>
<td>1</td>
</tr>
<tr>
<td>Steady</td>
<td>( u - \ell )</td>
<td>1</td>
</tr>
<tr>
<td>SteadySeq</td>
<td>( u - \ell )</td>
<td>0</td>
</tr>
<tr>
<td>SDecSeq</td>
<td>( u - \ell )</td>
<td>0</td>
</tr>
<tr>
<td>SIncSeq</td>
<td>( u - \ell )</td>
<td>0</td>
</tr>
<tr>
<td>Summit</td>
<td>( u - \ell )</td>
<td>1</td>
</tr>
<tr>
<td>Valley</td>
<td>( u - \ell )</td>
<td>1</td>
</tr>
<tr>
<td>Zigzag</td>
<td>( u - \ell )</td>
<td>( \begin{cases} 0 &amp; \text{if } u - \ell \leq \eta_\sigma \ 1 &amp; \text{otherwise} \end{cases} )</td>
</tr>
</tbody>
</table>

Table 3.54: Regular-expression short names \( \sigma \) and corresponding overlap between two consecutive pattern occurrences \( \circ \) and \( \bullet \) illustrated in red, i.e., \( \circ \) or \( \bullet \) (see Definition 13), where \( \eta_\sigma \) stands for the height characteristics of the corresponding \( \sigma \) (see Definition 8).
3. PATTERNS AND CORRESPONDING TABLES

3.3.6 Range

Definition 14. Consider a regular expression $\sigma$ and a time series length $n$. The range of $\sigma$ wrt $\langle n \rangle$, denoted by $\phi^{(n)}_\sigma$, is a function that maps an element of $\mathcal{R}_\Sigma \times \mathbb{N}^+$ to $\mathbb{N}$. It is defined by $\phi^{(n)}_\sigma = \min_{v \in \mathcal{L}_\sigma, |v| = n-1} \eta_\sigma(v)$, where $\eta_\sigma(v)$ is the height of the word $v$.

If $\mathcal{L}_\sigma$ does not contain any word of length $n - 1$, then the value of $\phi^{(n)}_\sigma$ is undefined.

<table>
<thead>
<tr>
<th>name $\sigma$</th>
<th>$\langle e_\sigma, c_\sigma \rangle$</th>
<th>illustration</th>
<th>$\phi^{(n)}_\sigma$</th>
</tr>
</thead>
</table>
| Bump          | undefined       | [diagram]     | $\begin{cases} 2 & \text{if } n = \omega_\sigma + 1 \\
                   & \text{undefined otherwise} \end{cases}$ |
| Dec           | undefined       | [diagram]     | $\begin{cases} 1 & \text{if } n = \omega_\sigma + 1 \\
                   & \text{undefined otherwise} \end{cases}$ |
| DecSeq        | $\langle 0, 1 \rangle$ | [diagram]     | $\begin{cases} 1 & \text{if } n = \omega_\sigma + 1 \\
                   & 2 & \text{if } n > \omega_\sigma + 1 \end{cases}$ |
| DecTer        | $\langle 0, 0 \rangle$ | [diagram]     | $2$ |
| Dip           | undefined       | [diagram]     | $\begin{cases} 2 & \text{if } n = \omega_\sigma + 1 \\
                   & \text{undefined otherwise} \end{cases}$ |
| Gorge         | $\langle 0, 1 \rangle$ | [diagram]     | $\begin{cases} 1 & \text{if } n = \omega_\sigma + 1 \\
                   & 2 & \text{if } n > \omega_\sigma + 1 \end{cases}$ |
| Inc           | undefined       | [diagram]     | $\begin{cases} 1 & \text{if } n = \omega_\sigma + 1 \\
                   & \text{undefined otherwise} \end{cases}$ |
| IncSeq        | $\langle 0, 1 \rangle$ | [diagram]     | $\begin{cases} 1 & \text{if } n = \omega_\sigma + 1 \\
                   & 2 & \text{if } n > \omega_\sigma + 1 \end{cases}$ |
| IncTer        | $\langle 0, 0 \rangle$ | [diagram]     | $2$ |
| Inflexion     | $\langle 0, 0 \rangle$ | [diagram]     | $1$ |
| Peak          | $\langle 0, 0 \rangle$ | [diagram]     | $1$ |
| Plain         | $\langle 0, 0 \rangle$ | [diagram]     | $1$ |
| Plateau       | $\langle 0, 0 \rangle$ | [diagram]     | $1$ |
| PropPlain     | $\langle 0, 0 \rangle$ | [diagram]     | $1$ |
| PropPlateau   | $\langle 0, 0 \rangle$ | [diagram]     | $1$ |
| Steady        | undefined       | [diagram]     | $\begin{cases} 0 & \text{if } n = \omega_\sigma + 1 \\
                   & \text{undefined otherwise} \end{cases}$ |
| SteadySeq     | $\langle 0, 0 \rangle$ | [diagram]     | $0$ |
| SDecSeq       | $\langle 1, 0 \rangle$ | [diagram]     | $n - 1$ |
### Table 3.55: Regular-expression short names $\sigma$ and corresponding range shown as thick orange vertical line segments (see Definition 14); for a non-fixed-length regular expression $\sigma$ and for any $n > \omega_\sigma + 1$, $\phi_\sigma^{(n)} = e_\sigma \cdot (n - 1 - \eta_\sigma) + c_\sigma + \eta_\sigma$, where $\omega_\sigma$ and $\eta_\sigma$ respectively correspond to the size (see Definition 18) and the height (see Definition 8) of the corresponding $\sigma$. 

<table>
<thead>
<tr>
<th>Name</th>
<th>Range $\langle r, w \rangle$</th>
<th>$n - 1$</th>
<th>Condition 1</th>
<th>Condition 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIncSeq</td>
<td>$\langle 1, 0 \rangle$</td>
<td>$n - 1$</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Summit</td>
<td>$\langle 0, 1 \rangle$</td>
<td></td>
<td>if $n = \omega_\sigma + 1$</td>
<td>if $n &gt; \omega_\sigma + 1$</td>
</tr>
<tr>
<td>Valley</td>
<td>$\langle 0, 0 \rangle$</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Zigzag</td>
<td>$\langle 0, 0 \rangle$</td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
3. PATTERNS AND CORRESPONDING TABLES

3.3.7 Set-of-inducing-words

**Definition 15.** Consider a disjunction-capsuled regular expression \( \sigma \). The (unique) non-empty shortest word of \( L_\sigma \) is the inducing word of \( L_\sigma \).

**Definition 16.** Consider a regular expression \( \sigma \) that is in the form of \( \sigma = \sigma_1 \mid \ldots \mid \sigma_t \) with \( t \geq 1 \), where every \( \sigma_i \) (with \( i \in [1, t] \)) is a disjunction-capsuled regular expression. The set of inducing words of \( \sigma \), denoted by \( \Theta_\sigma \), is a function that maps an element of \( R_\Sigma \) to \( P(\Sigma^*) \), where \( P(\Sigma^*) \) is the power set of \( \Sigma^* \). Each element \( v \) of \( \Theta_\sigma \) is a word, called an inducing word of \( \sigma \) such that all the following conditions are satisfied:

(i) \( v \) is in \( \{w_1, \ldots, w_t\} \), where \( w_i \) is the inducing word of \( \sigma_i \).

(ii) No word in \( \{w_1, w_2, \ldots, w_t\} \) is a proper factor of \( v \).

<table>
<thead>
<tr>
<th>name ( \sigma )</th>
<th>regular expression</th>
<th>( \Theta_\sigma )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bump</td>
<td>‘&gt;&gt;&gt;&lt;&lt;’</td>
<td>{‘&gt;&gt;&gt;&lt;&lt;’}</td>
</tr>
<tr>
<td>Dec</td>
<td>‘&gt;’</td>
<td>{‘&gt;’}</td>
</tr>
<tr>
<td>DecSeq</td>
<td>‘(&gt; (&gt;</td>
<td>=)*)’ &gt;’</td>
</tr>
<tr>
<td>DecTer</td>
<td>‘&gt;= ‘= ‘&gt;’</td>
<td>{‘&gt;= ‘= ‘&gt;’}</td>
</tr>
<tr>
<td>Dip</td>
<td>‘&lt;&lt;&gt;&lt;&lt;&lt;’</td>
<td>{‘&lt;&lt;&gt;&lt;&lt;&lt;’}</td>
</tr>
<tr>
<td>Gorge</td>
<td>‘(&gt; (&gt;</td>
<td></td>
</tr>
<tr>
<td>Inc</td>
<td>‘&lt;’</td>
<td>{‘&lt;’}</td>
</tr>
<tr>
<td>IncSeq</td>
<td>‘&lt;( &lt;</td>
<td>=)*)’ &lt;’</td>
</tr>
<tr>
<td>IncTer</td>
<td>‘=&lt; =’&lt;’</td>
<td>{‘=&lt; =’&lt;’}</td>
</tr>
<tr>
<td>Inflexion</td>
<td>‘&lt; ( &lt;</td>
<td>=)’ &gt;</td>
</tr>
<tr>
<td>Peak</td>
<td>‘&lt; (</td>
<td></td>
</tr>
<tr>
<td>Plain</td>
<td>‘&gt; =’&lt;’</td>
<td>{‘&gt; =’&lt;’}</td>
</tr>
<tr>
<td>Plateau</td>
<td>‘&lt; =’&gt;’</td>
<td>{‘&lt; =’&gt;’}</td>
</tr>
<tr>
<td>PropPlain</td>
<td>‘&gt;= =’&lt;’</td>
<td>{‘&gt;= =’&lt;’}</td>
</tr>
<tr>
<td>PropPlateau</td>
<td>‘&lt; =’&gt;’</td>
<td>{‘&lt; =’&gt;’}</td>
</tr>
<tr>
<td>Steady</td>
<td>‘=’</td>
<td>{‘=’}</td>
</tr>
<tr>
<td>SteadySeq</td>
<td>‘= =’</td>
<td>{‘=’}</td>
</tr>
<tr>
<td>SDecSeq</td>
<td>‘&gt; ‘&gt;’</td>
<td>{‘&gt; ‘&gt;’}</td>
</tr>
<tr>
<td>SIncSeq</td>
<td>‘&lt; ‘&lt;’</td>
<td>{‘&lt; ‘&lt;’}</td>
</tr>
<tr>
<td>Summit</td>
<td>‘&lt;( &lt;</td>
<td>=)’*’ &lt; (&gt;</td>
</tr>
<tr>
<td>Valley</td>
<td>‘&gt; (&gt;</td>
<td>=)’*&lt;( &lt;</td>
</tr>
<tr>
<td>Zigzag</td>
<td>‘&lt;(&gt;)* &lt;&lt;(&gt;</td>
<td>=)’ (&gt;</td>
</tr>
</tbody>
</table>

Table 3.56: Regular-expression short names \( \sigma \) and corresponding inducing words (see Definition 16)
3.3. TABLES OF REGULAR-EXPRESSION CHARACTERISTICS

3.3.8 Shift

Definition 17. Consider a regular expression $\sigma$ and an integer interval domain $[\ell,u]$. The shift of $\sigma$, denoted by $\delta_\sigma$, is a function that maps an element of $\mathcal{R}_\Sigma$ to $\mathbb{N}$. It is defined by 

$$\delta_\sigma = \min_{v \in L} \min_{t \in \Omega_{(0,|v|)}(\sigma)} \max_{i \in [1+b_\sigma,|t|-a_\sigma]} |v| - t_i,$$

where $\Omega_{(0,|v|)}(\sigma)$ is the set of supporting time series of $v$ wrt $[0,|v|]$, and $|t|$ is the length of the time series $t$.

<table>
<thead>
<tr>
<th>name $\sigma$</th>
<th>$\nu_\sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bump</td>
<td>2</td>
</tr>
<tr>
<td>Dec</td>
<td>1</td>
</tr>
<tr>
<td>DecSeq</td>
<td>1</td>
</tr>
<tr>
<td>DecTer</td>
<td>1</td>
</tr>
<tr>
<td>Dip</td>
<td>2</td>
</tr>
<tr>
<td>Gorge</td>
<td>1</td>
</tr>
<tr>
<td>Inc</td>
<td>1</td>
</tr>
<tr>
<td>IncSeq</td>
<td>1</td>
</tr>
<tr>
<td>IncTer</td>
<td>1</td>
</tr>
<tr>
<td>Inflexion</td>
<td>0</td>
</tr>
<tr>
<td>Peak</td>
<td>0</td>
</tr>
<tr>
<td>Plain</td>
<td>1</td>
</tr>
<tr>
<td>Plateau</td>
<td>0</td>
</tr>
<tr>
<td>PropPlain</td>
<td>1</td>
</tr>
<tr>
<td>PropPlateau</td>
<td>0</td>
</tr>
<tr>
<td>Steady</td>
<td>0</td>
</tr>
<tr>
<td>SteadySeq</td>
<td>0</td>
</tr>
<tr>
<td>SDecSeq</td>
<td>1</td>
</tr>
<tr>
<td>SIncSeq</td>
<td>1</td>
</tr>
<tr>
<td>Summit</td>
<td>0</td>
</tr>
<tr>
<td>Valley</td>
<td>1</td>
</tr>
<tr>
<td>Zigzag</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 3.57: Regular-expression short names $\sigma$ and corresponding shift of $\sigma$ (see Definitions 17)
### Size

**Definition 18.** Consider a regular expression $\sigma$. The size of $\sigma$, denoted by $\omega_\sigma$, is a function that maps an element of $R_\Sigma$ to $\mathbb{N}^+$. It is defined by

$$\omega_\sigma = \min_{v \in L_\sigma} |v|.$$ 

<table>
<thead>
<tr>
<th>name $\sigma$</th>
<th>regular expression</th>
<th>$\omega_\sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bump</td>
<td>'&gt;&gt;&gt;&lt;&gt;'</td>
<td>5</td>
</tr>
<tr>
<td>Dec</td>
<td>'&gt;'</td>
<td>1</td>
</tr>
<tr>
<td>DecSeq</td>
<td>'&gt;(&gt;</td>
<td>=)*'</td>
</tr>
<tr>
<td>DecTer</td>
<td>'&gt;= =*'</td>
<td>3</td>
</tr>
<tr>
<td>Dip</td>
<td>'&lt;&lt;&lt;&lt;'</td>
<td>5</td>
</tr>
<tr>
<td>Gorge</td>
<td>'(&gt;</td>
<td>&gt;</td>
</tr>
<tr>
<td>Inc</td>
<td>'&lt;'</td>
<td>1</td>
</tr>
<tr>
<td>IncSeq</td>
<td>'(&lt;&lt;</td>
<td>=)*'</td>
</tr>
<tr>
<td>IncTer</td>
<td>'&lt;&lt; =*'</td>
<td>3</td>
</tr>
<tr>
<td>Inflexion</td>
<td>'&lt;&lt; (</td>
<td>=)*'</td>
</tr>
<tr>
<td>Peak</td>
<td>'&lt; (</td>
<td>=)*'</td>
</tr>
<tr>
<td>Plain</td>
<td>'&gt;= '='</td>
<td>2</td>
</tr>
<tr>
<td>Plateau</td>
<td>'&lt; &lt;'</td>
<td>1</td>
</tr>
<tr>
<td>PropPlain</td>
<td>'&gt;= =*'</td>
<td>3</td>
</tr>
<tr>
<td>PropPlateau</td>
<td>'&lt;&lt; =*'</td>
<td>3</td>
</tr>
<tr>
<td>Steady</td>
<td>'='</td>
<td>1</td>
</tr>
<tr>
<td>SteadySeq</td>
<td>'=' =*'</td>
<td>1</td>
</tr>
<tr>
<td>SDecSeq</td>
<td>'&gt;' =*'</td>
<td>1</td>
</tr>
<tr>
<td>SIncSeq</td>
<td>'&lt; &lt;*'</td>
<td>1</td>
</tr>
<tr>
<td>Summit</td>
<td>'(&lt;</td>
<td>(&lt;</td>
</tr>
<tr>
<td>Valley</td>
<td>'&gt;(&gt;</td>
<td>=)*'</td>
</tr>
<tr>
<td>Zigzag</td>
<td>'(&gt;&lt;)*'</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 3.58: Regular-expression short names $\sigma$ and corresponding size (see Definition 18); within each regular expression subparts corresponding to a smallest length word are highlighted in yellow.
3.3. TABLES OF REGULAR-EXPRESSION CHARACTERISTICS

3.3.10 Smallest-variation-of-maxima

**Definition 19.** Consider a regular expression \( \sigma \) and an integer interval domain \([\ell, u]\).
The shift of a proper factor \( w \) in a word \( v \in L_\sigma \) wrt \((\ell, u)\), denoted by \( \varpi_\sigma^{(\ell,u)}(v, w, i) \), is a function that maps an element of \( R_\Sigma \times \mathbb{Z} \times \mathbb{Z} \times \Sigma^* \times \Sigma^* \times \mathbb{N} \) to \( \mathbb{N} \). It is defined by

\[
\varpi_\sigma^{(\ell,u)}(v, w, i) = \min_{t \in \mathbb{N}^{\ell,u}(v)} \min_{x \in \ell_i} (\max(t) - x),
\]

where \( \max(t) \) is the maximum value of a time series \( t \), a supporting time series of \( v \) wrt \((\ell, u)\), and \( t_{w_i} \) is a subseries of \( t \) corresponding to the \( i^{th} \) extended \( \sigma \)-pattern whose signature is \( w \). If \( w \) is not a proper factor of \( v \), or \( i \) is strictly greater than the number of occurrences of \( w \) in \( v \), then \( \varpi_\sigma^{(\ell,u)}(v, w, i) \) is undefined.

**Definition 20.** Consider a regular expression \( \sigma \) and an integer interval domain \([\ell, u]\).
The smallest variation of maxima of superpositions of two words \( w \) and \( v \) in \( L_\sigma \) wrt \((\ell, u)\), denoted by \( \delta_\sigma^{(\ell,u)}(v, w) \), is a function that maps an element of \( R_\Sigma \times \mathbb{Z} \times \mathbb{Z} \times \Sigma^* \times \Sigma^* \) to \( \mathbb{N} \). It is defined by

\[
\delta_\sigma^{(\ell,u)}(v, w) = \begin{cases} 
\varpi_\sigma(z_v, v, 1) - \varpi_\sigma(z_v, w, 1), & \text{if } v \neq w \text{ and } \Gamma_\sigma^{(\ell,u)}(v, w) \neq \emptyset \\
\varpi_\sigma(z_v, v, 1) - \varpi_\sigma(z_v, w, 2), & \text{if } v = w \text{ and } \Gamma_\sigma^{(\ell,u)}(v, w) \neq \emptyset \\
0, & \text{if } \Gamma_\sigma^{(\ell,u)}(v, w) = \emptyset
\end{cases}
\]

where the words \( z_v \) and \( z_v \) both belong to \( \Gamma_\sigma^{(\ell,u)}(v, w) \), and the value 

\[
\min_{z \in \Gamma_\sigma^{(\ell,u)}(v, w)} |\varpi_\sigma(z, v, 1) - \varpi_\sigma(z, w, 1)| \quad \text{(respectively } \min_{z \in \Gamma_\sigma^{(\ell,u)}(v, w)} |\varpi_\sigma(z, v, 1) - \varpi_\sigma(z, w, 2)) \text{ is reached when } z \text{ is } z_v \text{ (respectively } z_v).\]

**Definition 21.** Consider a regular expression \( \sigma \) and an integer interval domain \([\ell, u]\).
The smallest variation of maxima of \( \sigma \) wrt \((\ell, u)\), denoted by \( \delta_\sigma^{(\ell,u)} \), is a function that maps an element of \( R_\Sigma \times \mathbb{Z} \times \mathbb{Z} \) to \( \mathbb{N} \). It is defined by

\[
\delta_\sigma^{(\ell,u)} = \begin{cases} 
\delta_\sigma^{(\ell,u)}(v_v, w_v), & \text{if } \exists v, w \in L_\sigma \text{ such that } \delta_\sigma^{(\ell,u)}(v, w) \neq 0 \\
0, & \text{otherwise}
\end{cases}
\]

where the words \( v_v \) and \( w_v \) both belong to \( L_\sigma \) and the value 

\[
\min_{v, w \in L_\sigma} |\delta_\sigma^{(\ell,u)}(v, w)|
\]

is reached when \( v \) is \( v_v \) and \( w \) is \( w_v \).
Table 3.59: Regular-expression short names $\sigma$ and corresponding smallest variation of maxima (see Definition 21), where $\eta_\sigma$ stands for the height characteristics of the corresponding $\sigma$ (see Definition 8); maxima of two consecutive pattern occurrences • and ○ are shown in red, i.e., • or ○.

<table>
<thead>
<tr>
<th>name $\sigma$</th>
<th>illustration</th>
<th>$\delta^{(\ell,u)}_{\sigma}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bump</td>
<td>![Bump Illustration]</td>
<td>0</td>
</tr>
</tbody>
</table>
| Dec           | ![Dec Illustration] | \[
\begin{cases}
0 & \text{if } u - \ell \leq \eta_\sigma \\
-1 & \text{otherwise}
\end{cases}
\] |
| DecSeq        | ![DecSeq Illustration] | 0               |
| DecTer        | ![DecTer Illustration] | \[
\begin{cases}
0 & \text{if } u - \ell \leq \eta_\sigma \\
-1 & \text{otherwise}
\end{cases}
\] |
| Dip           | ![Dip Illustration] | 0               |
| Gorge         | ![Gorge Illustration] | 0               |
| Inc           | ![Inc Illustration] | \[
\begin{cases}
0 & \text{if } u - \ell \leq \eta_\sigma \\
1 & \text{otherwise}
\end{cases}
\] |
| IncSeq        | ![IncSeq Illustration] | 0               |
| IncTer        | ![IncTer Illustration] | \[
\begin{cases}
0 & \text{if } u - \ell \leq \eta_\sigma \\
1 & \text{otherwise}
\end{cases}
\] |
| Inflexion     | ![Inflexion Illustration] | 0               |
| Peak          | ![Peak Illustration] | 0               |
| Plain         | ![Plain Illustration] | 0               |
| Plateau       | ![Plateau Illustration] | 0               |
| PropPlain     | ![PropPlain Illustration] | 0               |
| PropPlateau   | ![PropPlateau Illustration] | 0               |
| Steady        | ![Steady Illustration] | 0               |
| SteadySeq     | ![SteadySeq Illustration] | 0               |
| SDecSeq       | ![SDecSeq Illustration] | 0               |
| SIncSeq       | ![SIncSeq Illustration] | 0               |
| Summit        | ![Summit Illustration] | 0               |
| Valley        | ![Valley Illustration] | 0               |
| Zigzag        | ![Zigzag Illustration] | 0               |
4

Global Constraint Catalogue

«Mon problème, avec les classements, c’est qu’ils ne durent pas ; à peine ai-je fini de mettre de l’ordre que cet ordre est déjà caduc.»

– Georges Perec, Penser/Classer

Contents

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_HEIGHT_DECREASING_TERRACE</td>
<td>158</td>
</tr>
<tr>
<td>ALL_EQUAL_HEIGHT_INCREASING_TERRACE</td>
<td>162</td>
</tr>
<tr>
<td>ALL_EQUAL_HEIGHT_PLAIN</td>
<td>166</td>
</tr>
<tr>
<td>ALL_EQUAL_HEIGHT_PLATEAU</td>
<td>170</td>
</tr>
<tr>
<td>ALL_EQUAL_HEIGHT_PROPER_PLAIN</td>
<td>174</td>
</tr>
<tr>
<td>ALL_EQUAL_HEIGHT_PROPER_PLATEAU</td>
<td>178</td>
</tr>
<tr>
<td>ALL_EQUAL_HEIGHT_STEADY</td>
<td>182</td>
</tr>
<tr>
<td>ALL_EQUAL_HEIGHT_STEADY_SEQUENCE</td>
<td>186</td>
</tr>
<tr>
<td>ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE</td>
<td>190</td>
</tr>
<tr>
<td>ALL_EQUAL_MAX_DECREASING</td>
<td>194</td>
</tr>
<tr>
<td>ALL_EQUAL_MAX_DECREASING_SEQUENCE</td>
<td>198</td>
</tr>
<tr>
<td>ALL_EQUAL_MAX_DIP_ON_INCREASINGSEQUENCE</td>
<td>202</td>
</tr>
<tr>
<td>ALL_EQUAL_MAX_INCREASING</td>
<td>206</td>
</tr>
<tr>
<td>ALL_EQUAL_MAX_INCREASING_SEQUENCE</td>
<td>210</td>
</tr>
<tr>
<td>ALL_EQUAL_MAX_INFLEXION</td>
<td>214</td>
</tr>
<tr>
<td>ALL_EQUAL_MAX_PEAK</td>
<td>218</td>
</tr>
<tr>
<td>ALL_EQUAL_MAX_STRICTLY_DECREASING_SEQUENCE</td>
<td>222</td>
</tr>
<tr>
<td>ALL_EQUAL_MAX_STRICTLY_INCREASINGSEQUENCE</td>
<td>226</td>
</tr>
<tr>
<td>ALL_EQUAL_MAX_SUMMIT</td>
<td>230</td>
</tr>
<tr>
<td>ALL_EQUAL_MAX_ZIGZAG</td>
<td>234</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE</td>
<td>238</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN_DECREASING</td>
<td>242</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN_DECREASING_SEQUENCE</td>
<td>246</td>
</tr>
<tr>
<td>Constraint</td>
<td>Page</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td>250</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN_GORGE</td>
<td>254</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN_INCREASING</td>
<td>258</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN_INCREASING_SEQUENCE</td>
<td>262</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN_INFLEXION</td>
<td>266</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN STRICTLY_DECREASING_SEQUENCE</td>
<td>270</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN.Strictly_INCREASING_SEQUENCE</td>
<td>274</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN_VALLEY</td>
<td>278</td>
</tr>
<tr>
<td>ALL_EQUAL_MIN_ZIGZAG</td>
<td>282</td>
</tr>
<tr>
<td>ALL_EQUAL_RANGE_Decreasing</td>
<td>286</td>
</tr>
<tr>
<td>ALL_EQUAL_RANGE_DECREASING_SEQUENCE</td>
<td>290</td>
</tr>
<tr>
<td>ALL_EQUAL_RANGE_INCREASING</td>
<td>294</td>
</tr>
<tr>
<td>ALL_EQUAL_RANGE_INCREASING_SEQUENCE</td>
<td>298</td>
</tr>
<tr>
<td>ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE</td>
<td>302</td>
</tr>
<tr>
<td>ALL_EQUAL_RANGE_STRICTLY_INCREASING_SEQUENCE</td>
<td>306</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
<td>310</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_DECREASING</td>
<td>314</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_Decreasing_SEQUENCE</td>
<td>318</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_DECREASING_TERRACE</td>
<td>322</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>326</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_GORGE</td>
<td>330</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_INCREASING</td>
<td>334</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_INCREASING_SEQUENCE</td>
<td>338</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_INCREASING_TERRACE</td>
<td>342</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_INFLEXION</td>
<td>346</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_PEAK</td>
<td>350</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_PLAIN</td>
<td>354</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_PLATEAU</td>
<td>358</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_PROPER_PLAIN</td>
<td>362</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_PROPER_TERRACE</td>
<td>366</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_STEADY</td>
<td>370</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_STEADY_SEQUENCE</td>
<td>374</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_STRICTLY_DECREASING_SEQUENCE</td>
<td>378</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_STRICTLY_INCREASING_SEQUENCE</td>
<td>382</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_SUMMIT</td>
<td>386</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_VALLEY</td>
<td>390</td>
</tr>
<tr>
<td>ALL_EQUAL_SURF_ZIGZAG</td>
<td>394</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_Decreasing_SEQUENCE</td>
<td>398</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_Decreasing_TERRACE</td>
<td>402</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_GORGE</td>
<td>406</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_INCREASING_SEQUENCE</td>
<td>410</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_INCREASING_TERRACE</td>
<td>414</td>
</tr>
<tr>
<td>ALL_EQUAL_WIDTH_INFLEXION</td>
<td>418</td>
</tr>
</tbody>
</table>
4. GLOBAL CONSTRAINT CATALOGUE

DECREASING_RANGE_DECREASING ................................. 594
DECREASING_RANGE_DECREASING_SEQUENCE ............. 598
DECREASING_RANGE_INCREASING ................................. 602
DECREASING_RANGE_INCREASING_SEQUENCE ............. 606
DECREASING_RANGE_STRICTLY_DECREASING_SEQUENCE .... 610
DECREASING_RANGE_STRICTLY_INCREASING_SEQUENCE .... 614
DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE .... 618
DECREASING_SURF_DECREASING ................................. 622
DECREASING_SURF_DECREASING_SEQUENCE ............. 626
DECREASING_SURF_DECREASING_TERRACE ................. 630
DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE .... 634
DECREASING_SURF_GORGE ................................. 638
DECREASING_SURF_INCREASING ................................. 642
DECREASING_SURF_INCREASING_SEQUENCE .......... 646
DECREASING_SURF_INCREASING_TERRACE ................. 650
DECREASING_SURF_INFLEXION ................................. 654
DECREASING_SURF_PEAK ................................. 658
DECREASING_SURF_PLAIN ................................. 662
DECREASING_SURF_PLATEAU ................................. 666
DECREASING_SURF_PROPER_PLAIN ............................ 670
DECREASING_SURF_PROPER_PLATEAU .......................... 674
DECREASING_SURF_STEADY ................................. 678
DECREASING_SURF_STEADY_SEQUENCE ................. 682
DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE .... 686
DECREASING_SURF_STRICTLY_INCREASING_SEQUENCE .... 690
DECREASING_SURF_SUMMIT ................................. 694
DECREASING_SURF_VALLEY ................................. 698
DECREASING_SURF_ZIGZAG ................................. 702
DECREASING_WIDTH_DECREASING_SEQUENCE ........... 706
DECREASING_WIDTH_DECREASING_TERRACE ............... 710
DECREASING_WIDTH_GORGE ................................. 714
DECREASING_WIDTH_INCREASING_SEQUENCE ........... 718
DECREASING_WIDTH_INCREASING_TERRACE ............... 722
DECREASING_WIDTH_INFLEXION ............................ 726
DECREASING_WIDTH_PEAK ................................. 730
DECREASING_WIDTH_PLAIN ................................. 734
DECREASING_WIDTH_PLATEAU .............................. 738
DECREASING_WIDTH_PROPER_PLAIN .......................... 742
DECREASING_WIDTH_PROPER_PLATEAU ...................... 746
DECREASING_WIDTH_STEADY_SEQUENCE ................. 750
DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE .... 754
DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE .... 758
DECREASING_WIDTH_SUMMIT ............................... 762
<table>
<thead>
<tr>
<th>Sequence Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_WIDTH_VALLEY</td>
<td>766</td>
</tr>
<tr>
<td>DECREASING_WIDTH_ZIGZAG</td>
<td>770</td>
</tr>
<tr>
<td>HEIGHT_DECREASING_TERRACE</td>
<td>774</td>
</tr>
<tr>
<td>HEIGHT_INCREASING_TERRACE</td>
<td>778</td>
</tr>
<tr>
<td>HEIGHT_PLAIN</td>
<td>782</td>
</tr>
<tr>
<td>HEIGHT_PLATEAU</td>
<td>786</td>
</tr>
<tr>
<td>HEIGHT_PROPER_PLAIN</td>
<td>790</td>
</tr>
<tr>
<td>HEIGHT_PROPER_PLATEAU</td>
<td>794</td>
</tr>
<tr>
<td>HEIGHT_STEADY</td>
<td>798</td>
</tr>
<tr>
<td>HEIGHT_STEADY_SEQUENCE</td>
<td>802</td>
</tr>
<tr>
<td>INCREASING_HEIGHT_DECREASING_TERRACE</td>
<td>806</td>
</tr>
<tr>
<td>INCREASING_HEIGHT_INCREASING_TERRACE</td>
<td>810</td>
</tr>
<tr>
<td>INCREASING_HEIGHT_PLAIN</td>
<td>814</td>
</tr>
<tr>
<td>INCREASING_HEIGHT_PROPER_PLAIN</td>
<td>822</td>
</tr>
<tr>
<td>INCREASING_HEIGHT_PROPER_PLATEAU</td>
<td>826</td>
</tr>
<tr>
<td>INCREASING_HEIGHT_STEADY</td>
<td>830</td>
</tr>
<tr>
<td>INCREASING_HEIGHT_STEADY_SEQUENCE</td>
<td>834</td>
</tr>
<tr>
<td>INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE</td>
<td>838</td>
</tr>
<tr>
<td>INCREASING_MAX_DECREASING</td>
<td>842</td>
</tr>
<tr>
<td>INCREASING_MAX_DECREASING_SEQUENCE</td>
<td>846</td>
</tr>
<tr>
<td>INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE</td>
<td>850</td>
</tr>
<tr>
<td>INCREASING_MAX_INCREASING</td>
<td>854</td>
</tr>
<tr>
<td>INCREASING_MAX_INCREASING_SEQUENCE</td>
<td>858</td>
</tr>
<tr>
<td>INCREASING_MAX_INFLEXION</td>
<td>862</td>
</tr>
<tr>
<td>INCREASING_MAX_PEAK</td>
<td>866</td>
</tr>
<tr>
<td>INCREASING_MAX STRICTLY_DECREASING_SEQUENCE</td>
<td>870</td>
</tr>
<tr>
<td>INCREASING_MAX STRICTLY_INCREASING_SEQUENCE</td>
<td>874</td>
</tr>
<tr>
<td>INCREASING_MAX_SUMMIT</td>
<td>878</td>
</tr>
<tr>
<td>INCREASING_MAX_ZIGZAG</td>
<td>882</td>
</tr>
<tr>
<td>INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE</td>
<td>886</td>
</tr>
<tr>
<td>INCREASING_MIN_DECREASING</td>
<td>890</td>
</tr>
<tr>
<td>INCREASING_MIN_DECREASING_SEQUENCE</td>
<td>894</td>
</tr>
<tr>
<td>INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td>898</td>
</tr>
<tr>
<td>INCREASING_MIN_GORGE</td>
<td>902</td>
</tr>
<tr>
<td>INCREASING_MIN_INCREASING</td>
<td>906</td>
</tr>
<tr>
<td>INCREASING_MIN_INCREASING_SEQUENCE</td>
<td>910</td>
</tr>
<tr>
<td>INCREASING_MIN_INFLEXION</td>
<td>914</td>
</tr>
<tr>
<td>INCREASING_MIN STRICTLY_DECREASING_SEQUENCE</td>
<td>918</td>
</tr>
<tr>
<td>INCREASING_MIN STRICTLY_INCREASING_SEQUENCE</td>
<td>922</td>
</tr>
<tr>
<td>INCREASING_MIN_RANGE_DECREASING</td>
<td>926</td>
</tr>
<tr>
<td>INCREASING_MIN_VALLEY</td>
<td>930</td>
</tr>
<tr>
<td>INCREASING_RANGE_DECREASING</td>
<td>934</td>
</tr>
<tr>
<td>Constraint</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>INCREASING_RANGE_DECENDING_SEQUENCE</td>
<td>938</td>
</tr>
<tr>
<td>INCREASING_RANGE_INCREASING</td>
<td>942</td>
</tr>
<tr>
<td>INCREASING_RANGE_INCREASING_SEQUENCE</td>
<td>946</td>
</tr>
<tr>
<td>INCREASING_RANGE.Strictly_DECENDING_SEQUENCE</td>
<td>950</td>
</tr>
<tr>
<td>INCREASING_RANGE.Strictly_INCREASING_SEQUENCE</td>
<td>954</td>
</tr>
<tr>
<td>INCREASING_SURF_BUMP_ON_DECENDING_SEQUENCE</td>
<td>958</td>
</tr>
<tr>
<td>INCREASING_SURF_DECRESSING</td>
<td>962</td>
</tr>
<tr>
<td>INCREASING_SURF_DECRESSING_SEQUENCE</td>
<td>966</td>
</tr>
<tr>
<td>INCREASING_SURF_DECRESSING_TERRACE</td>
<td>970</td>
</tr>
<tr>
<td>INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>974</td>
</tr>
<tr>
<td>INCREASING_SURF_GORGE</td>
<td>978</td>
</tr>
<tr>
<td>INCREASING_SURF_INCREASING</td>
<td>982</td>
</tr>
<tr>
<td>INCREASING_SURF_INCREASING_SEQUENCE</td>
<td>986</td>
</tr>
<tr>
<td>INCREASING_SURF_INCREASING_TERRACE</td>
<td>990</td>
</tr>
<tr>
<td>INCREASING_SURF_INFLEXION</td>
<td>994</td>
</tr>
<tr>
<td>INCREASING_SURF_PEAK</td>
<td>998</td>
</tr>
<tr>
<td>INCREASING_SURF_PLAIN</td>
<td>1002</td>
</tr>
<tr>
<td>INCREASING_SURF_PLATEAU</td>
<td>1006</td>
</tr>
<tr>
<td>INCREASING_SURF_PROPER_PLAIN</td>
<td>1010</td>
</tr>
<tr>
<td>INCREASING_SURF_PROPER_PLATEAU</td>
<td>1014</td>
</tr>
<tr>
<td>INCREASING_SURF_STEADY</td>
<td>1018</td>
</tr>
<tr>
<td>INCREASING_SURF_STEADY_SEQUENCE</td>
<td>1022</td>
</tr>
<tr>
<td>INCREASING_SURF_STRICTLY_DECRESSING_SEQUENCE</td>
<td>1026</td>
</tr>
<tr>
<td>INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE</td>
<td>1030</td>
</tr>
<tr>
<td>INCREASING_SURF_SUMMIT</td>
<td>1034</td>
</tr>
<tr>
<td>INCREASING_SURF_VALLEY</td>
<td>1038</td>
</tr>
<tr>
<td>INCREASING_SURF_ZIGZAG</td>
<td>1042</td>
</tr>
<tr>
<td>INCREASING_WIDTH_DECRESSING_SEQUENCE</td>
<td>1046</td>
</tr>
<tr>
<td>INCREASING_WIDTH_DECRESSING_TERRACE</td>
<td>1050</td>
</tr>
<tr>
<td>INCREASING_WIDTH_GORGE</td>
<td>1054</td>
</tr>
<tr>
<td>INCREASING_WIDTH_InCREASING_SEQUENCE</td>
<td>1058</td>
</tr>
<tr>
<td>INCREASING_WIDTH_INCREASING_TERRACE</td>
<td>1062</td>
</tr>
<tr>
<td>INCREASING_WIDTH_INFLEXION</td>
<td>1066</td>
</tr>
<tr>
<td>INCREASING_WIDTH_PEAK</td>
<td>1070</td>
</tr>
<tr>
<td>INCREASING_WIDTH_PLAIN</td>
<td>1074</td>
</tr>
<tr>
<td>INCREASING_WIDTH_PLATEAU</td>
<td>1078</td>
</tr>
<tr>
<td>INCREASING_WIDTH_PROPER_PLAIN</td>
<td>1082</td>
</tr>
<tr>
<td>INCREASING_WIDTH_PROPER_PLATEAU</td>
<td>1086</td>
</tr>
<tr>
<td>INCREASING_WIDTH_STEADY_SEQUENCE</td>
<td>1090</td>
</tr>
<tr>
<td>INCREASING_WIDTH_STRICTLY_DECRESSING_SEQUENCE</td>
<td>1094</td>
</tr>
<tr>
<td>INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE</td>
<td>1098</td>
</tr>
<tr>
<td>INCREASING_WIDTH_SUMMIT</td>
<td>1102</td>
</tr>
<tr>
<td>INCREASING_WIDTH_VALLEY</td>
<td>1106</td>
</tr>
<tr>
<td>Constraint</td>
<td>Reference</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>MAX_MAX_INCREASING_SEQUENCE</td>
<td>1282</td>
</tr>
<tr>
<td>MAX_MAX_INFLEXION</td>
<td>1286</td>
</tr>
<tr>
<td>MAX_MAX_PEAK</td>
<td>1290</td>
</tr>
<tr>
<td>MAX_MAX_STRICTLY_DECREASING_SEQUENCE</td>
<td>1294</td>
</tr>
<tr>
<td>MAX_MAX_STRICTLY_INCREASING_SEQUENCE</td>
<td>1298</td>
</tr>
<tr>
<td>MAX_MAX_SUMMIT</td>
<td>1302</td>
</tr>
<tr>
<td>MAX_MAX_ZIGZAG</td>
<td>1306</td>
</tr>
<tr>
<td>MAX_MIN_BUMP_ON_DECREASING_SEQUENCE</td>
<td>1314</td>
</tr>
<tr>
<td>MAX_MIN_DECREASING</td>
<td>1318</td>
</tr>
<tr>
<td>MAX_MIN_DECREASING_SEQUENCE</td>
<td>1322</td>
</tr>
<tr>
<td>MAX_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td>1326</td>
</tr>
<tr>
<td>MAX_MIN_GORGE</td>
<td>1330</td>
</tr>
<tr>
<td>MAX_MIN_INCREASING</td>
<td>1336</td>
</tr>
<tr>
<td>MAX_MIN_INCREASING_SEQUENCE</td>
<td>1340</td>
</tr>
<tr>
<td>MAX_MIN_INFLEXION</td>
<td>1344</td>
</tr>
<tr>
<td>MAX_MIN_STRICTLY_DECREASING_SEQUENCE</td>
<td>1348</td>
</tr>
<tr>
<td>MAX_MIN_STRICTLY_INCREASING_SEQUENCE</td>
<td>1352</td>
</tr>
<tr>
<td>MAX_MIN_VALLEY</td>
<td>1356</td>
</tr>
<tr>
<td>MAX_MIN_ZIGZAG</td>
<td>1360</td>
</tr>
<tr>
<td>MAX_PEAK</td>
<td>1368</td>
</tr>
<tr>
<td>MAX_RANGE_DECREASING</td>
<td>1372</td>
</tr>
<tr>
<td>MAX_RANGE_DECREASING_SEQUENCE</td>
<td>1376</td>
</tr>
<tr>
<td>MAX_RANGE_INCREASING</td>
<td>1380</td>
</tr>
<tr>
<td>MAX_RANGE_INCREASING_SEQUENCE</td>
<td>1384</td>
</tr>
<tr>
<td>MAX_RANGE_STRICTLY_DECREASING_SEQUENCE</td>
<td>1388</td>
</tr>
<tr>
<td>MAX_RANGE_STRICTLY_INCREASING_SEQUENCE</td>
<td>1392</td>
</tr>
<tr>
<td>MAX_STRICTLY_DECREASING_SEQUENCE</td>
<td>1396</td>
</tr>
<tr>
<td>MAX_STRICTLY_INCREASING_SEQUENCE</td>
<td>1400</td>
</tr>
<tr>
<td>MAX_SUMMIT</td>
<td>1404</td>
</tr>
<tr>
<td>MAX_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
<td>1408</td>
</tr>
<tr>
<td>MAX_SURF_DECREASING</td>
<td>1412</td>
</tr>
<tr>
<td>MAX_SURF_DECREASING_SEQUENCE</td>
<td>1416</td>
</tr>
<tr>
<td>MAX_SURF_DECREASING_TERRACE</td>
<td>1420</td>
</tr>
<tr>
<td>MAX_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>1424</td>
</tr>
<tr>
<td>MAX_SURF_GORGE</td>
<td>1428</td>
</tr>
<tr>
<td>MAX_SURF_INCREASING</td>
<td>1432</td>
</tr>
<tr>
<td>MAX_SURF_INCREASING_SEQUENCE</td>
<td>1436</td>
</tr>
<tr>
<td>MAX_SURF_INCREASING_TERRACE</td>
<td>1440</td>
</tr>
<tr>
<td>MAX_SURF_INFLEXION</td>
<td>1444</td>
</tr>
<tr>
<td>MAX_SURF_PEAK</td>
<td>1448</td>
</tr>
<tr>
<td>MAX_SURF_PLAIN</td>
<td>1452</td>
</tr>
<tr>
<td>MAX_SURF_PLATEAU</td>
<td>1456</td>
</tr>
<tr>
<td>MAX_SURF_PROPER_PLAIN</td>
<td>1460</td>
</tr>
</tbody>
</table>
MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX

MIN

MAX
4. GLOBAL CONSTRAINT CATALOGUE

MIN_MAX_DECREASING 1648
MIN_MAX_DECREASING_SEQUENCE 1652
MIN_MAX_DIP_ON_INCREASING_SEQUENCE 1656
MIN_MAX_INCREASING 1660
MIN_MAX_INCREASING_SEQUENCE 1664
MIN_MAX_INFLEXION 1668
MIN_MAX_PEAK 1672
MIN_MAX_STRICTLY_DECREASING_SEQUENCE 1676
MIN_MAX_STRICTLY_INCREASING_SEQUENCE 1680
MIN_MAX_SUMMIT 1684
MIN_MAX_ZIGZAG 1688
MIN_MIN_BUMP_ON_DECREASING_SEQUENCE 1696
MIN_MIN_DECREASING 1700
MIN_MIN_DECREASING_SEQUENCE 1704
MIN_MIN_DIP_ON_INCREASING_SEQUENCE 1708
MIN_MIN_GORGE 1712
MIN_MIN_INCREASING 1718
MIN_MIN_INCREASING_SEQUENCE 1722
MIN_MIN_INFLEXION 1726
MIN_MIN_STRICTLY_DECREASING_SEQUENCE 1730
MIN_MIN_STRICTLY_INCREASING_SEQUENCE 1734
MIN_MIN_VALLEY 1738
MIN_MIN_ZIGZAG 1742
MIN_RANGE_DECREASING 1750
MIN_RANGE_DECREASING_SEQUENCE 1754
MIN_RANGE_INCREASING 1758
MIN_RANGE_INCREASING_SEQUENCE 1762
MIN_RANGE_STRICTLY_DECREASING_SEQUENCE 1766
MIN_RANGE_STRICTLY_INCREASING_SEQUENCE 1770
MIN_STRICTLY_DECREASING_SEQUENCE 1774
MIN_STRICTLY_INCREASING_SEQUENCE 1778
MIN_SURF_BUMP_ON_DECREASING_SEQUENCE 1782
MIN_SURF_DECREASING 1786
MIN_SURF_DECREASING_SEQUENCE 1790
MIN_SURF_DECREASING_TERRACE 1794
MIN_SURF_DIP_ON_INCREASING_SEQUENCE 1798
MIN_SURF_GORGE 1802
MIN_SURF_INCREASING 1806
MIN_SURF_INCREASING_SEQUENCE 1810
MIN_SURF_INCREASING_TERRACE 1814
MIN_SURF_INFLEXION 1818
MIN_SURF_PEAK 1822
MIN_SURF_PLAIN 1826
<p>| MIN_Surf_Plateau                      | 1830 |
| MIN_Surf_Proper_Plateau              | 1834 |
| MIN_Surf_Steady                      | 1838 |
| MIN_Surf_Steady_Sequence             | 1842 |
| MIN_Surf_Strictly_Decreasing_Sequence| 1846 |
| MIN_Surf_Strictly_Increasing_Sequence| 1850 |
| MIN_Surf_Summit                      | 1854 |
| MIN_Surf_Proper_Plane                | 1858 |
| MIN_Surf_Proper_Plateau              | 1862 |
| MIN_Surf_Zigzag                      | 1866 |
| MIN_Valley                           | 1870 |
| MIN_Width_Decreasing_Sequence        | 1874 |
| MIN_Width_Decreasing_Terrace          | 1878 |
| MIN_Width_Gorge                       | 1882 |
| MIN_Width_Increasing_Sequence         | 1886 |
| MIN_Width_Increasing_Terrace          | 1890 |
| MIN_Width_Inflexion                   | 1894 |
| MIN_Width_Peak                        | 1898 |
| MIN_Width_Plain                       | 1902 |
| MIN_Width_Plateau                     | 1906 |
| MIN_Width_Plateau                     | 1910 |
| MIN_Width_Proper_Plain                | 1914 |
| MIN_Width_Proper_Plateau              | 1918 |
| MIN_Width_Steady_Sequence             | 1922 |
| MIN_Width_Strictly_Decreasing_Sequence| 1926 |
| MIN_Width_Strictly_Increasing_Sequence| 1930 |
| MIN_Width_Summit                      | 1934 |
| MIN_Width_Valley                      | 1938 |
| MIN_Width_Zigzag                      | 1942 |
| MIN_Zigzag                            | 1946 |
| NB_Bump_On_Decreasing_Sequence       | 1950 |
| NB_Decomposing                        | 1954 |
| NB_Decreasing                         | 1958 |
| NB_Decreasing_Sequence                | 1962 |
| NB_Decreasing_Terrace                 | 1966 |
| NB_Dip_On_Increasing_Sequence         | 1970 |
| NB_Gorge                              | 1974 |
| NB_Increasing                         | 1978 |
| NB_Increasing_Sequence                | 1982 |
| NB_Increasing_Terrace                 | 1986 |
| NB_Inflexion                          | 1990 |
| NB_Peak                               | 1994 |
| NB_Plain                              | 1998 |
| NB_Plateau                            | 2002 |
| NB_Proper_Plateau                     | 2006 |
| NB_Proper_Plain                       | 2010 |
| NB_Proper_Plateau                     | 2014 |
| NB_Peak                               | 2018 |
| NB_Plain                              | 2022 |
| NB_Plateau                            | 2026 |
| NB_Proper_Plateau                     | 2030 |</p>
<table>
<thead>
<tr>
<th>Constraint</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB_PROPER_PLATEAU</td>
<td>2040</td>
</tr>
<tr>
<td>NB_STEADY</td>
<td>2046</td>
</tr>
<tr>
<td>NB_STEADY_SEQUENCE</td>
<td>2052</td>
</tr>
<tr>
<td>NB STRICTLY_DECREASING_SEQUENCE</td>
<td>2058</td>
</tr>
<tr>
<td>NB STRICTLY_INCREASING_SEQUENCE</td>
<td>2064</td>
</tr>
<tr>
<td>NB_SUMMIT</td>
<td>2070</td>
</tr>
<tr>
<td>NB_VALLEY</td>
<td>2078</td>
</tr>
<tr>
<td>NB_ZIGZAG</td>
<td>2084</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_DECREASING_TERRACE</td>
<td>2092</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_INCREASING_TERRACE</td>
<td>2096</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_PLAIN</td>
<td>2100</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_PLATEAU</td>
<td>2104</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_PROPER_PLAIN</td>
<td>2108</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_STEADY</td>
<td>2112</td>
</tr>
<tr>
<td>POS_MAX_HEIGHT_STEADY_SEQUENCE</td>
<td>2116</td>
</tr>
<tr>
<td>POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE</td>
<td>2124</td>
</tr>
<tr>
<td>POS_MAX_MAX_DECREASING</td>
<td>2128</td>
</tr>
<tr>
<td>POS_MAX_MAX_DECREASING_SEQUENCE</td>
<td>2132</td>
</tr>
<tr>
<td>POS_MAX_MAX_DIP_ON_INCREASING_SEQUENCE</td>
<td>2136</td>
</tr>
<tr>
<td>POS_MAX_MAX_INCREASING</td>
<td>2140</td>
</tr>
<tr>
<td>POS_MAX_MAX_INCREASING_SEQUENCE</td>
<td>2144</td>
</tr>
<tr>
<td>POS_MAX_MAX_INFLEXION</td>
<td>2148</td>
</tr>
<tr>
<td>POS_MAX_MAX_PEAK</td>
<td>2152</td>
</tr>
<tr>
<td>POS_MAX_MAX.Strictly_DECREASING.SEQUENCE</td>
<td>2156</td>
</tr>
<tr>
<td>POS_MAX_MAX.Strictly_INCREASING.SEQUENCE</td>
<td>2160</td>
</tr>
<tr>
<td>POS_MAX_MAX_SUMMIT</td>
<td>2164</td>
</tr>
<tr>
<td>POS_MAX_MAX_ZIGZAG</td>
<td>2168</td>
</tr>
<tr>
<td>POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE</td>
<td>2172</td>
</tr>
<tr>
<td>POS_MAX_MIN_DECREASING</td>
<td>2176</td>
</tr>
<tr>
<td>POS_MAX_MIN_DECREASING_SEQUENCE</td>
<td>2180</td>
</tr>
<tr>
<td>POS_MAX_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td>2184</td>
</tr>
<tr>
<td>POS_MAX_MIN_GORGE</td>
<td>2188</td>
</tr>
<tr>
<td>POS_MAX_MIN_INCREASING</td>
<td>2192</td>
</tr>
<tr>
<td>POS_MAX_MIN_INCREASING_SEQUENCE</td>
<td>2196</td>
</tr>
<tr>
<td>POS_MAX_MIN_INFLEXION</td>
<td>2200</td>
</tr>
<tr>
<td>POS_MAX_MIN.Strictly_DECREASING.SEQUENCE</td>
<td>2204</td>
</tr>
<tr>
<td>POS_MAX_MIN.Strictly_INCREASING.SEQUENCE</td>
<td>2208</td>
</tr>
<tr>
<td>POS_MAX_MIN_VALLEY</td>
<td>2212</td>
</tr>
<tr>
<td>POS_MAX_MIN_ZIGZAG</td>
<td>2216</td>
</tr>
<tr>
<td>POS_MAX_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
<td>2220</td>
</tr>
<tr>
<td>POS_MAX_SURF_DECREASING</td>
<td>2224</td>
</tr>
<tr>
<td>POS_MAX_SURF_DECREASING_SEQUENCE</td>
<td>2228</td>
</tr>
</tbody>
</table>
POS_MAX_SURF_DECREASING_TERRACE 2232
POS_MAX_SURF_DIP_ON_INCREASING_SEQUENCE 2236
POS_MAX_SURF_GORGE 2240
POS_MAX_SURF_INCREASING 2244
POS_MAX_SURF_INCREASING_SEQUENCE 2248
POS_MAX_SURF_INCREASING_TERRACE 2252
POS_MAX_SURF_INFLEXION 2256
POS_MAX_SURF_PEAK 2260
POS_MAX_SURF_PLAIN 2264
POS_MAX_SURF_PLATEAU 2268
POS_MAX_SURF_PROPER_PLAIN 2272
POS_MAX_SURF_PROPER_PLATEAU 2276
POS_MAX_SURF_STEADY 2280
POS_MAX_SURF_STEADY_SEQUENCE 2284
POS_MAX_SURF_STRICTLY_DECREASING_SEQUENCE 2288
POS_MAX_SURF_STRICTLY_INCREASING_SEQUENCE 2292
POS_MAX_SURF_SUMMIT 2296
POS_MAX_SURF_VALLEY 2300
POS_MAX_SURF_ZIGZAG 2304
POS_MAX_WIDTH_DECREASING_SEQUENCE 2308
POS_MAX_WIDTH_DECREASING_TERRACE 2312
POS_MAX_WIDTH_GORGE 2316
POS_MAX_WIDTH_INCREASING_SEQUENCE 2320
POS_MAX_WIDTH_INCREASING_TERRACE 2324
POS_MAX_WIDTH_INFLEXION 2328
POS_MAX_WIDTH_PEAK 2332
POS_MAX_WIDTH_PLAIN 2336
POS_MAX_WIDTH_PLATEAU 2340
POS_MAX_WIDTH_PROPER_PLAIN 2344
POS_MAX_WIDTH_PROPER_PLATEAU 2348
POS_MAX_WIDTH_STEADY_SEQUENCE 2352
POS_MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE 2356
POS_MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE 2360
POS_MAX_WIDTH_SUMMIT 2364
POS_MAX_WIDTH_VALLEY 2368
POS_MAX_WIDTH_ZIGZAG 2372
POS_MIN_HEIGHT_DECREASING_TERRACE 2376
POS_MIN_HEIGHT_INCREASING_TERRACE 2380
POS_MIN_HEIGHT_PLAIN 2384
POS_MIN_HEIGHT_PLATEAU 2388
POS_MIN_HEIGHT_PROPER_PLAIN 2392
POS_MIN_HEIGHT_PROPER_PLATEAU 2396
POS_MIN_HEIGHT_STEADY 2400
4. GLOBAL CONSTRAINT CATALOGUE

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>POS_MIN_HEIGHT_STEADY_SEQUENCE</td>
<td>2404</td>
</tr>
<tr>
<td>POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE</td>
<td>2408</td>
</tr>
<tr>
<td>POS_MIN_MAX_DECREEASING</td>
<td>2412</td>
</tr>
<tr>
<td>POS_MIN_MAX_DECREEASING_SEQUENCE</td>
<td>2416</td>
</tr>
<tr>
<td>POS_MIN_MAX_DIP_ON_INCREASING_SEQUENCE</td>
<td>2420</td>
</tr>
<tr>
<td>POS_MIN_MAX_INCREASING</td>
<td>2424</td>
</tr>
<tr>
<td>POS_MIN_MAX_INCREASING_SEQUENCE</td>
<td>2428</td>
</tr>
<tr>
<td>POS_MIN_MAX_INFLEXION</td>
<td>2432</td>
</tr>
<tr>
<td>POS_MIN_MAX_PEEK</td>
<td>2436</td>
</tr>
<tr>
<td>POS_MIN_MAX_STRICTLY_DECREEASING_SEQUENCE</td>
<td>2440</td>
</tr>
<tr>
<td>POS_MIN_MAX_STRICTLY_INCREASING_SEQUENCE</td>
<td>2444</td>
</tr>
<tr>
<td>POS_MIN_MAX_SUMMIT</td>
<td>2448</td>
</tr>
<tr>
<td>POS_MIN_MAX_ZIGZAG</td>
<td>2452</td>
</tr>
<tr>
<td>POS_MIN_MIN_BUMP_ON_DECREEASING_SEQUENCE</td>
<td>2456</td>
</tr>
<tr>
<td>POS_MIN_MIN_DECREEASING</td>
<td>2460</td>
</tr>
<tr>
<td>POS_MIN_MIN_DECREEASING_SEQUENCE</td>
<td>2464</td>
</tr>
<tr>
<td>POS_MIN_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td>2468</td>
</tr>
<tr>
<td>POS_MIN_MIN_GORGE</td>
<td>2472</td>
</tr>
<tr>
<td>POS_MIN_MIN_INCREASING</td>
<td>2476</td>
</tr>
<tr>
<td>POS_MIN_MIN_INCREASING_SEQUENCE</td>
<td>2480</td>
</tr>
<tr>
<td>POS_MIN_MIN_INFLEXION</td>
<td>2484</td>
</tr>
<tr>
<td>POS_MIN_MIN_STRICTLY_DECREEASING_SEQUENCE</td>
<td>2488</td>
</tr>
<tr>
<td>POS_MIN_MIN_STRICTLY_INCREASING_SEQUENCE</td>
<td>2492</td>
</tr>
<tr>
<td>POS_MIN_MIN_VALLEY</td>
<td>2496</td>
</tr>
<tr>
<td>POS_MIN_MIN_ZIGZAG</td>
<td>2500</td>
</tr>
<tr>
<td>POS_MIN_SURF_BUMP_ON_DECREEASING_SEQUENCE</td>
<td>2504</td>
</tr>
<tr>
<td>POS_MIN_SURF_DECREEASING</td>
<td>2508</td>
</tr>
<tr>
<td>POS_MIN_SURF_DECREEASING_SEQUENCE</td>
<td>2512</td>
</tr>
<tr>
<td>POS_MIN_SURF_DECREEASING_TERRACE</td>
<td>2516</td>
</tr>
<tr>
<td>POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>2520</td>
</tr>
<tr>
<td>POS_MIN_SURF_GORGE</td>
<td>2524</td>
</tr>
<tr>
<td>POS_MIN_SURF_INCREASING</td>
<td>2528</td>
</tr>
<tr>
<td>POS_MIN_SURF_INCREASING_SEQUENCE</td>
<td>2532</td>
</tr>
<tr>
<td>POS_MIN_SURF_INCREASING_TERRACE</td>
<td>2536</td>
</tr>
<tr>
<td>POS_MIN_SURF_INFLEXION</td>
<td>2540</td>
</tr>
<tr>
<td>POS_MIN_SURF_PEEK</td>
<td>2544</td>
</tr>
<tr>
<td>POS_MIN_SURF.PLAIN</td>
<td>2548</td>
</tr>
<tr>
<td>POS_MIN_SURF_PLATEAU</td>
<td>2552</td>
</tr>
<tr>
<td>POS_MIN_SURF_PROPER.PLAIN</td>
<td>2556</td>
</tr>
<tr>
<td>POS_MIN_SURF_PROPER_PLATEAU</td>
<td>2560</td>
</tr>
<tr>
<td>POS_MIN_SURF_STEADY</td>
<td>2564</td>
</tr>
<tr>
<td>POS_MIN_SURF_STEADY_SEQUENCE</td>
<td>2568</td>
</tr>
<tr>
<td>POS_MIN_SURF_STRICTLY_DECREEASING_SEQUENCE</td>
<td>2572</td>
</tr>
</tbody>
</table>
POS_MIN_SURF_STRICTLY_INCREASING_SEQUENCE ............ 2576
POS_MIN_SURF_SUMMIT .................................. 2580
POS_MIN_SURF_VALLEY .................................. 2584
POS_MIN_SURF_ZIGZAG .................................. 2588
POS_MIN_WIDTH_DECREASINGSEQUENCE ................. 2592
POS_MIN_WIDTH_DECREASING_TERRACE ............... 2596
POS_MIN_WIDTH_GORGE .................................. 2600
POS_MIN_WIDTH_INCREASING_SEQUENCE ............... 2604
POS_MIN_WIDTH_INCREASING_TERRACE ............... 2608
POS_MIN_WIDTH_INFLEXION .................................. 2612
POS_MIN_WIDTH_PEAK .................................. 2616
POS_MIN_WIDTH_PLAIN .................................. 2620
POS_MIN_WIDTH_PLATEAU .................................. 2624
POS_MIN_WIDTH_PROPER_PLAIN ......................... 2628
POS_MIN_WIDTH_PROPER_PLATEAU ................. 2632
POS_MIN_WIDTH_STEADY_SEQUENCE ................. 2636
POS_MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE .... 2640
POS_MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE .... 2644
POS_MIN_WIDTH_SUMMIT .................................. 2648
POS_MIN_WIDTH_VALLEY .................................. 2652
POS_MIN_WIDTH_ZIGZAG .................................. 2656
SUM_HEIGHT_DECREASING_TERRACE ............... 2660
SUM_HEIGHT_INCREASING_TERRACE ............... 2666
SUM_HEIGHT_PLAIN .................................. 2672
SUM_HEIGHT_PLATEAU .................................. 2676
SUM_HEIGHT_PROPER_PLAIN ......................... 2680
SUM_HEIGHT_PROPER_PLATEAU ................. 2684
SUM_HEIGHT_STEADY .................................. 2688
SUM_HEIGHT_STEADY_SEQUENCE ............................. 2692
SUM_MAX_BUMP_ON_DECREASING_SEQUENCE ............. 2696
SUM_MAX_DECREASING .................................. 2700
SUM_MAX_DECREASING_SEQUENCE ..................... 2706
SUM_MAX_DIP_ON_INCREASING_SEQUENCE ............. 2710
SUM_MAX_INCREASING .................................. 2714
SUM_MAX_INCREASING_SEQUENCE ..................... 2720
SUM_MAX_INFLEXION .................................. 2724
SUM_MAX_PEAK .................................. 2728
SUM_MAX_STRICTLY_DECREASINGSEQUENCE .......... 2732
SUM_MAX_STRICTLY_INCREASINGSEQUENCE .......... 2736
SUM_MAX_SUMMIT .................................. 2740
SUM_MAX_ZIGZAG .................................. 2746
SUM_MIN_BUMP_ON_DECREASING_SEQUENCE ............. 2754
SUM_MIN_DECREASING .................................. 2758
<table>
<thead>
<tr>
<th>Constraint Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUM_MIN_DECREASING_SEQUENCE</td>
<td>2764</td>
</tr>
<tr>
<td>SUM_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td>2768</td>
</tr>
<tr>
<td>SUM_MIN_GORGE</td>
<td>2772</td>
</tr>
<tr>
<td>SUM_MIN_INCREASING</td>
<td>2778</td>
</tr>
<tr>
<td>SUM_MIN_INCREASING_SEQUENCE</td>
<td>2784</td>
</tr>
<tr>
<td>SUM_MIN_INFLEXION</td>
<td>2788</td>
</tr>
<tr>
<td>SUM_MIN STRICTLY DECREASING_SEQUENCE</td>
<td>2792</td>
</tr>
<tr>
<td>SUM_MIN STRICTLY INCREASING_SEQUENCE</td>
<td>2796</td>
</tr>
<tr>
<td>SUM_MIN_VALLEY</td>
<td>2800</td>
</tr>
<tr>
<td>SUM_MIN_ZIGZAG</td>
<td>2804</td>
</tr>
<tr>
<td>SUM_RANGE_DECREASING</td>
<td>2812</td>
</tr>
<tr>
<td>SUM_RANGE_Decreasing_sequence</td>
<td>2816</td>
</tr>
<tr>
<td>SUM_RANGE_INCREASING</td>
<td>2820</td>
</tr>
<tr>
<td>SUM_RANGE_INCREASING_SEQUENCE</td>
<td>2824</td>
</tr>
<tr>
<td>SUM_RANGE_STRICTLY DECREASING_SEQUENCE</td>
<td>2828</td>
</tr>
<tr>
<td>SUM_RANGE_STRICTLY INCREASING_SEQUENCE</td>
<td>2832</td>
</tr>
<tr>
<td>SUM_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
<td>2836</td>
</tr>
<tr>
<td>SUM_SURF_DECREASING</td>
<td>2840</td>
</tr>
<tr>
<td>SUM_SURF_DECREASING_SEQUENCE</td>
<td>2846</td>
</tr>
<tr>
<td>SUM_SURF_DECREASING_TERRACE</td>
<td>2850</td>
</tr>
<tr>
<td>SUM_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>2854</td>
</tr>
<tr>
<td>SUM_SURF_GORGE</td>
<td>2858</td>
</tr>
<tr>
<td>SUM_SURF_INCREASING</td>
<td>2864</td>
</tr>
<tr>
<td>SUM_SURF_INCREASING_SEQUENCE</td>
<td>2870</td>
</tr>
<tr>
<td>SUM_SURF_INCREASING_TERRACE</td>
<td>2874</td>
</tr>
<tr>
<td>SUM_SURF_INFLEXION</td>
<td>2878</td>
</tr>
<tr>
<td>SUM_SURF_PEAk</td>
<td>2882</td>
</tr>
<tr>
<td>SUM_SURF_PLAIN</td>
<td>2886</td>
</tr>
<tr>
<td>SUM_SURF_PLATEAU</td>
<td>2890</td>
</tr>
<tr>
<td>SUM_SURF_PROPER_PLAIN</td>
<td>2894</td>
</tr>
<tr>
<td>SUM_SURF_PROPER_PLATEAU</td>
<td>2898</td>
</tr>
<tr>
<td>SUM_SURF_STEADY</td>
<td>2902</td>
</tr>
<tr>
<td>SUM_SURF_STEADY_SEQUENCE</td>
<td>2906</td>
</tr>
<tr>
<td>SUM_SURF_STRICTLY DECREASING_SEQUENCE</td>
<td>2910</td>
</tr>
<tr>
<td>SUM_SURF_STRICTLY INCREASING_SEQUENCE</td>
<td>2914</td>
</tr>
<tr>
<td>SUM_SURF_SUMMIT</td>
<td>2918</td>
</tr>
<tr>
<td>SUM_SURF_VALLEY</td>
<td>2924</td>
</tr>
<tr>
<td>SUM_SURF_ZIGZAG</td>
<td>2928</td>
</tr>
<tr>
<td>SUM_WIDTH_DECREASING_SEQUENCE</td>
<td>2936</td>
</tr>
<tr>
<td>SUM_WIDTH_DECREASING_TERRACE</td>
<td>2942</td>
</tr>
<tr>
<td>SUM_WIDTH_GORGE</td>
<td>2948</td>
</tr>
<tr>
<td>SUM_WIDTH_INCREASING_SEQUENCE</td>
<td>2956</td>
</tr>
<tr>
<td>SUM_WIDTH_INCREASING_TERRACE</td>
<td>2962</td>
</tr>
<tr>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>SUM_WIDTH_INFLEXION</td>
<td>2968</td>
</tr>
<tr>
<td>SUM_WIDTH_PEAK</td>
<td>2974</td>
</tr>
<tr>
<td>SUM_WIDTH_PLAIN</td>
<td>2980</td>
</tr>
<tr>
<td>SUM_WIDTH_PLATEAU</td>
<td>2986</td>
</tr>
<tr>
<td>SUM_WIDTH_PROPER_PLAIN</td>
<td>2992</td>
</tr>
<tr>
<td>SUM_WIDTH_PROPER_PLATEAU</td>
<td>2998</td>
</tr>
<tr>
<td>SUM_WIDTH_STEADY_SEQUENCE</td>
<td>3004</td>
</tr>
<tr>
<td>SUM_WIDTH.Strictly Decreasing_sequence</td>
<td>3010</td>
</tr>
<tr>
<td>SUM_WIDTH.Strictly Increasing_sequence</td>
<td>3016</td>
</tr>
<tr>
<td>SUM_WIDTH.FAILURE</td>
<td>3022</td>
</tr>
<tr>
<td>SUM_WIDTH.PLAIN</td>
<td>3028</td>
</tr>
<tr>
<td>SUM_WIDTH.PLAIN</td>
<td>3030</td>
</tr>
<tr>
<td>SURF_BUMP_ON_INCREASING_SEQUENCE</td>
<td>3034</td>
</tr>
<tr>
<td>SURF_DECREASING</td>
<td>3040</td>
</tr>
<tr>
<td>SURF_DECREASING.Sequence</td>
<td>3054</td>
</tr>
<tr>
<td>SURF_DECREASING.TERRACE</td>
<td>3058</td>
</tr>
<tr>
<td>SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>3062</td>
</tr>
<tr>
<td>SURF_GORGE</td>
<td>3066</td>
</tr>
<tr>
<td>SURF_INCREMENT</td>
<td>3070</td>
</tr>
<tr>
<td>SURF_INCREMENT.Sequence</td>
<td>3074</td>
</tr>
<tr>
<td>SURF_INCREMENT.TERRACE</td>
<td>3078</td>
</tr>
<tr>
<td>SURF_INFLEXION</td>
<td>3082</td>
</tr>
<tr>
<td>SURF_PEAK</td>
<td>3086</td>
</tr>
<tr>
<td>SURF_PLAIN</td>
<td>3090</td>
</tr>
<tr>
<td>SURF_PLATEAU</td>
<td>3094</td>
</tr>
<tr>
<td>SURF_PROPER.PLAIN</td>
<td>3098</td>
</tr>
<tr>
<td>SURF_PROPER.PLATEAU</td>
<td>3102</td>
</tr>
<tr>
<td>SURF_STEADY</td>
<td>3106</td>
</tr>
<tr>
<td>SURF_STEADY.SEQUENCE</td>
<td>3110</td>
</tr>
<tr>
<td>SURF.Strictly.Decreasing.Sequence</td>
<td>3114</td>
</tr>
<tr>
<td>SURF.Strictly.Increasing.Sequence</td>
<td>3118</td>
</tr>
<tr>
<td>SURF_SUMMIT</td>
<td>3122</td>
</tr>
<tr>
<td>SURF_VALLEY</td>
<td>3126</td>
</tr>
<tr>
<td>SURF_ZIGZAG</td>
<td>3130</td>
</tr>
<tr>
<td>WIDTH.Decreasing.Sequence</td>
<td>3134</td>
</tr>
<tr>
<td>WIDTH.Decreasing.TERRACE</td>
<td>3138</td>
</tr>
<tr>
<td>WIDTH.GORGE</td>
<td>3142</td>
</tr>
<tr>
<td>WIDTH.Increasing.Sequence</td>
<td>3146</td>
</tr>
<tr>
<td>WIDTH.Increasing.TERRACE</td>
<td>3150</td>
</tr>
<tr>
<td>WIDTH.Inflection</td>
<td>3154</td>
</tr>
<tr>
<td>WIDTH_PEAK</td>
<td>3158</td>
</tr>
<tr>
<td>WIDTH.PLAIN</td>
<td>3162</td>
</tr>
<tr>
<td>WIDTH.PLATEAU</td>
<td>3166</td>
</tr>
</tbody>
</table>
4. GLOBAL CONSTRAINT CATALOGUE

WIDTH_PROPER.PLAIN    ........................................ 3170
WIDTH_PROPER.PLATEAU  ........................................ 3174
WIDTH_STEADY.SEQUENCE ........................................ 3178
WIDTH_STRICTLY_DECREASING_SEQUENCE  ................. 3182
WIDTH_STRICTLY_INCREASING_SEQUENCE .......... 3186
WIDTH_SUMMIT  .................................................. 3190
WIDTH_VALLEY  ................................................... 3194
WIDTH_ZIGZAG  .................................................... 3198
**Description**

**Origin**
Based on the **DECREASING_TERRACE** pattern.

**Constraint**

\[ \text{ALL\_EQUAL\_HEIGHT\_DECREASING\_TERRACE(VARIABLES)} \]

**Argument**

\[ \text{VARIABLES} : \text{collection(var-dvar)} \]

**Restriction**

\[ \text{required(VARIABLES,var)} \]

**Purpose**

Succeeds if the minima of the values in each occurrence of the **DECREASING_TERRACE** pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern **DECREASING_TERRACE** is the maximal subsequence which matches the regular expression ‘\(> = + >\)’.

Assume that the occurrence of the pattern **DECREASING_TERRACE** starts at position \(i\) and ends at position \(j\). The feature \(\text{MIN}\), called \(\text{HEIGHT}\) in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[ (6, 3, 3, 1, 5, 4, 3, 3, 2, 4, 4, 5, 3, 3, 1, 1) \]

Figure 4.1 provides an example where the **ALL\_EQUAL\_HEIGHT\_DECREASING\_TERRACE** \((6, 3, 3, 1, 5, 4, 3, 3, 2, 4, 4, 5, 3, 3, 1, 1)\) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(|\text{VARIABLES}.var|) > 2 \]
Figure 4.1: Illustrating the ALL_EQUAL_HEIGHT_DECREASING_TERRACE constraint of the Example slot
Figure 4.2 depicts the automaton associated with the constraint ALL_EQUAL_HEIGHT_DECREASING_TERRACE.

$$\begin{align*}
C &\leftarrow X \\
D &\leftarrow +\infty \\
F &\leftarrow X \\
R &\leftarrow 1 \\
&\{D \leftarrow \min(D, \text{VAR}_i)\}
\end{align*}$$

Figure 4.2: Automaton for the ALL_EQUAL_HEIGHT_DECREASING_TERRACE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING_TERRACE pattern
**ALL_EQUAL_HEIGHT_INCREASING_TERRACE**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ALL_EQUAL_HEIGHT_INCREASING_TERRACE</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>▶ ▷ ◄</td>
</tr>
</tbody>
</table>

**Origin**
Based on the `INCREASING_TERRACE` pattern.

**Constraint**

```
ALL_EQUAL_HEIGHT_INCREASING_TERRACE(VARIABLES)
```

**Argument**

```
VARIABLES : collection(var-dvar)
```

**Restriction**

```
required(VARIABLES, var)
```

Succeeds if the minima of the values in each occurrence of the `INCREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection are all the same.

An occurrence of the pattern `INCREASING_TERRACE` is the maximal subsequence which matches the regular expression `'<=+<'`.

**Purpose**
Assume that the occurrence of the pattern `INCREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `MIN`, called `HEIGHT` in the name of the constraint since all feature values are identical, computes the minimum of the values from index `i+1` to index `j`.

**Example**

```
((1, 4, 4, 6, 2, 3, 4, 4, 5, 3, 2, 4, 4, 6, 6))
```

Figure 4.3 provides an example where the `ALL_EQUAL_HEIGHT_INCREASING_TERRACE ([1, 4, 4, 6, 2, 3, 4, 4, 5, 3, 2, 4, 4, 6, 6])` constraint holds.

**Typical**

```
|VARIABLES| > 3
range(VARIABLES.var) > 2
```


Figure 4.3: Illustrating the ALL_EQUAL_HEIGHT_INCREASING_TERRACE constraint of the Example slot
Figure 4.4 depicts the automaton associated with the constraint `ALL_EQUAL_HEIGHT_INCREASING_TERRACE`.

Figure 4.4: Automaton for the `ALL_EQUAL_HEIGHT_INCREASING_TERRACE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING_TERRACE pattern.
### ALL_EQUAL_HEIGHTPlain

**Condition**

- **Feature:** ALL_EQUAL_HEIGHTPlain

**Pattern**

- ▶
  - ⊿
  - ◀

#### Description Automaton

- **Origin:** Based on the PLAIN pattern.
- **Constraint:** ALL_EQUAL_HEIGHTPlain(VARIABLES)
- **Argument:** VARIABLES : collection(var−dvar)
- **Restriction:** required(VARIABLES.var)

**Purpose**

Succeeds if the minima of the values in each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘\textgreater = \textless’.

Assume that the occurrence of the pattern PLAIN starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i+1$ to index $j$.

**Example**

```plaintext
((1,6,3,3,7,6,6,3,3,7,5,4,3,3,6,3))
```

Figure 4.5 provides an example where the ALL_EQUAL_HEIGHTPlain ($(1,6,3,3,7,6,6,3,3,7,5,4,3,3,6,3))$ constraint holds.

**Typical**

- $\mid$VARIABLES$\mid > 2$
- $\text{range}(\text{VARIABLES.var}) > 1$
Figure 4.5: Illustrating the ALL_EQUAL_HEIGHT_PLAIN constraint of the Example slot.
Figure 4.6 depicts the automaton associated with the constraint ALL_EQUAL_HEIGHTPLAIN.

\[ \begin{align*}
\text{If } R \land (F = C) \leq s, & \quad \Rightarrow (C \leftarrow X, D \leftarrow +\infty, F \leftarrow X, R \leftarrow 1) \\
\text{If } D \leftarrow \min(D, \text{VAR}_i), & \quad \Rightarrow (C \leftarrow \min(D, \text{VAR}_i), D \leftarrow +\infty, F \leftarrow \min(D, \text{VAR}_i), R \leftarrow R \land (F = \min(D, \text{VAR}_i))}
\end{align*} \]

Figure 4.6: Automaton for the ALL_EQUAL_HEIGHTPLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLAIN pattern.
Description

**Origin**
Based on the PLATEAU pattern.

**Constraint**

\[
\text{ALL\_EQUAL\_HEIGHT\_PLATEAU}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES} . \text{var})
\]

Succeeds if the minima of the values in each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**

An occurrence of the pattern PLATEAU is the *maximal* subsequence which matches the regular expression ‘\(<=\)’.

Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature \(\text{MIN}\), called \(\text{HEIGHT}\) in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
((7, 2, 5, 5, 1, 2, 2, 5, 3, 3, 4, 5, 5, 2, 5))
\]

Figure 4.7 provides an example where the ALL\_EQUAL\_HEIGHT\_PLATEAU \(((7, 2, 5, 5, 1, 2, 2, 5, 3, 3, 4, 5, 5, 2, 5))\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2
\]

\[
\text{range}(\text{VARIABLES} . \text{var}) > 1
\]
Figure 4.7: Illustrating the ALL_EQUAL_HEIGHT_PLATEAU constraint of the Example slot.
Automaton

Figure 4.8 depicts the automaton associated with the constraint ALL_EQUAL_HEIGHT_PLATEAU.

Figure 4.8: Automaton for the ALL_EQUAL_HEIGHT_PLATEAU constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLATEAU pattern
Origin: Based on the PROPER.PLAIN pattern.

Constraint: \( \text{ALL\_EQUAL\_HEIGHT\_PROPER\_PLAIN}(\text{VARIABLES}) \)

Argument: \( \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \)

Restriction: \( \text{required}(\text{VARIABLES} . \text{var}) \)

Purpose: Succeeds if the minima of the values in each occurrence of the \text{PROPER\_PLAIN} pattern in the time-series given by the \text{VARIABLES} collection are all the same.

An occurrence of the pattern \text{PROPER\_PLAIN} is the maximal subsequence which matches the regular expression \( \text{'}> = \text{'}+ <\text{'} \).

Assume that the occurrence of the pattern \text{PROPER\_PLAIN} starts at position \( i \) and ends at position \( j \). The feature \text{MIN}, called \text{HEIGHT} in the name of the constraint since all feature values are identical, computes the minimum of the values from index \( i + 1 \) to index \( j \).

Example: \( ((2, 7, 3, 6, 6, 3, 3, 3, 5, 6, 5, 3, 3, 5)) \)

Figure 4.9 provides an example where the \text{ALL\_EQUAL\_HEIGHT\_PROPER\_PLAIN} \( ([2, 7, 3, 6, 6, 3, 3, 3, 5, 6, 5, 3, 3, 5]) \) constraint holds.

Typical: \( |\text{VARIABLES}| > 3 \)
\( \text{range}(\text{VARIABLES} . \text{var}) > 1 \)
Figure 4.9: Illustrating the ALL_EQUAL_HEIGHT_PROPER_PLAIN constraint of the Example slot
Automaton

Figure 4.10 depicts the automaton associated with the constraint ALL_EQUAL_HEIGHT_PROPER_PLAIN.

\[
\begin{align*}
\text{Figure 4.10: Automaton for the ALL_EQUAL_HEIGHT_PROPER_PLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PROPER_PLAIN pattern}
\end{align*}
\]
ALL_EQUAL_HEIGHT_PROPER_PLATEAU

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_HEIGHT_PROPER_PLATEAU</td>
<td>ALL_EQUAL_HEIGHT_PROPER_PLATEAU(VARIABLES)</td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td>required(VARIABLES.var)</td>
<td>Succeeds if the minima of the values in each occurrence of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection are all the same. An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression '&lt;=+'.' Purpose</td>
<td></td>
</tr>
<tr>
<td>Assume that the occurrence of the pattern PROPER_PLATEAU starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i + 1$ to index $j$.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>((7, 1, 5, 5, 2, 2, 5, 1, 5, 5, 3, 2, 3, 5, 5, 3))</td>
<td>Figure 4.11 provides an example where the ALL_EQUAL_HEIGHT_PROPER_PLATEAU ((7, 1, 5, 5, 2, 2, 5, 1, 5, 5, 3, 2, 3, 5, 5, 3)) constraint holds.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
<td>$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\text{range}(\text{VARIABLES.var}) &gt; 1$</td>
</tr>
</tbody>
</table>
Figure 4.11: Illustrating the \texttt{ALL\_EQUAL\_HEIGHT\_PROPER\_PLATEAU} constraint of the \texttt{Example} slot
Automaton  

Figure 4.12 depicts the automaton associated with the constraint `ALL_EQUAL_HEIGHT_PROPER_PLATEAU`.

Figure 4.12: Automaton for the `ALL_EQUAL_HEIGHT_PROPER_PLATEAU` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `PROPER_PLATEAU` pattern.
### Description

**Origin**
Based on the STEADY pattern.

**Constraint**

```
ALL_EQUAL_HEIGHT_STEADY(VARIABLES)
```

**Argument**

```
VARIABLES : collection(var−dvar)
```

**Restriction**

```
required(VARIABLES,var)
```

**Purpose**
Succeeds if the minima of the values in each occurrence of the STEADY pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern STEADY is the subsequence which matches the regular expression '='.

Assume that the occurrence of the pattern STEADY starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

```
((4,4,4,2,6,4,4,4,3,2,1,4,4,3,4))
```

Figure 4.13 provides an example where the ALL_EQUAL_HEIGHT_STEADY ((4,4,4,2,6,4,4,4,3,2,1,4,4,3,4)) constraint holds.

**Typical**

```
|VARIABLES| > 1
```
Figure 4.13: Illustrating the ALL_EQUAL_HEIGHT_STEADY constraint of the Example slot
Figure 4.14 depicts the automaton associated with the constraint \textsc{ALL\_EQUAL\_HEIGHT\_STEADY}.

\begin{align*}
C & \leftarrow X \\
D & \leftarrow +\infty \\
F & \leftarrow X \\
R & \leftarrow 1 \\
\end{align*}

\[ R \land (F = C) \]

Figure 4.14: Automaton for the \textsc{ALL\_EQUAL\_HEIGHT\_STEADY} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{STEADY} pattern.
ALL_EQUAL_HEIGHT_STEADY_SEQUENCE

- **Origin**: Based on the STEADY_SEQUENCE pattern.
- **Constraint**: ALL_EQUAL_HEIGHT_STEADY_SEQUENCE(VARIABLES)
- **Argument**: VARIABLES : collection(var−dvar)
- **Restriction**: required(VARIABLES.var)
- **Purpose**: Succeeds if the minima of the values in each occurrence of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection are all the same. An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression ‘=+‘.
- **Purpose**: Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i$ to index $j + 1$.
- **Example**: ($4, 3, 5, 5, 3, 1, 5, 5, 5, 6, 5, 5, 5, 5, 5, 3, 5$)
- **Typical**: $|\text{VARIABLES}| > 1$

Figure 4.15 provides an example where the ALL_EQUAL_HEIGHT_STEADY_SEQUENCE ($(4, 3, 5, 5, 3, 1, 5, 5, 6, 5, 5, 5, 5, 3, 5)$) constraint holds.
Figure 4.15: Illustrating the ALL_EQUAL_HEIGHT_STEADY_SEQUENCE constraint of the Example slot
Automaton

Figure 4.16 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_HEIGHT_STEADY_SEQUENCE}.

\[
\begin{align*}
C & \leftarrow X \\
D & \leftarrow +\infty \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}
\]

Figure 4.16: Automaton for the \texttt{ALL_EQUAL_HEIGHT_STEADY_SEQUENCE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{STEADY_SEQUENCE} pattern.
ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE</td>
<td>&gt; &gt; &lt; &gt; &gt;</td>
<td></td>
</tr>
</tbody>
</table>

**Origin**
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**
ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the maxima of the values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**
An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence ‘>><<>’.
Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 2 to index j.

**Example**
((7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 6, 5, 4, 4))

Figure 4.17 provides an example where the ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE ((7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 6, 5, 4, 4)) constraint holds.

**Typical**
| VARIABLES | > 5 |
| range(VARIABLES.var) | > 2 |
Figure 4.17: Illustrating the **ALL_EQUAL_MAX_BUMP_ON DECREASING_SEQUENCE** constraint of the **Example** slot
Figure 4.18 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE}.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{automaton.png}
\caption{Automaton for the \texttt{ALL\_EQUAL\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} pattern}
\end{figure}
**ALL_EQUAL_MAX_DECREASING**

**Origin**
Based on the **DECREASING** pattern.

**Constraint**
ALL_EQUAL_MAX_DECREASING(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the maxima of the values in each occurrence of the **DECREASING** pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression ‘>’.

Assume that the occurrence of the pattern **DECREASING** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**
\[\langle 1, 6, 6, 1, 1, 4, 6, 6, 4, 4, 4, 5, 5, 6, 1, 5 \rangle\]

Figure 4.19 provides an example where the **ALL_EQUAL_MAX_DECREASING** constraint holds.

**Typical**
|VARIABLES| > 1
|range(VARIABLES.var)| > 1
Figure 4.19: Illustrating the ALL_EQUAL_MAX_DECREASING constraint of the Example slot
Figure 4.20 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_MAX\_DECREASING}.

![Automaton Diagram](image)

Figure 4.20: Automaton for the \texttt{ALL\_EQUAL\_MAX\_DECREASING} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{DECREASING} pattern.
ALL_EQUAL_MAX_DECREASING_SEQUENCE

### Description

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**

\[
\text{ALL_EQUAL_MAX_DECREASING_SEQUENCE}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the maxima of the values in each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\((> | =)^* > | >\)’.

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
(1, 6, 6, 1, 1, 4, 6, 6, 4, 4, 2, 1, 3, 6, 1, 5)
\]

Figure 4.21 provides an example where the ALL_EQUAL_MAX_DECREASING_SEQUENCE (\([1, 6, 6, 1, 1, 4, 6, 6, 4, 4, 2, 1, 3, 6, 1, 5]\)) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.21: Illustrating the ALL_EQUAL_MAX_DECREASING_SEQUENCE constraint of the Example slot
Figure 4.22 depicts the automaton associated with the constraint `ALL_EQUAL_MAX_DECREASING_SEQUENCE`.

\[
\begin{align*}
C &\leftarrow X \\
D &\leftarrow -\infty \\
F &\leftarrow X \\
R &\leftarrow 1 \\
\end{align*}
\]

Figure 4.22: Automaton for the `ALL_EQUAL_MAX_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `DECREASING_SEQUENCE` pattern.
<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE</strong></td>
<td>▶ ▶ ◀ ◀</td>
<td>(\ll&lt;\gg\ll&lt;\gg)</td>
</tr>
</tbody>
</table>

**Description Automaton**

**Origin**
Based on the **DIP_ON_INCREASING_SEQUENCE** pattern.

**Constraint**
\[
\text{ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE(VARIABLES)}
\]

**Argument**
\[
\text{VARIABLES : collection(var−dvar)}
\]

**Restriction**
\[
\text{required(VARIABLES.var)}
\]

**Purpose**
Succeeds if the maxima of the values in each occurrence of the **DIP_ON_INCREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection are all the same.

An occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** is the subsequence which matches the regular expression ‘\(\ll<\gg\ll<\gg\)’.

Assume that the occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i + 2\) to index \(j\).

**Example**
\[
((0,1,2,1,5,6,7,3,0,2,3,5,0,2,3,3))
\]

Figure 4.23 provides an example where the **ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE** constraint holds.

**Typical**
\[
\text{\(|VARIABLES| > 5\)} \quad \text{range(VARIABLES.var) > 2}
\]
Figure 4.23: Illustrating the \texttt{ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE} constraint of the \texttt{Example} slot
Figure 4.24 depicts the automaton associated with the constraint ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE.

Figure 4.24: Automaton for the ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern.
Origin

Based on the increasing pattern.

Constraint

\text{ALL\_EQUAL\_MAX\_INCREASING}(\text{VARIABLES})

Argument

\text{VARIABLES} : \text{collection(var\_dvar)}

Restriction

\text{required}(\text{VARIABLES}, \text{var})

Purpose

Succeeds if the maxima of the values in each occurrence of the increasing pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern increasing is the subsequence which matches the regular expression ‘<’.

Assume that the occurrence of the pattern increasing starts at position \(i\) and ends at position \(j\). The feature \text{MAX} computes the maximum of the values from index \(i\) to index \(j + 1\).

Example

\((6, 1, 1, 6, 6, 6, 6, 5, 3, 6, 4, 1, 6, 2)\)

Figure 4.25 provides an example where the ALL\_EQUAL\_MAX\_INCREASING (\([6, 1, 1, 6, 6, 6, 5, 3, 6, 4, 1, 6, 2]\)) constraint holds.

Typical

|VARIABLES| > 1
range(VARIABLES.var) > 1
Figure 4.25: Illustrating the ALL_EQUAL_MAX_INCREASING constraint of the Example slot
Figure 4.26 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_MAX_INCREASING}.

\[
\begin{align*}
C & \leftarrow X \\
D & \leftarrow -\infty \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow \max(D, \text{VAR}_i), \text{VAR}_{i+1}) < \\
D & \leftarrow -\infty \\
F & \leftarrow \max(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
R & \leftarrow R \land (F = \max(D, \text{VAR}_i), \text{VAR}_{i+1}))
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow X \\
D & \leftarrow -\infty \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}
\]

\[
R \land (F = C)
\]

Figure 4.26: Automaton for the \texttt{ALL_EQUAL_MAX_INCREASING} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{INCREASING} pattern.
ALL_EQUAL_MAX_INCREASING_SEQUENCE

**Description**

Origin: Based on the INCREASING_SEQUENCE pattern.

Constraint: ALL_EQUAL_MAX_INCREASING_SEQUENCE(VARIABLES)

Argument: VARIABLES : collection(var−dvar)

Restriction: required(VARIABLES.var)

Purpose: Succeeds if the maxima of the values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are all the same. An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘<( < | = )* < | <’ . Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i to index j+1.

Example: ((6,1,1,6,6,3,1,1,3,3,5,6,4,1,6,2))

Figure 4.27 provides an example where the ALL_EQUAL_MAX_INCREASING_SEQUENCE ((6,1,1,6,6,3,1,1,3,3,5,6,4,1,6,2)) constraint holds.

Typical: |

| VARIABLES | > 1 |

| range(VARIABLES.var) | > 1 |
Figure 4.27: Illustrating the ALL_EQUAL_MAX_INCREASING_SEQUENCE constraint of the Example slot
Figure 4.28 depicts the automaton associated with the constraint `ALL_EQUAL_MAX_INCREASING_SEQUENCE`.

\[
\begin{cases}
    C \leftarrow X \\
    D \leftarrow -\infty \\
    F \leftarrow X \\
    R \leftarrow 1
\end{cases}
\]

Figure 4.28: Automaton for the `ALL_EQUAL_MAX_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `INCREASING_SEQUENCE` pattern.
**ALL_EQUAL_MAX_INFLEXION**

### Description Automaton

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ALL_EQUAL_MAX_INFLEXION</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Origin
Based on the **INFLEXION** pattern.

#### Constraint

$$\text{ALL_EQUAL_MAX_INFLEXION}(< (< | =) | > | (> | =)^* | < )$$

#### Argument

$$\text{VARIABLES : collection(var - dvar)}$$

#### Restriction

$$\text{required(VARIABLES, var)}$$

#### Purpose
Succeeds if the maxima of the values in each occurrence of the **INFLEXION** pattern in the time-series given by the **VARIABLES** collection are all the same.

An occurrence of the pattern **INFLEXION** is the maximal subsequence which matches the regular expression ‘< (\(< | =) | > | (> | =)^* | < ’.

Assume that the occurrence of the pattern **INFLEXION** starts at position *i* and ends at position *j*. The feature **MAX** computes the maximum of the values from index *i* + 1 to index *j*.

#### Example

$$((5, 5, 4, 4, 3, 2, 2, 3, 3, 4, 4, 2, 2))$$

Figure 4.29 provides an example where the **ALL_EQUAL_MAX_INFLEXION** (**[5, 5, 4, 4, 3, 2, 2, 3, 3, 4, 4, 2, 2]**) constraint holds.

![Diagram](image-url)

Figure 4.29: Illustrating the **ALL_EQUAL_MAX_INFLEXION** constraint of the **Example** slot.
Typical

\[ |\text{VARIABLES}| > 2 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Automaton

Figure 4.30 depicts the automaton associated with the constraint \textsc{ALL\_EQUAL\_MAX\_INFLEXION}.

\hspace{1cm}

\begin{align*}
\begin{cases}
C & \leftarrow X \\
D & \leftarrow -\infty \\
F & \leftarrow X \\
R & \leftarrow 1
\end{cases}
\end{align*}

\begin{align*}
\leq & \hspace{1cm} \leq \\
\{D \leftarrow \max(D, \text{VAR}_i)\} & \hspace{1cm} \{D \leftarrow \max(D, \text{VAR}_i)\}
\end{align*}

\begin{align*}
\begin{cases}
C & \leftarrow \max(D, \text{VAR}_i) \\
D & \leftarrow -\infty \\
F & \leftarrow \max(D, \text{VAR}_i) \\
R & \leftarrow R \land (F = \max(D, \text{VAR}_i))
\end{cases}
\end{align*}

\begin{figure}
\centering
\includegraphics[width=\textwidth]{figure4.30.png}
\caption{Automaton for the \textsc{ALL\_EQUAL\_MAX\_INFLEXION} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{INFLEXION} pattern (transition \(r \rightarrow t\) has the same registers updates as transition \(t \rightarrow r\))}
\end{figure}
Based on the \textsc{peak} pattern.

\textbf{Constraint:} \textsc{all\_equal\_max\_peak}(\textsc{variables})

\textbf{Argument:} \textsc{variables} : \texttt{collection(var\_dvar)}

\textbf{Restriction:} \texttt{required(\textsc{variables}.var)}

Succeeds if the maxima of the values in each occurrence of the \textsc{peak} pattern in the time-series given by the \textsc{variables} collection are all the same.

\textbf{Purpose:}

An occurrence of the pattern \textsc{peak} is the maximal subsequence which matches the regular expression '=< (= | <)'(> | =)* >'.

Assume that the occurrence of the pattern \textsc{peak} starts at position \(i\) and ends at position \(j\). The feature \textsc{max} computes the maximum of the values from index \(i+1\) to index \(j\).

\textbf{Example:} \((\langle 1, 2, 2, 7, 7, 1, 2, 4, 5, 7, 1, 2, 3, 6, 7, 1 \rangle)\)

Figure 4.31 provides an example where the \textsc{all\_equal\_max\_peak} \((\{1, 2, 2, 7, 7, 1, 2, 4, 5, 7, 1, 2, 3, 6, 7, 1\})\) constraint holds.

\textbf{Typical:}

\(|\textsc{variables}| > 2\)

\texttt{range(\textsc{variables}.var) > 1}
Figure 4.31: Illustrating the \textit{ALL\_EQUAL\_MAX\_PEAK} constraint of the \textbf{Example} slot
Figure 4.32 depicts the automaton associated with the constraint \textsc{ALL\_EQUAL\_MAX\_PEAK}.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure4_32.png}
\caption{Automaton for the \textsc{ALL\_EQUAL\_MAX\_PEAK} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{PEAK} pattern}
\end{figure}
**ALL_EQUAL_MAX.StrictlyDecreasing_SEQUENCE**

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_MAX.StrictlyDecreasing_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Origins**: Based on the **StrictlyDecreasing_SEQUENCE** pattern.

**Constraint**: ALL_EQUAL_MAX.StrictlyDecreasing_SEQUENCE(VARIABLES)

**Argument**: VARIABLES : collection(var−dvar)

**Restriction**: required(VARIABLES.var)

Succeeds if the maxima of the values in each occurrence of the **StrictlyDecreasing_SEQUENCE** pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**: An occurrence of the pattern **StrictlyDecreasing_SEQUENCE** is the maximal sub-sequence which matches the regular expression ‘>’.
Assume that the occurrence of the pattern **StrictlyDecreasing_SEQUENCE** starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i to index j + 1.

**Example**: (⟨5,5,6,3,1,2,4,4,6,6,4,1,1,6,6,1⟩)

Figure 4.33 provides an example where the ALL_EQUAL_MAX.StrictlyDecreasing_SEQUENCE ([5,5,6,3,1,2,4,4,6,6,4,1,1,6,6,1]) constraint holds.

**Typical**

| VARIABLES | > 1 |
| range(VARIABLES.var) | > 1 |
Figure 4.33: Illustrating the `ALL_EQUAL_MAX_STRICTLY_DECREASING_SEQUENCE` constraint of the `Example` slot.
Figure 4.34 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_MAX\_STRICTLY\_DECREASING\_SEQUENCE}.

\[
\begin{align*}
C & \leftarrow X \\
D & \leftarrow -\infty \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}
\]

Figure 4.34: Automaton for the \texttt{ALL\_EQUAL\_MAX\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern.
ALL_EQUAL_MAX_STRICTLY_INCREASING_SEQUENCE

Origin: Based on the STRICTLY_INCREASING_SEQUENCE pattern.

Constraint: ALL_EQUAL_MAX_STRICTLY_INCREASING_SEQUENCE(VARIABLES)

Argument: VARIABLES : collection(var \(-\) dvar)

Restriction: required(VARIABLES.var)

Purpose: Succeeds if the maxima of the values in each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression '<+'.

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i to index j + 1.

Example: ((6, 1, 1, 6, 6, 3, 1, 1, 3, 4, 5, 6, 4, 1, 6, 2))

Figure 4.35 provides an example where the ALL_EQUAL_MAX_STRICTLY_INCREASING_SEQUENCE ((6, 1, 1, 6, 6, 3, 1, 1, 3, 4, 5, 6, 4, 1, 6, 2)) constraint holds.

Typical: |

| VARIABLES | > 1 |

| range(VARIABLES.var) | > 1 |
Figure 4.35: Illustrating the **ALL_EQUAL_MAX_STRICTLY_INCREASING_SEQUENCE** constraint of the **Example** slot
Figure 4.36 depicts the automaton associated with the constraint `ALL_EQUAL_MAX_STRICTLY_INCREASING_SEQUENCE`.

\[
\begin{align*}
C & \leftarrow X \\
D & \leftarrow -\infty \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}
\]

Figure 4.36: Automaton for the `ALL_EQUAL_MAX_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern.
**ALL_EQUAL_MAX_SUMMIT**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL</td>
<td>EQUAL</td>
<td>MAX</td>
</tr>
<tr>
<td>SUMMIT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DESCRIPTION AUTOMATON**

- **Origin**
  - Based on the **SUMMIT** pattern.

- **Constraint**
  - `ALL_EQUAL_MAX_SUMMIT(VARIABLES)`

- **Argument**
  - `VARIABLES : collection(var-dvar)`

- **Restriction**
  - `required(VARIABLES, var)`

  Succeeds if the maxima of the values in each occurrence of the **SUMMIT** pattern in the time-series given by the `VARIABLES` collection are all the same.

- **Purpose**
  - An occurrence of the pattern **SUMMIT** is the *maximal* subsequence which matches the regular expression `'(< | < (= | <)* <>) | > (= | >)* >)`. Assume that the occurrence of the pattern **SUMMIT** starts at position `i` and ends at position `j`. The feature MAX computes the maximum of the values from index `i + 1` to index `j`.

- **Example**
  - `((1,3,5,4,2,6,6,2,3,5,4,1,4,5,3,2))`

  Figure 4.37 provides an example where the **ALL_EQUAL_MAX_SUMMIT** `([1,3,5,4,2,6,6,2,3,5,4,1,4,5,3,2])` constraint holds.

- **Typical**
  - `|VARIABLES| > 2`
  - `range(VARIABLES, var) > 1`
Figure 4.37: Illustrating the ALL_EQUAL_MAX_SUMMIT constraint of the Example slot
Figure 4.38 depicts the automaton associated with the constraint `ALL_EQUAL_MAX_SUMMIT`.

Figure 4.38: Automaton for the `ALL_EQUAL_MAX_SUMMIT` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `SUMMIT` pattern (transition `u \rightarrow r` has the same register update as transition `r \rightarrow u`.)
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_MAX_ZIGZAG</td>
<td>▶ ▶ ◄</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

- **Origin**: Based on the ZIGZAG pattern.
- **Constraint**: \( \text{ALL_EQUAL_MAX_ZIGZAG}(\text{VARIABLES}) \)
- **Argument**: \( \text{VARIABLES} : \text{collection}(\text{var-dvar}) \)
- **Restriction**: \( \text{required}(\text{VARIABLES}.\text{var}) \)

**Purpose**: Succeeds if the maxima of the values in each occurrence of the ZIGZAG pattern in the time-series given by the \( \text{VARIABLES} \) collection are all the same.

An occurrence of the pattern ZIGZAG is the \textit{maximal} subsequence which matches the regular expression ‘\((<>)^{+}(<>|<>)|(<>)^{+}(>|<)\)’. Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i + 1 \) to index \( j \).

**Example**: \( ((1,6,1,2,7,5,1,6,3,4,2,5,6,3,4,7)) \)

Figure 4.39 provides an example where the \( \text{ALL_EQUAL_MAX_ZIGZAG} \) \( ([1,6,1,2,7,5,1,6,3,4,2,5,6,3,4,7]) \) constraint holds.

**Typical**
- \( |\text{VARIABLES}| > 3 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)
Figure 4.39: Illustrating the **ALL_EQUAL_MAX_ZIGZAG** constraint of the **Example** slot
Automaton Figure 4.40 depicts the automaton associated with the constraint ALL_EQUAL_MAX_ZIGZAG.
Figure 4.40: Automaton for the ALL\_EQUAL\_MAX\_ZIGZAG constraint obtained by applying decoration Table 3.38 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the register D is reset to its initial value; (3) on transitions from c, f to s the register F is reset to C, and the register R is updated wrt C and F.
<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE</strong></td>
<td>⌈ ⌉</td>
<td>⌈ ⌉</td>
</tr>
</tbody>
</table>

**Description Automaton**

**Origin**
Based on the `BUMP_ON_DECREASING_SEQUENCE` pattern.

**Constraint**

```
ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)
```

**Argument**

```
VARIABLES : collection(var−dvar)
```

**Restriction**

```
required(VARIABLES, var)
```

Succeeds if the minima of the values in each occurrence of the `BUMP_ON_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are all the same.

**Purpose**

An occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` is the subsequence `’>>><’`. Assume that the occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `MIN` computes the minimum of the values from index $i + 2$ to index $j$.

**Example**

```
((7, 6, 5, 6, 2, 1, 1, 5, 4, 3, 2, 6, 5, 4, 4, 5))
```

Figure 4.41 provides an example where the `ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE` constraint holds.

**Typical**

```
|VARIABLES| > 5
range(VARIABLES, var) > 2
```
Figure 4.41: Illustrating the ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Figure 4.42 depicts the automaton associated with the constraint \(\text{ALL\_EQUAL\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE}\).

Figure 4.42: Automaton for the \(\text{ALL\_EQUAL\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE}\) constraint obtained by applying decoration Table 3.38 to the seed transducer of the \(\text{BUMP\_ON\_DECREASING\_SEQUENCE}\) pattern.
ALL_EQUAL_MIN_DECREASING

**Description**

**Origin**
Based on the **DECREASING** pattern.

**Constraint**

\[
\text{ALL_EQUAL_MIN_DECREASING}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

**Purpose**

Succeeds if the minima of the values in each occurrence of the **DECREASING** pattern in the time-series given by the **VARIABLES** collection are all the same.

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression ‘>’.

Assume that the occurrence of the pattern **DECREASING** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
((1, 6, 6, 1, 1, 4, 4, 5, 5, 5, 5, 1, 3, 6, 1, 5))
\]

Figure 4.43 provides an example where the **ALL_EQUAL_MIN_DECREASING** (\([1, 6, 6, 1, 1, 4, 4, 5, 5, 5, 5, 1, 3, 6, 1, 5]\)) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.43: Illustrating the ALL_EQUAL_MIN_DECREASING constraint of the Example slot
Automaton

Figure 4.44 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_MIN\_DECREASING}.

Figure 4.44: Automaton for the \texttt{ALL\_EQUAL\_MIN\_DECREASING} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{DECREASING} pattern.
ALL_EQUAL_MIN_DECREASING_SEQUENCE

**Description**

Based on the **DECREASING_SEQUENCE** pattern.

**Constraint**

ALL_EQUAL_MIN_DECREASING_SEQUENCE(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the **DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**

An occurrence of the pattern **DECREASING_SEQUENCE** is the maximal subsequence which matches the regular expression `>(> | =)* > | >`. Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position $i$ and ends at position $j$. The feature **MIN** computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

```
((1, 6, 6, 1, 1, 4, 5, 5, 4, 4, 2, 1, 3, 4, 1, 5))
```

Figure 4.45 provides an example where the **ALL_EQUAL_MIN_DECREASING_SEQUENCE** constraint holds.

**Typical**

- $|\text{VARIABLES}| > 1$
- $\text{range(VARIABLES.var)} > 1$
Figure 4.45: Illustrating the ALL_EQUAL_MIN_DECREASING_SEQUENCE constraint of the Example slot
Figure 4.46 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_MIN\_DECREASING\_SEQUENCE}.

\begin{align*}
&\{ C \leftarrow X \} \\
&\{ D \leftarrow +\infty \} \\
&\{ R \leftarrow 1 \}
\end{align*}

Figure 4.46: Automaton for the \texttt{ALL\_EQUAL\_MIN\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{DECREASING\_SEQUENCE} pattern
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Origin**  
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**  
ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE(VARIABLES)

**Argument**  
VARIABLES : collection(var−dvar)

**Restriction**  
required(VARIABLES.var)

Succeeds if the minima of the values in each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**  
An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence ‘<<><<’. Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i+2$ to index $j$.

**Example**  
\[
((1,2,3,2,4,7,3,4,5,6,2,3,4,4,3))
\]

Figure 4.47 provides an example where the ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE \((1,2,3,2,4,7,3,4,5,6,2,3,4,4,3))\) constraint holds.

**Typical**  
| VARIABLES | > 5  
| range(VARIABLES.var) | > 2  


Figure 4.47: Illustrating the **ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE** constraint of the **Example** slot
Figure 4.48 depicts the automaton associated with the constraint 
ALL_EQUAL_MIN_DIP_ON_INCREMENTING_SEQUENCE.

Automaton

Figure 4.48: Automaton for the ALL_EQUAL_MIN_DIP_ON_INCREMENTING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DIP_ON_INCREMENTING_SEQUENCE pattern.
Origin
Based on the GORGE pattern.

Constraint
ALL_EQUAL_MIN_GORGE(VARIABLES)

Argument
VARIABLES : collection(var,dvar)

Restriction
required(VARIABLES,var)

Succeeds if the minima of the values in each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are all the same.

Purpose
An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression "(>||(=|>)*)<|<(=|)<)". Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i+1\) to index \(j\).

Example
\(((6,4,2,3,5,1,1,5,4,2,3,6,3,2,4,5))\)

Figure 4.49 provides an example where the ALL_EQUAL_MIN_GORGE ([6,4,2,3,5,1,1,5,4,2,3,6,3,2,4,5]) constraint holds.

Typical
\[|\text{VARIABLES}| > 2\]
\[\text{range}(\text{VARIABLES}.var) > 1\]
Figure 4.49: Illustrating the ALL_EQUAL_MIN_GORGE constraint of the Example slot
Automaton

Figure 4.50 depicts the automaton associated with the constraint 
ALL_EQUAL_MIN_GORGE.

Figure 4.50: Automaton for the ALL_EQUAL_MIN_GORGE constraint obtained by applying decoration Table 3.38 to the seed transducer of the GORGE pattern (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
**ALL_EQUAL_MIN_INCREASING**

- **Origin**: Based on the **INCREASING** pattern.
- **Constraint**: `ALL_EQUAL_MIN_INCREASING(VARIABLES)`
- **Argument**: `VARIABLES : collection(var−dvar)`
- **Restriction**: `required(VARIABLES,var)`

**Description**

Succeeds if the minima of the values in each occurrence of the **INCREASING** pattern in the time-series given by the **VARIABLES** collection are all the same.

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression `'<`.

Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i\) to index \(j+1\).

**Example**

\(((7,2,2,6,6,3,2,2,7,4,4,3,3,2,5,1))\)

Figure 4.51 provides an example where the `ALL_EQUAL_MIN_INCREASING` \(((7,2,2,6,6,3,2,2,7,4,4,3,3,2,5,1))\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 1\)

\(|\text{range(VARIABLES.var)}| > 1\)
Figure 4.51: Illustrating the ALL_EQUAL_MIN_INCREASING constraint of the Example slot
Automaton

Figure 4.52 depicts the automaton associated with the constraint ALL_EQUAL_MIN_INCREASING.

\[
\begin{cases}
C \leftarrow X \\
D \leftarrow +\infty \\
F \leftarrow X \\
R \leftarrow 1
\end{cases}
\]

\[
R \land (F = C) \geq \left\{ \begin{array}{l}
C \leftarrow \min(min(D, VAR_i), VAR_{i+1}) \\
D \leftarrow +\infty \\
F \leftarrow \min(min(D, VAR_i), VAR_{i+1}) \\
R \leftarrow R \land (F = \min(min(D, VAR_i), VAR_{i+1}))
\end{array} \right.
\]

Figure 4.52: Automaton for the ALL_EQUAL_MIN_INCREASING constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING pattern.
ALL_EQUAL_MIN_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_MIN_INCREASING_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Description**

**Origin**

Based on the **INCREASING_SEQUENCE** pattern.

**Constraint**

```
ALL_EQUAL_MIN_INCREASING_SEQUENCE(VARIABLES)
```

**Argument**

```
VARIABLES : collection(var−dvar)
```

**Restriction**

```
required(VARIABLES, var)
```

Succeeds if the minima of the values in each occurrence of the **INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**

An occurrence of the pattern **INCREASING_SEQUENCE** is the **maximal** subsequence which matches the regular expression ‘\(< \mid =)\ast \mid < \mid’

Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature \(\text{MIN}\) computes the minimum of the values from index \(i\) to index \(j+1\).

**Example**

```
((7,2,2,6,6,3,2,2,3,3,5,6,4,2,5,1))
```

Figure 4.53 provides an example where the **ALL_EQUAL_MIN_INCREASING_SEQUENCE** \(((7,2,2,6,6,3,2,2,3,3,5,6,4,2,5,1))\) constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 4.53: Illustrating the **ALL_EQUAL_MIN_INCREASING_SEQUENCE** constraint of the **Example** slot
Figure 4.54 depicts the automaton associated with the constraint ALL_EQUAL_MIN_INCREASING_SEQUENCE.

\[
\begin{align*}
C & \leftarrow X \\
D & \leftarrow +\infty \\
F & \leftarrow X \\
R & \leftarrow 1 \\
\end{align*}
\]

Figure 4.54: Automaton for the ALL_EQUAL_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING_SEQUENCE pattern.
ALL_EQUAL_MIN_INCREASING_SEQUENCE
### ALL_EQUAL_MIN_INFLEXION

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_MIN_INFLEXION</td>
<td>▶▷ ◄</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

#### Origin
Based on the INFLEXION pattern.

#### Constraint
ALL_EQUAL_MIN_INFLEXION(VARIABLES)

#### Argument
VARIABLES : collection(var − dvar)

#### Restriction
required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression '< (< | =)* > | > (> | =)* '<.

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

#### Purpose

#### Example

\(((2, 2, 3, 6, 6, 5, 4, 3, 4, 4, 5))\)

Figure 4.55 provides an example where the ALL_EQUAL_MIN_INFLEXION ([2, 2, 3, 6, 6, 5, 4, 3, 4, 4, 5]) constraint holds.

#### Typical

- \(|\text{VARIABLES}| > 2\)
- range(VARIABLES.var) > 1
Figure 4.55: Illustrating the ALL_EQUAL_MIN_INFLEXION constraint of the Example slot
Figure 4.56 depicts the automaton associated with the constraint ALL_EQUAL_MIN_INFLEXION.

Figure 4.56: Automaton for the ALL_EQUAL_MIN_INFLEXION constraint obtained by applying decoration Table 3.38 to the seed transducer of the INFLEXION pattern (transition $r \rightarrow t$ has the same registers updates as transition $t \rightarrow r$)
### ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Description**

- **Origin**: Based on the **STRICTLY_DECREASING_SEQUENCE** pattern.
- **Constraint**: `ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE(VARIABLES)`
- **Argument**: `VARIABLES : collection(var−dvar)`
- **Restriction**: `required(VARIABLES.var)`

**Purpose**

Succeeds if the minima of the values in each occurrence of the **STRICTLY_DECREASING_SEQUENCE** pattern in the time-series given by the `VARIABLES` collection are all the same.

An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the *maximal* sub-sequence which matches the regular expression `'^>``.

Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position `i` and ends at position `j`. The feature **MIN** computes the minimum of the values from index `i` to index `j + 1`.

**Example**

```plaintext
((5, 1, 4, 3, 1, 2, 4, 4, 5, 5, 4, 1, 1, 6, 6, 1))
```

Figure 4.57 provides an example where the **ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE** `(5, 1, 4, 3, 1, 2, 4, 4, 5, 5, 4, 1, 1, 6, 6, 1))` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 4.57: Illustrating the ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Figure 4.58 depicts the automaton associated with the constraint `ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE`.

```
{ C ← X
  D ← +∞
  F ← X
  R ← 1
}

> r
R ∧ (F = C)

{ D ← +∞
  F ← C
  R ← R ∧ (F = C)
}

≤ s

{ C ← min(min(D, VARi), VARi+1)
  D ← +∞
}

≤

{ C ← min(C, min(D, VARi+1))
  D ← +∞
}

≤

{ D ← +∞
  F ← C
  R ← R ∧ (F = C)
}
```

Figure 4.58: Automaton for the `ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern.
## ALL_EQUAL_MIN_STRICTLY_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_MIN_STRICTLY_INCREASING_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Description

**Origin**

Based on the \textit{STRICTLY_INCREASING_SEQUENCE} pattern.

**Constraint**

\texttt{ALL_EQUAL_MIN_STRICTLY_INCREASING_SEQUENCE(VARIABLES)}

**Argument**

\texttt{VARIABLES : collection(var\_dvar)}

**Restriction**

\texttt{required(VARIABLES.var)}

Succeeds if the minima of the values in each occurrence of the \textit{STRICTLY_INCREASING_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection are all the same.

**Purpose**

An occurrence of the pattern \textit{STRICTLY_INCREASING_SEQUENCE} is the \textit{maximal} sub-sequence which matches the regular expression \texttt{'<+'}. Assume that the occurrence of the pattern \textit{STRICTLY_INCREASING_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{MIN} computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
(7, 2, 2, 6, 6, 3, 2, 3, 4, 5, 6, 4, 2, 5, 1)
\]

Figure 4.59 provides an example where the \texttt{ALL_EQUAL_MIN_STRICTLY_INCREASING_SEQUENCE ([7, 2, 2, 6, 6, 3, 2, 3, 4, 5, 6, 4, 2, 5, 1])} constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES.var}) > 1
\]
Figure 4.59: Illustrating the \texttt{ALL\_EQUAL\_MIN\_STRICTLY\_INCREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Figure 4.60 depicts the automaton associated with the constraint \textsc{all\_equal\_min\_strictly\_increasing\_sequence}.

\begin{align*}
  &\begin{cases}
    C \leftarrow X \\
    D \leftarrow +\infty \\
    F \leftarrow X \\
    R \leftarrow 1
  \end{cases} & \geq s \\
  \begin{cases}
    D \leftarrow +\infty \\
    F \leftarrow C \\
    R \leftarrow R \land (F = C)
  \end{cases} & \geq \begin{cases}
    C \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
    D \leftarrow +\infty
  \end{cases} \\
  \begin{cases}
    C \leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \\
    D \leftarrow +\infty
  \end{cases} & < r R \land (F = C)
\end{align*}

Figure 4.60: Automaton for the \textsc{all\_equal\_min\_strictly\_increasing\_sequence} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{strictly\_increasing\_sequence} pattern.
ALL_EQUAL_MIN STRICTLY_INCREASING_SEQUENCE
ALL_EQUAL_MIN_VALLEY

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_MIN_VALLEY</td>
<td>▶ ▶ ◄</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

\[ > (|=|>)^*(<||=)^*< \]

Origin
Based on the VALLEY pattern.

Constraint
\[ \text{ALL}_\text{EQUAL}_\text{MIN}_\text{VALLEY}(\text{VARIABLES}) \]

Argument
\[ \text{VARIABLES} : \text{collection}(\text{var}\text{--dvar}) \]

Restriction
\[ \text{required}(\text{VARIABLES}.\text{var}) \]

Succeeds if the minima of the values in each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection are all the same.

Purpose
An occurrence of the pattern VALLEY is the \textit{maximal} subsequence which matches the regular expression \( > (|=|>)^*(<||=)^*< \).
Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

Example
\[ ((7,6,6,1,1,7,6,4,3,1,7,6,5,2,1,7)) \]

Figure 4.61 provides an example where the ALL_EQUAL_MIN_VALLEY \( ([7,6,6,1,1,7,6,4,3,1,7,6,5,2,1,7]) \) constraint holds.

Typical
\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 4.61: Illustrating the ALL_EQUAL_MIN_VALLEY constraint of the Example slot
Figure 4.62 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_MIN_VALLEY}.

\begin{align*}
&\{ C \leftarrow X \\
&\quad D \leftarrow +\infty \\
&\quad F \leftarrow X \\
&\quad R \leftarrow 1 \} \quad \leq s \\
&\quad \uparrow \\
&\leq t \quad \leq \\
&\quad \downarrow \\
&\quad \{ D \leftarrow \min(D, \text{VAR}_i) \} \\
&\quad \{ C \leftarrow \min(C, \min(D, \text{VAR}_i)) \leq \min(D, \text{VAR}_i) \} \\
&\quad \{ D \leftarrow +\infty \} \\
&\quad \{ D \leftarrow \min(D, \text{VAR}_i) \} \\
&\quad \{ C \leftarrow \min(C, \min(D, \text{VAR}_i)) \} \\
&\quad \{ D \leftarrow \min(D, \text{VAR}_i) \} \\
&\quad \{ D \leftarrow +\infty \} \\
&\quad \{ D \leftarrow \min(D, \text{VAR}_i) \} \\
&\quad \{ D \leftarrow +\infty \} \\
&\quad \{ D \leftarrow \min(D, \text{VAR}_i) \} \\
\end{align*}

\begin{equation}
R \land (F = C)
\end{equation}

Figure 4.62: Automaton for the \texttt{ALL_EQUAL_MIN_VALLEY} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{VALLEY} pattern.
### Condition

<table>
<thead>
<tr>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_MIN_ZIGZAG</td>
<td></td>
</tr>
</tbody>
</table>

#### Description

**Origin**
Based on the ZIGZAG pattern.

**Constraint**

\[
\text{ALL_EQUAL_MIN_ZIGZAG}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

**Purpose**

Succeeds if the minima of the values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern ZIGZAG is the *maximal* subsequence which matches the regular expression 
\[
'(<>)^* (< | <>) | (<>)^* (> | > <)'.
\]

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature \(\text{MIN}\) computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
((1,3,1,2,6,5,1,4,3,4,3,5,6,1,4,7))
\]

Figure 4.63 provides an example where the \text{ALL_EQUAL_MIN_ZIGZAG} \(([1,3,1,2,6,5,1,4,3,4,3,5,6,1,4,7])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 3
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.63: Illustrating the ALL_EQUAL_MIN_ZIGZAG constraint of the Example slot
Automaton Figure 4.64 depicts the automaton associated with the constraint \textit{ALL_EQUAL_MIN_ZIGZAG}. 
Figure 4.64: Automaton for the ALL_EQUAL_MIN_ZIGZAG constraint obtained by applying decoration Table 3.38 to the seed transducer of the ZIGZAG pattern: (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the register D is reset to its initial value; (3) on transitions from c, f to s the register F is reset to C, and the register R is updated wrt C and F.
ALL_EQUAL_RANGE_DECREASING

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>[_]</td>
<td></td>
<td>[_]</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

**Origin**
Based on the DECREASING pattern.

**Constraint**
\[
\text{ALL_EQUAL_RANGE_DECREASING(VARIABLES)}
\]

**Argument**
\[
\text{VARIABLES : collection(var\_dvar)}
\]

**Restriction**
\[
\text{required(VARIABLES.var)}
\]

Succeeds if the differences between the largest and smallest value in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**
An occurrence of the pattern DECREASING is the subsequence which matches the regular expression ‘>‘. Assume that the occurrence of the pattern DECREASING starts at position \(i\) and ends at position \(j\). The feature RANGE computes the range of the values from index \(i\) to index \(j + 1\).

**Example**
\[
((1, 4, 6, 2, 4, 4, 5, 5, 5, 1, 3, 6, 2, 5))
\]

Figure 4.65 provides an example where the ALL_EQUAL_RANGE_DECREASING \(((1, 4, 6, 2, 4, 4, 5, 5, 5, 1, 3, 6, 2, 5))\) constraint holds.

**Typical**
\[
|\text{VARIABLES}| > 1 \\
\text{range(VARIABLES.var)} > 1
\]
Figure 4.65: Illustrating the ALLEQUALRANGEDECREASING constraint of the Example slot
Figure 4.66 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_RANGE\_DECREASING}. 

Figure 4.66: Automaton for the \texttt{ALL\_EQUAL\_RANGE\_DECREASING} constraint obtained by applying decoration Table 3.49 to the seed transducer of the \texttt{DECREASING} pattern.
### ALL_EQUAL_RANGE_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_RANGE_DECREASING_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Origin
Based on the **DECREASING_SEQUENCE** pattern.

#### Constraint
```
ALL_EQUAL_RANGE_DECREASING_SEQUENCE(VARIABLES)
```

#### Argument
```
VARIABLES : collection(var−dvar)
```

#### Restriction
```
required(VARIABLES.var)
```

Succeeds if the differences between the largest and smallest value in each occurrence of the **DECREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection are the same.

#### Purpose
An occurrence of the pattern **DECREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression `'> (> | =)* > | >'`.

Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **RANGE** computes the range of the values from index \( i \) to index \( j + 1 \).

#### Example
```
((1, 6, 7, 2, 2, 4, 5, 5, 4, 2, 0, 3, 6, 1, 5))
```

Figure 4.67 provides an example where the **ALL_EQUAL_RANGE_DECREASING_SEQUENCE** ((1, 6, 7, 2, 2, 4, 5, 5, 4, 2, 0, 3, 6, 1, 5)) constraint holds.

#### Typical
```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 4.67: Illustrating the ALL_EQUAL_RANGE_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.68 depicts the automaton associated with the constraint `ALL_EQUAL_RANGE_DECREASING_SEQUENCE`.

Figure 4.68: Automaton for the `ALL_EQUAL_RANGE_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.49 to the seed transducer of the `DECREASING_SEQUENCE` pattern.
ALL_EQUAL_RANGE_DECREASING_SEQUENCE
ALL_EQUAL_RANGE_INCREASING

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_RANGE_INCREASING</td>
<td>↑</td>
<td>↑</td>
</tr>
</tbody>
</table>

### Description

#### Origin
Based on the **INCREASING** pattern.

#### Constraint

$$\text{ALL_EQUAL_RANGE_INCREASING}(\text{VARIABLES})$$

#### Argument

$$\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})$$

#### Restriction

$$\text{required}(\text{VARIABLES} . \text{var})$$

Succeeds if the differences between the largest and smallest value in each occurrence of the **INCREASING** pattern in the time-series given by the **VARIABLES** collection are the same.

#### Purpose

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression \( '<' \).

Assume that the occurrence of the pattern **INCREASING** starts at position \( i \) and ends at position \( j \). The feature **RANGE** computes the range of the values from index \( i \) to index \( j + 1 \).

#### Example

$$((7, 3, 3, 6, 6, 3, 2, 2, 5, 4, 4, 3, 1, 4, 1))$$

Figure 4.69 provides an example where the **ALL_EQUAL_RANGE_INCREASING** $$((7, 3, 3, 6, 6, 3, 2, 2, 5, 4, 4, 3, 1, 4, 1))$$ constraint holds.

#### Typical

- $$|\text{VARIABLES}| > 1$$
- $$\text{range}(\text{VARIABLES}.\text{var}) > 1$$
Figure 4.69: Illustrating the ALL_EQUAL_RANGE_INCREASING constraint of the Example slot
Figure 4.70 depicts the automaton associated with the constraint ALL_EQUAL_RANGE_INCREASING.

Figure 4.70: Automaton for the ALL_EQUAL_RANGE_INCREASING constraint obtained by applying decoration Table 3.49 to the seed transducer of the INCREASING pattern
Origin: Based on the \textit{INCREASING\_SEQUENCE} pattern.

Constraint: \textsc{ALL\_EQUAL\_RANGE\_INCREASING\_SEQUENCE} (\textsc{VARIABLES})

Argument: \textsc{VARIABLES}: \textit{collection} (\textit{var}–\textit{dvar})

Restriction: \textsc{required} (\textsc{VARIABLES}.\textit{var})

Purpose: Succeeds if the differences between the largest and smallest value in each occurrence of the \textit{INCREASING\_SEQUENCE} pattern in the time-series given by the \textsc{VARIABLES} collection are the same.

An occurrence of the pattern \textit{INCREASING\_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression ‘\texttt{<(<|=)*<|<}’.

Assume that the occurrence of the pattern \textit{INCREASING\_SEQUENCE} starts at position \textit{i} and ends at position \textit{j}. The feature \textsc{RANGE} computes the range of the values from index \textit{i} to index \textit{j} + 1.

Example: \texttt{((7, 2, 2, 6, 6, 3, 1, 1, 2, 3, 3, 5, 4, 2, 6, 1))}

Figure 4.71 provides an example where the \textsc{ALL\_EQUAL\_RANGE\_INCREASING\_SEQUENCE} \{[7, 2, 2, 6, 6, 3, 1, 1, 2, 3, 3, 5, 4, 2, 6, 1]\} constraint holds.

Typical: \begin{align*}
|\textsc{VARIABLES}| & > 1 \\
\textsc{range}(\textsc{VARIABLES} \cdot \textit{var}) & > 1
\end{align*}
Figure 4.71: Illustrating the ALL_EQUAL_RANGE_INCREASING_SEQUENCE constraint of the Example slot
Automaton Figure 4.72 depicts the automaton associated with the constraint ALL_EQUAL_RANGE_INCREASING_SEQUENCE.

\[ \begin{align*}
C & \leftarrow X \\
F & \leftarrow X \\
H & \leftarrow \text{VAR}_1 \\
R & \leftarrow 1 \\
\text{s} & \geq \{H \leftarrow \text{VAR}_{i+1}\} \\
\{F \leftarrow |H - \text{VAR}_i|\} & \geq \{H \leftarrow \text{VAR}_{i+1}\} \\
\{H \leftarrow \text{VAR}_{i+1}\} & < \{C \leftarrow |H - \text{VAR}_{i+1}|\} \\
\leq t & = \{C \leftarrow |H - \text{VAR}_{i+1}|\}
\end{align*} \]

Figure 4.72: Automaton for the ALL_EQUAL_RANGE_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.49 to the seed transducer of the INCREASING_SEQUENCE pattern.
ALL_EQUAL_RANGE_INCREASING_SEQUENCE
Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

Constraint: `ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE(VARIABLES)`

Argument: `VARIABLES : collection(var−dvar)`

Restriction: `required(VARIABLES.var)`

Succeeds if the differences between the largest and smallest value in each occurrence of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are the same.

Purpose: An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the *maximal* sub-sequence which matches the regular expression `'>'`. Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `RANGE` computes the range of the values from index \(i\) to index \(j + 1\).

Example: `((4, 1, 4, 3, 1, 2, 4, 4, 5, 5, 4, 2, 2, 6, 6, 3))`

Figure 4.73 provides an example where the `ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE` \(([4, 1, 4, 3, 1, 2, 4, 4, 5, 5, 4, 2, 2, 6, 6, 3])\) constraint holds.

Typical: `|VARIABLES| > 1`  
`range(VARIABLES.var) > 1`
Figure 4.73: Illustrating the **ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE** constraint of the **Example** slot
Automaton

Figure 4.74 depicts the automaton associated with the constraint `ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE`.

Figure 4.74: Automaton for the `ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.49 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern.
ALL_EQUAL_RANGE_STRICTLY_INCREASING_SEQUENCE

Origin
Based on the STRICTLY_INCREASINGSEQUENCE pattern.

Constraint
ALL_EQUAL_RANGE_STRICTLY_INCREASING_SEQUENCE(VARIABLES)

Argument
VARIABLES : collection(var-dvar)

Restriction
required(VARIABLES.var)

Purpose
Succeeds if the differences between the largest and smallest value in each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are the same.

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature RANGE computes the range of the values from index \(i\) to index \(j + 1\).

Example
\((7, 2, 6, 6, 3, 1, 1, 2, 3, 4, 5, 4, 0, 4, 1)\)

Figure 4.75 provides an example where the ALL_EQUAL_RANGE_STRICTLY_INCREASING_SEQUENCE \(((7, 2, 6, 6, 3, 1, 1, 2, 3, 4, 5, 4, 0, 4, 1))\) constraint holds.

Typical
|VARIABLES| > 1
\(\text{range}(\text{VARIABLES.var}) > 1\)
Figure 4.75: Illustrating the **ALL_EQUAL_RANGE_STRICTLY_INCREASING_SEQUENCE** constraint of the **Example** slot
Figure 4.76 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_RANGE_STRICTLY_INCREASING_SEQUENCE}.

\begin{align*}
C &\leftarrow X \\
F &\leftarrow X \\
H &\leftarrow \text{VAR}_i \\
R &\leftarrow 1 \\
\end{align*}

\begin{align*}
\geq &\quad \{ H \leftarrow \text{VAR}_{i+1} \} \\
\geq &\quad \{ F \leftarrow |H - \text{VAR}_i| \} \\
\leq &\quad \{ C \leftarrow |H - \text{VAR}_i| \} \\
\leq &\quad \{ C \leftarrow |H - \text{VAR}_{i+1}| \} \\
\end{align*}

Figure 4.76: Automaton for the \texttt{ALL_EQUAL_RANGE_STRICTLY_INCREASING_SEQUENCE} constraint obtained by applying decoration Table 3.49 to the seed transducer of the \texttt{STRICTLY_INCREASING_SEQUENCE} pattern.
ALL_EQUAL_RANGE.Strictly_Increasing_Sequence
### Condition

<table>
<thead>
<tr>
<th>ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description Automaton</td>
</tr>
</tbody>
</table>

### Origin

Based on the **BUMP_ON_DECREASING_SEQUENCE** pattern.

### Constraint

\[
\text{ALL\_EQUAL\_SURF\_BUMP\_ON\_DECREASING\_SEQUENCE(VARIABLES)}
\]

### Argument

\[
\text{VARIABLES : collection(var\_dvar)}
\]

### Restriction

\[
\text{required(VARIABLES.var)}
\]

Succeeds if the surface of all occurrences of the **BUMP_ON_DECREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection are the same.

### Purpose

An occurrence of the pattern **BUMP_ON_DECREASING_SEQUENCE** is the subsequence which matches the regular expression `>>><>`.

Assume that the occurrence of the pattern **BUMP_ON_DECREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i + 2\) to index \(j\).

### Example

\[
((7, 6, 5, 6, 5, 4, 2, 3, 7, 4, 3, 2, 8, 6, 4, 4))
\]

Figure 4.77 provides an example where the **ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE** \([(7, 6, 5, 6, 5, 4, 2, 3, 7, 4, 3, 2, 8, 6, 4, 4)]\) constraint holds.

### Typical

\[
|\text{VARIABLES}| > 5 \\
\text{range(VARIABLES.var)} > 2
\]
Figure 4.77: Illustrating the ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Figure 4.78 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_SURF\_BUMP\_ON\_DECREASING\_SEQUENCE}.

Figure 4.78: Automaton for the \texttt{ALL\_EQUAL\_SURF\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} pattern.
ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_SURF_DECREASING</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Description

**Origin**
Based on the **DECREASING** pattern.

**Constraint**

$$\text{ALL_EQUAL_SURF_DECREASING}(\text{VARIABLES})$$

**Argument**

$$\text{VARIABLES} : \text{collection}(\text{var-dvar})$$

**Restriction**

$$\text{required}(\text{VARIABLES}.\text{var})$$

Succeeds if the surface of all occurrences of the **DECREASING** pattern in the time-series given by the **VARIABLES** collection are the same.

**Purpose**

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression ‘>’.

Assume that the occurrence of the pattern **DECREASING** starts at position $$i$$ and ends at position $$j$$. The feature **SURF** computes the sum of the values from index $$i$$ to index $$j + 1$$.

**Example**

$$\langle 1, 5, 5, 2, 2, 3, 4, 6, 1, 2, 3, 4, 5, 2, 2, 3 \rangle$$

Figure 4.79 provides an example where the **ALL_EQUAL_SURF_DECREASING** ($$\langle 1, 5, 5, 2, 2, 3, 4, 6, 1, 2, 3, 4, 5, 2, 2, 3 \rangle$$) constraint holds.

**Typical**

$$|\text{VARIABLES}| > 1$$

$$\text{range}(\text{VARIABLES}.\text{var}) > 1$$
Figure 4.79: Illustrating the ALL_EQUAL_SURF_DECREASING constraint of the Example slot
**Automaton**

Figure 4.80 depicts the automaton associated with the constraint `ALL_EQUAL_SURF_DECREASING`.

\[
\begin{align*}
\text{C} & \leftarrow X \\
\text{D} & \leftarrow 0 \\
\text{F} & \leftarrow X \\
\text{R} & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
\text{C} & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
\text{D} & \leftarrow 0 \\
\text{F} & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
\text{R} & \leftarrow \text{R} \land (F = D + \text{VAR}_i + \text{VAR}_{i+1})
\end{align*}
\]

Figure 4.80: Automaton for the `ALL_EQUAL_SURF_DECREASING` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `DECREASING` pattern.
ALL_EQUAL_SURF_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_SURF_DECREASING_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ > (> | =)* > | \]

**Description Automaton**

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**
\[ \text{ALL\_EQUAL\_SURF\_DECREASING\_SEQUENCE(VARIABLES)} \]

**Argument**
\[ \text{VARIABLES : collection(var-dvar)} \]

**Restriction**
\[ \text{required(VARIABLES.var)} \]

Succeeds if the surface of all occurrences of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**
An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\( (> | =)* > | > \)’.

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i \) to index \( j+1 \).

**Example**
\[ ((1, 6, 6, 4, 3, 3, 2, 2, 5, 6, 6, 5, 5, 2, 3, 3)) \]

Figure 4.81 provides an example where the ALL_EQUAL_SURF_DECREASING_SEQUENCE ((1, 6, 6, 4, 3, 3, 2, 2, 5, 6, 6, 5, 5, 2, 3, 3)) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 1 \]
\[ \text{range(VARIABLES.var)} > 1 \]
Figure 4.81: Illustrating the ALL_EQUAL_SURF_DECREASING_SEQUENCE constraint of the Example slot
Figure 4.82 depicts the automaton associated with the constraint `ALL_EQUAL_SURF_DECREASING_SEQUENCE`.

Figure 4.82: Automaton for the `ALL_EQUAL_SURF_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `DECREASING_SEQUENCE` pattern
### Description

- **Origin**: Based on the `DECREASING_TERRACE` pattern.
- **Constraint**: `ALL_EQUAL_SURF_DECREASING_TERRACE(VARIABLES)`
- **Argument**: `VARIABLES` : `collection(var−dvar)`
- **Restriction**: `required(VARIABLES, var)`

### Purpose

Succeeds if the surface of all occurrences of the `DECREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection are the same. An occurrence of the pattern `DECREASING_TERRACE` is the *maximal* subsequence which matches the regular expression `">=+"`.

Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position $i$ and ends at position $j$. The feature `SURF` computes the sum of the values from index $i + 1$ to index $j$.

### Example

```plaintext
((5, 4, 4, 1, 6, 6, 2, 2, 2, 1, 6, 4, 4, 2, 2))
```

Figure 4.83 provides an example where the `ALL_EQUAL_SURF_DECREASING_TERRACE` `((5, 4, 4, 1, 6, 6, 2, 2, 2, 1, 6, 4, 4, 2, 2))` constraint holds.

### Typical

- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 2`
Figure 4.83: Illustrating the \texttt{ALL\_EQUAL\_SURF\_DECREASING\_TERRACE} constraint of the \texttt{Example} slot
Figure 4.84 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_SURF\_DECREASING\_TERRACE}.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{automaton.pdf}
\caption{Automaton for the \texttt{ALL\_EQUAL\_SURF\_DECREASING\_TERRACE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{DECREASING\_TERRACE} pattern}
\end{figure}
Origin
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

Constraint
ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES, var)

Purpose
Succeeds if the surface of all occurrences of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence ‘<<><<’. Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 2 \) to index \( j \).

Example
\((1,2,3,2,3,4,6,5,1,4,5,6,0,2,4,4)\)

Figure 4.85 provides an example where the ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE (\([1,2,3,2,3,4,6,5,1,4,5,6,0,2,4,4]) constraint holds.

Typical
\(|\text{VARIABLES}| > 5\\ \text{range(}\text{VARIABLES.var}) > 2\)
Figure 4.85: Illustrating the \texttt{ALL\_EQUAL\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Figure 4.86 depicts the automaton associated with the constraint
\texttt{ALL\_EQUAL\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE}.

Figure 4.86: Automaton for the \texttt{ALL\_EQUAL\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE}
constraint obtained by applying decoration Table 3.38 to the seed transducer of the
\texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern
ALL_EQUAL_SURF_GORGE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_SURF_GORGE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DESCRIPTION AUTOMATON

Origin
Based on the GORGE pattern.

Constraint
ALL_EQUAL_SURF_GORGE(VARIABLES)

Argument
VARIABLES : collection(var–dvar)

Restriction
required(VARIABLES, var)

Purpose
Succeeds if the surface of all occurrences of the GORGE pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression "(>|>(=>|>)*)(<|<(=|>)*)".

Assume that the occurrence of the pattern GORGE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example

\[ ((6, 4, 2, 3, 5, 1, 1, 5, 4, 2, 3, 6, 3, 2, 4, 5)) \]

Figure 4.87 provides an example where the ALL_EQUAL_SURF_GORGE ([6, 4, 2, 3, 5, 1, 1, 5, 4, 2, 3, 6, 3, 2, 4, 5]) constraint holds.

Typical

\[ |\text{VARIABLES}| > 2 \]

\[ \text{range}(\text{VARIABLES.var}) > 1 \]
Figure 4.87: Illustrating the ALL_EQUAL_SURF_GORGE constraint of the Example slot
Figure 4.88 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_SURF\_GORGE}.

Figure 4.88: Automaton for the \texttt{ALL\_EQUAL\_SURF\_GORGE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{GORGE} pattern (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
### Description

**Origin**
Based on the **INCREASING** pattern.

**Constraint**

\[
\text{ALL_EQUAL_SURF_INCREASING}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

### Purpose
Succeeds if the surface of all occurrences of the **INCREASING** pattern in the time-series given by the **VARIABLES** collection are the same.

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression \('<'\).

Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i\) to index \(j + 1\).

### Example

\[
((6, 1, 1, 7, 4, 4, 2, 2, 6, 4, 2, 6, 5, 3, 5, 5))
\]

Figure 4.89 provides an example where the **ALL_EQUAL_SURF_INCREASING** \([(6, 1, 1, 7, 4, 4, 2, 2, 6, 4, 2, 6, 5, 3, 5, 5)]\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.89: Illustrating the ALL_EQUAL_SURF_INCREASING constraint of the Example slot
Figure 4.90 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_SURF_INCREASING}.

\begin{itemize}
  \item \( C \leftarrow X \)
  \item \( D \leftarrow 0 \)
  \item \( F \leftarrow X \)
  \item \( R \leftarrow 1 \)
\end{itemize}

\( R \land (F = C) \) \geq \begin{cases} \\
C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
D \leftarrow 0 \\
F \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
R \leftarrow R \land (F = D + \text{VAR}_i + \text{VAR}_{i+1}) 
\end{cases}

Figure 4.90: Automaton for the \texttt{ALL_EQUAL_SURF_INCREASING} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{INCREASING} pattern.
ALL_EQUAL_SURF_INCREASING 337
### ALL_EQUAL_SURF_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_SURF_INCREASING_SEQUENCE</td>
<td>▶ △ ◄</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

#### Origin
Based on the **INCREASING_SEQUENCE** pattern.

#### Constraint

\[
\text{ALL_EQUAL_SURF_INCREASING_SEQUENCE(VARIABLES)}
\]

#### Argument
\[
\text{VARIABLES : collection(var–dvar)}
\]

#### Restriction
\[
\text{required(VARIABLES.var)}
\]

Succeeds if the surface of all occurrences of the **INCREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection are the same.

An occurrence of the pattern **INCREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression ‘\(< (< | =)^* < | <\)’.

Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j + 1\).

#### Purpose

#### Example
\[
\langle 6, 1, 3, 4, 4, 5, 2, 1, 5, 5, 6, 4, 4 \rangle
\]

Figure 4.91 provides an example where the **ALL_EQUAL_SURF_INCREASING_SEQUENCE** \(\langle 6, 1, 1, 3, 4, 4, 5, 2, 1, 1, 5, 6, 4, 4 \rangle\) constraint holds.

#### Typical
\[
|\text{VARIABLES}| > 1 \\
\text{range(VARIABLES.var)} > 1
\]
Figure 4.91: Illustrating the ALL_EQUAL_SURF_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.92 depicts the automaton associated with the constraint $\text{ALL_EQUAL_SURF_INCREASING_SEQUENCE}$.

$\begin{align*}
C &\leftarrow X \\
D &\leftarrow 0 \\
F &\leftarrow X \\
R &\leftarrow 1 \\
&\geq s \\
&\leq t \\
&\land (F = C)
\end{align*}$

$\begin{align*}
D &\leftarrow 0 \\
F &\leftarrow C \\
R &\leftarrow R \land (F = C) \\
&\geq s \\
&\leq t
\end{align*}$

$\begin{align*}
C &\leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
D &\leftarrow 0 \\
&\leq t \\
\end{align*}$

Figure 4.92: Automaton for the $\text{ALL_EQUAL_SURF_INCREASING_SEQUENCE}$ constraint obtained by applying decoration Table 3.38 to the seed transducer of the $\text{INCREASING_SEQUENCE}$ pattern
### ALL_EQUAL_SURF_INCREASING_TERRACE

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_SURF_INCREASING_TERRACE</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

#### Origin
Based on the `INCREASING_TERRACE` pattern.

#### Constraint

\[
\text{ALL\_EQUAL\_SURF\_INCREASING\_TERRACE}(\text{VARIABLES})
\]

#### Argument

\[
\text{VARIABLES} : \text{collection}(\text{var}-\text{dvar})
\]

#### Restriction

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the surface of all occurrences of the `INCREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection are the same.

An occurrence of the pattern `INCREASING_TERRACE` is the maximal subsequence which matches the regular expression `\'<=\ '+'\'`.

Assume that the occurrence of the pattern `INCREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i+1` to index `j`.

#### Purpose

**Example**

\[
(2, 3, 3, 6, 1, 1, 2, 2, 2, 4, 6, 1, 3, 3, 5, 5)
\]

Figure 4.93 provides an example where the `ALL\_EQUAL\_SURF\_INCREASING\_TERRACE` ([2, 3, 3, 6, 1, 1, 2, 2, 2, 4, 6, 1, 3, 3, 5, 5]) constraint holds.

#### Typical

\[
|\text{VARIABLES}| > 3
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]
Figure 4.93: Illustrating the ALL_EQUAL_SURF_INCREASING_TERRACE constraint of the Example slot
Figure 4.94 depicts the automaton associated with the constraint ALL_EQUAL_SURF_INCREASING_TERRACE.

\[
\begin{aligned}
C &\leftarrow X \\
D &\leftarrow 0 \\
F &\leftarrow X \\
R &\leftarrow 1
\end{aligned}
\]

\[\geq s\]

\[
\begin{aligned}
C &\leftarrow D + \text{VAR}_{i} \\
D &\leftarrow 0 \\
F &\leftarrow D + \text{VAR}_{i} \\
R &\leftarrow R \land (F = D + \text{VAR}_{i})
\end{aligned}
\]

\[\geq \]

\[<
\]

\[<\]

\[=
\]

\[\{D \leftarrow D + \text{VAR}_{i}\}\]

\[>\]

\[>\]

\[<\]

\[\{D \leftarrow 0\}\]

Figure 4.94: Automaton for the ALL_EQUAL_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING_TERRACE pattern
ALL_EQUAL_SURF_INFLEXION

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_SURF_INFLEXION</td>
<td>▶ ▶ ▶</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the INFLEXION pattern.

**Constraint**
ALL_EQUAL_SURF_INFLEXION(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the surface of all occurrences of the INFLEXION pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**
An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression '< (< | | =)∗ > | > | =)∗ <'.

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**
\((2,2,3,6,6,5,5,3,2,4,5,6,3,3,3))\)

Figure 4.95 provides an example where the ALL_EQUAL_SURF_INFLEXION (\([2,2,3,6,6,5,5,3,2,4,5,6,3,3,3]\)) constraint holds.

**Typical**
\(|\text{VARIABLES}| > 2\)
range(VARIABLES.var) > 1
Figure 4.95: Illustrating the ALL_EQUAL_SURF_INFLEXION constraint of the Example slot
Figure 4.96 depicts the automaton associated with the constraint 
ALL_EQUAL_SURF_INFLEXION.

\[
\begin{align*}
&= s \\
&= \\
&= \\
\end{align*}
\]

Figure 4.96: Automaton for the ALL_EQUAL_SURF_INFLEXION constraint obtained by applying decoration Table 3.38 to the seed transducer of the INFLEXION pattern (transition \( r \rightarrow t \) has the same registers updates as transition \( t \rightarrow r \))
ALL_EQUAL_SURF_PEAK

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_SURF_PEAK</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Description Automaton**

**Origin**
Based on the **PEAK** pattern.

**Constraint**
ALL_EQUAL_SURF_PEAK(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the surface of all occurrences of the **PEAK** pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**
An occurrence of the pattern **PEAK** is the *maximal* subsequence which matches the regular expression '< (= | <) (> | =>)* >'.
Assume that the occurrence of the pattern **PEAK** starts at position i and ends at position j. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Example**

```
((1,2,2,7,7,1,2,4,5,7,1,2,3,6,7,1))
```

Figure 4.97 provides an example where the ALL_EQUAL_SURF_PEAK ([1,2,2,7,7,1,2,4,5,7,1,2,3,6,7,1]) constraint holds.

**Typical**

- $|\text{VARIABLES}| > 2$
- range(VARIABLES.var) $> 1$

Figure 4.97: Illustrating the ALL_EQUAL_SURF_PEAK constraint of the Example slot
Figure 4.98 depicts the automaton associated with the constraint ALL_EQUAL_SURF_PEAK.

\[
\begin{align*}
C &\leftarrow X \\
D &\leftarrow 0 \\
F &\leftarrow X \\
R &\leftarrow 1 \\
\end{align*}
\]
\[\geq s\]
\[
\begin{align*}
C &\leftarrow D + \text{VAR}_i \\
D &\leftarrow 0 \\
\end{align*}
\]
\[\leq \{D \leftarrow D + \text{VAR}_i\}\]

\[
\begin{align*}
C &\leftarrow C + D + \text{VAR}_i \\
D &\leftarrow 0 \\
\end{align*}
\]
\[\geq t\]
\[
\begin{align*}
D &\leftarrow 0 \\
F &\leftarrow C \\
R &\leftarrow R \land (F = C) \\
\end{align*}
\]

Figure 4.98: Automaton for the ALL_EQUAL_SURF_PEAK constraint obtained by applying decoration Table 3.38 to the seed transducer of the PEAK pattern.
ALL_EQUAL_SURF_PLAIN

Origin: Based on the PLAIN pattern.

Constraint: ALL_EQUAL_SURF_PLAIN(VARIABLES)

Argument: VARIABLES : collection(var−dvar)

Restriction: required(VARIABLES.var)

Purpose: Succeeds if the surface of all occurrences of the PLAIN pattern in the time-series given by the VARIABLES collection are the same. An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘\( >=^*< \)’. Assume that the occurrence of the pattern PLAIN starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i+1 \) to index \( j \).

Example: \[((1,6,3,3,7,6,6,3,3,5,4,3,3,6,3))\]

Figure 4.99 provides an example where the ALL_EQUAL_SURF_PLAIN \(((1,6,3,3,7,6,6,3,3,5,4,3,3,6,3))\) constraint holds.

Typical: |VARIABLES| > 2
         range(VARIABLES.var) > 1
Figure 4.99: Illustrating the ALL_EQUAL_SURF_PLAIN constraint of the Example slot
Figure 4.100 depicts the automaton associated with the constraint ALL_EQUAL_SURFPLAIN.

**Automaton**

Figure 4.100: Automaton for the ALL_EQUAL_SURFPLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLAIN pattern.
### Description

**Origin**

Based on the PLATEAU pattern.

**Constraint**

\[
\text{ALL\_EQUAL\_SURF\_PLATEAU}((\text{VARIABLES}))
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}((\text{VARIABLES}.\text{var}))
\]

Succeeds if the surface of all occurrences of the PLATEAU pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**

An occurrence of the pattern PLATEAU is the *maximal* subsequence which matches the regular expression ‘\(<=?\)’.

Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
((7, 2, 5, 5, 1, 2, 2, 5, 5, 3, 3, 4, 5, 5, 2, 5))
\]

Figure 4.101 provides an example where the ALL\_EQUAL\_SURF\_PLATEAU \(((7, 2, 5, 5, 1, 2, 2, 5, 5, 3, 3, 4, 5, 5, 2, 5))\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.101: Illustrating the \texttt{ALL\_EQUAL\_SURF\_PLATEAU} constraint of the \texttt{Example} slot
Automaton

Figure 4.102 depicts the automaton associated with the constraint `ALL_EQUAL_SURF_PLATEAU`.

Figure 4.102: Automaton for the `ALL_EQUAL_SURF_PLATEAU` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `PLATEAU` pattern.
ALL_EQUAL_SURF_PROPERPLAIN

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_SURF_PROPERPLAIN</td>
<td>PROPERPLAIN</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

- **Origin**: Based on the PROPERPLAIN pattern.

- **Constraint**: ALL_EQUAL_SURF_PROPERPLAIN(VARIABLES)

- **Argument**: VARIABLES : collection(var−dvar)

- **Restriction**: required(VARIABLES.var)

- **Purpose**: Succeeds if the surface of all occurrences of the PROPERPLAIN pattern in the time-series given by the VARIABLES collection are the same. An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression ‘> =+ <’. Assume that the occurrence of the pattern PROPERPLAIN starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

- **Example**: $((2, 7, 3, 6, 6, 3, 7, 3, 5, 6, 5, 3, 3, 5))$

  Figure 4.103 provides an example where the ALL_EQUAL_SURF_PROPERPLAIN ($(2, 7, 3, 6, 6, 3, 7, 3, 5, 6, 5, 3, 3, 5))$ constraint holds.

- **Typical**

  $|\text{VARIABLES}| > 3$

  $\text{range}(\text{VARIABLES.var}) > 1$
Figure 4.103: Illustrating the ALL_EQUAL_SURF_PROPER_PLAIN constraint of the Example slot
Automaton

Figure 4.104 depicts the automaton associated with the constraint ALL_EQUAL_SURF_PROPER.PLAIN.

Figure 4.104: Automaton for the ALL_EQUAL_SURF_PROPER/plain constraint obtained by applying decoration Table 3.38 to the seed transducer of the PROPER.PLAIN pattern.
ALL_EQUAL_SURF_PROPER_PLAIN
ALL_EQUAL_SURF_PROPER_PLATEAU

### Origin
Based on the PROPER_PLATEAU pattern.

### Constraint
ALL_EQUAL_SURF_PROPER_PLATEAU(VARIABLES)

### Argument
VARIABLES : collection(var−dvar)

### Restriction
required(VARIABLES.var)

Succeeds if the surface of all occurrences of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression ‘< =+ >’.

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

### Purpose

### Example
```
((7, 1, 5, 5, 2, 2, 5, 1, 5, 3, 2, 3, 5, 5, 3))
```

Figure 4.105 provides an example where the ALL_EQUAL_SURF_PROPER_PLATEAU ([(7, 1, 5, 5, 2, 2, 5, 1, 5, 3, 2, 3, 5, 5, 3)]) constraint holds.

### Typical
```
| VARIABLES| > 3
range(VARIABLES.var) > 1
```
Figure 4.105: Illustrating the ALL_EQUAL_SURF_PROPER_PLATEAU constraint of the Example slot
Figure 4.106 depicts the automaton associated with the constraint `ALL_EQUAL_SURF_PROPER_PLATEAU`.

Automaton

Figure 4.106: Automaton for the `ALL_EQUAL_SURF_PROPER_PLATEAU` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `PROPER_PLATEAU` pattern
ALL_EQUAL_SURF_STEADY

**Description**

**Origin**
Based on the **STEADY** pattern.

**Constraint**

\[
\text{ALL_EQUAL_SURF_STEADY} (\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var-dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the surface of all occurrences of the **STEADY** pattern in the time-series given by the **VARIABLES** collection are the same.

**Purpose**

An occurrence of the pattern **STEADY** is the subsequence which matches the regular expression '\='.

Assume that the occurrence of the pattern **STEADY** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i\) to index \(j+1\).

**Example**

\[
((4, 4, 4, 2, 6, 4, 4, 4, 3, 2, 1, 4, 4, 3, 4))
\]

Figure 4.107 provides an example where the **ALL_EQUAL_SURF_STEADY** ((4, 4, 4, 2, 6, 4, 4, 4, 3, 2, 1, 4, 4, 3, 4)) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]
Figure 4.107: Illustrating the ALL_EQUAL_SURF_STEADY constraint of the Example slot
Figure 4.108 depicts the automaton associated with the constraint ALL_EQUAL_SURF_STEADY.

\[\begin{align*}
C & \leftarrow X \\
D & \leftarrow 0 \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}\]

\[R \land (F = C)\]

Figure 4.108: Automaton for the ALL_EQUAL_SURF_STEADY constraint obtained by applying decoration Table 3.38 to the seed transducer of the STEADY pattern
### ALL_EQUAL_SURF_STEADY_SEQUENCE

<table>
<thead>
<tr>
<th>CONSIDERED FEATURE PATTERNS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ALL_EQUAL_SURF_STEADY_SEQUENCE</strong></td>
</tr>
</tbody>
</table>

**DESCRIPTION AUTOMATON**

- **Origin**: Based on the **STEADY_SEQUENCE** pattern.
- **Constraint**: `ALL_EQUAL_SURF_STEADY_SEQUENCE(VARIABLES)`
- **Argument**: `VARIABLES : collection(var−dvar)`
- **Restriction**: `required(VARIABLES.var)`

**Purpose**: Succeeds if the surface of all occurrences of the **STEADY_SEQUENCE** pattern in the time-series given by the `VARIABLES` collection are the same. An occurrence of the pattern **STEADY_SEQUENCE** is the *maximal* subsequence which matches the regular expression `'=+'`. Assume that the occurrence of the pattern **STEADY_SEQUENCE** starts at position `i` and ends at position `j`. The feature **SURF** computes the sum of the values from index `i` to index `j + 1`.

**Example**: `((6, 6, 5, 3, 3, 1, 4, 4, 4, 5, 6, 6, 7, 2))`

Figure 4.109 provides an example where the `ALL_EQUAL_SURF_STEADY_SEQUENCE` ((6, 6, 5, 3, 3, 1, 4, 4, 4, 5, 6, 6, 7, 2)) constraint holds.

**Typical**: `|VARIABLES| > 1`
Figure 4.109: Illustrating the ALL_EQUAL_SURF_STEADY_SEQUENCE constraint of the Example slot
Figure 4.110 depicts the automaton associated with the constraint `ALL_EQUAL_SURF_STEADY_SEQUENCE`.

\[\begin{align*}
&C \leftarrow X \\
&D \leftarrow 0 \\
&F \leftarrow X \\
&R \leftarrow 1
\end{align*}\]

\begin{align*}
&D \leftarrow 0 \\
&F \leftarrow C \\
&R \leftarrow R \land (F = C)
\end{align*}

\begin{align*}
&C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
&D \leftarrow 0
\end{align*}

\begin{align*}
&D \leftarrow 0 \\
&F \leftarrow C + D + \text{VAR}_{i+1}
\end{align*}

Figure 4.110: Automaton for the `ALL_EQUAL_SURF_STEADY_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `STEADY_SEQUENCE` pattern.
ALL_EQUAL_SURF_STEADY_SEQUENCE
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_SURF_STRICTLY_DECREASING_SEQUENCE</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**
\[
\text{ALL_EQUAL_SURF_STRICTLY_DECREASING_SEQUENCE(VARIABLES)}
\]

**Argument**
\[
\text{VARIABLES : collection}(\text{var} - \text{dvar})
\]

**Restriction**
\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the surface of all occurrences of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are the same.

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the \textit{maximal} sub-sequence which matches the regular expression `\texttt{>^+}`.

**Purpose**
Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `SURF` computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**
\[
((3, 6, 5, 4, 5, 5, 6, 5, 3, 1, 2, 6, 5, 3, 1))
\]

Figure 4.111 provides an example where the `ALL_EQUAL_SURF_STRICTLY_DECREASING_SEQUENCE` \([3, 6, 5, 4, 5, 5, 6, 5, 3, 1, 2, 6, 5, 3, 1]\)) constraint holds.

**Typical**
\[
|\text{VARIABLES}| > 1 \\
|\text{range}(\text{VARIABLES}.\text{var})| > 1
\]
Figure 4.111: Illustrating the ALL_EQUAL_SURF_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Figure 4.112 depicts the automaton associated with the constraint \textit{ALL\_EQUAL\_SURF\_STRICTLY\_DECREASING\_SEQUENCE}.

Figure 4.112: Automaton for the \textit{ALL\_EQUAL\_SURF\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textit{STRICTLY\_DECREASING\_SEQUENCE} pattern.
**ALL_EQUAL_SURF_STRICTLY_INCREASING_SEQUENCE**

<table>
<thead>
<tr>
<th><strong>Origin</strong></th>
<th>Based on the <code>STRICTLY_INCREASING_SEQUENCE</code> pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Constraint</strong></td>
<td><code>ALL_EQUAL_SURF_STRICTLY_INCREASING_SEQUENCE(VARIABLES)</code></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : <code>collection(var−dvar)</code></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td><code>required(VARIABLES.var)</code></td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the surface of all occurrences of the `STRICTLY_INCREASING_SEQUENCE` pattern in the time-series given by the VARIABLES collection are the same. An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the *maximal* sub-sequence which matches the regular expression `'<+'`. Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j+1$.

**Example**

```plaintext
((6, 1, 1, 2, 3, 4, 5, 5, 2, 1, 1, 3, 5, 6, 4, 4))
```

Figure 4.113 provides an example where the `ALL_EQUAL_SURF_STRICTLY_INCREASING_SEQUENCE` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 4.113: Illustrating the ALL_EQUAL_SURF STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Figure 4.114 depicts the automaton associated with the constraint \textsc{ALL\_EQUAL\_SURF\_STRICTLY\_INCREASING\_SEQUENCE}.

\[ \begin{array}{l}
C \leftarrow X \\
D \leftarrow 0 \\
F \leftarrow X \\
R \leftarrow 1 \\
\end{array} \]

\[ \geq s \]

Figure 4.114: Automaton for the \textsc{ALL\_EQUAL\_SURF\_STRICTLY\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table \textsc{3.38} to the seed transducer of the \textsc{STRICTLY\_INCREASING\_SEQUENCE} pattern.
**ALL_EQUAL_SURF_SUMMIT**

**Description**

- **Origin**: Based on the SUMMIT pattern.
- **Constraint**: \( \text{ALL}\_\text{EQUAL}\_\text{SURF}\_\text{SUMMIT}(\text{VARIABLES}) \)
- **Argument**: \( \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \)
- **Restriction**: \( \text{required}(\text{VARIABLES}, \text{var}) \)
- **Purpose**: Succeeds if the surface of all occurrences of the SUMMIT pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern SUMMIT is the *maximal* subsequence which matches the regular expression \( '(\langle | < (\geq | <) \rangle | > | > (\geq | >) \rangle)' \).

Assume that the occurrence of the pattern SUMMIT starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**

\( (1,3,5,4,2,6,2,3,5,4,1,4,5,3,2) \)

Figure 4.115 provides an example where the \( \text{ALL}\_\text{EQUAL}\_\text{SURF}\_\text{SUMMIT}([1,3,5,4,2,6,2,3,5,4,1,4,5,3,2]) \) constraint holds.

**Typical**

\( |\text{VARIABLES}| > 2 \)
\( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)
Figure 4.115: Illustrating the ALL_EQUAL_SURF_SUMMIT constraint of the Example slot
Figure 4.116 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_SURF\_SUMMIT}.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{automaton.png}
\caption{Automaton for the \texttt{ALL\_EQUAL\_SURF\_SUMMIT} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{SUMMIT} pattern (transition \(u \rightarrow r\) has the same register update as transition \(r \rightarrow u\)).}
\end{figure}
### ALL_EQUAL_SURF_VALLEY

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ALL_EQUAL_SURF_VALLEY</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Description

- **Origin**: Based on the VALLEY pattern.
- **Constraint**: `ALL_EQUAL_SURF_VALLEY(VARIABLES)`
- **Argument**: `VARIABLES : collection(var−dvar)`
- **Restriction**: `required(VARIABLES.var)`

#### Purpose

Succeeds if the surface of all occurrences of the VALLEY pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern `VALLEY` is the *maximal* subsequence which matches the regular expression `\( > (= | >)^* (< | =)^* < \)`.

Assume that the occurrence of the pattern `VALLEY` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 1` to index `j`.

#### Example

```
((7,6,6,1,1,7,6,4,3,1,7,6,5,2,1,7))
```

Figure 4.117 provides an example where the `ALL_EQUAL_SURF_VALLEY` `([7,6,6,1,1,7,6,4,3,1,7,6,5,2,1,7])` constraint holds.

- **Typical**: `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`
Figure 4.117: Illustrating the ALL_EQUAL_SURF_VALLEY constraint of the Example slot
Automaton Figure 4.118 depicts the automaton associated with the constraint ALL_EQUAL_SURF_VALLEY.

Figure 4.118: Automaton for the ALL_EQUAL_SURF_VALLEY constraint obtained by applying decoration Table 3.38 to the seed transducer of the VALLEY pattern.
ALL_EQUAL_SURF_VALLEY
ALL_EQUAL_SURF_ZIGZAG

**Description**

- **Origin**: Based on the ZIGZAG pattern.
- **Constraint**: \( \text{ALL_EQUAL_SURF_ZIGZAG(}\text{VARIABLES}\text{)} \)
- **Argument**: \( \text{VARIABLES} : \text{collection(var\text{-}dvar)} \)
- **Restriction**: required(\(\text{VARIABLES.\text{var}}\))

**Purpose**

Succeeds if the surface of all occurrences of the ZIGZAG pattern in the time-series given by the \(\text{VARIABLES}\) collection are the same.

An occurrence of the pattern ZIGZAG is the \textit{maximal} subsequence which matches the regular expression \( '(<>^* (< | <>)) | (<>^* (> | ><>))' \).

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature \text{SURF} computes the sum of the values from index \(i+1\) to index \(j\).

**Example**

\( ((1, 6, 1, 2, 6, 5, 1, 2, 1, 4, 5, 2, 4, 7)) \)

Figure 4.119 provides an example where the \( \text{ALL_EQUAL_SURF_ZIGZAG} \) constraint holds.

**Typical**

\( |\text{VARIABLES}| > 3 \)
\( \text{range(VARIABLES.\text{var})} > 1 \)
Figure 4.119: Illustrating the ALL_EQUAL_SURF_ZIGZAG constraint of the Example slot
Automaton

Figure 4.120 depicts the automaton associated with the constraint ALL_EQUAL_SURF_ZIGZAG.
Figure 4.120: Automaton for the `ALL_EQUAL_SURF_ZIGZAG` constraint obtained by applying decoration Table 3.38 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from `a`, `b`, `c`, `d`, `e`, `f` to `s` are labelled by `=`; (2) on transitions from `b`, `c`, `e`, `f` to `s` the register `D` is reset to its initial value; (3) on transitions from `c`, `f` to `s` the register `F` is reset to `C`, and the register `R` is updated wrt `C` and `F`
**ALL_EQUAL_WIDTH_DECREASING_SEQUENCE**

**Origin**
Based on the `DECREASING_SEQUENCE` pattern.

**Constraint**
`ALL_EQUAL_WIDTH_DECREASING_SEQUENCE(VARIABLES)`

**Argument**
`VARIABLES : collection(var−dvar)`

**Restriction**
`required(VARIABLES.var)`

Succeeds if the width of all occurrences of the `DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are the same.

**Purpose**
An occurrence of the pattern `DECREASING_SEQUENCE` is the *maximal* subsequence which matches the regular expression `'(> | =)* > | >'`. Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

**Example**
`((1,3,3,2,2,1,1,4,6,6,4,4,2,2,3,4))`

Figure 4.121 provides an example where the `ALL_EQUAL_WIDTH_DECREASING_SEQUENCE` `([1,3,3,2,2,1,1,4,6,6,4,4,2,2,3,4])` constraint holds.

**Typical**
- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 4.121: Illustrating the ALL_EQUAL_WIDTH_DECREASING_SEQUENCE constraint of the Example slot
Figure 4.122 depicts the automaton associated with the constraint `ALL_EQUAL_WIDTH_DECREASING_SEQUENCE`.

Figure 4.122: Automaton for the `ALL_EQUAL_WIDTH_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `DECREASING_SEQUENCE` pattern.
ALL_EQUAL_WIDTH_DECREASING_SEQUENCE
**Condition**

```
> = >
```

**Feature**

```
DESCRIPTION
```

**Pattern**

```
AUTOMATON
```

### Origin

Based on the `DECREASING_TERRACE` pattern.

### Constraint

```
ALL_EQUAL_WIDTH_DECREASING_TERRACE(VARIABLES)
```

### Argument

```
VARIABLES : collection(var\_dvar)
```

### Restriction

```
required(VARIABLES.var)
```

Succeeds if the width of all occurrences of the `DECREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection are the same.

### Purpose

An occurrence of the pattern `DECREASING_TERRACE` is the maximal subsequence which matches the regular expression `'> = >'`.

Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j - i`.

### Example

```
((6, 5, 5, 4, 3, 3, 1, 3, 3, 5, 2, 2, 1, 1, 1))
```

Figure 4.123 provides an example where the `ALL_EQUAL_WIDTH_DECREASING_TERRACE` `([6, 5, 5, 4, 3, 3, 1, 1, 3, 3, 5, 2, 2, 1, 1, 1])` constraint holds.

### Typical

```
|VARIABLES| > 3
range(VARIABLES.var) > 2
```
Figure 4.123: Illustrating the \texttt{ALL_EQUAL\_WIDTH\_DECREASING\_TERRACE} constraint of the \texttt{Example} slot
Automaton Figure 4.124 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_WIDTH_DECREASING_TERRACE}.

Figure 4.124: Automaton for the \texttt{ALL_EQUAL_WIDTH_DECREASING_TERRACE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING_TERRACE pattern
ALL_EQUAL_WIDTH_DECREASING_TERRACE
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_WIDTH_GORGE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DESCRIPTION AUTOMATON**

- **Origin**: Based on the GORGE pattern.
- **Constraint**: $\text{ALL_EQUAL_WIDTH_GORGE}(\text{VARIABLES})$
- **Argument**: $\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})$
- **Restriction**: $\text{required}(\text{VARIABLES}\text{.var})$

**Purpose**

Succeeds if the width of all occurrences of the GORGE pattern in the time-series given by the $\text{VARIABLES}$ collection are the same.

An occurrence of the pattern GORGE is the *maximal* subsequence which matches the regular expression $'(> | > (|= >) > | < (|= <) <)'^{\ast}$.

Assume that the occurrence of the pattern GORGE starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$.

**Example**

```plaintext
((6, 4, 2, 3, 5, 1, 1, 5, 4, 2, 3, 6, 3, 2, 4, 5))
```

Figure 4.125 provides an example where the $\text{ALL_EQUAL_WIDTH_GORGE}([6, 4, 2, 3, 5, 1, 1, 5, 4, 2, 3, 6, 3, 2, 4, 5])$ constraint holds.

**Typical**

- $|\text{VARIABLES}| > 2$
- $\text{range}(\text{VARIABLES}\.\text{var}) > 1$
Figure 4.125: Illustrating the `ALL_EQUAL_WIDTH_GORGE` constraint of the `Example` slot
Automaton Figure 4.126 depicts the automaton associated with the constraint ALL_EQUAL_WIDTH_GORGE.

Figure 4.126: Automaton for the ALL_EQUAL_WIDTH_GORGE constraint obtained by applying decoration Table 3.38 to the seed transducer of the GORGE pattern (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
ALL_EQUAL_WIDTH_INCREASING_SEQUENCE

**Origin**
Based on the **INCREASING_SEQUENCE** pattern.

**Constraint**
\[ \text{ALL_EQUAL_WIDTH_INCREASING_SEQUENCE}(\text{VARIABLES}) \]

**Argument**
\[ \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \]

**Restriction**
\[ \text{required}(\text{VARIABLES}, \text{var}) \]

**Purpose**
Succeeds if the width of all occurrences of the **INCREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection are the same.

An occurrence of the pattern **INCREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression `'\(< \mid =\)* \(< \mid \)'`

Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position `i` and ends at position `j`. The feature **WIDTH** computes the value `j - i + 2`.

**Example**
\[ ((6, 4, 4, 5, 5, 6, 6, 3, 1, 1, 3, 3, 5, 5, 4, 3)) \]

Figure 4.127 provides an example where the **ALL_EQUAL_WIDTH_INCREASING_SEQUENCE** constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}, \text{var}) > 1 \]
Figure 4.127: Illustrating the ALL_EQUAL_WIDTH_INCREASING_SEQUENCE constraint of the Example slot
Automaton Figure 4.128 depicts the automaton associated with the constraint
\texttt{ALL\_EQUAL\_WIDTH\_INCREASING\_SEQUENCE}.

\begin{align*}
\{ & C \leftarrow X \\
D & \leftarrow 0 \\
F & \leftarrow X \\
R & \leftarrow 1 \\
\} \\
\end{align*}

\begin{align*}
\geq & \ s \\
\geq & \ \\
\{ & D \leftarrow 0 \\
F & \leftarrow C \\
R & \leftarrow R \land (F = C) \\
\} \\
\rangle & \langle \\
\geq & \ \\
\{ & C \leftarrow D + 2 \\
D & \leftarrow 0 \\
\} \\
\leq & \ \\
\{ & D \leftarrow D + 1 \\
\} \\
\langle & \ \\
\leq & \ \\
\{ & C \leftarrow C + D + 1 \\
D & \leftarrow 0 \\
\} \\
\leq & \ \\
\end{align*}

Figure 4.128: Automaton for the \texttt{ALL\_EQUAL\_WIDTH\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern
ALL_EQUAL_WIDTH_INCREASING_TERRACE

**DESCRIPTION**

**Origin**
Based on the INCREASING_TERRACE pattern.

**Constraint**
ALL_EQUAL_WIDTH_INCREASING_TERRACE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the width of all occurrences of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**
An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression '<=+'.'
Assume that the occurrence of the pattern INCREASING_TERRACE starts at position i and ends at position j. The feature WIDTH computes the value j − i.

**Example**

```
((1, 2, 2, 3, 4, 6, 6, 4, 4, 2, 5, 5, 6, 6, 6))
```

Figure 4.129 provides an example where the ALL_EQUAL_WIDTH_INCREASING_TERRACE ((1, 2, 2, 3, 4, 6, 6, 4, 4, 2, 5, 5, 6, 6)) constraint holds.

**Typical**

| VARIABLES | > 3
range(VARIABLES.var) | > 2
Figure 4.129: Illustrating the ALL_EQUAL_WIDTH_INCREASING_TERRACE constraint of the Example slot
Figure 4.130 depicts the automaton associated with the constraint \textsc{ALL\_EQUAL\_WIDTH\_INCREASING\_TERRACE}.

Figure 4.130: Automaton for the \textsc{ALL\_EQUAL\_WIDTH\_INCREASING\_TERRACE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{INCREASING\_TERRACE} pattern
**ALL_EQUAL_WIDTH_INFLEXION**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_WIDTH_INFLEXION</td>
<td>▶</td>
<td>&lt; (&lt;</td>
</tr>
</tbody>
</table>

**Description (Automaton)**

- **Origin**: Based on the `INFLEXION` pattern.
- **Constraint**: `ALL_EQUAL_WIDTH_INFLEXION(VARIABLES)`
- **Argument**: `VARIABLES : collection(var−dvar)`
- **Restriction**: `required(VARIABLES, var)`
- **Purpose**: Succeeds if the width of all occurrences of the `INFLEXION` pattern in the time-series given by the `VARIABLES` collection are the same.

An occurrence of the pattern `INFLEXION` is the *maximal* subsequence which matches the regular expression `<( < | =)> | > (< | =)> <`.

Assume that the occurrence of the pattern `INFLEXION` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

**Example**

```
((2, 2, 3, 6, 5, 3, 3, 4, 4, 5, 3, 3, 3, 4, 6))
```

Figure 4.131 provides an example where the `ALL_EQUAL_WIDTH_INFLEXION` constraint holds.

**Typical**

- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`
Figure 4.131: Illustrating the ALL_EQUAL_WIDTH_INFLEXION constraint of the Example slot
Figure 4.132 depicts the automaton associated with the constraint
\textsc{All\_Equal\_Width\_Inflexion}.

Figure 4.132: Automaton for the \textsc{All\_Equal\_Width\_Inflexion} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{Inflexion} pattern (transition $r \rightarrow t$ has the same registers updates as transition $t \rightarrow r$)
ALL_EQUAL_WIDTH_INFLEXION
### ALL_EQUAL_WIDTH_PEAK

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_WIDTH_PEAK</td>
<td>( \langle (=</td>
<td>&lt;)\rangle (&gt;</td>
</tr>
</tbody>
</table>

#### Description

**Origin**
Based on the **PEAK** pattern.

**Constraint**
\[ \text{ALL_EQUAL_WIDTH_PEAK}(\text{VARIABLES}) \]

**Argument**
\[ \text{VARIABLES} : \text{collection}(\text{var}−\text{dvar}) \]

**Restriction**
\[ \text{required}(\text{VARIABLES}.\text{var}) \]

#### Purpose
Succeeds if the width of all occurrences of the **PEAK** pattern in the time-series given by the **VARIABLES** collection are the same.

An occurrence of the pattern **PEAK** is the maximal subsequence which matches the regular expression '\( \langle (= | <)\rangle (> | =)\rangle \)'.

Assume that the occurrence of the pattern **PEAK** starts at position \( i \) and ends at position \( j \). The feature **WIDTH** computes the value \( j - i \).

**Example**
\[ (\langle 1, 2, 2, 7, 7, 1, 2, 4, 5, 7, 1, 2, 3, 6, 7, 1 \rangle) \]

Figure 4.133 provides an example where the **ALL_EQUAL_WIDTH_PEAK** \( \langle [1, 2, 2, 7, 7, 1, 2, 4, 5, 7, 1, 2, 3, 6, 7, 1] \rangle \) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 4.133: Illustrating the ALL_EQUAL_WIDTH_PEAK constraint of the Example slot.
Figure 4.134 depicts the automaton associated with the constraint \( \text{ALL\_EQUAL\_WIDTH\_PEAK} \).

Automaton

\[
\begin{align*}
&\{ C \leftarrow X \\
&D \leftarrow 0 \\
&P \leftarrow X \\
&R \leftarrow 1 \\
&L \leftarrow 1 \\
&\geq s \\
&\leq r \\
&\geq t
\end{align*}
\]

\[
\begin{align*}
&R \land (F = C) \\
&\geq
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + 1 \\
&D \leftarrow 0 \\
\}
\end{align*}
\]

\[
\begin{align*}
&D \leftarrow 0 \\
&P \leftarrow C \\
&R \leftarrow R \land (F = C) \\
\}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow C + D + 1 \\
&D \leftarrow 0 \\
\}
\end{align*}
\]

\[
\begin{align*}
&D \leftarrow 0 \\
&P \leftarrow C \\
&R \leftarrow R \land (F = C) \\
\}
\end{align*}
\]

Figure 4.134: Automaton for the \( \text{ALL\_EQUAL\_WIDTH\_PEAK} \) constraint obtained by applying decoration Table 3.38 to the seed transducer of the PEAK pattern.
**ALL_EQUAL_WIDTH_PLAIN**

**Description**

Based on the *PLAIN* pattern.

**Constraint**

```
ALL_EQUAL_WIDTH_PLAIN(VARIABLES)
```

**Argument**

```
VARIABLES : collection(var−dvar)
```

**Restriction**

```
required(VARIABLES.var)
```

Succeeds if the width of all occurrences of the *PLAIN* pattern in the time-series given by the VARIABLES collection are the same.

Purpose

An occurrence of the pattern *PLAIN* is the *maximal* subsequence which matches the regular expression ‘\( \geq \leq \)’.

Assume that the occurrence of the pattern *PLAIN* starts at position \( i \) and ends at position \( j \). The feature \( \text{WIDTH} \) computes the value \( j - i \).

**Example**

\[
(\langle 1, 6, 3, 3, 7, 6, 3, 3, 7, 6, 3, 3, 5, 4, 3, 3, 6, 3 \rangle)
\]

Figure 4.135 provides an example where the *ALL_EQUAL_WIDTH_PLAIN* \(((1, 6, 3, 3, 7, 6, 3, 3, 7, 6, 3, 3, 5, 4, 3, 3, 6, 3))\) constraint holds.

**Typical**

```
|VARIABLES| > 2
range(VARIABLES.var) > 1
```
Figure 4.135: Illustrating the ALL_EQUAL_WIDTH_PLAIN constraint of the Example slot
Figure 4.136 depicts the automaton associated with the constraint **ALL_EQUAL_WIDTHPLAIN**.

Figure 4.136: Automaton for the **ALL_EQUAL_WIDTHPLAIN** constraint obtained by applying decoration Table 3.38 to the seed transducer of the **PLAIN** pattern.
### ALL_EQUAL_WIDTH_PLATEAU

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_WIDTH_PLATEAU</td>
<td>(&lt;=)&gt;</td>
<td>(&lt;=)&gt;</td>
</tr>
</tbody>
</table>

#### Origin
Based on the PLATEAU pattern.

#### Constraint
\[ \text{ALL_EQUAL_WIDTH_PLATEAU(VARIABLES)} \]

#### Argument
\[ \text{VARIABLES : collection(var - dvar)} \]

#### Restriction
\[ \text{required(VARIABLES, var)} \]

Succeeds if the width of all occurrences of the PLATEAU pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**
An occurrence of the pattern PLATEAU is the *maximal* subsequence which matches the regular expression \(<=\>\). Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature \text{WIDTH} computes the value \(j - i\).

**Example**
\[ ((7, 2, 5, 5, 1, 2, 2, 5, 3, 3, 4, 5, 5, 2, 5)) \]

Figure 4.137 provides an example where the \text{ALL_EQUAL_WIDTH_PLATEAU} \((\{7, 2, 5, 5, 1, 2, 2, 5, 3, 3, 4, 5, 5, 2, 5\})\) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 2 \]
\[ \text{range(VARIABLES.var)} > 1 \]
Figure 4.137: Illustrating the ALL_EQUAL_WIDTH_PLATEAU constraint of the Example slot
Figure 4.138 depicts the automaton associated with the constraint `ALL_EQUAL_WIDTH_PLATEAU`.

Figure 4.138: Automaton for the `ALL_EQUAL_WIDTH_PLATEAU` constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLATEAU pattern.
Based on the `PROPER_PLAIN` pattern.

**Constraint**

\[
\text{ALL}_\text{EQUAL}_\text{WIDTH}_\text{PROPER}_\text{PLAIN}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the width of all occurrences of the `PROPER_PLAIN` pattern in the time-series given by the `VARIABLES` collection are the same.

**Purpose**

An occurrence of the pattern `PROPER_PLAIN` is the maximal subsequence which matches the regular expression `’> =+ <’`.

Assume that the occurrence of the pattern `PROPER_PLAIN` starts at position \(i\) and ends at position \(j\). The feature `WIDTH` computes the value \(j - i\).

**Example**

\[
((2, 7, 3, 6, 6, 3, 7, 3, 3, 5, 6, 5, 3, 3, 5))
\]

Figure 4.139 provides an example where the `ALL_EQUAL_WIDTH_PROPER_PLAIN` ([2, 7, 3, 6, 6, 3, 7, 3, 3, 5, 6, 5, 3, 3, 5]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 3
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.139: Illustrating the ALL_EQUAL_WIDTH_PROPER_PLAIN constraint of the Example slot
Automaton  Figure 4.140 depicts the automaton associated with the constraint *ALL_EQUAL_WIDTH_PROPERPLAIN*. 

Figure 4.140: Automaton for the *ALL_EQUAL_WIDTH_PROPERPLAIN* constraint obtained by applying decoration Table 3.38 to the seed transducer of the *PROPERPLAIN* pattern.
### Origin
Based on the `PROPER_PLATEAU` pattern.

### Constraint
\[
\text{ALL\_EQUAL\_WIDTH\_PROPER\_PLATEAU}(\text{VARIABLES})
\]

### Argument
\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

### Restriction
\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the width of all occurrences of the `PROPER_PLATEAU` pattern in the time-series given by the `VARIABLES` collection are the same.

### Purpose
An occurrence of the pattern `PROPER_PLATEAU` is the maximal subsequence which matches the regular expression `\(<=+>\)`.
Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position \(i\) and ends at position \(j\). The feature `WIDTH` computes the value \(j - i\).

### Example
\[
((7,1,5,5,2,2,5,1,5,5,3,2,3,5,5,3))
\]

Figure 4.141 provides an example where the `ALL\_EQUAL\_WIDTH\_PROPER\_PLATEAU` \(([7,1,5,5,2,2,5,1,5,5,3,2,3,5,5,3])\) constraint holds.

### Typical
\[
|\text{VARIABLES}| > 3
\]
\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.141: Illustrating the \texttt{ALL\_EQUAL\_WIDTH\_PROPER\_PLATEAU} constraint of the \texttt{Example} slot
Automaton  

Figure 4.142 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_WIDTH\_PROPER\_PLATEAU}.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure4_142.png}
\caption{Automaton for the \texttt{ALL\_EQUAL\_WIDTH\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{PROPER\_PLATEAU} pattern}
\end{figure}
ALL_EQUAL_WIDTH_PROPER_PLATEAU
### Origin
Based on the `STEADYSEQUENCE` pattern.

### Constraint
```
ALL_EQUAL_WIDTH_STEADY_SEQUENCE(VARIABLES)
```

### Argument
```
VARIABLES : collection(var−dvar)
```

### Restriction
```
required(VARIABLES, var)
```

### Purpose
Succeeds if the width of all occurrences of the `STEADYSEQUENCE` pattern in the time-series given by the `VARIABLES` collection are the same.

An occurrence of the pattern `STEADYSEQUENCE` is the maximal subsequence which matches the regular expression `”)+`.

Assume that the occurrence of the pattern `STEADYSEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

### Example
```
((3, 5, 5, 3, 1, 4, 4, 4, 6, 5, 3, 2, 2, 2, 5))
```

Figure 4.143 provides an example where the `ALL_EQUAL_WIDTH_STEADY_SEQUENCE` `([3, 5, 5, 3, 1, 4, 4, 4, 6, 5, 3, 2, 2, 2, 5])` constraint holds.

### Typical
```
|VARIABLES| > 1
```
Figure 4.143: Illustrating the ALL_EQUAL_WIDTH_STEADY_SEQUENCE constraint of the Example slot
Figure 4.144: Automaton for the `ALL_EQUAL_WIDTH_STEADY_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `STEADY_SEQUENCE` pattern.
<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the \texttt{STRICTLY_DECREASING_SEQUENCE} pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>\texttt{ALL_EQUAL_WIDTH_STRICTLY_DECREASING_SEQUENCE(VARIABLES)}</td>
</tr>
<tr>
<td>Argument</td>
<td>\texttt{VARIABLES : collection(var–dvar)}</td>
</tr>
<tr>
<td>Restriction</td>
<td>\texttt{required(VARIABLES.var)}</td>
</tr>
</tbody>
</table>

Succeeds if the width of all occurrences of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection are the same.

An occurrence of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} is the \textit{maximal} sub-sequence which matches the regular expression '\texttt{>\+}'.

Assume that the occurrence of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} starts at position $i$ and ends at position $j$. The feature \texttt{WIDTH} computes the value $j - i + 2$.

**Example**

$$((4,3,2,2,4,4,6,6,4,1,1,2,2,4,3,1))$$

Figure 4.145 provides an example where the \texttt{ALL\_EQUAL\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE} ($([4,3,2,2,4,4,6,6,4,1,1,2,2,4,3,1])$) constraint holds.

**Typical**

$$|\texttt{VARIABLES}| > 1$$

$$\texttt{range(VARIABLES.var)} > 1$$
Figure 4.145: Illustrating the \texttt{ALL\_EQUAL\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Figure 4.146 depicts the automaton associated with the constraint ALL_EQUAL_WIDTH_STRICTLY_DECREASING_SEQUENCE.

\[
\begin{align*}
&C \leftarrow X \\
&D \leftarrow 0 \\
&F \leftarrow X \\
&R \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
&D \leftarrow 0 \\
&F \leftarrow C \\
&R \leftarrow R \wedge (F = C)
\end{align*}
\]

\[
\begin{align*}
&C \leftarrow D + 2 \\
&D \leftarrow 0
\end{align*}
\]

\[
\begin{align*}
&D \leftarrow 0 \\
&F \leftarrow C \\
&R \leftarrow R \wedge (F = C)
\end{align*}
\]

\[
\begin{align*}
&C \leftarrow C + D + 1 \\
&D \leftarrow 0
\end{align*}
\]

Figure 4.146: Automaton for the ALL_EQUAL_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern.
**ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DESCRIPTION AUTOMATON**

**Origin**
Based on the **STRICTLY_INCREASING_SEQUENCE** pattern.

**Constraint**

\[
\text{ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} \ldots \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the width of all occurrences of the **STRICTLY_INCREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection are the same.

**Purpose**

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the *maximal* sub-sequence which matches the regular expression ‘<+’.

Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i + 2\).

**Example**

\[
((6, 3, 3, 4, 5, 6, 6, 3, 1, 1, 2, 3, 5, 5, 4, 3))
\]

Figure 4.147 provides an example where the **ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE** \(([6, 3, 3, 4, 5, 6, 6, 3, 1, 1, 2, 3, 5, 5, 4, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.147: Illustrating the ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Figure 4.148 depicts the automaton associated with the constraint \textit{ALL\_EQUAL\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}.

\begin{align*}
\{ & C \leftarrow X \\
& D \leftarrow 0 \\
& F \leftarrow X \\
& R \leftarrow 1 \} \\
\geq & \quad s \\
\geq & \{ \} \\
\geq & \{ C \leftarrow D + 2 \}
\end{align*}

\begin{align*}
\{ & D \leftarrow 0 \\
& F \leftarrow C \\
& R \leftarrow R \land (F = C) \} \\
\geq & \{ \} \\
\geq & \{ C \leftarrow D + 1 \}
\end{align*}

Figure 4.148: Automaton for the \textit{ALL\_EQUAL\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textit{STRICTLY\_INCREASING\_SEQUENCE} pattern.
ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE
### ALL_EQUAL_WIDTH_SUMMIT

**Description**: Based on the SUMMIT pattern.

**Constraint**: ALL_EQUAL_WIDTH_SUMMIT(VARIABLES)

**Argument**: VARIABLES : collection(var−dvar)

**Restriction**: required(VARIABLES, var)

**Purpose**: Succeeds if the width of all occurrences of the SUMMIT pattern in the time-series given by the VARIABLES collection are the same. An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression ‘(< | < (|=|)< )>(>||=)> )’. Assume that the occurrence of the pattern SUMMIT starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$.

**Example**: $\langle 1,3,5,4,2,6,6,2,3,5,4,1,4,5,3,2 \rangle$

Figure 4.149 provides an example where the ALL_EQUAL_WIDTH_SUMMIT ($\langle 1,3,5,4,2,6,6,2,3,5,4,1,4,5,3,2 \rangle$) constraint holds.

**Typical**:
- $|\text{VARIABLES}| > 2$
- $\text{range}(\text{VARIABLES.var}) > 1$
Figure 4.149: Illustrating the \texttt{ALL_EQUAL_WIDTH_SUMMIT} constraint of the \texttt{Example} slot
Figure 4.150 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_WIDTH_SUMMIT}.

Figure 4.150: Automaton for the \texttt{ALL_EQUAL_WIDTH_SUMMIT} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{SUMMIT} pattern (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
### ALL_EQUAL_WIDTH_VALLEY

**Description Automaton**

**Origin**
Based on the `VALLEY` pattern.

**Constraint**

\[
\text{ALL_EQUAL_WIDTH_VALLEY}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the width of all occurrences of the `VALLEY` pattern in the time-series given by the `VARIABLES` collection are the same.

**Purpose**

An occurrence of the pattern `VALLEY` is the *maximal* subsequence which matches the regular expression \( > (= | >) (| =)* < \).

Assume that the occurrence of the pattern `VALLEY` starts at position \( i \) and ends at position \( j \). The feature `WIDTH` computes the value \( j - i \).

**Example**

\[
((7,6,6,1,1,7,6,4,3,1,7,6,5,2,1,7))
\]

Figure 4.151 provides an example where the `ALL_EQUAL_WIDTH_VALLEY` ((7, 6, 6, 1, 1, 7, 6, 4, 3, 1, 7, 6, 5, 2, 1, 7)) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2
\] \[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.151: Illustrating the ALL_EQUAL_WIDTH_VALLEY constraint of the Example slot
Figure 4.152 depicts the automaton associated with the constraint ALL_EQUAL_WIDTH_VALLEY.

\[
\begin{align*}
\text{Automaton} & \quad \text{Figure 4.152: Automaton for the \text{ALL_EQUAL_WIDTH_VALLEY} constraint obtained by applying decoration Table 3.38 to the seed transducer of the } \text{VALLEY} \text{ pattern}
\end{align*}
\]
<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_EQUAL_WIDTH_ZIGZAG</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Description

**Origin**

Based on the ZIGZAG pattern.

**Constraint**

\[
\text{ALL_EQUAL_WIDTH_ZIGZAG}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES : collection(var - dvar)}
\]

**Restriction**

\[
\text{required(\text{VARIABLES.var})}
\]

Succeeds if the width of all occurrences of the ZIGZAG pattern in the time-series given by the \text{VARIABLES} collection are the same.

**Purpose**

An occurrence of the pattern ZIGZAG is the \textit{maximal} subsequence which matches the regular expression \((<>)^+ (< | <>) | (><)^+ (| ><>)(+ | >>)^+\).

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\[
((3, 4, 1, 4, 3, 4, 2, 6, 3, 2, 0, 3, 1, 2, 3))
\]

Figure 4.153 provides an example where the \text{ALL_EQUAL_WIDTH_ZIGZAG}\((\{3, 4, 1, 4, 3, 4, 2, 6, 3, 2, 0, 3, 1, 2, 3\})\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 3
\]

\[
\text{range(\text{VARIABLES.var})} > 1
\]
Figure 4.153: Illustrating the ALL_EQUAL_WIDTH_ZIGZAG constraint of the Example slot
Figure 4.154 depicts the automaton associated with the constraint `ALL_EQUAL_WIDTH_ZIGZAG`.
Figure 4.154: Automaton for the ALL_EQUAL_WIDTH_ZIGZAG constraint obtained by applying decoration Table 3.38 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from $a,b,c,d,e,f$ to $s$ are labelled by $=; (2)$ on transitions from $b,c,e,f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c,f$ to $s$ the register $F$ is reset to $C$, and the register $R$ is updated wrt $C$ and $F$.
DECREASING_HEIGHT_DECREASING_TERRACE

**Origin**
Based on the **DECREASING_TERRACE** pattern.

**Constraint**
**DECREASING_HEIGHT_DECREASING_TERRACE**(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the minima of the values in each occurrence of the **DECREASING_TERRACE** pattern in the time-series given by the VARIABLES collection are decreasing. An occurrence of the pattern **DECREASING_TERRACE** is the maximal subsequence which matches the regular expression '‘\(> = + >\)’.

Assume that the occurrence of the pattern **DECREASING_TERRACE** starts at position \(i\) and ends at position \(j\). The feature **MIN**, called **HEIGHT** in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\(((7, 6, 6, 4, 3, 3, 2, 2, 4, 4, 6, 3, 3, 1, 1))\)

Figure 4.155 provides an example where the **DECREASING_HEIGHT_DECREASING_TERRACE** \(((7, 6, 6, 4, 3, 3, 2, 2, 4, 4, 6, 3, 3, 1, 1))\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 3\)

\(\text{range}(\text{VARIABLES}.var) > 2\)
DECREASING HEIGHT DECREASING TERRACE

Figure 4.155: Illustrating the DECREASING_HEIGHT_DECREASING_TERRACE constraint of the Example slot
Automaton

Figure 4.156 depicts the automaton associated with the constraint `DECREASING_HEIGHT_DECREASING_TERRACE`.

\[
\begin{align*}
C &\leftarrow +\infty \\
D &\leftarrow +\infty \\
F &\leftarrow +\infty \\
R &\leftarrow 1
\end{align*}
\]

\[
\begin{align*}
&D = \min(D, \text{VAR}_i) \\
&C = \min(D, \text{VAR}_i) \\
&D = +\infty \\
&F = \min(D, \text{VAR}_i) \\
&R = R \land (F \geq \min(D, \text{VAR}_i))
\end{align*}
\]

Figure 4.156: Automaton for the `DECREASING_HEIGHT_DECREASING_TERRACE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `DECREASING_TERRACE` pattern.
### DESCRIPTION AUTOMATON

#### Origin
Based on the `INCREASING_TERRACE` pattern.

#### Constraint
`DECREASING_HEIGHT_INCREASING_TERRACE(VARIABLES)`

#### Argument
`VARIABLES : collection(var−dvar)`

#### Restriction
`required(VARIABLES.var)`

Succeeds if the minima of the values in each occurrence of the `INCREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection are decreasing.

An occurrence of the pattern `INCREASING_TERRACE` is the maximal subsequence which matches the regular expression `'<=+<'`.

#### Purpose
Assume that the occurrence of the pattern `INCREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `MIN`, called `HEIGHT` in the name of the constraint since all feature values are identical, computes the minimum of the values from index `i + 1` to index `j`.

#### Example
```plaintext
((2, 5, 5, 6, 2, 3, 4, 4, 5, 3, 1, 2, 2, 3))
```

Figure 4.157 provides an example where the `DECREASING_HEIGHT_INCREASING_TERRACE` constraint holds.

#### Typical
```plaintext
|VARIABLES| > 3  
range(VARIABLES.var) > 2
```
Figure 4.157: Illustrating the DECREASING_HEIGHT_INCREASING_TERRACE constraint of the Example slot
Figure 4.158 depicts the automaton associated with the constraint DECREASING_HEIGHT_INCREASING_TERRACE.

Automaton

Figure 4.158: Automaton for the DECREASING_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING_TERRACE pattern.
DECREASING_HEIGHT_INCREASING_TERRACE
### DECREASING_HEIGHTPLAIN

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ ▷ ◀ ◁</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

#### Origin
Based on the PLAIN pattern.

#### Constraint
DECREASING_HEIGHTPLAIN(VARIABLES)

#### Argument
VARIABLES : collection(var–dvar)

#### Restriction
required(VARIABLES.var)

Succeeds if the minima of the values in each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection are decreasing.

#### Purpose
An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘>`=' '<' .

Assume that the occurrence of the pattern PLAIN starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i + 1$ to index $j$.

#### Example

$((2,3,6,5,5,7,6,6,4,5,5,4,3,6,6,3))$

Figure 4.159 provides an example where the DECREASING_HEIGHTPLAIN ($[2,3,6,5,5,7,6,6,4,5,5,4,3,6,6,3]$) constraint holds.

#### Typical

- $|\text{VARIABLES}| > 2$
- $\text{range}(\text{VARIABLES}.\text{var}) > 1$
Figure 4.159: Illustrating the DECREASING_HEIGHT_PLAIN constraint of the Example slot
Automaton

Figure 4.160 depicts the automaton associated with the constraint DECREASING_HEIGHT_PLAIN.

Figure 4.160: Automaton for the DECREASING_HEIGHT_PLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLAIN pattern.
### DECREASING_HEIGHT_PLATEAU

#### Description

- **Origin**: Based on the PLATEAU pattern.
- **Constraint**: \( \text{DECREASING\_HEIGHT\_PLATEAU}(\text{VARIABLES}) \)
- **Argument**: \( \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \)
- **Restriction**: \( \text{required}(\text{VARIABLES}.\text{var}) \)

#### Purpose

Succeeds if the minima of the values in each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression ‘< = * >’.

Assume that the occurrence of the pattern PLATEAU starts at position \( i \) and ends at position \( j \). The feature \( \text{MIN} \), called \( \text{HEIGHT} \) in the name of the constraint since all feature values are identical, computes the minimum of the values from index \( i+1 \) to index \( j \).

#### Example

\[
(\langle 5,2,5,5,4,3,3,4,2,2,1,3,2,5,7 \rangle)
\]

Figure 4.161 provides an example where the \( \text{DECREASING\_HEIGHT\_PLATEAU}([5,2,2,5,5,4,3,3,4,2,2,1,3,2,5,7]) \) constraint holds.

#### Typical

- \( |\text{VARIABLES}| > 2 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)
Figure 4.161: Illustrating the **DECREASING_HEIGHT_PLATEAU** constraint of the **Example** slot.
Automaton Figure 4.162 depicts the automaton associated with the constraint \textsc{decreasing}\_\textsc{height}\_\textsc{plateau}.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{automaton.png}
\caption{Automaton for the \textsc{decreasing}\_\textsc{height}\_\textsc{plateau} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{plateau} pattern}
\end{figure}
<table>
<thead>
<tr>
<th>Feature</th>
<th>Pattern</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_HEIGHT_PROPER.PLAIN</td>
<td>PROPER.PLAIN</td>
<td></td>
</tr>
</tbody>
</table>

**Description**

**Origin**

Based on the PROPER.PLAIN pattern.

**Constraint**

DECREASING_HEIGHT_PROPER.PLAIN(VARIABLES)

**Argument**

VARIABLES : collection(var-dvar)

**Restriction**

required(VARIABLES, var)

**Purpose**

Succeeds if the minima of the values in each occurrence of the PROPER.PLAIN pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PROPER.PLAIN is the maximal subsequence which matches the regular expression ‘>=+<’.

Assume that the occurrence of the pattern PROPER.PLAIN starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i+1$ to index $j$.

**Example**

$((2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5))$

Figure 4.163 provides an example where the DECREASING_HEIGHT_PROPER.PLAIN ($[(2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5)]$) constraint holds.

**Typical**

$|\text{VARIABLES}| > 3$

$\text{range}(|\text{VARIABLES}.\text{var}|) > 1$
Figure 4.163: Illustrating the DECREASING_HEIGHT_PROPER_PLAIN constraint of the Example slot
Automaton

Figure 4.164 depicts the automaton associated with the constraint DECREASING_HEIGHT_PROPER_PLAIN.

\[
\begin{align*}
C &\leftarrow +\infty \\
D &\leftarrow +\infty \\
F &\leftarrow +\infty \\
R &\leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
\{D \leftarrow \min(D, \text{VAR}_i)\} &\quad \leq \quad \leq \quad \leq \\
\{D \leftarrow +\infty\} &\quad = \quad > \quad > \\
\{D \leftarrow \min(D, \text{VAR}_i)\} &\quad = \quad > \quad > \\
\end{align*}
\]

\[
\begin{align*}
R \land (F \geq C) &\quad \leq \quad \leq \\
\end{align*}
\]

Figure 4.164: Automaton for the DECREASING_HEIGHT_PROPER_PLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PROPER_PLAIN pattern
DECREASING_HEIGHT_PROPER_PLATEAU

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_HEIGHT_PROPER_PLATEAU</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Description**: Based on the PROPER_PLATEAU pattern.

**Constraint**: DECREASING_HEIGHT_PROPER_PLATEAU(VARIABLES)

**Argument**: VARIABLES : collection(var−dvar)

**Restriction**: required(VARIABLES.var)

Succeeds if the minima of the values in each occurrence of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression ‘< =+ >’.

**Purpose**: Assume that the occurrence of the pattern PROPER_PLATEAU starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i + 1$ to index $j$.

**Example**: ((3, 5, 5, 3, 2, 3, 4, 1, 5, 2, 3, 3, 1, 7))

Figure 4.165 provides an example where the DECREASING_HEIGHT_PROPER_PLATEAU (([3, 5, 5, 3, 2, 3, 4, 1, 5, 2, 3, 3, 1, 7])) constraint holds.

**Typical**

| VARIABLES | > 3
range(VARIABLES.var) | > 1
Figure 4.165: Illustrating the DECREASING_HEIGHT_PROPER_PLATEAU constraint of the Example slot
Automaton

Figure 4.166 depicts the automaton associated with the constraint DECREASING_HEIGHT_PROPER_PLATEAU.

Figure 4.166: Automaton for the DECREASING_HEIGHT_PROPER_PLATEAU constraint obtained by applying decoration Table 3.38 to the seed transducer of the PROPER_PLATEAU pattern.
### Condition

**DECREASING_HEIGHT_STEADY**

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_HEIGHT_STEADY</td>
<td>HEIGHT</td>
<td>STEADY</td>
</tr>
</tbody>
</table>

### Description

**Origin**

Based on the **STEADY** pattern.

**Constraint**

`DECREASING_HEIGHT_STEADY(VARIABLES)`

**Argument**

`VARIABLES : collection(var−dvar)`

**Restriction**

`required(VARIABLES, var)`

**Purpose**

Succeeds if the minima of the values in each occurrence of the **STEADY** pattern in the time-series given by the **VARIABLES** collection are decreasing.

An occurrence of the pattern **STEADY** is the subsequence which matches the regular expression `='`.

Assume that the occurrence of the pattern **STEADY** starts at position `i` and ends at position `j`. The feature `MIN`, called **HEIGHT** in the name of the constraint since all feature values are identical, computes the minimum of the values from index `i` to index `j + 1`.

**Example**

```plaintext
((3, 6, 6, 2, 7, 5, 5, 6, 5, 5, 3, 3, 7))
```

Figure 4.167 provides an example where the **DECREASING_HEIGHT_STEADY** `([3, 6, 6, 2, 7, 5, 5, 6, 5, 5, 3, 3, 7])` constraint holds.

**Typical**

`|VARIABLES| > 1`
Figure 4.167: Illustrating the DECREASING_HEIGHT_STEADY constraint of the Example slot
Figure 4.168 depicts the automaton associated with the constraint DECREASING_HEIGHT_STEADY.

Figure 4.168: Automaton for the DECREASING_HEIGHT_STEADY constraint obtained by applying decoration Table 3.38 to the seed transducer of the STEADY pattern.
DECREASING_HEIGHT_STEADY
### Origin
Based on the **STEADY_SEQUENCE** pattern.

### Constraint
| **DECREASING_HEIGHT_STEADY_SEQUENCE(VARIABLES)** |

### Argument
| VARIABLES : collection(var−dvar) |

### Restriction
| required(VARIABLES,var) |

### Purpose
Succeeds if the minima of the values in each occurrence of the **STEADY_SEQUENCE** pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern **STEADY_SEQUENCE** is the maximal subsequence which matches the regular expression '\(^{-}+\)'.

Assume that the occurrence of the pattern **STEADY_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature \(\text{MIN}\), called **HEIGHT** in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i\) to index \(j+1\).

### Example

| ((4,3,2,5,5,5,3,2,4,4,6,5,3,2,2,6)) |

Figure 4.169 provides an example where the **DECREASING_HEIGHT_STEADY_SEQUENCE** ((4,3,2,5,5,5,3,2,4,4,6,5,3,2,2,6)) constraint holds.

### Typical
| | \(|\text{VARIABLES}| > 1\) |
DECREASING_HEIGHT_STEADY_SEQUENCE

Figure 4.169: Illustrating the DECREASING_HEIGHT_STEADY_SEQUENCE constraint of the Example slot
**Automaton**

Figure 4.170 depicts the automaton associated with the constraint `DECREASING_HEIGHT_STEADY_SEQUENCE`.

\[
\begin{align*}
C &\leftarrow +\infty \\
D &\leftarrow +\infty \\
F &\leftarrow +\infty \\
R &\leftarrow 1 \\
\end{align*}
\]

Figure 4.170: Automaton for the `DECREASING_HEIGHT_STEADY_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `STEADY_SEQUENCE` pattern.
DECREASING_HEIGHT_STEADYSEQUENCE
### Origin
Based on the `BUMP_ON_DECREASING_SEQUENCE` pattern.

### Constraint
`DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)`

### Argument
`VARIABLES : collection(var-dvar)`

### Restriction
`required(VARIABLES.var)`

Succeeds if the maxima of the values in each occurrence of the `BUMP_ON_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are decreasing.

### Purpose
An occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` is the subsequence which matches the regular expression `\'>\'>\'<\'>`. Assume that the occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `MAX` computes the maximum of the values from index $i + 2$ to index $j$.

### Example
```
((7, 6, 5, 6, 5, 4, 1, 2, 7, 5, 4, 2, 5, 4, 3, 3))
```

Figure 4.171 provides an example where the `DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE` constraint holds.

### Typical
- `|VARIABLES| > 5`
- `range(VARIABLES.var) > 2`
Figure 4.171: Illustrating the DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.172 depicts the automaton associated with the constraint \textsc{decreasing\_max\_bump\_on\_decreasing\_sequence}.

\[
\begin{align*}
\{ & C \leftarrow +\infty, \\
& D \leftarrow -\infty, \\
& F \leftarrow +\infty, \\
& R \leftarrow 1 \} \\
\{ & D \leftarrow \text{max}(D, \text{VAR}_i) \\
& R \leftarrow R \land (F \geq \text{max}(D, \text{VAR}_i)) \} \\
\{ & D \leftarrow -\infty \} \\
\{ & C \leftarrow \text{max}(D, \text{VAR}_i) \\
& D \leftarrow -\infty \\
& F \leftarrow \text{max}(D, \text{VAR}_i) \\
& R \leftarrow R \land (F \geq \text{max}(D, \text{VAR}_i)) \} \\
\{ & D \leftarrow -\infty \} \\
\{ & R \land (F \geq C) \}
\end{align*}
\]
DECREASING_MAX_DECENDING

**CONDITION**
- **Feature**: DECREASING
- **Pattern**: MAX

**DESCRIPTION AUTOMATON**

**Origin**
Based on the DECREASING pattern.

**Constraint**
DECREASING_MAX_DECENDING(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the maxima of the values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression ‘>’. Assume that the occurrence of the pattern DECREASING starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i$ to index $j + 1$.

**Example**

$$\langle 1, 1, 3, 4, 4, 5, 7, 7, 4, 4, 2, 2, 2, 5, 5 \rangle$$

Figure 4.173 provides an example where the DECREASING_MAX_DECENDING ($\langle 1, 1, 3, 4, 4, 5, 7, 7, 5, 4, 2, 2, 5, 5 \rangle$) constraint holds.

**Typical**

$|\text{VARIABLES}| > 1$

$\text{range}(\text{VARIABLES, var}) > 1$
Figure 4.173: Illustrating the DECREASING_MAX_DECREASING constraint of the Example slot
Automaton Figure 4.174 depicts the automaton associated with the constraint DECREASING_MAX_DECREASING.

Figure 4.174: Automaton for the DECREASING_MAX_DECREASING constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING pattern
DECREASING_MAX_DECREASING

505
DECREASING_MAX DECREASING_SEQUENCE

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**
DECREASING_MAX DECREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the maxima of the values in each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘> (> | =)* > | >’. Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i \) to index \( j + 1 \).

**Example**
\((4, 7, 4, 2, 3, 4, 4, 6, 5, 4, 3, 2, 2, 3, 2)\)

Figure 4.175 provides an example where the DECREASING_MAX DECREASING_SEQUENCE \(([4, 7, 4, 2, 3, 4, 4, 6, 5, 4, 3, 2, 2, 3, 2])\) constraint holds.

**Typical**
|VARIABLES| > 1
range(VARIABLES.var) > 1
Figure 4.175: Illustrating the DECREASING_MAX_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.176 depicts the automaton associated with the constraint DECREASING_MAX_DECREASING_SEQUENCE.

\[
\begin{align*}
\{ & C \leftarrow +\infty \\
& D \leftarrow -\infty \\
& F \leftarrow +\infty \\
& R \leftarrow 1 \}\ 
\end{align*}
\]

\[
\begin{align*}
\leq s & \leq
\end{align*}
\]

\[
\begin{align*}
\{ & D \leftarrow -\infty \\
& F \leftarrow C \\
& R \leftarrow R \land (F \geq C) \}
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow \max(D, \text{VAR}_i), \text{VAR}_{i+1} \}
\end{align*}
\]

\[
\begin{align*}
\leq t \leq
\end{align*}
\]

\[
\begin{align*}
\{ & D \leftarrow \max(D, \text{VAR}_{i+1}) \}
\end{align*}
\]

\[
\begin{align*}
\leq & \leq
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \\
& D \leftarrow -\infty \}
\end{align*}
\]

\[
\begin{align*}
< & <
\end{align*}
\]

\[
\begin{align*}
\{ & D \leftarrow \max(D, \text{VAR}_{i+1}) \}
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \\
& D \leftarrow -\infty \}
\end{align*}
\]

\[
\begin{align*}
\leq & \leq
\end{align*}
\]

\[
\begin{align*}
\{ & D \leftarrow \max(D, \text{VAR}_{i+1}) \}
\end{align*}
\]

\[
\begin{align*}
\leq & \leq
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \\
& D \leftarrow -\infty \}
\end{align*}
\]

Figure 4.176: Automaton for the DECREASING_MAX_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING_SEQUENCE pattern.
### Origin
Based on the `DIP_ON_INCREASING_SEQUENCE` pattern.

### Constraint
\[
\text{DECREASING\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE(\text{VARIABLES})}
\]

### Argument
\[
\text{VARIABLES} : \text{collection(var\text{-}dvar)}
\]

### Restriction
\[
\text{required(\text{VARIABLES.var})}
\]

Succeeds if the maxima of the values in each occurrence of the `DIP_ON_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are decreasing.

**Purpose**

An occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` is the subsequence that matches the regular expression `’<<><<’`. Assume that the occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `MAX` computes the maximum of the values from index \(i + 2\) to index \(j\).

**Example**

\[
((4, 5, 6, 0, 2, 4, 4, 1, 2, 3, 2, 3, 4, 6, 5, 1))
\]

Figure 4.177 provides an example where the `DECREASING\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE` constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 5
\]
\[
\text{range(\text{VARIABLES.var})} > 2
\]
DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE

Figure 4.177: Illustrating the DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE

Automaton

Figure 4.178 depicts the automaton associated with the constraint DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE.

Figure 4.178: Automaton for the DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern.
### Description Automaton

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_MAX_INCREASING</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Origin
Based on the `INCREASING` pattern.

#### Constraint
`DECREASING_MAX_INCREASING(VARIABLES)`

#### Argument
`VARIABLES : collection(var-dvar)`

#### Restriction
`required(VARIABLES, var)`

#### Purpose
Succeeds if the maxima of the values in each occurrence of the `INCREASING` pattern in the time-series given by the `VARIABLES` collection are decreasing.

An occurrence of the pattern `INCREASING` is the subsequence which matches the regular expression `'<'`.

Assume that the occurrence of the pattern `INCREASING` starts at position $i$ and ends at position $j$. The feature `MAX` computes the maximum of the values from index $i$ to index $j + 1$.

#### Example

```
((4, 4, 5, 5, 4, 3, 3, 1, 4, 2, 4, 3, 3, 1, 2, 1))
```

Figure 4.179 provides an example where the `DECREASING_MAX_INCREASING` constraint holds.

#### Typical

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 4.179: Illustrating the DECREASING_MAX_INCREASING constraint of the Example slot
Figure 4.180 depicts the automaton associated with the constraint `DECREASING_MAX_INCREASING`.

\[
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow -\infty \\
F & \leftarrow +\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
R \land (F \geq C)
\]

Figure 4.180: Automaton for the `DECREASING_MAX_INCREASING` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `INCREASING` pattern.
DECREASING_MAX_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_MAX_INCREASING_SEQUENCE</td>
<td>▶ ▷ ◀ ◆</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the INCREASING_SEQUENCE pattern.

**Constraint**
DECREASING_MAX_INCREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the maxima of the values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘<( | =)* < | <’. Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i \) to index \( j + 1 \).

**Example**
\((3, 4, 5, 5, 4, 3, 3, 1, 2, 2, 4, 3, 3, 1, 3, 1)\)

Figure 4.181 provides an example where the DECREASING_MAX_INCREASING_SEQUENCE \((3, 4, 5, 5, 4, 3, 3, 1, 2, 2, 4, 3, 3, 1, 3, 1)\) constraint holds.

**Typical**
| \( |VARIABLES| > 1\) |
| \( \text{range}(\text{VARIABLES}.\text{var}) > 1\) |
Figure 4.181: Illustrating the DECREASING_MAX_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.182 depicts the automaton associated with the constraint DECREASING_MAX_INCREASING_SEQUENCE.

\[
\begin{align*}
&\{ \begin{array}{l}
C \leftarrow +\infty \\
D \leftarrow -\infty \\
F \leftarrow +\infty \\
R \leftarrow 1
\end{array} \} \\
&\{ \begin{array}{l}
D \leftarrow -\infty \\
F \leftarrow C \\
R \leftarrow R \land (F \geq C)
\end{array} \} \\
&\{ D \leftarrow \max(D, \text{VAR}_{i+1}) \} \\
&\{ C \leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \} \\
&\{ D \leftarrow -\infty \}
\end{align*}
\]

Figure 4.182: Automaton for the DECREASING_MAX_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING_SEQUENCE pattern.
DECREASING_MAX_INFLEXION

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_MAX_INFLEXION</td>
<td>▶ ▶ ◀ ◀</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

Description Automaton

Origin
Based on the INFLEXION pattern.

Constraint
DECREASING_MAX_INFLEXION(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES.var)

Purpose
Succeeds if the maxima of the values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression '\<(<?|\=)\*\>|\>(\>|\=)\*\>'

Assume that the occurrence of the pattern INFLEXION starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i + 1$ to index $j$.

Example

$((3, 4, 3, 2, 2, 1, 2, 2, 5, 5))$

Figure 4.183 provides an example where the DECREASING_MAX_INFLEXION ($(3, 4, 3, 2, 2, 1, 2, 2, 5, 5)$) constraint holds.

Typical

$|\text{VARIABLES}| > 2$

$\text{range}(\text{VARIABLES.var}) > 1$
Figure 4.183: Illustrating the DECREASING_MAX_INFLEXION constraint of the Example slot
Figure 4.184 depicts the automaton associated with the constraint \textit{DECREASING\_MAX\_INFLEXION}.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{automaton}
\caption{Automaton for the \textit{DECREASING\_MAX\_INFLEXION} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textit{INFLEXION} pattern (transition $r \rightarrow t$ has the same registers updates as transition $t \rightarrow r$)}
\end{figure}
DECREASING_MAX_PEAK

**Origin**  
Based on the **PEAK** pattern.

**Constraint**  
\text{DECREASING\_MAX\_PEAK}(\text{VARIABLES})

**Argument**  
\text{VARIABLES} : \text{collection}(\text{var}−\text{dvar})

**Restriction**  
\text{required}(\text{VARIABLES}.\text{var})

Succeeds if the maxima of the values in each occurrence of the **PEAK** pattern in the time-series given by the \text{VARIABLES} collection are decreasing.

**Purpose**  
An occurrence of the pattern **PEAK** is the maximal subsequence which matches the regular expression '< (\text{=} | <)∗ (>| =)∗ >'.

Assume that the occurrence of the pattern **PEAK** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**  
\((1,6,6,6,2,5,4,3,2,2,3,2,1,5,5,7))\)

Figure 4.185 provides an example where the \text{DECREASING\_MAX\_PEAK} \((\{1,6,6,6,2,5,4,3,2,2,3,2,1,5,5,7\})\) constraint holds.

**Typical**  
\(|\text{VARIABLES}| > 2\)  
\(\text{range}(\text{VARIABLES}.\text{var}) > 1\)
Figure 4.185: Illustrating the DECREASING_MAX_PEAK constraint of the Example slot
Automaton

Figure 4.186 depicts the automaton associated with the constraint DECREASING_MAX_PEAK.

Figure 4.186: Automaton for the DECREASING_MAX_PEAK constraint obtained by applying decoration Table 3.38 to the seed transducer of the PEAK pattern
DECREASING_MAX_STRICTLY_DECREASING_SEQUENCE

Condition: Based on the STRICTLY_DECREASING_SEQUENCE pattern.

Constraint: DECREASING_MAX_STRICTLY_DECREASING_SEQUENCE(VARIABLES)

Argument: VARIABLES : collection(var−dvar)

Restriction: required(VARIABLES.var)

Purpose:
Succeeds if the maxima of the values in each occurrence of the
STRICTLY_DECREASING_SEQUENCE pattern in the time-series given by the
VARIABLES collection are decreasing.
An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal
sub-sequence which matches the regular expression ‘>’. Assume that the occurrence of the pattern
STRICTLY_DECREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the
values from index i to index j + 1.

Example: ((6, 4, 6, 5, 5, 3, 5, 4, 4, 3, 2, 2, 3, 4))

Figure 4.187 provides an example where the DECREASING_MAX_STRICTLY_DECREASING_SEQUENCE ((6, 4, 6, 5, 5, 3, 5, 4, 4, 3, 2, 2, 3, 4))
constraint holds.

Typical:
|VARIABLES| > 1
range(VARIABLES.var) > 1
Figure 4.187: Illustrating the DECREASING_MAX.Strictly_DEECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.188 depicts the automaton associated with the constraint DECREASING_MAX.Strictly DECREASING_SEQUENCE.

\[
\begin{align*}
C &\leftarrow +\infty \\
D &\leftarrow -\infty \\
F &\leftarrow +\infty \\
R &\leftarrow 1
\end{align*}
\]

Figure 4.188: Automaton for the DECREASING_MAX.Strictly DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STRICTLY DECREASING SEQUENCE pattern.
DECREASING_MAX.Strictly_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_MAX.Strictly_INCREASING_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DESCRIPTION AUTOMATON**

**Origin**
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

**Constraint**
DECREASING_MAX.Strictly_INCREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the maxima of the values in each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression '<+'.
Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i$ to index $j + 1$.

**Example**

```
((3, 4, 5, 4, 3, 3, 1, 2, 3, 4, 3, 3, 1, 3, 1))
```

Figure 4.189 provides an example where the DECREASING_MAX.Strictly_INCREASING_SEQUENCE\{(3, 4, 5, 4, 3, 3, 1, 2, 3, 4, 3, 3, 1, 3, 1)\} constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
DECREASING\_MAX\_STRRICTLY\_INCREASING\_SEQUENCE

Figure 4.189: Illustrating the DECREASING\_MAX\_STRRICTLY\_INCREASING\_SEQUENCE constraint of the Example slot
Automaton

Figure 4.190 depicts the automaton associated with the constraint DECREASING_MAX_STRICTLY_INCREASING_SEQUENCE.

\[
\begin{align*}
&\left\{ 
\begin{array}{l}
C \leftarrow +\infty \\
D \leftarrow -\infty \\
F \leftarrow +\infty \\
R \leftarrow 1
\end{array}
\right. \\
\geq s
\end{align*}
\]

\[
\begin{align*}
&\left\{ 
\begin{array}{l}
D \leftarrow -\infty \\
F \leftarrow C \\
R \leftarrow R \land (F \geq C)
\end{array}
\right. \\
\geq
\end{align*}
\]

\[
\begin{align*}
&\left\{ 
\begin{array}{l}
C \leftarrow \max(D, VAR_i), VAR_{i+1} \\
D \leftarrow -\infty
\end{array}
\right. \\
<
\end{align*}
\]

\[
\begin{align*}
&\left\{ 
\begin{array}{l}
C \leftarrow \max(C, \max(D, VAR_{i+1})) \\
D \leftarrow -\infty
\end{array}
\right. \\
<
\end{align*}
\]

Figure 4.190: Automaton for the DECREASING_MAX_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern.
DECREASING_MAX_SUMMIT

Condition

Feature

Pattern

DECREASING_MAX_SUMMIT

▶ ▷ ◄ ◄

DESCRIPTION

AUTOMATON

(< | < (|= | <)* <) (>| > (|= | >)* >)

Origin

Based on the SUMMIT pattern.

Constraint

DECREASING_MAX_SUMMIT(VARIABLES)

Argument

VARIABLES : collection(var−dvar)

Restriction

required(VARIABLES, var)

Purpose

Succeeds if the maxima of the values in each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression ‘(< | < (|= | <)* <) (>| > (|= | >)* >)’.

Assume that the occurrence of the pattern SUMMIT starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 1 to index j.

Example

((1, 4, 4, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 2, 4, 2))

Figure 4.191 provides an example where the DECREASING_MAX_SUMMIT ((1, 4, 4, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 2, 4, 2)) constraint holds.

Typical

|VARIABLES| > 2
range(VARIABLES, var) > 1
Figure 4.191: Illustrating the DECREASING_MAX_SUMMIT constraint of the **Example** slot
Automaton

Figure 4.192 depicts the automaton associated with the constraint DECREASING_MAX_SUMMIT.

Figure 4.192: Automaton for the DECREASING_MAX_SUMMIT constraint obtained by applying decoration Table 3.38 to the seed transducer of the SUMMIT pattern (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_MAX_ZIGZAG</td>
<td>▶ ▶ ◀ ◀</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the ZIGZAG pattern.

**Constraint**
DECREASING_MAX_ZIGZAG(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the maxima of the values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern ZIGZAG is the \textit{maximal} subsequence which matches the regular expression \((<>)^+ (<>^+ | <>) | (<>)^+ (>| <>)^+ | >)\).
Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature \texttt{MAX} computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**
\[(6, 5, 7, 4, 5, 1, 1, 6, 3, 6, 4, 3, 5, 2, 7, 7)\]

Figure 4.193 provides an example where the DECREASING_MAX_ZIGZAG \(([6, 5, 7, 4, 5, 1, 1, 6, 3, 6, 4, 3, 5, 2, 7, 7])\) constraint holds.

**Typical**
\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES, var}) > 1
\]
Figure 4.193: Illustrating the DECREASING_MAX_ZIGZAG constraint of the Example slot
Automaton

Figure 4.194 depicts the automaton associated with the constraint DECREASING_MAX_ZIGZAG.
Figure 4.194: Automaton for the DECREASING_MAX_ZIGZAG constraint obtained by applying decoration Table 3.38 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the register $D$ is reset to its initial value; (3) on transitions from c, f to s the register $F$ is reset to $C$, and the register $R$ is updated wrt $C$ and $F$. 

\[ \begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow -\infty \\
F & \leftarrow +\infty \\
R & \leftarrow 1 \\
\end{align*} \]

\[ s \]

\[ \begin{align*}
\{ C \leftarrow +\infty \} \\
\{ D \leftarrow -\infty \} \\
\{ F \leftarrow +\infty \} \\
\{ R \leftarrow 1 \} \\
\end{align*} \]

\[ < a \]

\[ \{ D \leftarrow < \} \]

\[ \{ D \leftarrow -\infty \} \]

\[ \{ D \leftarrow \max(D, \text{VAR}_i) \} \]

\[ \{ D \leftarrow -\infty \} \]

\[ > d \]

\[ \{ D \leftarrow -\infty \} \]

\[ \{ D \leftarrow \max(D, \text{VAR}_i) \} \]

\[ \{ D \leftarrow -\infty \} \]

\[ < b \]

\[ \{ D \leftarrow -\infty \} \]

\[ \{ D \leftarrow \max(D, \text{VAR}_i) \} \]

\[ \{ D \leftarrow -\infty \} \]

\[ > b \]

\[ \{ D \leftarrow -\infty \} \]

\[ \{ D \leftarrow \max(D, \text{VAR}_i) \} \]

\[ \{ D \leftarrow -\infty \} \]

\[ < f \]

\[ \{ C \leftarrow \max(C, \max(D, \text{VAR}_i)) \} \\
\{ D \leftarrow -\infty \} \]

\[ > f \]

\[ \{ C \leftarrow \max(C, \max(D, \text{VAR}_i)) \} \\
\{ D \leftarrow -\infty \} \]
Origin: Based on the \textit{BUMP\_ON\_DECREASING\_SEQUENCE} pattern.

Constraint: \texttt{DECREASING\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE(VARIABLES)}

Argument: \texttt{VARIABLES: collection(var\_dvar)}

Restriction: \texttt{required(VARIABLES.var)}

Purpose: Succeeds if the minima of the values in each occurrence of the \textit{BUMP\_ON\_DECREASING\_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection are decreasing.

An occurrence of the pattern \textit{BUMP\_ON\_DECREASING\_SEQUENCE} is the subsequence `>><<>.'

Assume that the occurrence of the pattern \textit{BUMP\_ON\_DECREASING\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{MIN} computes the minimum of the values from index \(i + 2\) to index \(j\).

Example: \texttt{((7,6,5,6,4,1,1,5,5,4,2,6,5,4,4,5))}

Figure 4.195 provides an example where the \texttt{DECREASING\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint holds.

Typical: \texttt{|VARIABLES| > 5 }
\texttt{range(VARIABLES.var) > 2}
Figure 4.195: Illustrating the DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
**Automaton**

Figure 4.196 depicts the automaton associated with the constraint **DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE**.
DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE
DECREASING_MIN_DECREASING

**Description**

- **Origin**: Based on the DECREASING pattern.
- **Constraint**: \texttt{DECREASING\_MIN\_DECREASING(VARIABLES)}
- **Argument**: \texttt{VARIABLES : \textit{collection(var-dvar)}}
- **Restriction**: \texttt{required(VARIABLES.var)}

**Purpose**

Succeeds if the minima of the values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression ‘\texttt{>}’.

Assume that the occurrence of the pattern DECREASING starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i$ to index $j+1$.

**Example**

\begin{verbatim}
((4,7,6,4,5,5,6,4,4,3,3,2,2,3,1))
\end{verbatim}

Figure 4.197 provides an example where the \texttt{DECREASING\_MIN\_DECREASING ([4,7,6,4,5,5,6,4,4,3,3,2,2,3,1])} constraint holds.

- **Typical**

  |VARIABLES| > 1
  \texttt{range(VARIABLES.var)} > 1
Figure 4.197: Illustrating the DECREASING_MIN_DECENDING constraint of the Example slot
Automaton Figure 4.198 depicts the automaton associated with the constraint \textsc{DECREASING\_MIN\_DECREASING}.

\begin{equation*}
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow +\infty \\
F & \leftarrow +\infty \\
R & \leftarrow 1 \\
R \land (F \geq C) \\
\end{align*}
\end{equation*}

Figure 4.198: Automaton for the \textsc{DECREASING\_MIN\_DECREASING} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{DECREASING} pattern
DECREASING_MIN_DECREASING
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING</td>
<td>MIN</td>
<td>DECREASING_SEQUENCE</td>
</tr>
</tbody>
</table>

### Description

**Origin**

Based on the `DECREASING_SEQUENCE` pattern.

**Constraint**

`DECREASING_MIN_DECREASING_SEQUENCE(VARIABLES)`

**Argument**

`VARIABLES : collection(var-dvar)`

**Restriction**

`required(VARIABLES.var)`

**Purpose**

Succeeds if the minima of the values in each occurrence of the `DECREASING_MIN_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are decreasing.

An occurrence of the pattern `DECREASING_SEQUENCE` is the *maximal* subsequence which matches the regular expression `'(>|=)*>|'`. Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

```
((4, 7, 6, 4, 5, 5, 6, 5, 4, 4, 2, 3, 1))
```

Figure 4.199 provides an example where the `DECREASING_MIN_DECREASING_SEQUENCE ((4, 7, 6, 4, 5, 5, 6, 5, 4, 4, 2, 3, 1))` constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 4.199: Illustrating the \texttt{DECREASING\_MIN\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Figure 4.200 depicts the automaton associated with the constraint \textsc{decreasing\_min\_decreasing\_sequence}.

\begin{align*}
\text{Automaton} & \quad \text{Figure 4.200} \text{ depicts the automaton associated with the constraint } \textsc{decreasing\_min\_decreasing\_sequence}.

\begin{array}{l}
C \leftarrow +\infty \\
D \leftarrow +\infty \\
F \leftarrow +\infty \\
R \leftarrow 1
\end{array}
\begin{array}{c}
\leq s \\
\leq t \land (F \geq C)
\end{array}
\begin{array}{c}
D \leftarrow +\infty \\
F \leftarrow C \\
R \leftarrow R \land (F \geq C)
\end{array}
\begin{array}{c}
C \leftarrow \min\{\min(D, \text{VAR}_i), \text{VAR}_{i+1}\} \\
D \leftarrow +\infty
\end{array}
\begin{array}{l}
\leq s \\
\leq t \land (F \geq C)
\end{array}
\begin{array}{c}
D \leftarrow +\infty \\
F \leftarrow C \\
R \leftarrow R \land (F \geq C)
\end{array}
\begin{array}{c}
\{D \leftarrow \min(D, \text{VAR}_{i+1})\} \\
= \{\text{VAR}_i \leftarrow \min(D, \text{VAR}_{i+1})\}
\end{array}
\begin{array}{c}
C \leftarrow \min\{\min(D, \text{VAR}_{i+1})\} \\
D \leftarrow +\infty
\end{array}

\end{array}

Figure 4.200: Automaton for the \textsc{decreasing\_min\_decreasing\_sequence} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{decreasing\_sequence} pattern.
DECREASING_MIN_DECREASING_SEQUENCE
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DESCRIPTION**

**AUTOMATON**

**Origin**

Based on the `DIP_ON_INCREASING_SEQUENCE` pattern.

**Constraint**

`DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE(VARIABLES)`

**Argument**

`VARIABLES : collection(var–dvar)`

**Restriction**

`required(VARIABLES.var)`

Succeeds if the minima of the values in each occurrence of the `DIP_ON_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are decreasing.

**Purpose**

An occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` is the subsequence `<<><<<`.

Assume that the occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i + 2` to index `j`.

**Example**

```
((4,5,6,1,2,4,4,1,2,3,0,3,4,6,5,1))
```

Figure 4.201 provides an example where the `DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE` `((4,5,6,1,2,4,4,1,2,3,0,3,4,6,5,1))` constraint holds.

**Typical**

```
|VARIABLES| > 5
range(VARIABLES.var) > 2
```
Figure 4.201: Illustrating the DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.202 depicts the automaton associated with the constraint DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE.

\[
\begin{align*}
\{ & C \leftarrow +\infty, \\
& D \leftarrow +\infty, \\
& F \leftarrow +\infty, \\
& R \leftarrow 1 \} \\
\geq & s \\
\geq & < r \\
\geq & \leq < u \\
\leq & R \land (F \geq C) \\
\geq & \{ D \leftarrow \min(D, VAR_i) \} \\
\geq & \{ D \leftarrow +\infty \} \\
\leq & \{ D \leftarrow \min(D, VAR_i) \} \\
< & \{ D \leftarrow \min(D, VAR_i) \} \\
< & v \\
< & t
\end{align*}
\]

Figure 4.202: Automaton for the DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern.
**DECREASING_MIN_GORGE**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_MIN_GORGE</td>
<td>▸ ▷ ◀ ◄</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the GORGE pattern.

**Constraint**
DECREASING_MIN_GORGE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the minima of the values in each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression 

`(> | > (= | >)°)>(< | < (= | <)°)°'`

Assume that the occurrence of the pattern GORGE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

**Example**

```
((6,2,3,4,5,6,1,1,5,4,2,3,6,1,4,5))
```

Figure 4.203 provides an example where the DECREASING_MIN_GORGE ((6,2,3,4,5,6,1,1,5,4,2,3,6,1,4,5)) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
|\text{range}(\text{VARIABLES.var})| > 1
\]
Figure 4.203: Illustrating the DECREASING_MIN_GORGE constraint of the Example slot
Automaton

Figure 4.204 depicts the automaton associated with the constraint `DECREASING_MIN_GORGE`.

\[
\begin{align*}
\text{If } s \leq s & \Rightarrow \{ D \leftarrow \min(D, \text{VAR}_i) \} \\
\text{If } s > s & \Rightarrow \{ C \leftarrow \min(D, \text{VAR}_i) \} \\
\text{If } \leq s & \Rightarrow \{ D \leftarrow \min(D, \text{VAR}_i) \} \\
\text{If } \leq s & \Rightarrow \{ D \leftarrow +\infty \} \\
\text{If } \leq S & \Rightarrow \{ D \leftarrow +\infty \} \\
\text{If } \leq S & \Rightarrow \{ D \leftarrow +\infty \} \\
\end{align*}
\]

Figure 4.204: Automaton for the `DECREASING_MIN_GORGE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `GORGE` pattern (transition `u \rightarrow r` has the same register update as transition `r \rightarrow u`).
DECREASING_MIN_INCREASING

### Condition

**Variation**

**Pattern**

**DECREASING_MIN_INCREASING**

### Description

**Origin**

Based on the **INCREASING** pattern.

**Constraint**

DECREASING_MIN_INCREASING(VARIABLES)

**Argument**

VARIABLES : collection(var-dvar)

**Restriction**

required(VARIABLES.var)

Succeeds if the minima of the values in each occurrence of the **INCREASING** pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression '\'<\'.

Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i\) to index \(j + 1\).

**Purpose**

**Example**

\[((4, 4, 5, 5, 4, 3, 3, 2, 4, 3, 3, 2, 2, 1, 3, 1))\]

Figure 4.205 provides an example where the **DECREASING_MIN_INCREASING** \[([4, 4, 5, 5, 4, 3, 3, 2, 4, 3, 3, 2, 2, 1, 3, 1])\] constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]

\[\text{range(VARIABLES.var)} > 1\]
Figure 4.205: Illustrating the DECREASING_MIN_INCREASING constraint of the Example slot
Figure 4.206 depicts the automaton associated with the constraint \texttt{DECREASING\_MIN\_INCREASING}.

```
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow +\infty \\
F & \leftarrow +\infty \\
R & \leftarrow 1 \\
\end{align*}
\begin{align*}
C & \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D & \leftarrow +\infty \\
F & \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
R & \leftarrow R \land (F \geq \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}))
\end{align*}
```

Figure 4.206: Automaton for the \texttt{DECREASING\_MIN\_INCREASING} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{INCREASING} pattern
### Description

**Origin**
Based on the **INCREASING_SEQUENCE** pattern.

**Constraint**

\[
\text{DECREASING\_MIN\_INCREASING\_SEQUENCE}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var\_dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

**Purpose**

Succeeds if the minima of the values in each occurrence of the **INCREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection are decreasing. An occurrence of the pattern **INCREASING_SEQUENCE** is the maximal subsequence which matches the regular expression ‘\(< (< | =)^* < | \)'.

Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
\langle 3, 4, 5, 5, 4, 3, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1 \rangle
\]

Figure 4.207 provides an example where the **DECREASING\_MIN\_INCREASING\_SEQUENCE** (\(\langle 3, 4, 5, 5, 4, 3, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1 \rangle\)) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
DECREASING_MIN_INCREASING_SEQUENCE

Figure 4.207: Illustrating the DECREASING_MIN_INCREASING_SEQUENCE constraint of the Example slot
Automaton Figure 4.208 depicts the automaton associated with the constraint DECREASING_MIN_INCREASING_SEQUENCE.

\[
\begin{align*}
&\{ C \leftarrow +\infty \} \\
&\{ D \leftarrow +\infty \} \\
&\{ F \leftarrow +\infty \} \\
&\{ R \leftarrow 1 \} \\
\geq s \quad \geq \\
\{ D \leftarrow +\infty \} \\
\{ F \leftarrow C \} \\
\{ R \leftarrow R \land (F \geq C) \} \\
\leq t \quad \leq \\
\{ D \leftarrow \min(D, \text{VAR}_i+1) \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \} \\
\{ \text{VAR}_i \} \\
\{ \text{VAR}_i+1 \}
\end{align*}
\]

Figure 4.208: Automaton for the DECREASING_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING_SEQUENCE pattern.
<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_MIN_INFLEXION</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Description**

**Origin**
Based on the INFLEXION pattern.

**Constraint**

\[
\text{DECREASING\_MIN\_INFLEXION}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the minima of the values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (< | =) | > | > | =) ^* <\).

**Purpose**

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
((3, 4, 3, 2, 1, 1, 2, 2, 5, 5))
\]

Figure 4.209 provides an example where the DECREASING_MIN_INFLEXION ((3, 4, 3, 2, 1, 1, 2, 2, 5, 5)) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.209: Illustrating the DECREASING_MIN_INFLEXION constraint of the Example slot
Automaton

Figure 4.210 depicts the automaton associated with the constraint `DECREASING_MIN_INFLEXION`.

\[
\begin{align*}
\{ C \leftarrow +\infty \} \\
\{ D \leftarrow +\infty \} \\
\{ F \leftarrow +\infty \} \\
R \leftarrow 1
\end{align*}
\]

\[
R \land (F \geq C)
\]

\[
\begin{align*}
\{ D \leftarrow \min(D, \text{VAR}_i) \} & \quad \leq \quad \{ D \leftarrow \min(D, \text{VAR}_i) \} \\
C \leftarrow \min(D, \text{VAR}_i) \\
D \leftarrow +\infty \\
F \leftarrow \min(D, \text{VAR}_i) \\
R \leftarrow R \land (F \geq \min(D, \text{VAR}_i))
\end{align*}
\]

Figure 4.210: Automaton for the `DECREASING_MIN_INFLEXION` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `INFLEXION` pattern (transition \( r \rightarrow t \) has the same registers updates as transition \( t \rightarrow r \)).
### Condition: DECREASING_MIN_STRICTLY_DECREASING_SEQUENCE

#### Description

**Origin**: Based on the **STRICTLY_DECREASING_SEQUENCE** pattern.

**Constraint**: \( \text{DECREASING\_MIN\_STRICTLY\_DECREASING\_SEQUENCE}(\text{VARIABLES}) \)

**Argument**: \( \text{VARIABLES} : \text{collection}(\text{var-dvar}) \)

**Restriction**: \( \text{required}(\text{VARIABLES.\text{var}}) \)

**Purpose**: Succeeds if the minima of the values in each occurrence of the **STRICTLY_DECREASING_SEQUENCE** pattern in the time-series given by the \( \text{VARIABLES} \) collection are decreasing.

An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the maximal sub-sequence which matches the regular expression ‘\( >^+ \)’.

Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature \( \text{MIN} \) computes the minimum of the values from index \( i \) to index \( j + 1 \).

**Example**: \( \{(6, 5, 6, 5, 4, 4, 3, 6, 2, 4, 3, 2, 2, 3, 4)\} \)

Figure 4.211 provides an example where the **DECREASING_MIN_STRICTLY_DECREASING_SEQUENCE** \( \{6, 5, 6, 5, 4, 4, 3, 6, 2, 4, 3, 2, 2, 3, 4\} \) constraint holds.

**Typical**: \( |\text{VARIABLES}| > 1 \)

\( \text{range}(\text{VARIABLES.\text{var}}) > 1 \)
Figure 4.211: Illustrating the DECREASING_MIN.Strictly_DECREASING_SEQUENCE constraint of the Example slot
Automaton Figure 4.212 depicts the automaton associated with the constraint DECREASING_MIN_STRICTLY_DECREASING_SEQUENCE.

\[
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow +\infty \\
F & \leftarrow +\infty \\
R & \leftarrow 1
\end{align*}
\]

\[\leq s\]
\[
\begin{align*}
C & \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_i + 1) \\
D & \leftarrow +\infty
\end{align*}
\]

\[\leq\]
\[
\begin{align*}
C & \leftarrow \min(C, \min(D, \text{VAR}_i + 1)) \\
D & \leftarrow +\infty
\end{align*}
\]

\[\leq\]
\[
\begin{align*}
D & \leftarrow +\infty \\
F & \leftarrow C \\
R & \leftarrow R \land (F \geq C)
\end{align*}
\]

\[> r \]
\[
\begin{align*}
C & \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_i + 1) \\
D & \leftarrow +\infty
\end{align*}
\]

\[>\]

\[
\begin{align*}
C & \leftarrow \min(C, \min(D, \text{VAR}_i + 1)) \\
D & \leftarrow +\infty
\end{align*}
\]

Figure 4.212: Automaton for the DECREASING_MIN_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern.
Decreasing-min_strictly_decreasing_sequence
**Origin**
Based on the `STRICTLY_INCREASING_SEQUENCE` pattern.

**Constraint**
`DECREASING_MIN_STRICTLY_INCREASING_SEQUENCE(VARIABLES)`

**Argument**
`VARIABLES : collection(var−dvar)`

**Restriction**
`required(VARIABLES.var)`

Succeeds if the minima of the values in each occurrence of the `STRICTLY_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are decreasing.

**Purpose**
An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the **maximal** sub-sequence which matches the regular expression `'<+'`. Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

```
((3, 4, 5, 4, 3, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))
```

Figure 4.213 provides an example where the `DECREASING_MIN_STRICTLY_INCREASING_SEQUENCE` constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 4.213: Illustrating the \textsc{decreasing\_min\_strictly\_increasing\_sequence} constraint of the \textit{Example} slot
Figure 4.214 depicts the automaton associated with the constraint DECREASING_MIN_STRICTLY_INCREASING_SEQUENCE.

\[
\begin{align*}
\left\{ \begin{array}{l}
C &\leftarrow +\infty \\
D &\leftarrow +\infty \\
F &\leftarrow +\infty \\
R &\leftarrow 1
\end{array} \right. \\
\geq s
\end{align*}
\]

\[
\begin{align*}
\left\{ \begin{array}{l}
D &\leftarrow +\infty \\
F &\leftarrow C \\
R &\leftarrow R \land (F \geq C)
\end{array} \right. \\
\geq 8
\end{align*}
\]

\[
\left\{ \begin{array}{l}
C &\leftarrow \min\left(\min(D, \text{VAR}_i), \text{VAR}_{i+1}\right) \\
D &\leftarrow +\infty \\
F &\leftarrow C \\
R &\leftarrow R \land (F \geq C)
\end{array} \right. \\
< R
\]

\[
\left\{ \begin{array}{l}
C &\leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \\
D &\leftarrow +\infty \\
F &\leftarrow C \\
R &\leftarrow R \land (F \geq C)
\end{array} \right. \\
< s
\]

Figure 4.214: Automaton for the DECREASING_MIN_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern.
DECREASING_MIN_STRICTLY_INCREASING_SEQUENCE
DECREASING_MIN_VALLEY

Origin
Based on the VALLEY pattern.

Constraint
DECREASING_MIN_VALLEY(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES.var)

Purpose
Succeeds if the minima of the values in each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern VALLEY is the \textit{maximal} subsequence which matches the regular expression \textquoteleft > (\textquoteleft = | \textgreater \textquoteleft)' \textquoteleft \textless \textquoteleft (\textquoteleft < | \textless \textquoteleft)\textquoteleft \textless \textquoteleft'.

Assume that the occurrence of the pattern VALLEY starts at position \textit{i} and ends at position \textit{j}. The feature MIN computes the minimum of the values from index \textit{i} + 1 to index \textit{j}.

Example
((1,3,3,7,6,5,4,6,6,5,4,3,6,2,2,7))

Figure 4.215 provides an example where the DECREASING_MIN_VALLEY ([1,3,3,7,6,5,4,6,6,5,4,3,6,2,2,7]) constraint holds.

Typical
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
Figure 4.215: Illustrating the DECREASING_MIN_VALLEY constraint of the Example slot.
Automaton Figure 4.216 depicts the automaton associated with the constraint **DECREASING_MIN_VALLEY**.

\[
\begin{align*}
C &\leftarrow +\infty \\
D &\leftarrow +\infty \\
F &\leftarrow +\infty \\
R &\leftarrow 1 \\
\{ D \leftarrow \min(D, \text{VAR}_i) \} &\quad \text{if } s \leq r \\
\{ C \leftarrow \min(C, \min(D, \text{VAR}_i)) \} &\quad \text{if } D \geq +\infty
\end{align*}
\]

Figure 4.216: Automaton for the **DECREASING_MIN_VALLEY** constraint obtained by applying decoration Table 3.38 to the seed transducer of the **VALLEY** pattern.
DECREASING_MIN_ZIGZAG

DECREASING_MIN_ZIGZAG

▶ ▶ ▶ ▶ DESCRIPTION AUTOMATON ▶ ▶ ▶ ▶

Origin
Based on the ZIGZAG pattern.

Constraint
DECREASING_MIN_ZIGZAG(VARIABLES)

Argument
VAR: VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES.var)

Purpose
Succeeds if the minima of the values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘(<>)(< | <>)(<>)(< | <>)’.

Assume that the occurrence of the pattern ZIGZAG starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

Example
((6, 5, 7, 4, 5, 1, 1, 6, 3, 6, 4, 3, 5, 2, 7, 7))

Figure 4.217 provides an example where the DECREASING_MIN_ZIGZAG ((6, 5, 7, 4, 5, 1, 1, 6, 3, 6, 4, 3, 5, 2, 7, 7)) constraint holds.

Typical
|VAR| > 3
range(VARIABLES.var) > 1
Figure 4.217: Illustrating the DECREASING_MIN_ZIGZAG constraint of the Example slot
Automaton Figure 4.218 depicts the automaton associated with the constraint DECREASING_MIN_ZIGZAG.
Figure 4.218: Automaton for the **DECREASING_MIN_ZIGZAG** constraint obtained by applying decoration Table 3.38 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the register $F$ is reset to $C$, and the register $R$ is updated wrt $C$ and $F$
### Description

**Origin**
Based on the **DECREASING** pattern.

**Constraint**

\[
\text{DECREASING\_RANGE\_DECREASING} (\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection} (\text{var-dvar})
\]

**Restriction**

\[
\text{required} (\text{VARIABLES}.\text{var})
\]

Succeeds if the differences between the largest and smallest value in each occurrence of the **DECREASING** pattern in the time-series given by the **VARIABLES** collection are decreasing.

**Purpose**

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression `'>'`.
Assume that the occurrence of the pattern **DECREASING** starts at position \(i\) and ends at position \(j\). The feature **RANGE** computes the range of the values from index \(i\) to index \(j+1\).

**Example**

\[
((4, 8, 6, 4, 5, 5, 8, 6, 6, 5, 5, 4, 4, 6, 5))
\]

Figure 4.219 provides an example where the **DECREASING\_RANGE\_DECREASING** \(((4, 8, 6, 4, 5, 5, 8, 6, 6, 5, 5, 4, 4, 6, 5))\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]
\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.219: Illustrating the DECREASING_RANGE_DECREASING constraint of the Example slot
Automaton. Figure 4.220 depicts the automaton associated with the constraint \textsc{decreasing\_range\_decreasing}.

![Automaton Diagram]

Figure 4.220: Automaton for the \textsc{decreasing\_range\_decreasing} constraint obtained by applying decoration Table 3.49 to the seed transducer of the \textsc{decreasing} pattern.
### Description

**Pattern**: \( \text{DECREASING\_RANGE\_DECREASING\_SEQUENCE} \)

**Constraint**: \( \text{DECREASING\_RANGE\_DECREASING\_SEQUENCE}(\text{VARIABLES}) \)

**Argument**: \( \text{VARIABLES} : \text{collection}(\text{var-dvar}) \)

**Restriction**: \( \text{required}(\text{VARIABLES}.\text{var}) \)

Succeeds if the differences between the largest and smallest value in each occurrence of the \text{DECREASING\_SEQUENCE} pattern in the time-series given by the \text{VARIABLES} collection are decreasing.

**Purpose**

An occurrence of the pattern \text{DECREASING\_SEQUENCE} is the maximal subsequence which matches the regular expression ‘\( (> | =)^* > | > \)’.

Assume that the occurrence of the pattern \text{DECREASING\_SEQUENCE} starts at position \( i \) and ends at position \( j \). The feature \text{RANGE} computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**

\[ ((4, 7, 6, 1, 5, 5, 6, 5, 4, 4, 2, 2, 3, 1)) \]

Figure 4.221 provides an example where the \text{DECREASING\_RANGE\_DECREASING\_SEQUENCE} constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 4.221: Illustrating the **DECREASING_RANGE** \_ **DECREASING_SEQUENCE** constraint of the **Example** slot
Automaton

Figure 4.222 depicts the automaton associated with the constraint \texttt{DECREASING\_RANGE\_DECREASING\_SEQUENCE}.

\begin{align*}
C & \leftarrow +\infty \\
F & \leftarrow +\infty \\
H & \leftarrow \text{VAR}_1 \\
R & \leftarrow 1
\end{align*}

\begin{align*}
G & \leq s \\
\{ H \leftarrow \text{VAR}_{i+1} \} \\
\{ F \leftarrow |H - \text{VAR}_i| \} & \leq \{ C \leftarrow |H - \text{VAR}_{i+1}| \} \\
\{ H \leftarrow \text{VAR}_{i+1} \} & > \{ C \leftarrow |H - \text{VAR}_{i+1}| \} \\
\{ C \leftarrow |H - \text{VAR}_{i+1}| \} & = \{ R \leftarrow R \land (F \geq C) \} \\
H & \leftarrow \text{VAR}_{i+1} \\
R & \leftarrow R \land (F \geq C) \\
F & \leftarrow |H - \text{VAR}_i|
\end{align*}

Figure 4.222: Automaton for the \texttt{DECREASING\_RANGE\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.49 to the seed transducer of the \texttt{DECREASING\_SEQUENCE} pattern.
DECREASING RANGE DECREASING SEQUENCE
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_RANGE_INCREASING</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Description**

- **Origin**: Based on the INCREASING pattern.
- **Constraint**: `DECREASING_RANGE_INCREASING(VARIABLES)`
- **Argument**: `VARIABLES : collection(var - dvar)`
- **Restriction**: `required(VARIABLES.var)`

**Purpose**

Succeeds if the differences between the largest and smallest value in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression '<'.

Assume that the occurrence of the pattern INCREASING starts at position $i$ and ends at position $j$. The feature RANGE computes the range of the values from index $i$ to index $j + 1$.

**Example**

```plaintext
([4, 6, 8, 4, 4, 5, 5, 6, 6, 7, 5, 5, 5, 5, 6, 4])
```

Figure 4.223 provides an example where the `DECREASING_RANGE_INCREASING(([4, 6, 8, 4, 4, 5, 5, 6, 6, 7, 5, 5, 5, 5, 6, 4]))` constraint holds.

**Typical**

- $|\text{VARIABLES}| > 1$
- $\text{range}(\text{VARIABLES.var}) > 1$
Figure 4.223: Illustrating the DECREASING_RANGE_INCREASING constraint of the Example slot
Automaton

Figure 4.224 depicts the automaton associated with the constraint DECREASING_RANGE_INCREASING.

\[
\begin{align*}
C & \leftarrow +\infty \\
F & \leftarrow +\infty \\
H & \leftarrow \text{VAR}_1 \\
R & \leftarrow 1 \\
\end{align*}
\]

\[R \land (F \geq H) \geq \{H \leftarrow \text{VAR}_{i+1}\}\]

\[R \land (F \geq C)\]

Figure 4.224: Automaton for the DECREASING_RANGE_INCREASING constraint obtained by applying decoration Table 3.49 to the seed transducer of the INCREASING pattern.
### DECREASING\_RANGE\_INCREASING\_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶</td>
<td>▶</td>
<td>▶</td>
</tr>
</tbody>
</table>

**Description**

**Origin**

Based on the **INCREASING\_SEQUENCE** pattern.

**Constraint**

**DECREASING\_RANGE\_INCREASING\_SEQUENCE**(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES.var)

**Purpose**

Succeeds if the differences between the largest and smallest value in each occurrence of the **INCREASING\_SEQUENCE** pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern **INCREASING\_SEQUENCE** is the maximal subsequence which matches the regular expression ‘(*(< | =)> | < | ‹)*’. Assume that the occurrence of the pattern **INCREASING\_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **RANGE** computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\(([1, 4, 5, 5, 4, 3, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1])\)

Figure 4.225 provides an example where the **DECREASING\_RANGE\_INCREASING\_SEQUENCE** constraint holds.

**Typical**

\(|\text{VARIABLES}| > 1\)

\(\text{range(VARIABLES.var)} > 1\)
Figure 4.225: Illustrating the DECREASING_RANGE_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.226 depicts the automaton associated with the constraint `DECREASING_RANGE_INCREASING_SEQUENCE`.

\[
\begin{align*}
C &\leftarrow \infty \\
F &\leftarrow \infty \\
H &\leftarrow \text{VAR}_1 \\
R &\leftarrow 1 \\
\end{align*}
\]

\[\geq s\]

\[
\begin{align*}
F &\leftarrow |H - \text{VAR}_i| \\
H &\leftarrow \text{VAR}_{i+1} \\
R &\leftarrow R \land (F \geq C) \\
\end{align*}
\]

\[\leq t\]

\[
\begin{align*}
C &\leftarrow |H - \text{VAR}_{i+1}| \\
\end{align*}
\]

Figure 4.226: Automaton for the `DECREASING_RANGE_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.49 to the seed transducer of the `INCREASING_SEQUENCE` pattern.
### Description

**Origin**
Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**

\[ \text{DECREASING\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE}(\text{VARIABLES}) \]

**Argument**

\[ \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \]

**Restriction**

\[ \text{required}(\text{VARIABLES}.\text{var}) \]

Succeeds if the differences between the largest and smallest value in each occurrence of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are decreasing.

**Purpose**

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the *maximal* subsequence which matches the regular expression `^>`.

Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position \( i \) and ends at position \( j \). The feature `RANGE` computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**

\[ ((6,3,3,5,4,2,7,6,5,6,4,3,3,2,6)) \]

Figure 4.227 provides an example where the `DECREASING\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE` \(([6,3,3,5,4,2,7,6,5,6,4,3,3,2,6])\) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 4.227: Illustrating the \textsc{decreasing\_range}\_\textsc{strictly\_decreasing\_sequence} constraint of the \texttt{Example} slot
Automaton Figure 4.228 depicts the automaton associated with the constraint `DECREASING_RANGE_STRICTLY_DECREASING_SEQUENCE`.

Figure 4.228: Automaton for the `DECREASING_RANGE_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.49 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern.
DECREASING\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE
DECREASING_RANGE STRICTLY INCREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the **STRICTLY_INCREASING_SEQUENCE** pattern.

**Constraint**

\[
\text{DECREASING\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}, \text{var})
\]

Succeeds if the differences between the largest and smallest value in each occurrence of the **STRICTLY_INCREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection are decreasing.

**Purpose**

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the **maximal** sub-sequence which matches the regular expression ‘<+’.

Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **RANGE** computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
((1, 4, 5, 5, 4, 1, 0, 2, 3, 3, 6, 2, 3, 4, 4, 6))
\]

Figure 4.229 provides an example where the **DECREASING\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE** ([1, 4, 5, 5, 4, 1, 0, 2, 3, 3, 6, 2, 3, 4, 4, 6]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.229: Illustrating the \texttt{DECREASING\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Automaton  Figure 4.230 depicts the automaton associated with the constraint DECREASING_RANGE_STRINGLY_INCREASING_SEQUENCE.

\[
\begin{align*}
    C & \leftarrow +\infty \\
    F & \leftarrow +\infty \\
    H & \leftarrow \text{VAR}_1 \\
    R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
    F & \leftarrow |H - \text{VAR}_i| \\
    H & \leftarrow \text{VAR}_{i+1} \\
    R & \leftarrow R \land (F \geq C)
\end{align*}
\]

\[
\begin{align*}
    C & \leftarrow |H - \text{VAR}_{i+1}|
\end{align*}
\]

Figure 4.230: Automaton for the DECREASING_RANGE_STRINGLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.49 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern.
DECREASING\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE
DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
<td>▶ ▶ &lt; ▶</td>
<td>&gt; &gt; &lt; &gt; &gt;</td>
</tr>
</tbody>
</table>

DESCRIPTION AUTOMATON

Origin
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

Constraint
DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

Purpose
An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence ‘>><<<<<<<<’. Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 2$ to index $j$.

Example

```plaintext
((7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3))
```

Figure 4.231 provides an example where the DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE ((7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3)) constraint holds.

Typical

- $|\text{VARIABLES}| > 5$
- $\text{range}(\text{VARIABLES}\_\text{var}) > 2$
DECREASING
SURF
BUMP
ON
DECREASING
SEQUENCE

Figure 4.231: Illustrating the DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.232 depicts the automaton associated with the constraint DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE.

Figure 4.232: Automaton for the DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern.
DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE
DECREASING_SURF_DECREASING

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING</td>
<td>SURF</td>
<td>DECREASING</td>
</tr>
</tbody>
</table>

**Description**

**Automaton**

**Origin**

Based on the **DECREASING** pattern.

**Constraint**

DECREASING_SURF_DECREASING(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the **DECREASING** pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression ‘>’.

Assume that the occurrence of the pattern **DECREASING** starts at position \( i \) and ends at position \( j \). The feature **SURF** computes the sum of the values from index \( i \) to index \( j + 1 \).

**Purpose**

**Example**

\(((2, 6, 5, 2, 3, 3, 3, 2, 4, 4, 5, 7, 7))\)

Figure 4.233 provides an example where the **DECREASING_SURF_DECREASING** \(((2, 6, 5, 2, 3, 3, 4, 3, 3, 2, 4, 4, 5, 7, 7))\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 1\)

\(\text{range(VARIABLES.var)} > 1\)
Figure 4.233: Illustrating the \textbf{DECREASING\_SURF\_DECREASING} constraint of the \textbf{Example} slot
Automaton

Figure 4.234 depicts the automaton associated with the constraint `DECREASING_SURF_DECREASING`.

\[\begin{align*}
\{ & C \leftarrow +\infty \\
& D \leftarrow 0 \\
& F \leftarrow +\infty \\
& R \leftarrow 1 \}
\end{align*}\]

\[\begin{align*}
\leq & S \\
\geq & \end{align*}\]

\[R \land (F \geq C)\]

Figure 4.234: Automaton for the `DECREASING_SURF_DECREASING` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `DECREASING` pattern.
DECREASING_SURF_DECREASING_SEQUENCE

Origin: Based on the DECREASING_SEQUENCE pattern.

Constraint: DECREASING_SURF_DECREASING_SEQUENCE(VARIABLES)

Argument: VARIABLES : collection(var−dvar)

Restriction: required(VARIABLES.var)

Purpose: Succeeds if the values denoting the surface of each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘>(>|=)∗>|>’.
Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

Example: $((2, 6, 5, 2, 3, 3, 4, 3, 2, 4, 4, 5, 7, 5))$

Figure 4.235 provides an example where the DECREASING_SURF_DECREASING_SEQUENCE constraint holds.

Typical: $|\text{VARIABLES}| > 1$
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]
Figure 4.235: Illustrating the DECREASING_SURF_DECREASING_SEQUENCE constraint of the Example slot
Figure 4.236 depicts the automaton associated with the constraint `DECREASING_SURF_DECREASING_SEQUENCE`.

\[
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow 0 \\
F & \leftarrow +\infty \\
R & \leftarrow 1 \\
\end{align*}
\]

\[\leq s\]

\[\leq \]

\[\begin{align*}
D & \leftarrow 0 \\
F & \leftarrow C \\
R & \leftarrow R \land (F \geq C) \\
\end{align*}\]

\[\{ D \leftarrow D + \text{VAR}_i \}\]

\[\leq\]

\[\geq t\]

\[\begin{align*}
C & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
D & \leftarrow 0 \\
\end{align*}\]

\[\{ C \leftarrow C + D + \text{VAR}_{i+1} \}\]

Figure 4.236: Automaton for the `DECREASING_SURF_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `DECREASING_SEQUENCE` pattern.
Origin | Based on the **DECREASING_TERRACE** pattern.
Constraint | **DECREASING_SURF** | **DECREASING** | **TERRACE**<br>(**VARIABLES**)<br>
Argument | **VARIABLES** : collection(var−dvar)
Restriction | required(**VARIABLES.var**)<br>
Purpose | Succeeds if the values denoting the surface of each occurrence of the **DECREASING_TERRACE** pattern in the time-series given by the **VARIABLES** collection are decreasing.<br><br>An occurrence of the pattern **DECREASING_TERRACE** is the *maximal* subsequence which matches the regular expression `'>=+'`.<br><br>Assume that the occurrence of the pattern **DECREASING_TERRACE** starts at position *i* and ends at position *j*. The feature **SURF** computes the sum of the values from index *i* + 1 to index *j*.
Example | `((7,6,6,6,4,3,3,2,2,4,4,6,3,3,1,1))`
Typical | `|**VARIABLES**| > 3`<br>`range(**VARIABLES.var**) > 2`
Figure 4.237: Illustrating the `DECREASING_SURF_DECREASING_TERRACE` constraint of the `Example` slot
Automaton

Figure 4.238 depicts the automaton associated with the constraint DECREASING_SURF_DECRESING_TERRACE.

Figure 4.238: Automaton for the DECREASING_SURF_DECRESING_TERRACE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING_TERRACE pattern.
DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE

**Condition**

**Feature**

**Pattern**

DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE

▶ ▶ ◄ ◄ DESCRIPTION AUTOMATON

---

**Origin**

Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**

DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES.var)

Succeeds if the values denoting the surface of each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence ‘<<><<‘. Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 2 to index j.

**Example**

`((4,5,6,0,3,4,4,1,2,3,2,3,4,6,5,1))`

Figure 4.239 provides an example where the DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE `((4,5,6,0,3,4,4,1,2,3,2,3,4,6,5,1))` constraint holds.

**Typical**

| VARIABLES | > 5

range(VARIABLES.var) > 2
Figure 4.239: Illustrating the DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Figure 4.240 depicts the automaton associated with the constraint DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE.

Figure 4.240: Automaton for the DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern.
DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE
### Condition

**DECREASING_SURF_GORGE**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ ▶ &lt; ◄</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

#### Origin

Based on the GORGE pattern.

#### Constraint

DECREASING_SURF_GORGE(VARIABLES)

#### Argument

VARIABLES : collection(var−dvar)

#### Restriction

required(VARIABLES.var)

Succeeds if the values denoting the surface of each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are decreasing.

#### Purpose

An occurrence of the pattern **GORGE** is the *maximal* subsequence which matches the regular expression "(< | > (= | >)* >)(< | < (= | <)* <)". Assume that the occurrence of the pattern **GORGE** starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

#### Example

```plaintext
((6, 2, 3, 4, 5, 6, 1, 1, 5, 4, 2, 3, 6, 1, 4, 5))
```

Figure 4.241 provides an example where the **DECREASING_SURF_GORGE** ((6, 2, 3, 4, 5, 6, 1, 1, 5, 4, 2, 3, 6, 1, 4, 5)) constraint holds.

#### Typical

- $|\text{VARIABLES}| > 2$
- $\text{range}(\text{VARIABLES}.var) > 1$
Figure 4.241: Illustrating the **DECREASING_SURF_GORGE** constraint of the **Example** slot
Automaton **Figure 4.242** depicts the automaton associated with the constraint DECREASING_SURF_GORGE.

**Figure 4.242:** Automaton for the DECREASING_SURF_GORGE constraint obtained by applying decoration Table 3.38 to the seed transducer of the GORGE pattern (transition \( u \rightarrow r \) has the same register update as transition \( r \rightarrow u \)).
DECREASING_SURF_INCREASING

**CONDITION**

**DECREASING_SURF_INCREASING**

**DESCRIPTION**

**AUTOMATON**

**Origin**

Based on the **INCREASING** pattern.

**Constraint**

DECREASING_SURF_INCREASING(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES.var)

Succeeds if the values denoting the surface of each occurrence of the **INCREASING** pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression '<'.

Assume that the occurrence of the pattern **INCREASING** starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

**Example**

$((4, 4, 5, 4, 3, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))$

Figure 4.243 provides an example where the DECREASING_SURF_INCREASING ($([4, 4, 5, 4, 3, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1])$) constraint holds.

**Typical**

$|\text{VARIABLES}| > 1$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$
DECREASING_SURF_INCREASING

Figure 4.243: Illustrating the DECREASING_SURF_INCREASING constraint of the Example slot
Automaton

Figure 4.244 depicts the automaton associated with the constraint `DECREASING_SURF_INCREASING`.

\[
\begin{align*}
\{ & C \leftarrow +\infty \\
& D \leftarrow 0 \\
& F \leftarrow +\infty \\
& R \leftarrow 1 \\
\} \\
\{ & C \leftarrow D + \text{VAR}_{i} + \text{VAR}_{i+1} \\
& D \leftarrow 0 \\
& F \leftarrow D + \text{VAR}_{i} + \text{VAR}_{i+1} \\
& R \leftarrow R \land (F \geq D + \text{VAR}_{i} + \text{VAR}_{i+1}) \\
\} \\
\end{align*}
\]

Figure 4.244: Automaton for the `DECREASING_SURF_INCREASING` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `INCREASING` pattern.
DECREASING_SURF_INCREASING
### CONDITION  
**DECREASING_SURF_INCREASING_SEQUENCE**

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
</table>

| Origin | Based on the **INCREASING_SEQUENCE** pattern. |
| Constraint | **DECREASING_SURF_INCREASING_SEQUENCE**(VARIABLES) |
| Argument | VARIABLES : collection(var−dvar) |
| Restriction | required(VARIABLES.var) |

Succeeds if the values denoting the surface of each occurrence of the **INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern **INCREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression `< (< | =)∗ < | <`. Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i\) to index \(j+1\).

**Example**

\(((3,4,5,5,4,3,3,1,4,3,3,2,2,1,3,1))\)

Figure 4.245 provides an example where the **DECREASING_SURF_INCREASING_SEQUENCE** ((3, 4, 5, 5, 4, 3, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1)) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]
\[\text{range}(|\text{VARIABLES}.\text{var}|) > 1\]
Figure 4.245: Illustrating the DECREASING_SURF_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.246 depicts the automaton associated with the constraint `DECREASING_SURF_INCREASING_SEQUENCE`.

- δ: Transition function
- Σ: Input alphabet
- Q: State set
- F: Final states
- s: Start state
- t: Final state

- Q = \{q_0, q_1, q_2, q_3\}
- F = \{q_2, q_3\}
- s = q_0
- t = \{q_2, q_3\}

- Transition function δ:
  - δ(q_0, a) = q_1
  - δ(q_1, b) = q_2
  - δ(q_2, c) = q_3
  - δ(q_3, d) = q_3

- Initial state: q_0
- Final states: q_2, q_3

- Input alphabet Σ = \{a, b, c, d\}

Figure 4.246: Automaton for the `DECREASING_SURF_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `INCREASING_SEQUENCE` pattern.
### Origin
Based on the **INCREASING_TERRACE** pattern.

### Constraint
**DECREASING_SURF_INCREASING_TERRACE**(VARIABLES)

### Argument
VARIABLES : collection(var−dvar)

### Restriction
required(VARIABLES.var)

Succeeds if the values denoting the surface of each occurrence of the **INCREASING_TERRACE** pattern in the time-series given by the VARIABLES collection are decreasing.

### Purpose
An occurrence of the pattern **INCREASING_TERRACE** is the maximal subsequence which matches the regular expression ‘\(<=+<\)’. Assume that the occurrence of the pattern **INCREASING_TERRACE** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i + 1\) to index \(j\).

### Example

\[
(2, 5, 5, 6, 2, 3, 4, 4, 5, 3, 3, 1, 2, 2, 3)
\]

Figure 4.247 provides an example where the **DECREASING_SURF_INCREASING_TERRACE** \((2, 5, 5, 6, 2, 3, 4, 4, 5, 3, 3, 1, 2, 2, 3)\) constraint holds.

### Typical

\[
|\text{VARIABLES}| > 3
\]

\[
\text{range(VARIABLES.var)} > 2
\]
Figure 4.247: Illustrating the \textsc{DECREASING\_SURF\_INCREASING\_TERRACE} constraint of the \textbf{Example} slot
Automaton

Figure 4.248 depicts the automaton associated with the constraint DECREASING_SURF_INCREASING_TERRACE.

\[
\begin{align*}
&\left\{ C \leftarrow +\infty \\
&\{ D \leftarrow 0 \} \\
&\{ F \leftarrow +\infty \} \\
&\{ R \leftarrow 1 \} \\
\right. \\
\left. \begin{array}{c}
&\geq s \\
&\geq \\
&\geq \\
&\geq \\
&\geq \\
&\geq \\
&\geq \\
&\geq \\
&\geq \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
&\left\{ D \leftarrow D +\text{VAR}_i \} \\
&\{ \{ D \leftarrow D +\text{VAR}_i \} \\
\right. \\
\left. \begin{array}{c}
&\leq \\
&\leq \\
&\leq \\
&\leq \\
&\leq \\
&\leq \\
&\leq \\
&\leq \\
\end{array}
\end{align*}
\]

Figure 4.248: Automaton for the DECREASING_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING_TERRACE pattern.
DECREASING_SURF_INFLEXION

CONDITION  FEATURE  PATTERN

DECREASING_SURF_INFLEXION

▶ ▷ ◄ ◀  DESCRIPTION  AUTOMATON

< (< | =)> | > (> | =)> <

Origin
Based on the INFLEXION pattern.

Constraint
DECREASING_SURF_INFLEXION(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES,var)

Succeeds if the values denoting the surface of each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are decreasing.

Purpose
An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression ' < (< | =)> | > (> | =)> < '.
Assume that the occurrence of the pattern INFLEXION starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
((6,5,5,4,3,2,3,3,4,6,3,3,1,2,1,1))

Figure 4.249 provides an example where the DECREASING_SURF_INFLEXION ((6,5,5,4,3,2,3,3,4,6,3,3,1,2,1,1)) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES,var) > 1
Figure 4.249: Illustrating the DECREASING_SURF_INFLEXION constraint of the Example slot
Figure 4.250 depicts the automaton associated with the constraint DECREASING_SURF_INFLEXION.

\[
\begin{align*}
&\begin{cases}
C &\leftarrow +\infty \\
D &\leftarrow 0 \\
F &\leftarrow +\infty \\
R &\leftarrow 1
\end{cases} \\
&\leq r \ \geq t \\
&\quad \Rightarrow \\
&\quad \Rightarrow \\
&\quad \Rightarrow \\
&\quad \Rightarrow \\
&\{ D &\leftarrow D + \text{VAR}_i \} \\
&\quad \Rightarrow \\
&\quad \Rightarrow \\
&\quad \Rightarrow \\
&\quad \Rightarrow \\
&\{ D &\leftarrow D + \text{VAR}_i \}
\end{align*}
\]

Figure 4.250: Automaton for the DECREASING_SURF_INFLEXION constraint obtained by applying decoration Table 3.38 to the seed transducer of the INFLEXION pattern (transition \( r \rightarrow t \) has the same registers updates as transition \( t \rightarrow r \))
<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_SURF_PEAK</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DESCRIPTION**

**Origin**
Based on the PEAK pattern.

**Constraint**
DECREASING_SURF_PEAK(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the values denoting the surface of each occurrence of the PEAK pattern in the time-series given by the VARIABLES collection are decreasing. An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression '< (|=)< (>|=)* >'. Assume that the occurrence of the pattern PEAK starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Example**

```
(1, 6, 6, 6, 2, 5, 4, 3, 2, 2, 3, 2, 1, 5, 5, 7)
```

Figure 4.251 provides an example where the DECREASING_SURF_PEAK ([1, 6, 6, 6, 2, 5, 4, 3, 2, 2, 3, 2, 1, 5, 5, 7]) constraint holds.

**Typical**

| VARIABLES | > 2  
| range(VARIABLES.var) | > 1  

```
| VARIABLES | > 2  
| range(VARIABLES.var) | > 1  
```
Figure 4.251: Illustrating the DECREASING_SURF_PEAK constraint of the Example slot
Automaton

Figure 4.252 depicts the automaton associated with the constraint DECREASING_SURF_PEAK.

\[
\begin{align*}
C &\leftarrow +\infty \\
D &\leftarrow 0 \\
F &\leftarrow +\infty \\
R &\leftarrow 1
\end{align*}
\]

\[
\begin{align*}
R \land (F \geq C)
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow D + \text{VAR}_i \\
D &\leftarrow 0
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow C + D + \text{VAR}_i \\
D &\leftarrow 0
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow 0 \\
F &\leftarrow C \\
R &\leftarrow R \land (F \geq C)
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow +\infty \\
D &\leftarrow 0 \\
F &\leftarrow +\infty \\
R &\leftarrow 1
\end{align*}
\]

\[
\begin{align*}
R \land (F \geq C)
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow C + D + \text{VAR}_i \\
D &\leftarrow 0
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

Figure 4.252: Automaton for the DECREASING_SURF_PEAK constraint obtained by applying decoration Table 3.38 to the seed transducer of the PEAK pattern.
DECREASING_SURFPLAIN

DESCRIPTION

Origin
Based on the PLAIN pattern.

Constraint
DECREASING_SURFPLAIN(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES, var)

Purpose
Succeeds if the values denoting the surface of each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection are decreasing.
An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘\(\geq <\)’.
Assume that the occurrence of the pattern PLAIN starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+1\) to index \(j\).

Example
\(((2, 3, 6, 5, 5, 7, 6, 6, 4, 5, 5, 4, 3, 6, 6, 3))\)

Figure 4.253 provides an example where the DECREASING_SURFPLAIN ([2, 3, 6, 5, 5, 7, 6, 6, 4, 5, 5, 4, 3, 6, 6, 3]) constraint holds.

Typical
\(|\text{VARIABLES}| > 2
\range(\text{VARIABLES, var}) > 1\)
Figure 4.253: Illustrating the DECREASING_SURF_PLAIN constraint of the Example slot
Automaton: Figure 4.254 depicts the automaton associated with the constraint `DECREASING_SURF_PLAIN`.

Figure 4.254: Automaton for the `DECREASING_SURF_PLAIN` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `PLAIN` pattern.
### DECREASING_SURF_PLATEAU

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_SURF_PLATEAU</td>
<td>↑</td>
<td>↑</td>
</tr>
</tbody>
</table>

#### Description

**Origin**
Based on the PLATEAU pattern.

**Constraint**

DECREASING_SURF_PLATEAU(VARIABLES)

**Argument**

VARIABLES : collection(var-dvar)

**Restriction**

required(VARIABLES.var)

Succeeds if the values denoting the surface of each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern **PLATEAU** is the *maximal* subsequence which matches the regular expression '='<='. Assume that the occurrence of the pattern **PLATEAU** starts at position \( i \) and ends at position \( j \). The feature **SURF** computes the sum of the values from index \( i + 1 \) to index \( j \).

**Purpose**

Succeeds if the values denoting the surface of each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are decreasing.

**Example**

\[
(5, 2, 2, 5, 5, 4, 3, 3, 4, 2, 2, 1, 3, 2, 5, 7)
\]

Figure 4.25 provides an example where the DECREASING_SURF_PLATEAU \((5, 2, 2, 5, 5, 4, 3, 3, 4, 2, 2, 1, 3, 2, 5, 7)\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
DECREASING_SURF_PLATEAU

Figure 4.255: Illustrating the DECREASING_SURF_PLATEAU constraint of the Example slot
Figure 4.256 depicts the automaton associated with the constraint DECREASING_SURF_PLATEAU.

Automaton

Figure 4.256: Automaton for the DECREASING_SURF_PLATEAU constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLATEAU pattern.
Origin

Based on the PROPER_PLAIN pattern.

Constraint

DECREASING_SURF_PROPER_PLAIN(VARIABLES)

Argument

VARIABLES : collection(var−dvar)

Restriction

required(VARIABLES.var)

Purpose

Succeeds if the values denoting the surface of each occurrence of the PROPER_PLAIN pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PROPER_PLAIN is the maximal subsequence which matches the regular expression ‘> =+ <’.

Assume that the occurrence of the pattern PROPER_PLAIN starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i+1$ to index $j$.

Example

$((2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5))$

Figure 4.257 provides an example where the DECREASING_SURF_PROPER_PLAIN $((2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5))$ constraint holds.

Typical

$|\text{VARIABLES}| > 3$

$\text{range} (\text{VARIABLES.var}) > 1$
Figure 4.257: Illustrating the DECREASING_SURF_PROPER_PLAIN constraint of the Example slot
Automaton Figure 4.258 depicts the automaton associated with the constraint DECREASING_SURF_PROPER.PLAIN.

Figure 4.258: Automaton for the DECREASING_SURF_PROPER.PLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PROPER.PLAIN pattern.
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_SURF_PROPER_PLATEAU</td>
<td>▶ &gt;&gt; ◄ ◄</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the PROPER_PLATEAU pattern.

**Constraint**
`DECREASING_SURF_PROPER_PLATEAU(VARIABLES)`

**Argument**
`VARIABLES : collection(var−dvar)`

**Restriction**
`required(VARIABLES.var)`

Succeeds if the values denoting the surface of each occurrence of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection are decreasing. An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression `'<\=+>'`.

**Purpose**
Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>((3,5,5,5,3,2,3,4,4,1,5,2,3,3,1,7))</code></td>
</tr>
</tbody>
</table>

Figure 4.259 provides an example where the `DECREASING_SURF_PROPER_PLATEAU` constraint holds.

**Typical**

<table>
<thead>
<tr>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>`</td>
</tr>
<tr>
<td><code>range(VARIABLES.var) &gt; 1</code></td>
</tr>
</tbody>
</table>
Figure 4.259: Illustrating the **DECREASING_SURF_PROPER_PLATEAU** constraint of the **Example** slot
Automaton

Figure 4.260 depicts the automaton associated with the constraint \texttt{DECREASING\_SURF\_PROPER\_PLATEAU}:

Figure 4.260: Automaton for the \texttt{DECREASING\_SURF\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{PROPER\_PLATEAU} pattern.
DECREASING_SURF_STEADY

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_SURF_STEADY</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Origin**
Based on the STEADY pattern.

**Constraint**
DECREASING_SURF_STEADY(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the STEADY pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern STEADY is the subsequence which matches the regular expression '='.

Assume that the occurrence of the pattern STEADY starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

**Purpose**

**Example**

```
((3, 6, 6, 2, 7, 5, 5, 6, 5, 5, 3, 3))
```

Figure 4.261 provides an example where the DECREASING_SURF_STEADY ([3, 6, 6, 2, 7, 5, 5, 6, 5, 5, 3, 3]) constraint holds.

**Typical**
$|\text{VARIABLES}| > 1$
Figure 4.261: Illustrating the DECREASING_SURF_STEADY constraint of the Example slot.
Automaton

Figure 4.262 depicts the automaton associated with the constraint DECREASING_SURF_STEADY.

\[
\begin{cases}
C \leftarrow +\infty \\
D \leftarrow 0 \\
F \leftarrow +\infty \\
R \leftarrow 1
\end{cases}
\]

\[R \land (F \geq C)\]

Figure 4.262: Automaton for the DECREASING_SURF_STEADY constraint obtained by applying decoration Table 3.38 to the seed transducer of the STEADY pattern
**DECREASING_SURF_STEADY_SEQUENCE**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ ▽ ◁ ◅</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the **STEADY_SEQUENCE** pattern.

**Constraint**

\[
\text{DECREASING\_SURF\_STEADY\_SEQUENCE}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

**Purpose**

Succeeds if the values denoting the surface of each occurrence of the **STEADY_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection are decreasing.

An occurrence of the pattern **STEADY_SEQUENCE** is the *maximal* subsequence which matches the regular expression '\=\+\'.

Assume that the occurrence of the pattern **STEADY_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
((4,3,2,5,5,3,2,4,4,6,5,3,2,2,6))
\]

Figure 4.263 provides an example where the **DECREASING_SURF_STEADY_SEQUENCE** \([(4,3,2,5,5,3,2,4,4,6,5,3,2,2,6)]\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]
Figure 4.263: Illustrating the `DECREASING_SURF_STEADY_SEQUENCE` constraint of the `Example` slot
Automaton} Figure 4.264 depicts the automaton associated with the constraint `DECREASING_SURF_STEADYSEQUENCE`.

\[
\begin{align*}
&C ← +∞ \\
&D ← 0 \\
&F ← +∞ \\
&R ← 1 \\
&= r \\
&∧ (F ≥ C)
\end{align*}
\]

\[\{ \begin{array}{c}
D ← 0 \\
F ← C \\
R ← R ∧ (F ≥ C)
\end{array} \] \[\{ \begin{array}{c}
C ← D + \text{VAR}_i + \text{VAR}_{i+1} \\
D ← 0
\end{array} \]

\[\{ \begin{array}{c}
C ← C + D + \text{VAR}_{i+1} \\
D ← 0
\end{array} \]

Figure 4.264: Automaton for the `DECREASING_SURF_STEADYSEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `STEADYSEQUENCE` pattern.
DECREASING_SURF_STEADY_SEQUENCE
Origin
Based on the \textit{STRICTLY\_DECREASING\_SEQUENCE} pattern.

Constraint
\begin{align*}
\text{DECREASING\_SURF\_STRICTLY\_DECREASING\_SEQUENCE}(&) \text{(VARIABLES)}
\end{align*}

Argument
\begin{align*}
\text{VARIABLES} & : \text{collection(\text{var} - \text{dvar})} \\
\end{align*}

Restriction
\begin{align*}
\text{required}(&) \text{(VARIABLES.var)}
\end{align*}

Purpose
\begin{enumerate}
\item Succeeds if the values denoting the surface of each occurrence of the \textit{STRICTLY\_DECREASING\_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection are decreasing.
\item An occurrence of the pattern \textit{STRICTLY\_DECREASING\_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression ‘\texttt{>}\texttt{}’.
\item Assume that the occurrence of the pattern \textit{STRICTLY\_DECREASING\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{SURF} computes the sum of the values from index \(i\) to index \(j + 1\).
\end{enumerate}

Example
\begin{align*}
[(7, 4, 6, 5, 5, 4, 4, 3, 6, 1, 4, 2, 1, 2, 3, 4)]
\end{align*}

Typical
\begin{align*}
|\text{VARIABLES}| & > 1 \\
\text{range(\text{VARIABLES.var})} & > 1
\end{align*}

Figure 4.265 provides an example where the \texttt{DECREASING\_SURF\_STRICTLY\_DECREASING\_SEQUENCE ([7, 4, 6, 5, 5, 4, 4, 3, 6, 1, 4, 2, 1, 2, 3, 4])} constraint holds.
Figure 4.265: Illustrating the DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton Figure 4.266 depicts the automaton associated with the constraint `DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE`.

\[
\begin{align*}
\text{LET } & C \leftarrow +\infty \\
\text{LET } & D \leftarrow 0 \\
\text{LET } & F \leftarrow +\infty \\
\text{LET } & R \leftarrow 1 \\
\text{IF } & \leq s \\
\text{THEN } & \{ C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
\text{AND } & D \leftarrow 0 \}
\end{align*}
\]

Figure 4.266: Automaton for the `DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern.
DECREASING_SURF STRICTLY DECREASING SEQUENCE
### DESCRIPTION

**Origin**
Based on the \textit{STRICTLY_INCREASING_SEQUENCE} pattern.

**Constraint**
\texttt{DECREASING\_SURF\_STRICTLY\_INCREASING\_SEQUENCE(VARIABLES)}

**Argument**
\texttt{VARIABLES : collection(var−dvar)}

**Restriction**
\texttt{required(VARIABLES.var)}

Succeeds if the values denoting the surface of each occurrence of the \textit{STRICTLY_INCREASING_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection are decreasing.

**Purpose**
An occurrence of the pattern \textit{STRICTLY_INCREASING_SEQUENCE} is the \textit{maximal} sub-sequence which matches the regular expression '\(<^+\)'.

Assume that the occurrence of the pattern \textit{STRICTLY_INCREASING_SEQUENCE} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{SURF} computes the sum of the values from index \texttt{i} to index \texttt{j + 1}.

**Example**
\texttt{((3, 4, 5, 5, 4, 3, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))}

Figure 4.267 provides an example where the \textit{DECREASING\_SURF\_STRICTLY\_INCREASING\_SEQUENCE} constraint holds.

**Typical**
\texttt{|VARIABLES| > 1}

\texttt{range(VARIABLES.var) > 1}
Figure 4.267: Illustrating the DECREASING_SURF_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Automaton: Figure 4.268 depicts the automaton associated with the constraint \textsc{decreasing_surf_strictly_increasing_sequence}.

\[
\begin{align*}
& C \leftarrow +\infty \\
& D \leftarrow 0 \\
& F \leftarrow +\infty \\
& R \leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
& \geq s \\
& \geq \\
& \geq \\
& < r \land (F \geq C) \\
& \geq \{ \\
& \begin{align*}
& D \leftarrow 0 \\
& F \leftarrow C \\
& R \leftarrow R \land (F \geq C) \\
& \end{align*} \\
& < \{ \\
& \begin{align*}
& C \leftarrow D + \text{VAR} + \text{VAR}_{i+1} \\
& D \leftarrow 0 \\
& \end{align*} \\
& < \{ \\
& \begin{align*}
& C \leftarrow C + D + \text{VAR}_{i+1} \\
& D \leftarrow 0 \\
& \end{align*}
\end{align*}
\]

Figure 4.268: Automaton for the \textsc{decreasing_surf_strictly_increasing_sequence} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{strictly_increasing_sequence} pattern.


<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DECREASING_SURF_SUMMIT</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Description Automaton**

```
(\< \< (\= \<) \< (\> \> (\= \>)) >)
```

**Origin**

Based on the **SUMMIT** pattern.

**Constraint**

`DECREASING_SURF_SUMMIT(VARIABLES)`

**Argument**

VARIABLES : `collection(var-dvar)`

**Restriction**

`required(VARIABLES, var)`

Succeeds if the values denoting the surface of each occurrence of the **SUMMIT** pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern **SUMMIT** is the maximal subsequence which matches the regular expression `'(\< \< (\= \<) \< (\> \> (\= \>)) >)'`.

Assume that the occurrence of the pattern **SUMMIT** starts at position `\(i\)` and ends at position `\(j\)`. The feature **SURF** computes the sum of the values from index `\(i + 1\)` to index `\(j\)`.

**Example**

\[\langle 1, 4, 4, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 2, 4, 2 \rangle\]

Figure 4.269 provides an example where the `DECREASING_SURF_SUMMIT` constraint holds.

**Typical**

\[|\text{VARIABLES}| > 2\]

\[\text{range(\text{VARIABLES}.var)} > 1\]
Figure 4.269: Illustrating the DECREASING_SURF_SUMMIT constraint of the Example slot
Automaton

Figure 4.270 depicts the automaton associated with the constraint DECREASING_SURF_SUMMIT.

\[
\begin{align*}
&\begin{cases}
C &\leftarrow +\infty \\
D &\leftarrow 0 \\
R &\leftarrow +\infty
\end{cases} \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\geq \\
&\geq s \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\leq \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\leq \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\geq \land (F \geq C) \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow 0 \}
\end{align*}
\]
### Condition

**DECREASING_SURF_VALLEY**

<table>
<thead>
<tr>
<th>&gt;</th>
<th>&gt;&gt;</th>
<th>&lt;</th>
<th>◀</th>
</tr>
</thead>
</table>

### Description

Automaton: 

\[
> (= | >)^*(< | =)^* <
\]

### Origin

Based on the **VALLEY** pattern.

### Constraint

**DECREASING_SURF_VALLEY**(VARIABLES)

### Argument

**VARIABLES**: collection(var−dvar)

### Restriction

**required**(VARIABLES.var)

Succeeds if the values denoting the surface of each occurrence of the **VALLEY** pattern in the time-series given by the **VARIABLES** collection are decreasing.

### Purpose

An occurrence of the pattern **VALLEY** is the *maximal* subsequence which matches the regular expression '

```
> (| >)^*(< | =)^* <
```

Assume that the occurrence of the pattern **VALLEY** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i+1\) to index \(j\).

### Example

\[
((1, 3, 3, 7, 6, 5, 4, 6, 5, 4, 3, 6, 2, 2, 7))
\]

Figure 4.271 provides an example where the **DECREASING_SURF_VALLEY**

\[
([1, 3, 3, 7, 6, 5, 4, 6, 5, 4, 3, 6, 2, 2, 7])
\]

constraint holds.

### Typical

- \(|\text{VARIABLES}| > 2\)
- \(\text{range(VARIABLES.var)} > 1\)
DECREASING_SURF_VALLEY

Figure 4.271: Illustrating the DECREASING_SURF_VALLEY constraint of the Example slot
Automaton  

Figure 4.272 depicts the automaton associated with the constraint `DECREASING_SURF_VALLEY`.

**Figure 4.272:** Automaton for the `DECREASING_SURF_VALLEY` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `VALLEY` pattern
**Condition**

**Feature**

**Pattern**

**DECREASING_SURF_ZIGZAG**

**Description**

**Automaton**

\[(<>)^+ (< | <>)(> | <>)^+ (> | <>)\]

**Origin**

Based on the ZIGZAG pattern.

**Constraint**

`DECREASING_SURF_ZIGZAG(VARIABLES)`

**Argument**

`VARIABLES : collection(var−dvar)`

**Restriction**

`required(VARIABLES.var)`

Succeeds if the values denoting the surface of each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern ZIGZAG is the *maximal* subsequence which matches the regular expression `'(<>)^+ (< | <>)(> | <>)^+ (> | <>)'`.

Assume that the occurrence of the pattern ZIGZAG starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 1` to index `j`.

**Example**

\[((6, 5, 7, 4, 5, 1, 1, 6, 3, 6, 4, 3, 5, 2, 7, 7))\]

Figure 4.273 provides an example where the `DECREASING_SURF_ZIGZAG` \([(6, 5, 7, 4, 5, 1, 1, 6, 3, 6, 4, 3, 5, 2, 7, 7)]\) constraint holds.

**Typical**

- \[|VARIABLES| > 3\]
- \[range(VARIABLES.var) > 1\]
Figure 4.273: Illustrating the DECREASING_SURF_ZIGZAG constraint of the Example slot
Automaton Figure 4.274 depicts the automaton associated with the constraint DECREASING_SURF_ZIGZAG.
DECREASING_SURF_ZIGZAG

\[
\begin{align*}
\text{Figure 4.274: Automaton for the DECREASING_SURF_ZIGZAG constraint obtained by applying decoration Table 3.38 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from } a, b, c, d, e, f \text{ to } s \text{ are labelled by } =; (2) \text{ on transitions from } b, c, e, f \text{ to } s \text{ the register } D \text{ is reset to its initial value; (3) on transitions from } c, f \text{ to } s \text{ the register } F \text{ is reset to } C, \text{ and the register } R \text{ is updated wrt } C \text{ and } F
\end{align*}
\]
## DECREASING_WIDTH_DECREASING_SEQUENCE

### Description
Based on the **DECREASING_SEQUENCE** pattern.

### Automaton
![Automaton Diagram]

### Origin
Based on the **DECREASING_SEQUENCE** pattern.

### Constraint
**DECREASING_WIDTH_DECREASING_SEQUENCE**(VARIABLES)

### Argument
**VARIABLES**: `collection(var−dvar)`

### Restriction
**required**(VARIABLES.var)

### Purpose
Succeeds if the values denoting the width of each occurrence of the **DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern **DECREASING_SEQUENCE** is the maximal subsequence which matches the regular expression `'>(>|=)^>*|>'`

Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position `i` and ends at position `j`. The feature **WIDTH** computes the value `j − i + 2`.

### Example
`((2, 6, 5, 2, 3, 3, 4, 3, 2, 3, 4, 4, 5, 7, 5))`

Figure 4.275 provides an example where the **DECREASING_WIDTH_DECREASING_SEQUENCE** constraint holds.

### Typical
- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 4.275: Illustrating the DECREASING_WIDTH_DECREASING_SEQUENCE constraint of the Example slot.
Automaton Figure 4.276 depicts the automaton associated with the constraint DECREASING_WIDTH_DECREASING_SEQUENCE.

\[
\begin{align*}
\{ C &\leftarrow +\infty \} \\
\{ D &\leftarrow 0 \} \\
\{ F &\leftarrow +\infty \} \\
\{ R &\leftarrow 1 \}
\end{align*}
\]

\[
\leq s
\]

\[
\begin{align*}
\{ D &\leftarrow 0 \} \\
\{ F &\leftarrow C \} \\
\{ R &\leftarrow R \land (F \geq C) \}
\end{align*}
\]

\[
\leq \begin{cases} 
C &\leftarrow D + 2 \\
D &\leftarrow 0
\end{cases}
\]

\[
\geq t
\]

\[
\begin{cases} 
D &\leftarrow D + 1
\end{cases}
\]

\[
\begin{align*}
\{ D &\leftarrow D + 1 \} \\
\{ C &\leftarrow C + D + 1 \} \\
\{ D &\leftarrow 0 \}
\end{align*}
\]

Figure 4.276: Automaton for the DECREASING_WIDTH_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING_SEQUENCE pattern.
### Description

#### Condition

**DECREASING_WIDTH**

#### Feature

**DECREASING_TERRACE**

#### Pattern

**DECREASING_WIDTH**

**DECREASING_TERRACE**

### Automaton

![Automaton Diagram]

### Origin

Based on the **DECREASING_TERRACE** pattern.

### Constraint

**DECREASING_WIDTH**

**DECREASING_TERRACE**(VARIABLES)

### Argument

**VARIABLES** : collection(var\(\rightarrow\)dvar)

### Restriction

required\(\left[VARIABLES, \text{var}\right]\)

### Purpose

Succeeds if the values denoting the width of each occurrence of the **DECREASING_TERRACE** pattern in the time-series given by the **VARIABLES** collection are decreasing.

An occurrence of the pattern **DECREASING_TERRACE** is the **maximal** subsequence which matches the regular expression ‘\(>=^+>\)’.

Assume that the occurrence of the pattern **DECREASING_TERRACE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i\).

### Example

\(\langle 7, 6, 6, 4, 3, 3, 2, 2, 4, 4, 6, 3, 3, 1, 1 \rangle\)

Figure 4.277 provides an example where the **DECREASING_WIDTH**\_**DECREASING_TERRACE** \(\langle 7, 6, 6, 4, 3, 3, 2, 2, 4, 4, 6, 3, 3, 1, 1 \rangle\) constraint holds.

### Typical

\(|VARIABLES| > 3\)

\(\text{range}(VARIABLES.\text{var}) > 2\)
Figure 4.277: Illustrating the \texttt{DECREASING\_WIDTH\_DECREASING\_TERRACE} constraint of the \texttt{Example} slot
Automaton  

Figure 4.278 depicts the automaton associated with the constraint DECREASING_WIDTH_DECREASING_TERRACE.

\[
\begin{align*}
C &\leftarrow +\infty \\
D &\leftarrow 0 \\
F &\leftarrow +\infty \\
R &\leftarrow 1 \\
\{D \leftarrow D + 1\} \quad \{D \leftarrow D + 1\} \\
\{C \leftarrow D + 1\} &\leftarrow \{D \leftarrow D + 1\} \\
\{F \leftarrow D + 1\} &\leftarrow \{D \leftarrow D + 1\} \\
R &\leftarrow R \land (F \geq D + 1) \\
\end{align*}
\]

Figure 4.278: Automaton for the DECREASING_WIDTH_DECREASING_TERRACE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING_TERRACE pattern
DECREASING_WIDTH_GORGE

DECREASING_WIDTH_GORGE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶</td>
<td>▶</td>
<td>▶</td>
</tr>
</tbody>
</table>

**Description Automaton**

**Origin**
Based on the GORGE pattern.

**Constraint**
DECREASING_WIDTH_GORGE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the values denoting the width of each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression "(>|>|(|>|>||)<|<(<(|>|>||)<|<(<(|>|>||)<|<(|>|>||)<|<(<(|>|>||)<|<(<|". Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature \(\text{WIDTH}\) computes the value \(j-i\).

**Example**

\[ ((6, 2, 3, 4, 5, 6, 1, 1, 5, 4, 2, 3, 6, 1, 4, 5)) \]

Figure 4.279 provides an example where the DECREASING_WIDTH_GORGE ([6, 2, 3, 4, 5, 6, 1, 1, 5, 4, 2, 3, 6, 1, 4, 5]) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 2 \]

\[ \text{range}(\text{VARIABLES.var}) > 1 \]
Figure 4.279: Illustrating the DECREASING_WIDTH_GORGE constraint of the Example slot
Automaton Figure 4.280 depicts the automaton associated with the constraint DECREASING_WIDTH_GORGE.

![Automaton Diagram]

Figure 4.280: Automaton for the DECREASING_WIDTH_GORGE constraint obtained by applying decoration Table 3.38 to the seed transducer of the GORGE pattern (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
DECREASING_WIDTH_GORGE
## DECREASING_WIDTH_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_WIDTH_INCREASING_SEQUENCE</td>
<td>&lt; (&lt;</td>
<td>=)* &lt;</td>
</tr>
</tbody>
</table>

### Description

- **Origin**: Based on the INCREASING_SEQUENCE pattern.
- **Constraint**: DECREASING_WIDTH_INCREASING_SEQUENCE(VARIABLES)
- **Argument**: VARIABLES : collection(var–dvar)

### Purpose

Succeeds if the values denoting the width of each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

### Restriction

required(VARIABLES.var)

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘< (< | =)* < | <’.

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i + 2$.

### Example

$((3, 4, 5, 5, 4, 3, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))$

Figure 4.281 provides an example where the DECREASING_WIDTH_INCREASING_SEQUENCE constraint holds.

### Typical

- $|\text{VARIABLES}| > 1$
- $\text{range}(\text{VARIABLES}.\text{var}) > 1$
Figure 4.281: Illustrating the DECREASING_WIDTH_INCREMENTING_SEQUENCE constraint of the Example slot
Automaton Figure 4.282 depicts the automaton associated with the constraint \texttt{DECREASING_WIDTH_INCREASING_SEQUENCE}.

\begin{align*}
\{ C &\leftarrow +\infty \} \\
\{ D &\leftarrow 0 \} \\
\{ F &\leftarrow +\infty \} \\
\{ R &\leftarrow 1 \}
\end{align*}

\begin{align*}
\{ D &\leftarrow 0 \} \\
\{ F &\leftarrow C \} \\
\{ R &\leftarrow R \land (F \geq C) \}
\end{align*}

\begin{align*}
\{ D &\leftarrow D + 1 \}
\end{align*}

\begin{align*}
\{ C &\leftarrow +\infty \} \\
\{ D &\leftarrow 0 \} \\
\{ F &\leftarrow +\infty \} \\
\{ R &\leftarrow 1 \}
\end{align*}

\begin{align*}
\{ D &\leftarrow 0 \} \\
\{ F &\leftarrow C \} \\
\{ R &\leftarrow R \land (F \geq C) \}
\end{align*}

\begin{align*}
\{ D &\leftarrow D + 1 \}
\end{align*}

\begin{align*}
\{ C &\leftarrow +\infty \} \\
\{ D &\leftarrow 0 \} \\
\{ F &\leftarrow +\infty \} \\
\{ R &\leftarrow 1 \}
\end{align*}

\begin{align*}
\{ D &\leftarrow 0 \} \\
\{ F &\leftarrow C \} \\
\{ R &\leftarrow R \land (F \geq C) \}
\end{align*}

\begin{align*}
\{ D &\leftarrow D + 1 \}
\end{align*}

Figure 4.282: Automaton for the \texttt{DECREASING_WIDTH_INCREASING_SEQUENCE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{INCREASING_SEQUENCE} pattern.
### Description

**Origin**
Based on the **increasing Terrace** pattern.

**Constraint**
阴道**DECREASING_WIDTH_INCREMENTING_TERRACE(VARIABLES)**

**Argument**
```
VARIABLES : collection(var–dvar)
```

**Restriction**
```
required(VARIABLES.var)
```

**Purpose**
A occurrence of the pattern **increasing Terrace** is the maximal subsequence which matches the regular expression ‘< =+ <’.
Assume that the occurrence of the pattern **increasing Terrace** starts at position i and ends at position j. The feature **Width** computes the value j – i.

**Example**
```
((2, 5, 5, 6, 2, 3, 4, 5, 3, 3, 1, 2, 2, 3))
```

Figure 4.283 provides an example where the **DECREASING_WIDTH_INCREMENTING_TERRACE** ((2, 5, 5, 6, 2, 3, 4, 5, 3, 3, 1, 2, 2, 3)) constraint holds.

**Typical**
```
|VARIABLES| > 3
range(VARIABLES.var) > 2
```
Figure 4.283: Illustrating the DECREASING_WIDTH_INCREASING_TERRACE constraint of the Example slot
Automaton

Figure 4.284 depicts the automaton associated with the constraint \textsc{decreasing\_width\_increasing\_terrace}.

Figure 4.284: Automaton for the \textsc{decreasing\_width\_increasing\_terrace} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{increasing\_terrace} pattern
DECREASING_WIDTH_INCREASING_TERRACE

725
DECREASING_WIDTH_INFLEXION

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ ▶ ▶ ▶</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

< (| | )* > | > (| | )* <

Origin
Based on the INFLEXION pattern.

Constraint
DECREASING_WIDTH_INFLEXION(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES, var)

Succeeds if the values denoting the width of each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are increasing.

Purpose
An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression ' < (| | )* > | > (| | )* < '/. Assume that the occurrence of the pattern INFLEXION starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example
((6, 5, 5, 4, 3, 2, 3, 3, 4, 6, 3, 3, 1, 2, 1, 1))

Figure 4.285 provides an example where the DECREASING_WIDTH_INFLEXION ((6, 5, 5, 4, 3, 2, 3, 3, 4, 6, 3, 3, 1, 2, 1, 1)) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES, var) > 1
Figure 4.285: Illustrating the `DECREASING_WIDTH_INFLEXION` constraint of the `Example` slot
Automaton

Figure 4.286 depicts the automaton associated with the constraint DECREASING_WIDTH_INFLEXION.

\[
\begin{align*}
&\{C \gets +\infty, D \gets 0, F \gets +\infty, R \gets 1\} \\
&\{D \gets D + 1\} \\
&\{C \gets D + 1, D \gets 0, F \gets D + 1, R \gets R \land (F \geq D + 1)\}
\end{align*}
\]

Figure 4.286: Automaton for the DECREASING_WIDTH_INFLEXION constraint obtained by applying decoration Table 3.38 to the seed transducer of the INFLEXION pattern (transition \(r \rightarrow t\) has the same registers updates as transition \(t \rightarrow r\))
DECREASING_WIDTH_INFLEXION
**DECREASING_WIDTH_PEAK**

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; (=</td>
<td>&lt;)* (&gt;</td>
<td>=)* &gt;</td>
</tr>
</tbody>
</table>

**Origin**
Based on the **PEAK** pattern.

**Constraint**

\[
\text{DECREASING_WIDTH_PEAK}({\text{VARIABLES}})
\]

**Argument**

\[
\text{VARIABLES : } \text{collection}({\text{var-dvar}})
\]

**Restriction**

\[
\text{required}({\text{VARIABLES}.\text{var}})
\]

Succeeds if the values denoting the width of each occurrence of the **PEAK** pattern in the time-series given by the **VARIABLES** collection are decreasing.

**Purpose**

An occurrence of the pattern **PEAK** is the *maximal* subsequence which matches the regular expression ‘\(< (= | <)^* (> | =)^* >\)’.
Assume that the occurrence of the pattern **PEAK** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i\).

**Example**

\[
[(1, 6, 6, 2, 5, 4, 3, 2, 3, 2, 1, 5, 5, 7)]
\]

Figure 4.287 provides an example where the **DECREASING_WIDTH_PEAK** ([1, 6, 6, 2, 5, 4, 3, 2, 3, 2, 1, 5, 5, 7]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range}({\text{VARIABLES}.\text{var}}) > 1
\]
Figure 4.287: Illustrating the DECREASING_WIDTH_PEAK constraint of the Example slot
Figure 4.288 depicts the automaton associated with the constraint \textsc{DECREASING\_WIDTH\_PEAK}.

\begin{center}
\begin{tikzpicture}[node distance=2.5cm,>=latex]
  \node (init) {$\leq t$};
  \node (state1) [right of=init, below of=init] {$\geq s$};
  \node (state2) [right of=state1, below of=state1] {$\geq r$};
  \node (state3) [right of=state2, below of=state2] {$\geq s$};

  \path (init) edge [loop above] node {$\{D \leftarrow D + 1\}$} (init)
  (init) edge [loop below] node {$\{D \leftarrow 0\}$} (init)
  (init) edge [above] node {$\{C \leftarrow D + 1\}$} (state1)
  (state1) edge [above] node {$\{C \leftarrow +\infty\}$} (init)
  (state1) edge [below] node {$R \land (F \geq C)$} (state2)
  (state2) edge [above] node {$\{C \leftarrow +\infty\}$} (state1)
  (state2) edge [below] node {$\{D \leftarrow +\infty\}$} (state3)
  (state3) edge [above] node {$\{C \leftarrow +\infty\}$} (state2)
  (state3) edge [below] node {$\{D \leftarrow +\infty\}$} (state1);
\end{tikzpicture}
\end{center}

Figure 4.288: Automaton for the \textsc{DECREASING\_WIDTH\_PEAK} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{PEAK} pattern.
### DECREASING_WIDTHPLAIN

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;(\geq)&quot;</td>
<td>&quot;(&lt;)&quot;</td>
<td>&quot;(\text{PLAIN})&quot;</td>
</tr>
</tbody>
</table>

#### Description

**Origin**
Based on the **PLAIN** pattern.

**Constraint**

\[
\text{DECREASING\_WIDTH\_PLAIN}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var\_dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES, var})
\]

Succeeds if the values denoting the width of each occurrence of the **PLAIN** pattern in the time-series given by the **VARIABLES** collection are decreasing.

**Purpose**

An occurrence of the pattern **PLAIN** is the *maximal* subsequence which matches the regular expression ‘\(\geq\)\(\ast\)\(<\)’. Assume that the occurrence of the pattern **PLAIN** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i\).

**Example**

\[
(\langle 2, 3, 6, 5, 7, 6, 6, 4, 5, 4, 3, 6, 6, 3 \rangle)
\]

Figure 4.289 provides an example where the **DECREASING\_WIDTH\_PLAIN** ((\(2, 3, 6, 5, 7, 6, 6, 4, 5, 4, 3, 6, 6, 3\)) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2
\]

\[
\text{range}(\text{VARIABLES.var}) > 1
\]
DECREASING_WIDTH_PLAIN

Figure 4.289: Illustrating the DECREASING_WIDTH_PLAIN constraint of the Example slot
Automaton  

Figure 4.290 depicts the automaton associated with the constraint `DECREASING_WIDTH.PLAIN`. 

![Automaton Diagram](image)

Figure 4.290: Automaton for the `DECREASING_WIDTH.PLAIN` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `PLAIN` pattern.
Origin: Based on the PLATEAU pattern.

Constraint: \texttt{DECREASING_WIDTH_PLATEAU(VARIABLES)}

Argument: \texttt{VARIABLES : collection(var\text{−}dvar)}

Restriction: \texttt{required(VARIABLES, var)}

Purpose:
Succeeds if the values denoting the width of each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PLATEAU is the \textit{maximal} subsequence which matches the regular expression ‘\texttt{<\textasciitilde\textasciitilde}’.

Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature \texttt{WIDTH} computes the value \(j - i\).

Example: \((5, 2, 2, 5, 5, 4, 3, 3, 4, 2, 2, 1, 3, 2, 5, 7)\)

Figure 4.291 provides an example where the \texttt{DECREASING_WIDTH_PLATEAU} \((\langle 5, 2, 2, 5, 5, 4, 3, 3, 4, 2, 2, 1, 3, 2, 5, 7\rangle )\) constraint holds.

Typical:
\[|\text{VARIABLES}| > 2\]
\[\text{range}(\text{VARIABLES.var}) > 1\]
Figure 4.291: Illustrating the DECREASING_WIDTH_PLATEAU constraint of the Example slot
Automaton

Figure 4.292 depicts the automaton associated with the constraint DECREASING_WIDTH_PLATEAU.

Figure 4.292: Automaton for the DECREASING_WIDTH_PLATEAU constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLATEAU pattern.
DECREASING_WIDTH_PLATEAU
DECREASING_WIDTH_PROPER_PLAIN

**Origin**
Based on the PROPER_PLAIN pattern.

**Constraint**
DECREASING_WIDTH_PROPER_PLAIN(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the values denoting the width of each occurrence of the PROPER_PLAIN pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PROPER_PLAIN is the maximal subsequence which matches the regular expression ‘\(> =^+ <\)’.

Assume that the occurrence of the pattern PROPER_PLAIN starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**
\[
((2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5))
\]

Figure 4.293 provides an example where the DECREASING_WIDTH_PROPER_PLAIN ([2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5]) constraint holds.

**Typical**
\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES.var}) > 1
\]
Figure 4.293: Illustrating the DECREASING_WIDTH_PROPERPLAIN constraint of the Example slot
Automaton

Figure 4.294 depicts the automaton associated with the constraint DECREASING_WIDTH_PROPER_PLAIN.

Figure 4.294: Automaton for the DECREASING_WIDTH_PROPER_PLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PROPER_PLAIN pattern
DECREASING_WIDTH_PROPER_PLAIN
DECREASING_WIDTH_PROPER_PLATEAU

Origin: Based on the PROPER_PLATEAU pattern.

Constraint: \text{DECREASING_WIDTH_PROPER_PLATEAU}(\text{VARIABLES})

Argument: \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})

Restriction: \text{required}(\text{VARIABLES}.\text{var})

Purpose: Succeeds if the values denoting the width of each occurrence of the PROPER_PLATEAU pattern in the time-series given by the \text{VARIABLES} collection are decreasing.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression '\text{< =+ >}'.

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \(i\) and ends at position \(j\). The feature \text{WIDTH} computes the value \(j - i\).

Example: \((3, 5, 5, 3, 2, 3, 4, 1, 5, 2, 3, 3, 1, 7)\)

Figure 4.295 provides an example where the \text{DECREASING_WIDTH_PROPER_PLATEAU} \((3, 5, 5, 3, 2, 3, 4, 1, 5, 2, 3, 3, 1, 7)\) constraint holds.

Typical: \(|\text{VARIABLES}| > 3\)

\text{range}(\text{VARIABLES}.\text{var}) > 1\)
Figure 4.295: Illustrating the DECREASING_WIDTH_PROPER_PLATEAU constraint of the Example slot
Automaton

Figure 4.296 depicts the automaton associated with the constraint \textsc{decreasing\_width\_proper\_plateau}.

\begin{itemize}
\item \(s\geq t\) if \(C\leftarrow D+1\), \(D\leftarrow 0\), \(F\leftarrow D+1\), and \(R\leftarrow R\wedge(F\geq D+1)\).
\item \(s\geq R\wedge(F\geq C)\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\item \(s\geq\) if \(s\geq\).
\end{itemize}
**DECREASING_WIDTH_STEADY_SEQUENCE**

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DECREASING_WIDTH_STEADY_SEQUENCE</strong></td>
<td>▶▷◁◀</td>
<td><strong>DESCRIPTION</strong></td>
</tr>
</tbody>
</table>

**Origin**
Based on the **STEADY_SEQUENCE** pattern.

**Constraint**

```
DECREASING_WIDTH_STEADY_SEQUENCE(VARIABLES)
```

**Argument**

```
VARIABLES : collection(var−dvar)
```

**Restriction**
```
required(VARIABLES, var)
```

**Purpose**
Succeeds if the values denoting the width of each occurrence of the **STEADY_SEQUENCE** pattern in the time-series given by the VARIABLES collection are decreasing. An occurrence of the pattern **STEADY_SEQUENCE** is the *maximal* subsequence which matches the regular expression ‘=+’. Assume that the occurrence of the pattern **STEADY_SEQUENCE** starts at position *i* and ends at position *j*. The feature **WIDTH** computes the value *j* − *i* + 2.

**Example**

```
((4,3,2,5,5,5,3,2,4,4,6,5,3,2,2,6))
```

Figure 4.297 provides an example where the **DECREASING_WIDTH_STEADY_SEQUENCE** ((4,3,2,5,5,5,3,2,4,4,6,5,3,2,2,6)) constraint holds.

**Typical**

```
|VARIABLES| > 1
```
Figure 4.297: Illustrating the DECREASING_WIDTH_STEADY_SEQUENCE constraint of the Example slot
Automaton

Figure 4.298 depicts the automaton associated with the constraint DECREASING_WIDTH_STEADY_SEQUENCE.

\[
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow 0 \\
F & \leftarrow +\infty \\
R & \leftarrow 1
\end{align*}
\]

Figure 4.298: Automaton for the DECREASING_WIDTH_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STEADY_SEQUENCE pattern
### DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

#### Origin
Based on the **STRICTLY_DECREASING_SEQUENCE** pattern.

#### Constraint
**DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE**(VARIABLES)

#### Argument
VARIABLES : collection(var – dvar)

#### Restriction
required(VARIABLES, var)

Succeeds if the values denoting the width of each occurrence of the **STRICTLY_DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are decreasing.

#### Purpose
An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression ‘\>\+\’.

Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position $i$ and ends at position $j$. The feature **WIDTH** computes the value $j - i + 2$.

#### Example

```
(⟨1, 1, 2, 2, 3, 2, 1, 5, 6, 6, 5, 4, 4, 6, 6, 1⟩)
```

Figure 4.299 provides an example where the **DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE** ((1, 1, 2, 2, 3, 2, 1, 5, 6, 6, 5, 4, 4, 6, 6, 1)) constraint holds.

#### Typical
$|\text{VARIABLES}| > 1$

$\text{range}(\text{VARIABLES}\.\text{var}) > 1$
Figure 4.299: Illustrating the \textsc{decreasing} \_ \textsc{width} \_ \textsc{strictly} \_ \textsc{decreasing} \_ \textsc{sequence} constraint of the \textbf{Example} slot
Automaton

Figure 4.300 depicts the automaton associated with the constraint `DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE`.

```
C ← +∞
D ← 0
F ← +∞
R ← 1
```

Figure 4.300: Automaton for the `DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern.
DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE
**DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE**

**DESCRIPTION**

- **Origin**: Based on the `STRICTLY_INCREASING_SEQUENCE` pattern.
- **Constraint**: `DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE(VARIABLES)`
- **Argument**: `VARIABLES : collection(var−dvar)`
- **Restriction**: `required(VARIABLES, var)`

**Purpose**

Succeeds if the values denoting the width of each occurrence of the `STRICTLY_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are decreasing.

An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the **maximal** sub-sequence which matches the regular expression `'<+'`. Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

**Example**

`((3, 4, 5, 5, 4, 3, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))`

Figure 4.301 provides an example where the `DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE` `((3, 4, 5, 5, 4, 3, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 4.301: Illustrating the **DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE** constraint of the **Example** slot
Figure 4.302 depicts the automaton associated with the constraint DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE.

Figure 4.302: Automaton for the DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern.
DECREASING_WIDTH_SUMMIT

**Origin**
Based on the SUMMIT pattern.

**Constraint**
DECREASING_WIDTH_SUMMIT(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the values denoting the width of each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression ‘(< | < (= | <)∗ <) | > (|= | >)∗ >)’.
Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j − i\).

**Example**
\(((1, 4, 4, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 2, 4, 2))\)

Figure 4.303 provides an example where the DECREASING_WIDTH_SUMMIT \(([1, 4, 4, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 2, 4, 2])\) constraint holds.

**Typical**
\(|VARIABLES| > 2\\
range(VARIABLES.var) > 1\\
DECREASING_WIDTH_SUMMIT

Figure 4.303: Illustrating the DECREASING_WIDTH_SUMMIT constraint of the Example slot
Automaton

Figure 4.304 depicts the automaton associated with the constraint DECREASING_WIDTH_SUMMIT.

\[
\begin{align*}
\{ C ← +∞ \\
D ← 0 \\
F ← +∞ \\
R ← 1 \\
\} & \quad \geq s \\
\{ D ← D + 1 \}
\end{align*}
\]

\[
\begin{align*}
\{ \} & \quad \langle \rangle
\end{align*}
\]

\[
\begin{align*}
\{ C ← D + 1 \\
D ← 0 \}
\end{align*}
\]

\[
\begin{align*}
\{ D ← D + 1 \}
\end{align*}
\]

\[
\begin{align*}
\{ D ← 0 \}
\end{align*}
\]

\[
\begin{align*}
\{ D ← D + 1 \}
\end{align*}
\]

\[
\begin{align*}
\{ D ← D + 1 \}
\end{align*}
\]

\[
\begin{align*}
\{ D ← D + 1 \}
\end{align*}
\]

\[
\begin{align*}
\{ D ← D + 1 \}
\end{align*}
\]

Figure 4.304: Automaton for the DECREASING_WIDTH_SUMMIT constraint obtained by applying decoration Table 3.38 to the seed transducer of the SUMMIT pattern (transition \( u \rightarrow r \) has the same register update as transition \( r \rightarrow u \))
DECREASING_WIDTH_VALLEY

**Description Automaton**

- **Origin**: Based on the VALLEY pattern.
- **Constraint**: `DECREASING_WIDTH_VALLEY(VARIABLES)`
- **Argument**: `VARIABLES : collection(var-dvar)`
- **Restriction**: `required(VARIABLES.var)`
- **Purpose**: Succeeds if the values denoting the width of each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection are decreasing.
  An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression `'> (= | >)" (< | =)" <`.
  Assume that the occurrence of the pattern VALLEY starts at position `i` and ends at position `j`. The feature WIDTH computes the value `j - i`.
- **Example**: `((1,3,3,7,6,5,4,6,6,5,4,3,6,2,2,7))`

Figure 4.305 provides an example where the DECREASING_WIDTH_VALLEY `((1,3,3,7,6,5,4,6,6,5,4,3,6,2,2,7))` constraint holds.

**Typical**
- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`
Figure 4.305: Illustrating the DECREASING_WIDTH_VALLEY constraint of the Example slot
Automaton Figure 4.306 depicts the automaton associated with the constraint DECREASING_WIDTH_VALLEY.

Figure 4.306: Automaton for the DECREASING_WIDTH_VALLEY constraint obtained by applying decoration Table 3.38 to the seed transducer of the VALLEY pattern
DECREASING_WIDTH_VALLEY
Origin: Based on the ZIGZAG pattern.

Constraint: \texttt{DECREASING_WIDTH_ZIGZAG(VARIABLES)}

Argument: \texttt{VARIABLES : collection(var−dvar)}

Restriction: \texttt{required(VARIABLES.var)}

Purpose: Succeeds if the values denoting the width of each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are decreasing. An occurrence of the pattern ZIGZAG is the \textit{maximal} subsequence which matches the regular expression ‘’\texttt{’(<>)(< | <>)(<>)+>|<>)+’’}. Assume that the occurrence of the pattern ZIGZAG starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{WIDTH} computes the value \texttt{j − i}.

Example: \texttt{((6,5,7,4,5,1,1,6,3,6,4,3,5,2,7,7))}

Figure 4.307 provides an example where the \texttt{DECREASING_WIDTH_ZIGZAG} \texttt{([6,5,7,4,5,1,1,6,3,6,4,3,5,2,7,7])} constraint holds.

Typical: \texttt{|VARIABLES| > 3}

\texttt{range(VARIABLES.var) > 1}
Figure 4.307: Illustrating the DECREASING_WIDTH_ZIGZAG constraint of the Example slot
Automaton

Figure 4.308 depicts the automaton associated with the constraint DECREASING_WIDTH_ZIGZAG.
Figure 4.308: Automaton for the DECREASING_WIDTH_ZIGZAG constraint obtained by applying decoration Table 3.38 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the register D is reset to its initial value; (3) on transitions from c, f to s the register F is reset to C, and the register R is updated wrt C and F.
### Description

**HEIGHT_DECREASING_TERRACE**

- **Origin**: Based on the **DECREASING_TERRACE** pattern.

- **Constraint**: 
  
  ```
  HEIGHT_DECREASING_TERRACE(VARIABLES, FEATURES, DEFAULT)
  ```

- **Arguments**:
  - **VARIABLES**: `collection(var−dvar)`
  - **FEATURES**: `collection(var−dvar)`
  - **DEFAULT**: `int`

- **Restrictions**:
  ```
  required(VARIABLES.var)
  required(FEATURES.var)
  sv = |FEATURES|
  sv ≤ 3 ∨ rv ≤ 2 ⇒ FEATURES.var = DEFAULT
  FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv + 1
  FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv - 1
  DEFAULT < minv + 1 ∨ DEFAULT > maxv - 1
  ```

  Where
  - `maxv = maxval(VARIABLES.var)`
  - `minv = minval(VARIABLES.var)`
  - `sv = |VARIABLES|`
  - `rv = range(VARIABLES.var)`

- **Purpose**
  Consider the time-series given by the **VARIABLES** collection. If position `i` does not correspond to the first position in **VARIABLES** where an occurrence of **DECREASING_TERRACE** is identified then **FEATURES[i]** is the default value **DEFAULT**; otherwise **FEATURES[i]** gives the feature value of the corresponding occurrence of **DECREASING_TERRACE**.

  An occurrence of the pattern **DECREASING_TERRACE** is the maximal subsequence which matches the regular expression ```>+=+>```.

  Assume that the occurrence of the pattern **DECREASING_TERRACE** starts at position `i` and ends at position `j`. The feature **MIN** called **HEIGHT** in the name of the constraint since all feature values are identical, computes the minimum of the values from index `i + 1` to index `j`.

- **Example**
  Figure 4.309 provides an example where the **HEIGHT_DECREASING_TERRACE** ([6, 4, 4, 5, 2, 1, 3, 3, 5, 4, 4, 3, 3], [0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 4, 0, 0], 0) constraint holds.

- **Typical**
  ```
  |VARIABLES| > 3
  range(VARIABLES.var) > 2
  ```
Figure 4.309: Illustrating the HEIGHT_DECREASING_TERRACE constraint of the Example slot

Arg. properties: Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
## Description

**Origin**
Based on the `INCREASING_TERRACE` pattern.

**Constraint**

`HEIGHT_INCREASING_TERRACE(VARIABLES, FEATURES, DEFAULT)`

### Arguments

- **VARIABLES**: `collection(var−dvar)`
- **FEATURES**: `collection(var−dvar)`
- **DEFAULT**: `int`

### Restrictions

```
required(VARIABLES.var)
required(FEATURES.var)
sv = |FEATURES|
sv ≤ 3 ∨ rv ≤ 2 ⇒ FEATURES.var = DEFAULT
FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv + 1
FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv − 1
DEFAULT < minv + 1 ∨ DEFAULT > maxv − 1
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

Consider the time-series given by the `VARIABLES` collection. If position `i` does not correspond to the first position in `VARIABLES` where an occurrence of `INCREASING_TERRACE` is identified then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `INCREASING_TERRACE`.

### Purpose

An occurrence of the pattern `INCREASING_TERRACE` is the maximal subsequence which matches the regular expression `'<\text{=}+<'.

Assume that the occurrence of the pattern `INCREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `MIN`, called `HEIGHT` in the name of the constraint since all feature values are identical, computes the minimum of the values from index `i + 1` to index `j`.

### Example

Figure 4.310 provides an example where the `HEIGHT_INCREASING_TERRACE ([1,3,3,2,5,6,4,4,2,3,3,3,4,4],[0,0,0,0,0,0,5,0,0,0,0,0,0,0,0,0,3,0,0],0)` constraint holds.

### Typical

- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 2`
Figure 4.310: Illustrating the HEIGHT_INCREASING_TERRACE constraint of the Example slot

Arg. properties: Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
**Description**

**Origin**
Based on the **PLAIN** pattern.

**Constraint**

\[
\text{HEIGHT\_PLAIN}(\text{VARIABLES, FEATURES, DEFAULT})
\]

**Arguments**

- **VARIABLES**: \(\text{collection}(\text{var}−\text{dvar})\)
- **FEATURES**: \(\text{collection}(\text{var}−\text{dvar})\)
- **DEFAULT**: \(\text{int}\)

**Restrictions**

- \(\text{required}(\text{VARIABLES, var})\)
- \(\text{required}(\text{FEATURES, var})\)
- \(\text{sv} = |\text{FEATURES}|\)
- \(\text{sv} ≤ 2 \lor \text{rv} ≤ 1 \Rightarrow \text{FEATURES.var} = \text{DEFAULT}\)
- \(\text{FEATURES.var} = \text{DEFAULT} \lor \text{FEATURES_var} ≥ \text{minv}\)
- \(\text{FEATURES.var} = \text{DEFAULT} \lor \text{FEATURES.var} ≤ \text{maxv} - 1\)
- \(\text{DEFAULT} < \text{minv} \lor \text{DEFAULT} > \text{maxv} - 1\)

where

\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES.var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES.var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES.var})
\end{align*}
\]

Consider the time-series given by the **VARIABLES** collection. If position \(i\) does not correspond to the first position in **VARIABLES** where an occurrence of **PLAIN** is identified then **FEATURES[i]** is the default value **DEFAULT**; otherwise **FEATURES[i]** gives the feature value of the corresponding occurrence of **PLAIN**.

**Purpose**

An occurrence of the pattern **PLAIN** is the **maximal** subsequence which matches the regular expression ‘\(>\Rightarrow<\)’.

Assume that the occurrence of the pattern **PLAIN** starts at position \(i\) and ends at position \(j\). The feature **MIN**, called **HEIGHT** in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

Figure 4.311 provides an example where the **HEIGHT\_PLAIN** constraint holds.

- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES.var}) > 1\)

**Typical**

**Arg. properties**

Functional dependency: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.311: Illustrating the HEIGHT_PLAIN constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
**Origin**
Based on the PLATEAU pattern.

**Constraint**

\[ \text{HEIGHT\_PLATEAU} (\text{VARIABLES, FEATURES, DEFAULT}) \]

**Arguments**

- \( \text{VARIABLES} : \) collection(\(\text{var} \rightarrow \text{dvar}\) )
- \( \text{FEATURES} : \) collection(\(\text{var} \rightarrow \text{dvar}\) )
- \( \text{DEFAULT} : \) int

**Restrictions**

\[
\begin{align*}
\text{required} (\text{VARIABLES}, \text{var}) \\
\text{required} (\text{FEATURES}, \text{var}) \\
\text{sv} &= |\text{FEATURES}| \\
\text{sv} &\leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{FEATURES}.\text{var} = \text{DEFAULT} \\
\text{FEATURES}.\text{var} &= \text{DEFAULT} \lor \text{FEATURES}.\text{var} \geq \text{minv} + 1 \\
\text{FEATURES}.\text{var} &= \text{DEFAULT} \lor \text{FEATURES}.\text{var} \leq \text{maxv} \\
\text{DEFAULT} &\lt \text{minv} + 1 \lor \text{DEFAULT} > \text{maxv} \\
\text{where} \\
\text{maxv} &= \text{maxval} (\text{VARIABLES}.\text{var}) \\
\text{minv} &= \text{minval} (\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range} (\text{VARIABLES}.\text{var})
\end{align*}
\]

Consider the time-series given by the \( \text{VARIABLES} \) collection. If position \( i \) does not correspond to the first position in \( \text{VARIABLES} \) where an occurrence of \( \text{PLATEAU} \) is identified then \( \text{FEATURES}[i] \) is the default value \( \text{DEFAULT} \); otherwise \( \text{FEATURES}[i] \) gives the feature value of the corresponding occurrence of \( \text{PLATEAU} \).

**Purpose**

An occurrence of the pattern \( \text{PLATEAU} \) is the maximal subsequence which matches the regular expression '\'<\(=\)*>'.

Assume that the occurrence of the pattern \( \text{PLATEAU} \) starts at position \( i \) and ends at position \( j \). The feature \( \text{MIN} \), called \( \text{HEIGHT} \) in the name of the constraint since all feature values are identical, computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**

Figure 4.312 provides an example where the \( \text{HEIGHT\_PLATEAU} \) constraint holds.

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Typical**

**Arg. properties**

Functional dependency: \( \text{FEATURES} \) determined by \( \text{VARIABLES} \) and \( \text{DEFAULT} \).
Figure 4.312: Illustrating the \texttt{HEIGHT\_PLATEAU} constraint of the \textbf{Example} slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
Origin

Based on the PROPERPLAIN pattern.

Constraint

HEIGHT_PROPERPLAIN(VARIABLES, FEATURES, DEFAULT)

Arguments

VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int

Restrictions

required(VARIABLES.var)
required(FEATURES.var)
sv = |FEATURES|
sv ≤ 3 ∧ rv ≤ 1 ⇒ FEATURES.var = DEFAULT
FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv
FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv − 1
DEFAULT < minv ∨ DEFAULT > maxv − 1

where

maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Consider the time-series given by the VARIABLES collection. If position i does not correspond to the first position in VARIABLES where an occurrence of PROPERPLAIN is identified then FEATURES[i] is the default value DEFAULT; otherwise FEATURES[i] gives the feature value of the corresponding occurrence of PROPERPLAIN. An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression ‘>=’ <’.

Assume that the occurrence of the pattern PROPERPLAIN starts at position i and ends at position j. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index i + 1 to index j.

Example

Figure 4.313 provides an example where the HEIGHT_PROPERPLAIN ([2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5, 1, 0, 0, 5, 0, 0, 0, 0, 4, 0, 0, 0, 0, 3, 0, 0]) constraint holds.

Typical

|VARIABLES| > 3
range(VARIABLES.var) > 1

Arg. properties

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.313: Illustrating the HEIGHT_PROPER_PLAIN constraint of the Example slot
Use the decoration table 3.32 to synthesise the corresponding register automaton.
HEIGHT_PROPER_PLATEAU

DESCRIPTION

Origin
Based on the PROPER_PLATEAU pattern.

Constraint
HEIGHT_PROPER_PLATEAU(VARIABLES, FEATURES, DEFAULT)

Arguments
- VARIABLES : collection(var–dvar)
- FEATURES : collection(var–dvar)
- DEFAULT : int

Restrictions
required(VARIABLES.var)
required(FEATURES.var)
sv = |FEATURES|
sv ≤ 3 ∧ rv ≤ 1 ⇒ FEATURES.var = DEFAULT
FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv + 1
FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv
DEFAULT < minv + 1 ∨ DEFAULT > maxv

where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
Consider the time-series given by the VARIABLES collection. If position \( i \) does not correspond to the first position in VARIABLES where an occurrence of PROPER_PLATEAU is identified then FEATURES[\( i \)] is the default value DEFAULT; otherwise FEATURES[\( i \)] gives the feature value of the corresponding occurrence of PROPER_PLATEAU.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression ‘\(<\leq + \rangle\)’.

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \( i \) and ends at position \( j \). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \( i + 1 \) to index \( j \).

Example
Figure 4.314 provides an example where the HEIGHT_PROPER_PLATEAU ([7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3], [0, 0, 0, 3, 0, 0, 0, 0, 4, 0, 0, 0, 0, 5, 0], 0) constraint holds.

Typical
- \(|\text{VARIABLES}| > 3\)
- range(VARIABLES.var) > 1

Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.314: Illustrating the \textsc{height\_proper\_plateau} constraint of the \textbf{Example} slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
HEIGHT_PROPER_PLATEAU
### Origin
Based on the `STEADY` pattern.

### Constraint
\[
\text{HEIGHT\_STEADY}(\text{VARIABLES}, \text{FEATURES}, \text{DEFAULT})
\]

### Arguments
- **VARIABLES**: \text{collection}(\text{var}−\text{dvar})
- **FEATURES**: \text{collection}(\text{var}−\text{dvar})
- **DEFAULT**: \text{int}

### Restrictions
\[
\begin{align*}
\text{required}(\text{VARIABLES}.\text{var}) \\
\text{required}(\text{FEATURES}.\text{var}) \\
\text{sv} = |\text{FEATURES}| \\
\text{sv} \leq 1 \Rightarrow \text{FEATURES}.\text{var} = \text{DEFAULT} \\
\text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \geq \text{minv} \\
\text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \leq \text{maxv} \\
\text{DEFAULT} < \text{minv} \lor \text{DEFAULT} > \text{maxv}
\end{align*}
\]

where
\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Consider the time-series given by the `VARIABLES` collection. If position \(i\) does not correspond to the first position in `VARIABLES` where an occurrence of `STEADY` is identified then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `STEADY`.

### Purpose
An occurrence of the pattern `STEADY` is the subsequence which matches the regular expression `\'\='`. Assume that the occurrence of the pattern `STEADY` starts at position \(i\) and ends at position \(j\). The feature `MIN`, called `HEIGHT` in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i\) to index \(j+1\).

### Example
Figure 4.315 provides an example where the `HEIGHT\_STEADY` constraint holds.

### Typical
\[|\text{VARIABLES}| > 1\]

### Arg. properties
- **Functional dependency**: `FEATURES` determined by `VARIABLES` and `DEFAULT`. 
Figure 4.315: Illustrating the **HEIGHT_STEADY** constraint of the **Example** slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
Origin
Based on the **STEADY_SEQUENCE** pattern.

**Constraint**

HEIGHT_STEADY_SEQUENCE(VARIABLES, FEATURES, DEFAULT)

**Arguments**

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>collection(var−dvar)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FEATURES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>int</td>
</tr>
</tbody>
</table>

**Restrictions**

- `required(VARIABLES.var)`
- `required(FEATURES.var)`
- `sv = |FEATURES|`
- `sv ≤ 1 ⇒ FEATURES.var = DEFAULT`
- `FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv`
- `FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv`
- `DEFAULT ≤ minv ∨ DEFAULT > maxv`

where

- `maxv = maxval(VARIABLES.var)`
- `minv = minval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**

Consider the time-series given by the VARIABLES collection. If position \( i \) does not correspond to the first position in VARIABLES where an occurrence of **STEADY_SEQUENCE** is identified (even if this occurrence of pattern is not complete) then FEATURES[\( i \)] is the default value DEFAULT; otherwise FEATURES[\( i \)] gives the feature value of the corresponding occurrence of **STEADY_SEQUENCE**.

Assume that the occurrence of the pattern **STEADY_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \( i \) to index \( j + 1 \).

**Example**

Figure 4.316 provides an example where the HEIGHT_STEADY_SEQUENCE constraint holds.

**Typical**

| VARIABLES | > 1 |

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.316: Illustrating the HEIGHT_STEADY_SEQUENCE constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
### INCREASING_HEIGHT_DECREASING_TERRACE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_HEIGHT</td>
<td>DECREASING_TERRACE</td>
<td>↑</td>
</tr>
</tbody>
</table>

#### Description

**Origin**
Based on the `DECREASING_TERRACE` pattern.

**Constraint**

`INCREASING_HEIGHT_DECREASING_TERRACE(VARIABLES)`

**Argument**

`VARIABLES : collection(var-dvar)`

**Restriction**

`required(VARIABLES.var)`

**Purpose**

Succeeds if the minima of the values in each occurrence of the `DECREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection are increasing. An occurrence of the pattern `DECREASING_TERRACE` is the maximal subsequence which matches the regular expression `‘>=’‘+’‘>’`. Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position $i$ and ends at position $j$. The feature `MIN`, called `HEIGHT` in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i+1$ to index $j$.

**Example**

```
(⟨5, 2, 1, 5, 4, 3, 3, 2, 4, 4, 6, 5, 5, 4⟩)
```

Figure 4.317 provides an example where the `INCREASING_HEIGHT_DECREASING_TERRACE` constraint holds.

**Typical**

```
|VARIABLES| > 3
range(VARIABLES.var) > 2
```
Figure 4.317: Illustrating the INCREASING_HEIGHT_DECREASING_TERRACE constraint of the Example slot
Figure 4.318: Automaton for the increasing_height_decreasing_terrace constraint obtained by applying decoration Table 3.38 to the seed transducer of the decreasing_terrace pattern
**CONDITION**

**FEATURE**

**PATTERN**

INCREASING_HEIGHT_INCREASING_TERRACE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
<td>Based on the INCREASING_TERRACE pattern.</td>
</tr>
<tr>
<td>Constraint</td>
<td>INCREASING_HEIGHT_INCREASING_TERRACE(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES.var)</td>
</tr>
</tbody>
</table>

Succeeds if the minima of the values in each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are increasing. An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression ‘< =+ <’.

Purpose

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature \( \text{MIN} \), called \( \text{HEIGHT} \) in the name of the constraint since all feature values are identical, computes the minimum of the values from index \( i + 1 \) to index \( j \).

Example

\[ ((1, 2, 2, 4, 5, 5, 6, 6, 4, 4, 2, 5, 5, 7, 7)) \]

Figure 4.319 provides an example where the INCREASING_HEIGHT_INCREASING_TERRACE ((1, 2, 2, 4, 5, 5, 6, 6, 4, 4, 2, 5, 5, 7, 7)) constraint holds.

Typical

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range(VARIABLES.var)} > 2 \]
Figure 4.319: Illustrating the INCREASING_HEIGHT_INCREASING_TERRACE constraint of the Example slot
INCREASING_HEIGHT_INCREASING_TERRACE

Automaton

Figure 4.320 depicts the automaton associated with the constraint INCREASING_HEIGHT_INCREASING_TERRACE.

\[
\begin{align*}
\{ C \leftarrow -\infty \} &\quad \{ D \leftarrow +\infty \} &\quad \{ F \leftarrow -\infty \} &\quad \{ R \leftarrow 1 \} \\
\{ D \leftarrow \min(D, \text{VAR}_i) \} &\quad \{ D \leftarrow \min(D, \text{VAR}_i) \} &\quad \{ C \leftarrow \min(D, \text{VAR}_i) \} &\quad \{ D \leftarrow +\infty \} \\
\{ C \leftarrow \min(D, \text{VAR}_i) \} &\quad \{ D \leftarrow +\infty \} &\quad \{ F \leftarrow \min(D, \text{VAR}_i) \} &\quad \{ R \leftarrow R \land (F \leq \min(D, \text{VAR}_i)) \}
\end{align*}
\]

Figure 4.320: Automaton for the INCREASING_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING_TERRACE pattern
INCREASING_HEIGHTPLAIN

**Origin**
Based on the PLAIN pattern.

**Constraint**
INCREASING_HEIGHTPLAIN(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘> =* <’. Assume that the occurrence of the pattern PLAIN starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i + 1$ to index $j$.

**Example**

```
((3, 6, 6, 3, 4, 5, 5, 4, 6, 6, 7, 5, 5, 6, 3, 2))
```

Figure 4.321 provides an example where the INCREASING_HEIGHTPLAIN ((3, 6, 6, 3, 4, 5, 5, 4, 6, 6, 7, 5, 5, 6, 3, 2)) constraint holds.

**Typical**

$|\text{VARIABLES}| > 2$

$\text{range(}\text{VARIABLES.var}) > 1$
INCREASING_HEIGHT_PLAIN

Figure 4.321: Illustrating the INCREASING_HEIGHT_PLAIN constraint of the Example slot
Figure 4.322 depicts the automaton associated with the constraint INCREASING_HEIGHT.PLAIN.

Figure 4.322: Automaton for the INCREASING_HEIGHT.PLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLAIN pattern.
### INCREASING\_HEIGHT\_PLATEAU

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ ▶ ▶ ▶</td>
<td>Description</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the PLATEAU pattern.

**Constraint**

```plaintext
INCREASING\_HEIGHT\_PLATEAU(VARIABLES)
```

**Argument**

```plaintext
VARIABLES : collection(var−dvar)
```

**Restriction**

```plaintext
required(VARIABLES, var)
```

Succeeds if the minima of the values in each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression `< = >`.

**Purpose**
Assume that the occurrence of the pattern PLATEAU starts at position `i` and ends at position `j`. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index `i + 1` to index `j`.

**Example**

```plaintext
((7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5))
```

Figure 4.323 provides an example where the INCREASING\_HEIGHT\_PLATEAU ((7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5)) constraint holds.

**Typical**

- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`
Figure 4.323: Illustrating the INCREASING_HEIGHT_PLATEAU constraint of the Example slot
Automaton Figure 4.324 depicts the automaton associated with the constraint INCREASING_HEIGHT_PLATEAU.

Figure 4.324: Automaton for the INCREASING_HEIGHT_PLATEAU constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLATEAU pattern
### Origin
Based on the `PROPER.PLAIN` pattern.

### Constraint
`INCREASING_HEIGHT_PROPERPLAIN(VARIABLES)`

### Argument
`VARIABLES : collection(var−dvar)`

### Restriction
`required(VARIABLES.var)`

### Purpose
Succeeds if the minima of the values in each occurrence of the `PROPER.PLAIN` pattern in the time-series given by the `VARIABLES` collection are increasing.

An occurrence of the pattern `PROPER.PLAIN` is the **maximal** subsequence which matches the regular expression `'>= '+' '<'`.

Assume that the occurrence of the pattern `PROPER.PLAIN` starts at position $i$ and ends at position $j$. The feature `MIN`, called `HEIGHT` in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i+1$ to index $j$.

### Example
```
((5,3,3,5,6,5,4,4,7,3,6,5,5,7,2))
```

Figure 4.325 provides an example where the `INCREASING_HEIGHT_PROPERPLAIN` `((5,3,3,5,6,5,4,4,7,3,6,5,5,7,2))` constraint holds.

### Typical
- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 1`
Figure 4.325: Illustrating the INCREASING_HEIGHT_PROPER_PLAIN constraint of the Example slot
Automaton Figure 4.326 depicts the automaton associated with the constraint INCREASING_HEIGHT_PROPER_PLAIN.

Figure 4.326: Automaton for the INCREASING_HEIGHT_PROPER_PLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PROPER_PLAIN pattern.
INCREASING_HEIGHT_PROPER_PLATEAU

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_HEIGHT_PROPER_PLATEAU</td>
<td>▶ ▶ ▶ ▶</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the PROPER_PLATEAU pattern.

**Constraint**
INCREASING_HEIGHT_PROPER_PLATEAU(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression '<=+>'.

**Purpose**
Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \(i\) and ends at position \(j\). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**
\[((7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 5, 3))\]

Figure 4.327 provides an example where the INCREASING_HEIGHT_PROPER_PLATEAU \(((7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 5, 3))\) constraint holds.

**Typical**
\[|\text{VARIABLES}| > 3\]
\[\text{range}(|\text{VARIABLES}.var|) > 1\]
INCREASING_HEIGHT_PROPER_PLATEAU

Figure 4.327: Illustrating the INCREASING_HEIGHT_PROPER_PLATEAU constraint of the Example slot
**Figure 4.328** depicts the automaton associated with the constraint `INCREASING_HEIGHT_PROPER_PLATEAU`.

**Automaton**

\[
\begin{align*}
&\{D \leftarrow \min(D, \text{VAR}_i)\} \\
&\{D \leftarrow +\infty\} \\
&\{D \leftarrow \text{VAR}_i\} \\
&\{D \leftarrow +\infty\} \\
&\{D \leftarrow \min(D, \text{VAR}_i)\} \\
&\{D \leftarrow \min(D, \text{VAR}_i)\} < \\
&\{D \leftarrow +\infty\} <
\end{align*}
\]

Figure 4.328: Automaton for the `INCREASING_HEIGHT_PROPER_PLATEAU` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `PROPER_PLATEAU` pattern
INCREASING_HEIGHT_PROPER_PLATEAU 829
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_HEIGHT_STEADY</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the **STEADY** pattern.

**Constraint**
INCREASING_HEIGHT_STEADY(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the minima of the values in each occurrence of the **STEADY** pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern **STEADY** is the subsequence which matches the regular expression ‘\='.
Assume that the occurrence of the pattern **STEADY** starts at position \(i\) and ends at position \(j\). The feature **MIN**, called **HEIGHT** in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
((1,1,1,7,3,3,5,5,5,6,5,5,7,2,6,6))
\]

Figure 4.329 provides an example where the **INCREASING_HEIGHT_STEADY** \(((1,1,7,3,3,5,5,6,5,5,5,7,2,6,6))\) constraint holds.

**Typical**

\(|VARIABLES| > 1\)
Figure 4.329: Illustrating the INCREASING_HEIGHT_STEADY constraint of the Example slot
Figure 4.330 depicts the automaton associated with the constraint INCREASING_HEIGHT_STEADY.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow +\infty \\
F & \leftarrow -\infty \\
R & \leftarrow 1 \\
R \land (F \leq C) & \\
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D & \leftarrow +\infty \\
F & \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
R & \leftarrow R \land (F \leq \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1})) \\
\end{align*}
\]

Figure 4.330: Automaton for the INCREASING_HEIGHT_STEADY constraint obtained by applying decoration Table 3.38 to the seed transducer of the STEADY pattern.
INCREASING_HEIGHT_STEADY_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the STEADY_SEQUENCE pattern.

**Constraint**

\[
\text{INCREASING\_HEIGHT\_STEADY\_SEQUENCE}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

**Purpose**

Succeeds if the minima of the values in each occurrence of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression `^\=\+$'.

Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
((6, 2, 2, 3, 5, 6, 4, 4, 2, 3, 5, 5, 2, 3, 4))
\]

Figure 4.331 provides an example where the INCREASING_HEIGHT_STEADY_SEQUENCE ((6, 2, 2, 3, 5, 6, 4, 4, 2, 3, 5, 5, 2, 3, 4)) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]
Figure 4.331: Illustrating the INCREASING_HEIGHT_STEADY_SEQUENCE constraint of the Example slot
Automaton

Figure 4.332 depicts the automaton associated with the constraint \textsc{increasing\_height\_steady\_sequence}.

\[
\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow +\infty \\
F &\leftarrow -\infty \\
R &\leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
D &\leftarrow +\infty \\
F &\leftarrow C \\
R &\leftarrow R \land (F \leq C) \\
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow \min(D, \text{VAR}_i, \text{VAR}_{i+1}) \\
D &\leftarrow +\infty \\
\end{align*}
\]

Figure 4.332: Automaton for the \textsc{increasing\_height\_steady\_sequence} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{steady\_sequence} pattern.
INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE

**Origin**
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**
INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the maxima of the values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence ‘>><<’.
Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index $i+2$ to index $j$.

**Example**
$((7, 5, 4, 2, 5, 4, 3, 3, 5, 7, 6, 5, 6, 5, 4, 1))$

Figure 4.333 provides an example where the INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE ($[(7, 5, 4, 2, 5, 4, 3, 3, 5, 7, 6, 5, 6, 5, 4, 1)]$) constraint holds.

**Typical**
$|\text{VARIABLES}| > 5$
$\text{range(VARIABLES.var)} > 2$
INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE

Figure 4.333: Illustrating the INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.334 depicts the automaton associated with the constraint INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE.

Figure 4.334: Automaton for the INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern
### INCREASING_MAX_DECENDING

**Description**

Based on the `DECREASING` pattern.

**Constraint**

`INCREASING_MAX_DECENDING(VARIABLES)`

**Argument**

`VARIABLES : collection(var-dvar)`

**Restriction**

`required(VARIABLES.var)`

Succeeds if the maxima of the values in each occurrence of the `DECREASING` pattern in the time-series given by the `VARIABLES` collection are increasing. An occurrence of the pattern `DECREASING` is the subsequence which matches the regular expression `'>'`. Assume that the occurrence of the pattern `DECREASING` starts at position $i$ and ends at position $j$. The feature `MAX` computes the maximum of the values from index $i$ to index $j + 1$.

#### Purpose

**Example**

```
((2, 1, 1, 1, 3, 2, 2, 3, 1, 5, 3, 4, 5, 6, 5))
```

Figure 4.335 provides an example where the `INCREASING_MAX_DECENDING` constraint holds.

#### Typical

```
|VARIALES| > 1
range(VARIABLES.var) > 1
```
Figure 4.335: Illustrating the INCREASING_MAX_DECREASING constraint of the Example slot
Figure 4.336 depicts the automaton associated with the constraint \textsc{increasing\_max\_decreasing}.  

\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow -\infty \\
F &\leftarrow -\infty \\
R &\leftarrow 1
\end{align*}

\begin{align*}
C &\leftarrow \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D &\leftarrow -\infty \\
F &\leftarrow \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
R &\leftarrow R \land (F \leq \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}))
\end{align*}

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{automaton.png}
\caption{Automaton for the \textsc{increasing\_max\_decreasing} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{decreasing} pattern}
\end{figure}
### Condition

<table>
<thead>
<tr>
<th>Condition</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_MAX_DECREASING_SEQUENCE</td>
<td>INCREASING_MAX_DECREASING_SEQUENCE</td>
</tr>
</tbody>
</table>

### Description

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**

\[
\text{INCREASING\_MAX\_DECREASING\_SEQUENCE}(\text{VAR})
\]

**Argument**

\[
\text{VAR} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required} (\text{VAR}.\text{var})
\]

**Purpose**

Succeeds if the maxima of the values in each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VAR collection are increasing.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression `(> | =) * > | >`.

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
((4, 3, 2, 3, 4, 4, 6, 5, 3, 4, 4, 6, 4, 6))
\]

Figure 4.337 provides an example where the INCREASING_MAX_DECREASING_SEQUENCE \([(4, 3, 2, 2, 3, 4, 4, 6, 5, 3, 4, 4, 6, 4, 6)]\) constraint holds.

**Typical**

\[
|\text{VAR}| > 1
\]

\[
\text{range}(\text{VAR}.\text{var}) > 1
\]
Figure 4.337: Illustrating the INCREASING_MAX_DECREASING_SEQUENCE constraint of the Example slot
Figure 4.338 depicts the automaton associated with the constraint INCREASING_MAX_DECREASING_SEQUENCE.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow -\infty \\
F & \leftarrow -\infty \\
R & \leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
\leq s & \Rightarrow \\
\leq t & \Rightarrow \\
C & \leftarrow \max(D, \text{VAR}_i, \text{VAR}_{i+1}) \\
D & \leftarrow -\infty \\
\end{align*}
\]

\[
\begin{align*}
\geq s & \Rightarrow \\
\geq t & \Rightarrow \\
C & \leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \\
D & \leftarrow -\infty \\
\end{align*}
\]

Figure 4.338: Automaton for the INCREASING_MAX_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING_SEQUENCE pattern.
INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**
INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the maxima of the values in each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence ‘<<><<’.
Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index $i + 2$ to index $j$.

**Example**

```plaintext
((1, 2, 3, 2, 3, 4, 6, 5, 1, 4, 5, 6, 0, 2, 4, 4))
```

Figure 4.339 provides an example where the INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE $([1, 2, 3, 2, 3, 4, 6, 5, 1, 4, 5, 6, 0, 2, 4, 4])$ constraint holds.

**Typical**

$|\text{VARIABLES}| > 5$

range(VARIABLES.var) > 2
Figure 4.339: Illustrating the INCREASING_MAX_DIP_ON_INCREMENTING_SEQUENCE constraint of the Example slot
Automaton  Figure 4.340 depicts the automaton associated with the constraint INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE.

Figure 4.340: Automaton for the INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern
INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE
### INCREASING_MAX_INCREASING

**Description**

**Origin**

Based on the **INCREASING** pattern.

**Constraint**

INCREASING_MAX_INCREASING(VARIABLES)

**Argument**

VARIABLES : collection(var–dvar)

**Restriction**

required(VARIABLES.var)

Succeeds if the maxima of the values in each occurrence of the **INCREASING** pattern in the time-series given by the VARIABLES collection are increasing. An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression '<'. Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[(2, 1, 2, 0, 2, 3, 1, 1, 3, 4, 2, 1, 1, 0, 6, 7)\]

Figure 4.341 provides an example where the INCREASING_MAX_INCREASING \([(2, 1, 2, 0, 2, 3, 1, 1, 3, 4, 2, 1, 1, 0, 6, 7)]\) constraint holds.

**Typical**

| VARIABLES | > 1
range(VARIABLES.var) | > 1
Figure 4.341: Illustrating the INCREASING_MAX_INCREASING constraint of the Example slot
Figure 4.342 depicts the automaton associated with the constraint INCREASING_MAX_INCREASING.

\[
\begin{aligned}
C \leftarrow & -\infty \\
D \leftarrow & -\infty \\
F \leftarrow & -\infty \\
R \leftarrow & 1 \\
\end{aligned}
\]

\[
R \land (F \leq C) \geq
\]

\[
\begin{aligned}
C \leftarrow & \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D \leftarrow & -\infty \\
F \leftarrow & \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
R \leftarrow & R \land (F \leq \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1})) \\
\end{aligned}
\]

Figure 4.342: Automaton for the INCREASING_MAX_INCREASING constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING pattern.
### CONDITION

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>INCREASING_MAX_INCREASING_SEQUENCE</strong></td>
<td><img src="image" alt="DESCRIPTION AUTOMATON" /></td>
</tr>
</tbody>
</table>

### Origin
Based on the INCREASING_SEQUENCE pattern.

### Constraint

\[ \text{INCREASING_MAX_INCREASING_SEQUENCE}(\text{VARIABLES}) \]

### Argument

\[ \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \]

### Restriction

\[ \text{required}(\text{VARIABLES}.\text{var}) \]

### Purpose

Succeeds if the maxima of the values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\(< (< | =)\)* < | <\)’.

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

### Example

\[ (4, 1, 4, 5, 5, 4, 4, 2, 3, 3, 4, 5, 6, 6, 5, 6) \]

Figure 4.343 provides an example where the INCREASING_MAX_INCREASING_SEQUENCE \([4, 1, 4, 5, 5, 4, 4, 2, 3, 3, 4, 5, 6, 6, 5, 6]\) constraint holds.

### Typical

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 4.343: Illustrating the `INCREASING_MAX_INCREASING_SEQUENCE` constraint of the `Example` slot.
Automaton Figure 4.344 depicts the automaton associated with the constraint INCREASING_MAX_INCREASING_SEQUENCE.

\[ \begin{align*}
&\begin{cases}
    C \leftarrow -\infty \\
    D \leftarrow -\infty \\
    F \leftarrow -\infty \\
    R \leftarrow 1
\end{cases} \\
&\geq s
\end{align*} \]

\[ \begin{align*}
&\begin{cases}
    D \leftarrow -\infty \\
    F \leftarrow C \\
    R \leftarrow R \land (F \leq C)
\end{cases} \\
&\geq \}
\]

\[ \begin{align*}
&\begin{cases}
    D \leftarrow \max(D, \text{VAR}_i+1) \\
    C \leftarrow \max(C, \max(D, \text{VAR}_i+1)) \\
    D \leftarrow -\infty
\end{cases} \\
&\geq \}
\]

\[ \begin{align*}
&\begin{cases}
    D \leftarrow \max(D, \text{VAR}_i) \\
    C \leftarrow \max(C, \min(D, \text{VAR}_i)) \\
    D \leftarrow -\infty
\end{cases} \\
&\leq \}
\]

Figure 4.344: Automaton for the INCREASING_MAX_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING_SEQUENCE pattern.
INCREASING_MAX_INFLEXION

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_MAX_INFLEXION</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the INFLEXION pattern.

**Constraint**
INCREASING_MAX_INFLEXION(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the maxima of the values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression '< (< | =) > | > (> | =) > <'.

Assume that the occurrence of the pattern INFLEXION starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 1 to index j.

**Example**

\[(5, 5, 2, 2, 1, 1, 2, 2, 3, 4, 3)\]

Figure 4.345 provides an example where the INCREASING_MAX_INFLEXION \((5, 5, 2, 2, 1, 1, 2, 2, 3, 4, 3)\) constraint holds.

**Typical**

\[|VARIABLES| > 2\]

\[\text{range}(\text{VARIABLES}\.var) > 1\]
INCREASING_MAX_INFLEXION

INCREASING_MAX_INFLEXION constraint of the Exam-
ple slot

Figure 4.345: Illustrating the INCREASING_MAX_INFLEXION constraint of the Example slot
Automaton Figure 4.346 depicts the automaton associated with the constraint INCREASING_MAX_INFLEXION.

Figure 4.346: Automaton for the INCREASING_MAX_INFLEXION constraint obtained by applying decoration Table 3.38 to the seed transducer of the INFLEXION pattern (transition $r \rightarrow t$ has the same registers updates as transition $t \rightarrow r$)
INCREASING_MAX_INFLEXION


**Origin**
Based on the **PEAK** pattern.

**Constraint**

\[
\text{INCREASING\_MAX\_PEAK} (\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the maxima of the values in each occurrence of the **PEAK** pattern in the time-series given by the **VARIABLES** collection are increasing.

**Purpose**

An occurrence of the pattern **PEAK** is the *maximal* subsequence which matches the regular expression 

\[
< (\geq | <)^* (> | \geq)^* >
\]

Assume that the occurrence of the pattern **PEAK** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
((7,5,5,1,2,3,2,2,3,4,5,2,6,6,6,1))
\]

Figure 4.347 provides an example where the **INCREASING\_MAX\_PEAK** \((\{7,5,5,1,2,3,2,2,3,4,5,2,6,6,6,1\})\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.347: Illustrating the INCREASING_MAX_PEAK constraint of the Example slot
Automaton

Figure 4.348 depicts the automaton associated with the constraint INCREASING_MAX_PEAK.

\[
\begin{align*}
\{ C \leftarrow -\infty \} & \quad \geq s \\
\{ D \leftarrow -\infty \} & \quad \geq t \\
\{ F \leftarrow -\infty \} & \quad \geq R \\
\{ R \leftarrow 1 \} & \quad \geq t \\
\{ D \leftarrow \max(D, \text{VAR}_i) \} & \quad \geq \}
\end{align*}
\]

Figure 4.348: Automaton for the INCREASING_MAX_PEAK constraint obtained by applying decoration Table 3.38 to the seed transducer of the PEAK pattern
INCREASING_MAX_PEAK

869
INCREASING_MAX.Strictly_DECREASING_SEQUENCE

**Description**
Based on the **STRICTLY_DECREASING_SEQUENCE** pattern.

**Constraint**
INCREASING_MAX.Strictly_DECREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the maxima of the values in each occurrence of the **STRICTLY_DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the *maximal* sub-sequence which matches the regular expression ‘>’.
Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position *i* and ends at position *j*. The feature MAX computes the maximum of the values from index *i* to index *j + 1*.

**Example**
((2, 3, 2, 2, 3, 4, 5, 6, 5, 4, 3, 2, 4, 7, 4))

Figure 4.349 provides an example where the INCREASING_MAX.Strictly_DECREASING_SEQUENCE((2, 3, 2, 2, 3, 4, 5, 6, 5, 4, 3, 2, 4, 7, 4)) constraint holds.

**Typical**
|VARIABLES| > 1
|range(VARIABLES.var) > 1
Figure 4.349: Illustrating the INCREASING_MAX_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.350 depicts the automaton associated with the constraint INCREASING\_MAX\_STRICTLY\_DECREASING\_SEQUENCE.

\[
\begin{align*}
&C \leftarrow -\infty \\
&D \leftarrow -\infty \\
&F \leftarrow -\infty \\
&R \leftarrow 1
\end{align*}
\]

Figure 4.350: Automaton for the INCREASING\_MAX\_STRICTLY\_DECREASING\_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STRICTLY\_DECREASING\_SEQUENCE pattern.
INCREASING_MAX.Strictly_DECREASING_SEQUENCE 873
Origin

Based on the STRICTLY_INCREASING_SEQUENCE pattern.

Constraint

INCREASING_MAX_STRICTLY_INCREASING_SEQUENCE(VARIABLES)

Argument

VARIABLES : collection(var−dvar)

Restriction

required(VARIABLES.var)

Purpose

Succeeds if the maxima of the values in each occurrence of the

STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES

collection are increasing.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal sub-

sequence which matches the regular expression '＜+'.

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at

position i and ends at position j. The feature MAX computes the maximum of the values

from index i to index j + 1.

Example

(((4, 1, 4, 5, 5, 4, 4, 2, 3, 4, 5, 6, 6, 5, 6)))

Figure 4.351 provides an example where the INCREAS-

ING_MAX_STRICTLY_INCREASING_SEQUENCE ([4, 1, 4, 5, 5, 4, 4, 2, 3, 4, 5, 6, 6, 5, 6])

constraint holds.

Typical

|VARIABLES| > 1

range(VARIABLES.var) > 1
Figure 4.351: Illustrating the increasing_max_strictly_increasing_sequence constraint of the Example slot
Automaton

Figure 4.352 depicts the automaton associated with the constraint \textsc{increasing\_max\_strictly\_increasing\_sequence}.

\[
\begin{align*}
& C \leftarrow -\infty \\
& D \leftarrow -\infty \\
& F \leftarrow -\infty \\
& R \leftarrow 1 \\
& \geq s
\end{align*}
\]

\[
\begin{align*}
& D \leftarrow -\infty \\
& F \leftarrow C \\
& R \leftarrow R \land (F \leq C) \\
& \geq
\end{align*}
\]

\[
\begin{align*}
& C \leftarrow \max(D, \text{VAR}_i, \text{VAR}_{i+1}) \\
& D \leftarrow -\infty \\
& \geq
\end{align*}
\]

\[
\begin{align*}
& C \leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \\
& D \leftarrow -\infty
\end{align*}
\]

Figure 4.352: Automaton for the \textsc{increasing\_max\_strictly\_increasing\_sequence} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{strictly\_increasing\_sequence} pattern.
INCREASING\_MAX\_STRICLY\_INCREASING\_SEQUENCE
**Condition**

INCREASING_MAX_SUMMIT

**Feature**

\((< | < (= | <)*) <)(> | > (= | >)*)\)

**Pattern**

**DESCRIPTION AUTOMATON**

**Origin**

Based on the SUMMIT pattern.

**Constraint**

INCREASING_MAX_SUMMIT(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES.var)

Succeeds if the maxima of the values in each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression \(’(< | < (= | <)*) <)(> | > (= | >)*)’\).

Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\(((1,5,2,1,6,6,2,3,5,4,1,4,6,4,3,2))\)

Figure 4.353 provides an example where the INCREASING_MAX_SUMMIT \(((1,5,2,1,6,6,2,3,5,4,1,4,6,4,3,2))\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 2
range(\text{VARIABLES.var}) > 1\)
Figure 4.353: Illustrating the INCREASING_MAX_SUMMIT constraint of the Example slot
Automaton Figure 4.354 depicts the automaton associated with the constraint INCREASING_MAX_SUMMIT.

Figure 4.354: Automaton for the INCREASING_MAX_SUMMIT constraint obtained by applying decoration Table 3.38 to the seed transducer of the SUMMIT pattern (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
CONDITION  FEATURE  PATTERN

INCREASING_MAX_ZIGZAG

⇒ △ ◤ DESCRIPTION  AUTOMATON  △ ◤ ◤

Origin  Based on the ZIGZAG pattern.

Constraint  INCREASING_MAX_ZIGZAG(VARIABLES)

Argument  VARIABLES : collection(var−dvar)

Restriction  required(VARIABLES.var)

Purpose  Succeeds if the maxima of the values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression "(<>)(< | <>)(<>)(>|<>)(|<>)."

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

Example  \(((7,7,2,5,3,4,6,3,6,1,1,5,4,7,5,6))\)

Figure 4.355 provides an example where the INCREASING_MAX_ZIGZAG \(((7,7,2,5,3,4,6,3,6,1,1,5,4,7,5,6))\) constraint holds.

Typical  |VARIABLES| > 3

range(VARIABLES.var) > 1
Figure 4.355: Illustrating the INCREASING_MAX_ZIGZAG constraint of the Example slot
Automaton

Figure 4.356 depicts the automaton associated with the constraint INCREASING_MAX_ZIGZAG.
Figure 4.356: Automaton for the INCREASING_MAX_ZIGZAG constraint obtained by applying decoration Table 3.38 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from a, b, c, d, e, f to s are labelled by $=\$; (2) on transitions from b, c, e, f to s the register $D$ is reset to its initial value; (3) on transitions from c, f to s the register $F$ is reset to $C$, and the register $R$ is updated wrt $C$ and $F$.
INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE

Origin
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

Constraint
INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES.var)

Purpose
Succeeds if the minima of the values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence ‘>>><’.
Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i+2$ to index $j$.

Example

\[
(5,5,4,2,6,5,4,4,5,7,6,5,6,4,1,1)
\]

Figure 4.357 provides an example where the INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE \([5, 5, 4, 2, 6, 5, 4, 4, 5, 7, 6, 5, 6, 4, 1, 1]\) constraint holds.

Typical

\[|\text{VARIABLES}| > 5\]
\[\text{range}(\text{VARIABLES.var}) > 2\]
Figure 4.357: Illustrating the INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.358 depicts the automaton associated with the constraint INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE.

Figure 4.358: Automaton for the INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern
INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE
INCREASING_MIN_DECREASING

**Description Automaton**

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_MIN_DECREASING</td>
<td>▶</td>
<td>▶</td>
</tr>
</tbody>
</table>

**Origin**
Based on the DECREASING pattern.

**Constraint**
INCREASING_MIN_DECREASING(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the minima of the values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression ‘>’.

Assume that the occurrence of the pattern DECREASING starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

$$((3, 6, 6, 1, 1, 3, 3, 4, 5, 6, 1, 5, 3, 3, 6, 5))$$

Figure 4.359 provides an example where the INCREASING_MIN_DECREASING ($((3, 6, 6, 1, 1, 3, 3, 4, 5, 6, 1, 5, 3, 3, 6, 5))$) constraint holds.

**Typical**

$|\text{VARIABLES}| > 1$

$\text{range(}\text{VARIABLES.var}) > 1$
Figure 4.359: Illustrating the INCREASING_MIN_DECREASING constraint of the Example slot
Automaton

Figure 4.360 depicts the automaton associated with the constraint INCREASING_MIN_DECREASING.

\[
\begin{align*}
&C \leftarrow -\infty \\
&D \leftarrow +\infty \\
&F \leftarrow -\infty \\
&R \leftarrow 1 \\
\end{align*}
\]

\[
R \land (F \leq C)
\]

Figure 4.360: Automaton for the INCREASING_MIN_DECREASING constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING pattern.
# Increasing Min Decreasing Sequence

## Condition

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Increasing Min Decreasing Sequence</strong></td>
<td>▶ ▶ &lt; ◄</td>
<td>△ ◄</td>
</tr>
</tbody>
</table>

## Description

**Origin**

Based on the *Decreasing Sequence* pattern.

**Constraint**

\[
\text{Increasing Min Decreasing Sequence(VARIABLES)}
\]

**Argument**

\[
\text{VARIABLES : collection(var\text{--}dvar)}
\]

**Restriction**

\[
\text{required(VARIABLES.var)}
\]

**Purpose**

Succeeds if the minima of the values in each occurrence of the *Decreasing Sequence* pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern *Decreasing Sequence* is the *maximal* subsequence which matches the regular expression `>(> | =)* > | >`.

Assume that the occurrence of the pattern *Decreasing Sequence* starts at position \(i\) and ends at position \(j\). The feature \(\text{MIN}\) computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
((4,3,2,2,3,4,4,6,3,4,4,5,5,6,4,6))
\]

Figure 4.361 provides an example where the *Increasing Min Decreasing Sequence* \(((4,3,2,2,3,4,4,6,3,4,4,5,5,6,4,6))\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
\text{range(VARIABLES.var)} > 1
\]
INCREASING_MIN_DECREASING_SEQUENCE

Figure 4.361: Illustrating the INCREASING_MIN_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.362 depicts the automaton associated with the constraint \textit{INCREASING\_MIN\_DECREASING\_SEQUENCE}.

\begin{align*}
&\text{\leq s} \\
&\text{\leq} \\
&\{ D \leftarrow +\infty \} \\
&\{ \text{VAR}_i \bullet C \leq} \\
&\{ R \leftarrow R \land (F \leq C) \} \\
&\{ C \leftarrow \min(D, \text{VAR}_i + 1) \} \\
&\{ D \leftarrow +\infty \} \\
&\text{\geq t} \\
&\{ D \leftarrow \min(D, \text{VAR}_i + 1) \} \\
&\{ \text{VAR}_i \bullet \} \\
&\{ \leq} \\
&\{ C \leftarrow \min(D, \text{VAR}_i + 1) \} \\
&\{ D \leftarrow +\infty \} \\
\end{align*}

Figure 4.362: Automaton for the \textit{INCREASING\_MIN\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textit{DECREASING\_SEQUENCE} pattern.
<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td>Based on the <strong>DIP_ON_INCREASING_SEQUENCE</strong> pattern.</td>
<td><img src="image" alt="Automaton" /></td>
</tr>
</tbody>
</table>

**Origin**

Based on the **DIP_ON_INCREASING_SEQUENCE** pattern.

**Constraint**

INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES.var)

Succeeds if the minima of the values in each occurrence of the **DIP_ON_INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** is the subsequence which matches the regular expression ‘<<><<’.

Assume that the occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i + 2\) to index \(j\).

**Example**

\(((1,2,3,0,3,4,6,5,1,4,5,6,1,2,4,4))\)

Figure 4.363 provides an example where the **INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE** constraint holds.

**Typical**

\(|\text{VARIABLES}| > 5\)

range(\text{VARIABLES.var}) > 2
Figure 4.363: Illustrating the \texttt{INCREASING\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE} constraint of the \textbf{Example} slot
Figure 4.364 depicts the automaton associated with the constraint INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE.

Figure 4.364: Automaton for the INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern.
### INCREASING_MIN_GORGE

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_MIN_GORGE</td>
<td>▶ ▷ ▶ &lt; ◄</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

#### Origin
Based on the GORGE pattern.

#### Constraint
$\text{INCREASING\_MIN\_GORGE(}\text{VARIABLES})$

#### Argument
$\text{VARIABLES : collection(var–dvar)}$

#### Restriction
$\text{required(}\text{VARIABLES.var})$

Succeeds if the minima of the values in each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are increasing.

#### Purpose
An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression $'(> | > (= | >)* >) (< | < (= | <)* <)'$. Assume that the occurrence of the pattern GORGE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

#### Example
$((6, 2, 5, 1, 1, 5, 4, 2, 3, 6, 5, 4, 4, 3, 5, 5))$

Figure 4.365 provides an example where the INCREASING_MIN_GORGE $([6, 2, 5, 1, 1, 5, 4, 2, 3, 6, 5, 4, 4, 3, 5, 5])$ constraint holds.

#### Typical
$|\text{VARIABLES}| > 2$
$\text{range(}\text{VARIABLES.var}) > 1$
Figure 4.365: Illustrating the INCREASING\_MIN\_GORGE constraint of the Example slot
Automaton

Figure 4.366 depicts the automaton associated with the constraint INCREASING_MIN_GORGE.

Figure 4.366: Automaton for the INCREASING_MIN_GORGE constraint obtained by applying decoration Table 3.38 to the seed transducer of the GORGE pattern (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
**Condition**

**Feature**

**Pattern**

**DESCRIPTION AUTOMATON**

**Origin**

Based on the INCREASING pattern.

**Constraint**

INCREASING_MIN_INCREASING(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression '<<'.

Assume that the occurrence of the pattern INCREASING starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i \) to index \( j+1 \).

**Purpose**

**Example**

\(((4, 1, 2, 4, 3, 3, 2, 3, 3, 4, 4, 6, 6, 5, 7))\)

Figure 4.367 provides an example where the INCREASING_MIN_INCREASING ([4, 1, 2, 4, 3, 3, 2, 3, 3, 4, 4, 6, 6, 5, 7]) constraint holds.

**Typical**

|VARIABLES| > 1
range(VARIABLES.var) > 1
Figure 4.367: Illustrating the INCREASING_MIN_INCREASING constraint of the Example slot
Figure 4.368 depicts the automaton associated with the constraint \texttt{INCREASING\_MIN\_INCREASING}.

Figure 4.368: Automaton for the \texttt{INCREASING\_MIN\_INCREASING} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{INCREASING} pattern.
### INCREASING_MIN_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_MIN_INCREASING_SEQUENCE</td>
<td>▶▷◁◀</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

#### Origin
Based on the INCREASING_SEQUENCE pattern.

#### Constraint
INCREASING_MIN_INCREASING_SEQUENCE(VARIABLES)

#### Argument
VARIABLES : collection(var−dvar)

#### Restriction
required(VARIABLES.var)

Succeeds if the minima of the values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘(\(<\mid=\)\)*\(<\mid\)’. Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

#### Purpose

#### Example

((4, 1, 2, 4, 3, 3, 2, 3, 3, 4, 4, 6, 6, 5, 7))

Figure 4.369 provides an example where the INCREASING_MIN_INCREASING_SEQUENCE ([(4, 1, 2, 4, 3, 3, 2, 3, 3, 4, 4, 6, 6, 5, 7)]) constraint holds.

#### Typical

| VARIABLES | > 1
range(VARIABLES.var) | > 1
Figure 4.369: Illustrating the INCREASING_MIN_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.370 depicts the automaton associated with the constraint `INCREASING_MIN_INCREASING_SEQUENCE`.

\[ \begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow +\infty \\
F &\leftarrow -\infty \\
R &\leftarrow 1
\end{align*} \]

\[ \begin{align*}
\geq s &
\begin{cases}
D \leftarrow +\infty \\
F \leftarrow C \\
R \leftarrow R \land (F \leq C)
\end{cases}
\end{align*} \]

\[ \begin{align*}
\leq t &
\begin{cases}
D \leftarrow \min(D, \text{VAR}_i+1) \\
C \leftarrow \min(C, \min(D, \text{VAR}_{i+1}))
\end{cases}
\end{align*} \]

Figure 4.370: Automaton for the `INCREASING_MIN_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `INCREASING_SEQUENCE` pattern.
Origin: Based on the INFLEXION pattern.

Constraint: `INCREASING_MIN_INFLEXION(VARIABLES)`

Argument: `VARIABLES : collection(var−dvar)`

Restriction: `required(VARIABLES, var)`

Purpose: Succeeds if the minima of the values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are increasing. An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression `'( < | = )* > | > ( > | = )* <'`. Assume that the occurrence of the pattern INFLEXION starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

Example: $((1,1,2,4,4,5,3,3,3,4,5,6))$

Figure 4.37 provides an example where the `INCREASING_MIN_INFLEXION (⟨1,1,2,4,4,5,3,3,3,4,5,6⟩)` constraint holds.

Typical: $|VARIABLES| > 2$

$\text{range}(\text{VARIABLES.var}) > 1$
Figure 4.37: Illustrating the INCREASING_MIN_INFLEXION constraint of the Example slot
Figure 4.372 depicts the automaton associated with the constraint INCREASING_MIN_INFLEXION.

Automaton

\[
\begin{align*}
&C \leftarrow -\infty \\
&D \leftarrow +\infty \\
&F \leftarrow -\infty \\
&R \leftarrow 1
\end{align*}
\]

\[C \leftarrow \min(D, \text{VAR}_i)\]
\[D \leftarrow +\infty\]
\[F \leftarrow \min(D, \text{VAR}_i)\]
\[R \leftarrow R \land (F \leq \min(D, \text{VAR}_i))\]

\[\leq\] \[\geq\]

\[\{D \leftarrow \min(D, \text{VAR}_i)\}\]

\[\leq\]
\[\geq\]

\[\{D \leftarrow \min(D, \text{VAR}_i)\}\]

\[\{D \leftarrow \min(D, \text{VAR}_i)\}\]
INCREASING_MIN STRICTLY DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_MIN STRICTLY DECREASING_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Description Automaton**

**Origin**
Based on the STRICTLY_DECREASING_SEQUENCE pattern.

**Constraint**
INCREASING_MIN STRICTLY DECREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

Succeeds if the minima of the values in each occurrence of the STRICTLY_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression ‘>+’. Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

**Example**

\[ ((1,3,2,2,4,4,5,5,6,3,5,5,4,6,7,4)) \]

Figure 4.373 provides an example where the INCREASING_MIN STRICTLY DECREASING_SEQUENCE \([1,3,2,4,4,5,5,6,3,5,5,4,6,7,4])\) constraint holds.

**Typical**
<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>range(VARIABLES.var)</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt; 1</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>
Figure 4.373: Illustrating the INCREASING_MIN STRICTLY DECREASING_SEQUENCE constraint of the Example slot
Figure 4.374 depicts the automaton associated with the constraint `INCREASING_MIN_STRICTLY_DECREASING_SEQUENCE`.

\[
\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow +\infty \\
F &\leftarrow -\infty \\
R &\leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
\{ D &\leftarrow +\infty \\
F &\leftarrow C \\
R &\leftarrow R \land (F \leq C) \} \\
\{ C &\leftarrow \min(D, \text{VAR}_i), \text{VAR}_{i+1} \} \\
\{ D &\leftarrow +\infty \}
\end{align*}
\]

\[
\begin{align*}
\{ C &\leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \} \\
\{ D &\leftarrow +\infty \}
\end{align*}
\]

Figure 4.374: Automaton for the `INCREASING_MIN_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern.
INCREASING_MIN_STRICTLY_DECREASING_SEQUENCE
INCREASING_MIN.Strictly_INCREASING_SEQUENCE

- **Origin**: Based on the STRICTLY_INCREASING_SEQUENCE pattern.
- **Constraint**: INCREASING_MIN.Strictly_INCREASING_SEQUENCE(VARIABLES)
- **Argument**: VARIABLES : collection(var−dvar)
- **Restriction**: required(VARIABLES.var)
- **Purpose**: Succeeds if the minima of the values in each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression ‘<+’.
Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

```plaintext
((4, 1, 2, 4, 3, 3, 2, 3, 3, 4, 5, 6, 6, 5, 7))
```

Figure 4.375 provides an example where the INCREASING_MIN.Strictly_INCREASING_SEQUENCE constraint holds.

**Typical**

- $|\text{VARIABLES}| > 1$
- $\text{range}(\text{VARIABLES.var}) > 1$
Figure 4.375: Illustrating the \textsc{increasing} \textsc{min} \textsc{strictly increasing} \textsc{sequence} constraint of the \textsc{Example} slot
Figure 4.376 depicts the automaton associated with the constraint INCREASING_MIN_STRICTLY_INCREASING_SEQUENCE. 

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow +\infty \\
F & \leftarrow -\infty \\
R & \leftarrow 1 \\
\{ & D \leftarrow +\infty \\
& F \leftarrow C \\
& R \leftarrow R \land (F \leq C) \\
\}
\end{align*}
\]

Figure 4.376: Automaton for the INCREASING_MIN_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern.
### INCREASING_MIN_Valley

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_MIN_Valley</td>
<td>↑</td>
<td>↑</td>
</tr>
</tbody>
</table>

#### Description

**Origin**
Based on the VALLEY pattern.

**Constraint**

\[
\text{INCREASING}_\text{MIN}_\text{VALLEY}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the minima of the values in each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \( > (= | >)^* (< | =)^* < \).

Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**

\[
((7, 2, 6, 3, 4, 5, 6, 4, 4, 6, 7, 3, 3, 1))
\]

Figure 4.377 provides an example where the INCREASING_MIN_VALLEY \((7, 2, 6, 3, 4, 5, 6, 4, 4, 6, 7, 3, 3, 1))\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.377: Illustrating the INCREASING_MIN_VALLEY constraint of the Example slot
Automaton Figure 4.378 depicts the automaton associated with the constraint INCREASING_MIN_VALLEY.

\[
\begin{align*}
\{ C \leftarrow -\infty \} & \quad \leq s \quad \leq \\
\{ D \leftarrow +\infty \} \quad \{ F \leftarrow -\infty \} \quad \{ R \leftarrow 1 \} & \quad > \quad R \land (F \leq C) \\
\{ D \leftarrow \min(D, \text{VAR}_i) \} & \quad \geq r \quad \leq t \\
\{ C \leftarrow \min(D, \text{VAR}_i) \} \quad \{ D \leftarrow +\infty \} & \quad \geq \quad \{ D \leftarrow \min(D, \text{VAR}_i) \} \\
\{ D \leftarrow +\infty \} \quad \{ F \leftarrow C \} \quad \{ R \leftarrow R \land (F \leq C) \} & \quad = \quad \{ D \leftarrow \min(D, \text{VAR}_i) \}
\end{align*}
\]

Figure 4.378: Automaton for the INCREASING_MIN_VALLEY constraint obtained by applying decoration Table 3.38 to the seed transducer of the VALLEY pattern.
INCREASING_MIN_ZIGZAG

**Origin**
Based on the ZIGZAG pattern.

**Constraint**
INCREASING_MIN_ZIGZAG(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the minima of the values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘(<>)*(<> | <>) | (><)>(>|><)’.

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**
\((7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6)\)

Figure 4.379 provides an example where the INCREASING_MIN_ZIGZAG ((7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6)) constraint holds.

**Typical**
\[|\text{VARIABLES}| > 3\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]
Figure 4.379: Illustrating the INCREASING_MIN_ZIGZAG constraint of the Example slot
Automaton Figure 4.380 depicts the automaton associated with the constraint INCREASING_MIN_ZIGZAG.
Figure 4.380: Automaton for the \textsc{increasing\_min\_zigzag} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \textsc{zigzag} pattern; (1) missing transitions from \(a, b, c, d, e, f\) to \(s\) are labelled by \(=\); (2) on transitions from \(b, c, e, f\) to \(s\) the register \(D\) is reset to its initial value; (3) on transitions from \(c, f\) to \(s\) the register \(F\) is reset to \(C\), and the register \(R\) is updated w.r.t. \(C\) and \(F\)
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING</td>
<td>RANGE</td>
<td>DECREASING</td>
</tr>
</tbody>
</table>

### Description

**Origin**: Based on the DECREASING pattern.

**Constraint**: \( \text{INCREASING\_RANGE\_DECREASING}(\text{VARIABLES}) \)

**Argument**: \( \text{VARIABLES} : \text{collection}(\text{var}-\text{dvar}) \)

**Restriction**: \( \text{required}(\text{VARIABLES}.\text{var}) \)

**Purpose**: Succeeds if the differences between the largest and smallest value in each occurrence of the DECREASING pattern in the time-series given by the \( \text{VARIABLES} \) collection are increasing.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression ‘\( > \)’.

Assume that the occurrence of the pattern DECREASING starts at position \( i \) and ends at position \( j \). The feature RANGE computes the range of the values from index \( i \) to index \( j+1 \).

**Example**: \( \langle 3, 6, 6, 5, 4, 4, 6, 4, 2, 2, 0, 5, 2, 6, 3, 3 \rangle \)

Figure 4.381 provides an example where the \( \text{INCREASING\_RANGE\_DECREASING} \) \( \langle [3, 6, 6, 5, 4, 4, 6, 4, 2, 2, 0, 5, 2, 6, 3, 3] \rangle \) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 4.381: Illustrating the INCREASING_RANGE_DECREASING constraint of the Example slot
Automaton

Figure 4.382 depicts the automaton associated with the constraint INCREASING_RANGE_DECREASING.

\[
\begin{align*}
C &\leftarrow -\infty \\
F &\leftarrow -\infty \\
H &\leftarrow \text{VAR} \\
R &\leftarrow 1 \\
\end{align*}
\]

\[
R \land (F \leq C)
\]

\[
\begin{align*}
C &\leftarrow |H - \text{VAR}_{i+1}| \\
F &\leftarrow |H - \text{VAR}_{i+1}| \\
H &\leftarrow \text{VAR}_{i+1} \\
R &\leftarrow R \land (F \leq |H - \text{VAR}_{i+1}|) \\
\end{align*}
\]

\[
H \leftarrow \text{VAR}_{i+1}
\]

\[
R \land (F \leq C)
\]

Figure 4.382: Automaton for the INCREASING_RANGE_DECREASING constraint obtained by applying decoration Table 3.49 to the seed transducer of the DECREASING pattern.
**INCREASING_RANGE_DECREEASING_SEQUENCE**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING RANGE</td>
<td>▶</td>
<td>DECREASING SEQUENCE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ ▽ ◁ ◤</td>
<td>▶ ▽ ◁ ◤</td>
</tr>
</tbody>
</table>

**Origin**
Based on the **DECREASING_SEQUENCE** pattern.

**Constraint**
**INCREASING_RANGE_DECREEASING_SEQUENCE**(VARIABLES)

**Argument**
VARIABLES : collection(var − dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the differences between the largest and smallest value in each occurrence of the **DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern **DECREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression ‘\( (> | =)* > | \)’. Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **RANGE** computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**

\( ((4, 3, 2, 2, 0, 0, 3, 4, 6, 2, 4, 4, 6, 5, 1)) \)

Figure 4.383 provides an example where the **INCREASING_RANGE_DECREEASING_SEQUENCE** \( ([4, 3, 2, 2, 0, 0, 3, 4, 6, 2, 4, 4, 6, 5, 1]) \) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}, \text{var}) > 1 \]
Figure 4.383: Illustrating the INCREASING_RANGE_DECREASING_SEQUENCE constraint of the Example slot
**Automaton**  
Figure 4.384 depicts the automaton associated with the constraint `INCREASING_RANGE_DECREASING_SEQUENCE`.

\[
\begin{align*}
&\{ \begin{array}{l}
C \leftarrow -\infty \\
F \leftarrow -\infty \\
H \leftarrow \text{VAR}_1 \\
R \leftarrow 1
\end{array} \} \\
\leq & \leq t \quad \{ H \leftarrow \text{VAR}_{i+1} \} \\
\{ F \leftarrow |H - \text{VAR}_i| \} & \{ H \leftarrow \text{VAR}_{i+1} \} \\
\{ R \leftarrow R \land (F \leq C) \} & \{ C \leftarrow |H - \text{VAR}_{i+1}| \} \\
= & \{ C \leftarrow |H - \text{VAR}_{i+1}| \} \\
\geq & \geq s \quad \{ H \leftarrow \text{VAR}_{i+1} \}
\end{align*}
\]

Figure 4.384: Automaton for the `INCREASING_RANGE_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.49 to the seed transducer of the `DECREASING_SEQUENCE` pattern.
**CONDITION**

**DESCRIPTION**

**AUTOMATON**

**INCREASING_RANGE_INCREASING**

**Origin**

Based on the **INCREASING** pattern.

**Constraint**

**INCREASING_RANGE_INCREASING**(VARIABLES)

**Argument**

VARIABLES : collection(var–dvar)

**Restriction**

required(VARIABLES.var)

**Purpose**

Succeeds if the differences between the largest and smallest value in each occurrence of the **INCREASING** pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression ‘<’.

Assume that the occurrence of the pattern **INCREASING** starts at position $i$ and ends at position $j$. The feature **RANGE** computes the range of the values from index $i$ to index $j + 1$.

**Example**

$((4, 5, 6, 3, 3, 4, 6, 2, 2, 1, 4, 7, 0, 5, 5))$

Figure 4.385 provides an example where the **INCREASING_RANGE_INCREASING** ($[4, 5, 6, 3, 3, 4, 6, 2, 2, 1, 4, 7, 0, 5, 5]$) constraint holds.

**Typical**

$|\text{VARIABLES}| > 1$

$\text{range} \{\text{VARIABLES.var}\} > 1$
Figure 4.385: Illustrating the INCREASING_RANGE_INCREASING constraint of the Example slot
Automaton

Figure 4.386 depicts the automaton associated with the constraint \texttt{INCREASING\_RANGE\_INCREASING}.

\[
\begin{aligned}
C &\leftarrow -\infty \\
F &\leftarrow -\infty \\
H &\leftarrow \text{VAR}_1 \\
R &\leftarrow 1 \\
\end{aligned}
\]

\[
\begin{aligned}
C &\leftarrow |H - \text{VAR}_{i+1}| \\
F &\leftarrow |H - \text{VAR}_{i+1}| \\
H &\leftarrow \text{VAR}_{i+1} \\
R &\leftarrow R \land (F \leq |H - \text{VAR}_{i+1}|) \\
\end{aligned}
\]

\[
\begin{aligned}
\{ &H \leftarrow \text{VAR}_{i+1} \} \\
\{ &R \land (F \leq C) \}
\end{aligned}
\]

Figure 4.386: Automaton for the \texttt{INCREASING\_RANGE\_INCREASING} constraint obtained by applying decoration Table 3.49 to the seed transducer of the \texttt{INCREASING} pattern
INCREASING RANGE INCREASING
### Description

**Pattern:**

\[
\text{\textasciitilde} < (< | =)^* < | < \text\textasciitilde
\]

**Condition:**

INCREASING\_RANGE\_INCREASING\_SEQUENCE

**Feature:**

RANGE

**Automaton**

- Based on the INCREASING\_SEQUENCE pattern.
- \[\text{\textasciitilde} < (< | =)^* < | < \text\textasciitilde\]

**Origin**

**Constraint**

INCREASING\_RANGE\_INCREASING\_SEQUENCE(VARIABLES)

**Argument**

VARIABLES : collection(var\_dvar)

**Restriction**

required(VARIABLES.var)

**Purpose**

Succeeds if the differences between the largest and smallest value in each occurrence of the INCREASING\_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INCREASING\_SEQUENCE is the maximal subsequence which matches the regular expression ‘\(< (< | =)^* < | < \)’.

Assume that the occurrence of the pattern INCREASING\_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature RANGE computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\[
\langle 4, 1, 2, 4, 3, 3, 2, 3, 3, 4, 4, 6, 3, 7 \rangle
\]

Figure 4.387 provides an example where the INCREASING\_RANGE\_INCREASING\_SEQUENCE constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]

\[\text{range(VARIABLES.var)} > 1\]
Figure 4.387: Illustrating the \texttt{INCREASING\_RANGE\_INCREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Automaton  Figure 4.388 depicts the automaton associated with the constraint INCREASING_RANGE_INCREASING_SEQUENCE.

\[
\begin{align*}
\{ C \leftarrow -\infty \} \\
\{ F \leftarrow -\infty \} \\
\{ H \leftarrow \text{VAR}_1 \} \\
\{ R \leftarrow 1 \}
\end{align*}
\]

\[
\begin{align*}
\geq s & \quad \{ H \leftarrow \text{VAR}_{i+1} \} \\
\{ F \leftarrow |H - \text{VAR}_i| \} \\
\{ H \leftarrow \text{VAR}_{i+1} \} \\
\{ R \leftarrow R \land (F \leq C) \}
\end{align*}
\]

\[
\leq t & \quad \{ C \leftarrow |H - \text{VAR}_{i+1}| \}
\]

Figure 4.388: Automaton for the INCREASING_RANGE_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.49 to the seed transducer of the INCREASING_SEQUENCE pattern.
**INCREASING_RANGE STRICTLY_DECREASING_SEQUENCE**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_RANGE STRICTLY_DECREASING_SEQUENCE</td>
<td>&gt;</td>
<td>+</td>
</tr>
</tbody>
</table>

**Description**

- **Origin**: Based on the STRICTLY_DECREASING_SEQUENCE pattern.
- **Constraint**: INCREASING_RANGE STRICTLY_DECREASING_SEQUENCE(VARIABLES)
- **Argument**: VARIABLES : collection(var−dvar)
- **Restriction**: required(VARIABLES.var)

**Purpose**

Succeeds if the differences between the largest and smallest value in each occurrence of the STRICTLY_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘>’.

Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature RANGE computes the range of the values from index $i$ to index $j + 1$.

**Example**

$$\langle 1, 3, 2, 2, 5, 6, 5, 3, 3, 0, 1, 3, 5, 1, 1, 2 \rangle$$

Figure 4.389 provides an example where the INCREASING_RANGE STRICTLY_DECREASING_SEQUENCE ($\langle 1, 3, 2, 2, 5, 6, 5, 3, 3, 0, 1, 3, 5, 1, 1, 2 \rangle$) constraint holds.

**Typical**

- $|\text{VARIABLES}| > 1$
- $\text{range}(\text{VARIABLES.var}) > 1$
Figure 4.389: Illustrating the INCREASING_RANGE.Strictly DECREASING_SEQUENCE constraint of the Example slot.
Figure 4.390 depicts the automaton associated with the constraint `INCREASING_RANGE_STRICTLY_DECREASING_SEQUENCE`.
INCREASING_RANGE STRICTLY DECREASING_SEQUENCE
### INCREASING_RANGE_STRICLY_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_RANGE_STRICLY_INCREASING_SEQUENCE</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

#### Origin
Based on the **STRICTLY_INCREASING_SEQUENCE** pattern.

#### Constraint
**INCREASING_RANGE_STRICLY_INCREASING_SEQUENCE**(VARIABLES)

#### Argument
VARIABLES : collection(var−dvar)

#### Restriction
```
required(VARIABLES.var)
```

Succeeds if the differences between the largest and smallest value in each occurrence of the **STRICTLY_INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are increasing.

#### Purpose
An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression ‘<+’.

Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position $i$ and ends at position $j$. The feature RANGE computes the range of the values from index $i$ to index $j + 1$.

#### Example
```
((4, 1, 2, 3, 0, 1, 4, 1, 1, 2, 3, 4, 4, 8))
```

Figure 4.391 provides an example where the **INCREASING_RANGE_STRICLY_INCREASING_SEQUENCE** ([(4, 1, 2, 3, 0, 1, 4, 1, 1, 2, 3, 4, 4, 8)]) constraint holds.

#### Typical
```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 4.391: Illustrating the `INCREASING_RANGE_STRICTLY_INCREASING_SEQUENCE` constraint of the `Example` slot
Automaton

Figure 4.392 depicts the automaton associated with the constraint \textsc{increasing\_range\_strictly\_increasing\_sequence}.

\begin{align*}
&\{ C \leftarrow -\infty \\ &F \leftarrow -\infty \\ &H \leftarrow \text{VAR}_1 \\ &R \leftarrow 1 \}
\end{align*}

\[
\begin{array}{c}
\geq s \\
\{ H \leftarrow \text{VAR}_{i+1} \}
\end{array}
\]

\[
\begin{array}{c}
F \leftarrow |H - \text{VAR}_i| \\
H \leftarrow \text{VAR}_{i+1} \\
R \leftarrow R \land (F \leq C)
\end{array}
\]

\[
\begin{array}{c}
\leq r \\
\{ C \leftarrow |H - \text{VAR}_{i+1}| \}
\end{array}
\]

Figure 4.392: Automaton for the \textsc{increasing\_range\_strictly\_increasing\_sequence} constraint obtained by applying decoration Table 3.49 to the seed transducer of the \textsc{strictly\_increasing\_sequence} pattern.
INCREASING_RANGE_STRICTLY_INCREASING_SEQUENCE
### CONDITION |
| FEATURE |
| PATTERN |

| **INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE** |
| **DESCRIPTION** |
| **AUTOMATON** |

| **Origin** | Based on the **BUMP_ON_DECREASING_SEQUENCE** pattern. |
| **Constraint** | **INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)** |
| **Argument** | VARIABLES : collection(var—dvar) |
| **Restriction** | required(VARIABLES, var) |

Succeeds if the values denoting the surface of each occurrence of the **BUMP_ON_DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are increasing.

| **Purpose** | An occurrence of the pattern **BUMP_ON_DECREASING_SEQUENCE** is the subsequence ‘>>><’.
Assume that the occurrence of the pattern **BUMP_ON_DECREASING_SEQUENCE** starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 2 to index j. |

| **Example** | ```
((7, 6, 4, 2, 5, 4, 3, 3, 5, 7, 6, 5, 6, 5, 4, 1))
``` |

Figure 4.393 provides an example where the **INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE** ((7, 6, 4, 2, 5, 4, 3, 3, 5, 7, 6, 5, 6, 5, 4, 1)) constraint holds.

| **Typical** | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
Figure 4.393: Illustrating the INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.394 depicts the automaton associated with the constraint \textsc{increasing\textunderscore surf\textunderscore bump\textunderscore on\textunderscore decreasing\textunderscore sequence}.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{automaton_diagram.png}
\caption{Automaton for the \texttt{increasing\textunderscore surf\textunderscore bump\textunderscore on\textunderscore decreasing\textunderscore sequence} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{bump\_on\_decreasing\_sequence} pattern}
\end{figure}

\begin{align*}
\text{\{ } & C \leftarrow -\infty, \\
\text{\{ } & D \leftarrow 0, \\
\text{\{ } & F \leftarrow -\infty, \\
\text{\{ } & R \leftarrow 1
\end{align*}
INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE
INCREASING_SURF_DECREASING

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING</td>
<td>SURF</td>
<td>DECREASING</td>
</tr>
</tbody>
</table>

**DESCRIPTION AUTOMATON**

- **Origin**: Based on the DECREASING pattern.
- **Constraint**: INCREASING_SURF_DECREASING(VARIABLES)
- **Argument**: VARIABLES : collection(var−dvar)
- **Restriction**: required(VARIABLES.var)
- **Purpose**: Succeeds if the values denoting the surface of each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection are increasing.
  An occurrence of the pattern DECREASING is the subsequence which matches the regular expression '>'.
  Assume that the occurrence of the pattern DECREASING starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.
- **Example**: $((3, 3, 2, 3, 4, 4, 4, 6, 3, 4, 4, 5, 6, 4, 6))$

Figure 4.395 provides an example where the INCREASING_SURF_DECREASING $([3, 3, 2, 3, 4, 4, 6, 3, 4, 4, 5, 6, 4, 6])$ constraint holds.

- **Typical**: $|\text{VARIABLES}| > 1$
  $\text{range}(\text{VARIABLES}.\text{var}) > 1$
Figure 4.395: Illustrating the INCREASING_SURF_DECREASING constraint of the Example slot
Automaton

Figure 4.396 depicts the automaton associated with the constraint INCREASING_SURF_DECREASING.

\[
\begin{align*}
\{ & C \leftarrow -\infty \\
 & D \leftarrow 0 \\
 & F \leftarrow -\infty \\
 & R \leftarrow 1 \\
\}\quad (F \leq C)
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
 & D \leftarrow 0 \\
 & F \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
 & R \leftarrow R \land (F \leq D + \text{VAR}_i + \text{VAR}_{i+1}) \\
\}\quad (R \land (F \leq C))
\end{align*}
\]

Figure 4.396: Automaton for the INCREASING_SURF_DECREASING constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING pattern.
**INCREASING_SURF_DECREASING_SEQUENCE**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING</td>
<td>SURF</td>
<td>DECREASING</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**

`INCREASING_SURF_DECREASING_SEQUENCE(VARIABLES)`

**Argument**

`VARIABLES : collection(var−dvar)`

**Restriction**

`required(VARIABLES.var)`

Succeeds if the values denoting the surface of each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘>` (> | =)* > | ’. Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

**Example**

`[(4,3,2,3,4,4,6,3,4,5,6,4,6)]`

Figure 4.397 provides an example where the INCREASING_SURF_DECREASING_SEQUENCE constraint holds.

**Typical**

`|VARIABLES| > 1`

`range(VARIABLES.var) > 1`
Figure 4.397: Illustrating the INCREASING_SURF_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.398 depicts the automaton associated with the constraint INCREASING_SURF_DECREASING_SEQUENCE.

\[
\begin{align*}
&C \leftarrow -\infty \\
&D \leftarrow 0 \\
&W \leftarrow -\infty \\
&R \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
\leq s & \Rightarrow \\
&D \leftarrow 0 \\
&W & \leftarrow C \\
&R & \leftarrow R \land (F \leq C)
\end{align*}
\]

\[
\begin{align*}
\geq t & \Rightarrow \\
&C & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
&D & \leftarrow 0
\end{align*}
\]

\[
\begin{align*}
&D & \leftarrow D + \text{VAR}_{i+1}
\end{align*}
\]

Figure 4.398: Automaton for the INCREASING_SURF_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING_SEQUENCE pattern
INCREASING_SURF_DECREASING_SEQUENCE
### Condition
**INCREASING_SURF_DECREEASING_TERRACE**

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <strong>DECREASING_TERRACE</strong> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><strong>INCREASING_SURF_DECREEASING_TERRACE</strong>(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Requirement</strong></td>
<td>required(VARIABLES.var)</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the surface of each occurrence of the **DECREASING_TERRACE** pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern **DECREASING_TERRACE** is the *maximal* subsequence which matches the regular expression ‘>=+>’.

Assume that the occurrence of the pattern **DECREASING_TERRACE** starts at position $i$ and ends at position $j$. The feature **SURF** computes the sum of the values from index $i+1$ to index $j$.

**Example**

\[(5, 2, 2, 1, 5, 4, 3, 3, 2, 4, 4, 6, 5, 5, 4)\]

Figure 4.399 provides an example where the **INCREASING_SURF_DECREEASING_TERRACE** \((5, 2, 2, 1, 5, 4, 3, 3, 2, 4, 4, 6, 5, 5, 4)\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 3\]

\[\text{range}(\text{VARIABLES.var}) > 2\]
Figure 4.399: Illustrating the INCREASING_SURF_DECREASING_TERRACE constraint of the Example slot
Automaton  Figure 4.400 depicts the automaton associated with the constraint INCREASING_SURF_DECREASING_TERRACE.

Figure 4.400: Automaton for the INCREASING_SURF_DECREASING_TERRACE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING_TERRACE pattern.
**CONDITION** | **FEATURE** | **PATTERN**
---|---|---
**INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE**

**Origin**
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**
INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the values denoting the surface of each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression ‘<<><<<’.

Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 2$ to index $j$.

**Example**

```
((1, 2, 3, 0, 3, 4, 6, 5, 1, 4, 5, 6, 1, 2, 4, 4))
```

Figure 4.401 provides an example where the INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE $((1, 2, 3, 0, 3, 4, 6, 5, 1, 4, 5, 6, 1, 2, 4, 4))$ constraint holds.

**Typical**

$|\text{VARIABLES}| > 5$

$\text{range(VARIABLES.var)} > 2$
Figure 4.401: Illustrating the increasing surf dip on increasing sequence constraint of the Example slot
Automaton

Figure 4.402 depicts the automaton associated with the constraint INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow 0 \\
F & \leftarrow -\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
&\{ D \leftarrow 0 \} \\
&\{ D > 0 \} \\
&\{ D < 0 \}
\end{align*}
\]

\[
\begin{align*}
&\{ D \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow D + \text{VAR}_i \} \\
&\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

Figure 4.402: Automaton for the INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern.
INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE

977
INCREASING_SURF_GORGE

Origin: Based on the GORGE pattern.

Constraint: INCREASING_SURF_GORGE(VARIABLES)

Argument: VARIABLES : collection(var−dvar)

Restriction: required(VARIABLES, var)

Purpose: Succeeds if the values denoting the surface of each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression 

\((> \mid > (|= >) > ) \langle < \mid < (|= <) < \rangle\).

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example: \(\langle 6, 2, 5, 1, 1, 5, 4, 2, 3, 6, 5, 4, 4, 3, 5, 5 \rangle\)

Figure 4.403 provides an example where the INCREASING_SURF_GORGE ((6, 2, 5, 1, 1, 5, 4, 2, 3, 6, 5, 4, 4, 3, 5, 5)) constraint holds.

Typical: |VARIABLES| > 2
range(VARIABLES.var) > 1
Figure 4.403: Illustrating the INCREASING_SURF_GORGE constraint of the Example slot
**Automaton**

Figure 4.404 depicts the automaton associated with the constraint `INCREASING_SURF_GORGE`.

![Automaton Diagram]

Figure 4.404: Automaton for the `INCREASING_SURF_GORGE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `GORGE` pattern (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
INCREASING_SURF_INCREASING

**Description**

**Origin**
Based on the INCREASING pattern.

**Constraint**

INCREASING_SURF_INCREASING(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES.var)

**Purpose**

Succeeds if the values denoting the surface of each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression '<'.

Assume that the occurrence of the pattern INCREASING starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

**Example**

$$((6, 1, 2, 3, 5, 5, 4, 5, 6, 4, 3, 2, 1, 1))$$

Figure 4.405 provides an example where the INCREASING_SURF_INCREASING ([6, 1, 2, 3, 5, 5, 4, 5, 6, 4, 3, 2, 1, 1]) constraint holds.

**Typical**

$|VARIABLES| > 1$

$\text{range}(\text{VARIABLES}\text{.var}) > 1$
Figure 4.405: Illustrating the INCREASING_SURF_INCREASING constraint of the Example slot
Automaton Figure 4.406 depicts the automaton associated with the constraint INCREASING_SURF_INCREASING.

Figure 4.406: Automaton for the INCREASING_SURF_INCREASING constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING pattern.
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_SURF_INCREASING_SEQUENCE</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the `INCREASING_SEQUENCE` pattern.

**Constraint**
`INCREASING_SURF_INCREASING_SEQUENCE(VARIABLES)`

**Argument**
`VARIABLES : collection(var−dvar)`

**Restriction**
`required(VARIABLES.var)`

Succeeds if the values denoting the surface of each occurrence of the `INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing.

**Purpose**
An occurrence of the pattern `INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `'< (< | =)* '< |'`. Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j+1`.

**Example**

```
((6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 3, 3, 4, 5, 5))
```

Figure 4.407 provides an example where the `INCREASING_SURF_INCREASING_SEQUENCE` constraint holds.

**Typical**
```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
INCREASING_SURF_INCREASING_SEQUENCE

Figure 4.407: Illustrating the INCREASING_SURF_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.408 depicts the automaton associated with the constraint `INCREASING_SURF_INCREASING_SEQUENCE`.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow 0 \\
F & \leftarrow -\infty \\
R & \leftarrow 1 \\
\end{align*}
\]

Figure 4.408: Automaton for the `INCREASING_SURF_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `INCREASING_SEQUENCE` pattern.
INCREASING_SURF_INCREASING_TERRACE

**Description**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_SURF_INCREASING_TERRACE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Origin**

Based on the INCREASING_TERRACE pattern.

**Constraint**

INCREASING_SURF_INCREASING_TERRACE(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES.var)

Succeeds if the values denoting the surface of each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression '<\=+<'.

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+1\) to index \(j\).

**Example**

\(((1, 2, 2, 4, 5, 5, 6, 6, 4, 4, 2, 5, 5, 5, 7, 7))\)

Figure 4.409 provides an example where the INCREASING_SURF_INCREASING_TERRACE (\([1, 2, 2, 4, 5, 5, 6, 6, 4, 4, 2, 5, 5, 5, 7, 7]\)) constraint holds.

**Typical**

\[|VARIABLES| > 3\]

\[\text{range}(\text{VARIABLES}.\text{var}) > 2\]
Figure 4.409: Illustrating the INCREASING_SURF_INCREASING_TERRACE constraint of the Example slot
Figure 4.410 depicts the automaton associated with the constraint INCREASING_SURF_INCREASING_TERRACE.

Automaton

Figure 4.410: Automaton for the INCREASING_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING_TERRACE pattern
<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_SURF_INFLEXION</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
<tr>
<td>▶ ▶ &lt; ◀</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Origin**
Based on the INFLEXION pattern.

**Constraint**
INCREASING_SURF_INFLEXION(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression "< (< | =) > | > (>|=)" <".
Assume that the occurrence of the pattern INFLEXION starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Example**

```
((1,1,2,3,3,2,2,2,2,1,6,6,5,5,5))
```

Figure 4.4.11 provides an example where the INCREASING_SURF_INFLEXION ((1,1,2,3,3,2,2,2,2,1,6,6,5,5,5)) constraint holds.

**Typical**

```
|VARIABLES| > 2
range(VARIABLES.var) > 1
```
Figure 4.411: Illustrating the \textsc{increasing\_surf\_inflexion} constraint of the \textit{Example} slot
Automaton

Figure 4.412 depicts the automaton associated with the constraint INCREASING_SURF_INFLEXION.

\[
\begin{align*}
\{ C \leftarrow -\infty \} & \quad \{ D \leftarrow 0 \} & \quad \{ R \leftarrow 1 \} \\
\{ F \leftarrow -\infty \} & \quad \{ \text{transition } r \rightarrow t \} & \quad \text{has the same registers updates as transition } t \rightarrow r
\end{align*}
\]

Figure 4.412: Automaton for the INCREASING_SURF_INFLEXION constraint obtained by applying decoration Table 3.38 to the seed transducer of the INFLEXION pattern (transition \( r \rightarrow t \) has the same registers updates as transition \( t \rightarrow r \))
INCREASING_SURF_PEAK

DESCRIPTION AUTOMATON

Origin  Based on the PEAK pattern.

Constraint  INCREASING_SURF_PEAK(VARIABLES)

Argument  VARIABLES : collection(var−dvar)

Restriction  required(VARIABLES.var)

Purpose  Succeeds if the values denoting the surface of each occurrence of the PEAK pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression '<(=|<)*(>|=)*>'.

Assume that the occurrence of the pattern PEAK starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

Example  $((7,5,5,1,2,3,2,3,4,5,2,6,6,1))$

Figure 4.413 provides an example where the INCREASING_SURF_PEAK ($([7,5,5,1,2,3,2,3,4,5,2,6,6,1])$) constraint holds.

Typical  $|\text{VARIABLES}| > 2$

$\text{range}(\text{VARIABLES.var}) > 1$
Figure 4.413: Illustrating the INCREASING_SURF_PEAK constraint of the Example slot
Automaton | Figure 4.414 depicts the automaton associated with the constraint `INCREASING_SURF_PEAK`.

```
\begin{align*}
\{ C \leftarrow -\infty \\
D \leftarrow 0 \\
F \leftarrow -\infty \\
R \leftarrow 1 \} & \xrightarrow{\geq 8} \geq \\
< & \\
R \land (F \leq C) & \\
\{ C \leftarrow D + \text{VAR}_i \\
D \leftarrow 0 \} & \xrightarrow{>} > \\
\leq r \geq t & \\
\{ C \leftarrow C + D + \text{VAR}_i \\
D \leftarrow 0 \} & \xrightarrow{=} = \\
\{ D \leftarrow D + \text{VAR}_i \} & \\
\{ D \leftarrow 0 \\
F \leftarrow C \\
R \leftarrow R \land (F \leq C) \} & \xrightarrow{<} < \\
\end{align*}
```

Figure 4.414: Automaton for the `INCREASING_SURF_PEAK` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `PEAK` pattern.
### INCREASING_SURF.PLAIN

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>➤ ▶ ▷ ◄ ◌</td>
<td>DESCRIPTION</td>
<td>AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the `PLAIN` pattern.

**Constraint**

\[
\text{INCREASING\_SURF\_PLAIN}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} \rightarrow \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the values denoting the surface of each occurrence of the `PLAIN` pattern in the time-series given by the `VARIABLES` collection are increasing.

**Purpose**

An occurrence of the pattern `PLAIN` is the maximal subsequence which matches the regular expression `>`=`<`.

Assume that the occurrence of the pattern `PLAIN` starts at position \(i\) and ends at position \(j\). The feature `SURF` computes the sum of the values from index \(i+1\) to index \(j\).

**Example**

\[
((3, 6, 6, 3, 4, 5, 5, 4, 6, 6, 7, 5, 6, 3, 2))
\]

Figure 4.415 provides an example where the `INCREASING\_SURF\_PLAIN` constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(&\text{VARIABLES}.\text{var}) > 1
\end{align*}
\]
Figure 4.415: Illustrating the INCREASING_SURF.PLAIN constraint of the Example slot
Automaton

Figure 4.416 depicts the automaton associated with the constraint INCREASING_SURF_PLAIN.

\[
\begin{align*}
\text{Automaton} & \quad \text{Figure 4.416: Automaton for the INCREASING_SURFPLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLAIN pattern}
\end{align*}
\]
INCREASING_SURF_PLATEAU

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_SURF_PLATEAU</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Origin**
Based on the PLATEAU pattern.

**Constraint**
INCREASING_SURF_PLATEAU(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the values denoting the surface of each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are increasing. An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression ‘<="\>’.
Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**
\(((7,5,2,3,1,2,2,4,3,4,5,5,2,2,5))\)

Figure 4.417 provides an example where the INCREASING_SURF_PLATEAU ((7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5)) constraint holds.

**Typical**
| VARIABLES | > 2
| range(VARIABLES.var) | > 1
Figure 4.417: Illustrating the INCREASING_SURF_PLATEAU constraint of the Example slot
Figure 4.418 depicts the automaton associated with the constraint INCREASING_SURF_PLATEAU.

\[
\begin{align*}
\text{Automaton} & \quad \text{Figure 4.418: Automaton for the INCREASING_SURF_PLATEAU constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLATEAU pattern}
\end{align*}
\]
INCREASING_SURF_PROPER.PLAIN

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_SURF_PROPER/plain</td>
<td>▶</td>
<td>▶</td>
</tr>
</tbody>
</table>

**Description**

- **Origin:** Based on the PROPERPLAIN pattern.
- **Constraint:** INCREASING_SURF_PROPER.PLAIN(VARIABLES)
- **Argument:** VARIABLES : collection(var−dvar)
- **Restriction:** required(VARIABLES.var)

**Purpose**

Succeeds if the values denoting the surface of each occurrence of the PROPER.PLAIN pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PROPER.PLAIN is the maximal subsequence which matches the regular expression ‘≥+<’.

Assume that the occurrence of the pattern PROPER.PLAIN starts at position i and ends at position j. The feature SURF computes the sum of the values from index i+1 to index j.

**Example**

```
((5,3,3,5,6,5,4,4,7,3,6,5,5,5,7,2))
```

Figure 4.419 provides an example where the INCREASING_SURF_PROPER.PLAIN \(((5,3,3,5,6,5,4,4,7,3,6,5,5,5,7,2))\) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}.var) > 1\)
Figure 4.419: Illustrating the INCREASING_SURF_PROPERPLAIN constraint of the Example slot
Automaton

Figure 4.420 depicts the automaton associated with the constraint INCREASING_SURF_PROPER.PLAIN.

Figure 4.420: Automaton for the INCREASING_SURF_PROPER.PLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PROPER.PLAIN pattern.
INCREASING_SURF_PROPER_PLATEAU

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING Thái</td>
<td>SURF Thái</td>
<td>PROPER Thái PLATEAU</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

**Origin**
Based on the PROPER_PLATEAU pattern.

**Constraint**
INCREASING_SURF_PROPER_PLATEAU(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression '<=+>.'

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Purpose**

**Example**

$((7, 1, 3, 2, 5, 1, 4, 3, 2, 3, 5, 5, 3))$

Figure 4.421 provides an example where the INCREASING_SURF_PROPER_PLATEAU($(7, 1, 3, 2, 5, 1, 4, 3, 2, 3, 5, 5, 3))$ constraint holds.

**Typical**

$|\text{VARIABLES}| > 3$

$\text{range}(\text{VARIABLES.var}) > 1$
Figure 4.421: Illustrating the INCREASING_SURF_PROPER_PLATEAU constraint of the Example slot
Automaton Figure 4.422 depicts the automaton associated with the constraint INCREASING_SURF_PROPER_PLATEAU.

Figure 4.422: Automaton for the INCREASING_SURF_PROPER_PLATEAU constraint obtained by applying decoration Table 3.38 to the seed transducer of the PROPER_PLATEAU pattern.
### Increasing_Surf_Steady

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the Steady pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>Increasing_Surf_Steady(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES.var)</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the surface of each occurrence of the Steady pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern Steady is the subsequence which matches the regular expression ‘=’. Assume that the occurrence of the pattern Steady starts at position $i$ and ends at position $j$. The feature Surf computes the sum of the values from index $i$ to index $j + 1$.

**Example**

```plaintext
((1,1,7,3,3,5,5,6,5,5,7,2,6,6))
```

Figure 4.423 provides an example where the Increasing_Surf_Steady ([1,1,7,3,3,5,5,6,5,5,7,2,6,6]) constraint holds.

**Typical**

$|\text{VARIABLES}| > 1$
Figure 4.423: Illustrating the INCREASING_SURF_STEADY constraint of the Example slot
Automaton

Figure 4.424 depicts the automaton associated with the constraint INCREASING_SURF_STEADY.

Figure 4.424: Automaton for the INCREASING_SURF_STEADY constraint obtained by applying decoration Table 3.38 to the seed transducer of the STEADY pattern.
**INCREASING_SURF_STEADY_SEQUENCE**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>INCREASING_SURF_STEADY_SEQUENCE</strong></td>
<td><strong>DESCRIPTION</strong></td>
<td><strong>AUTOMATON</strong></td>
</tr>
</tbody>
</table>

**Origin**

Based on the `STEADY_SEQUENCE` pattern.

**Constraint**

`INCREASING_SURF_STEADY_SEQUENCE(VARIABLES)`

**Argument**

`VARIABLES : collection(var−dvar)`

**Restriction**

`required(VARIABLES.var)`

Succeeds if the values denoting the surface of each occurrence of the `STEADY_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing.

**Purpose**

An occurrence of the pattern `STEADY_SEQUENCE` is the *maximal* subsequence which matches the regular expression `'=+'`.

Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `SURF` computes the sum of the values from index $i$ to index $j+1$.

**Example**

```
(6, 2, 2, 3, 5, 6, 4, 4, 2, 3, 5, 5, 2, 3, 4)
```

Figure 4.425 provides an example where the `INCREASING_SURF_STEADY_SEQUENCE` constraint holds.

**Typical**

$|VARIABLES| > 1$
Figure 4.425: Illustrating the INCREASING_SURF_STEADY_SEQUENCE constraint of the Example slot
Automaton

Figure 4.426 depicts the automaton associated with the constraint INCREASING_SURF_STEADY_SEQUENCE.

\[
\begin{align*}
&\{ C \leftarrow -\infty \\
&\{ D \leftarrow 0 \\
&\{ F \leftarrow -\infty \\
&\{ R \leftarrow 1 \\

&= r R \land (F \leq C) \\
&\{ C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
&\{ D \leftarrow 0
\end{align*}
\]

Figure 4.426: Automaton for the INCREASING_SURF_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STEADY_SEQUENCE pattern.
### CONDITION

**INCREASING_SURF STRICTLY DECREASING_SEQUENCE**

<table>
<thead>
<tr>
<th>▶</th>
<th>▷</th>
<th>◀</th>
<th>◀</th>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
</table>

### Origin

Based on the **STRICTLY DECREASING_SEQUENCE** pattern.

### Constraint

**INCREASING_SURF STRICTLY DECREASING_SEQUENCE(VARIABLES)**

### Argument

**VARIABLES : collection(var−dvar)**

### Restriction

**required**(VARIABLES.var)

Succeeds if the values denoting the surface of each occurrence of the **STRICTLY DECREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection are increasing.

### Purpose

An occurrence of the pattern **STRICTLY DECREASING_SEQUENCE** is the maximal sub-sequence which matches the regular expression ‘>’.

Assume that the occurrence of the pattern **STRICTLY DECREASING_SEQUENCE** starts at position $i$ and ends at position $j$. The feature **SURF** computes the sum of the values from index $i$ to index $j + 1$.

### Example

$\langle 1, 4, 3, 2, 7, 3, 3, 7, 3, 3, 4, 4, 5, 7, 5 \rangle$

Figure 4.427 provides an example where the **INCREASING_SURF STRICTLY DECREASING_SEQUENCE** $(1, 4, 3, 2, 7, 3, 3, 7, 3, 3, 4, 4, 5, 7, 5)$ constraint holds.

### Typical

$|VARIABLES| > 1$

$\text{range}(VARIABLES.var) > 1$
Figure 4.427: Illustrating the INCREASING_SURF_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.428 depicts the automaton associated with the constraint INCREASING_SURF.Strictly_DECREASINGSEQUENCE.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow 0 \\
F & \leftarrow -\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
\{ & \text{if } D < 0 \\
F & \leftarrow C \\
R & \leftarrow R \land (F \leq C)
\}
\{ & \text{if } D \geq 0 \\
C & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
D & \leftarrow 0
\}
\]

\[
\begin{align*}
\{ & \text{if } C \leq s \leq F \leq C \\
\}
\{ & \text{if } C > s > F > C
\}
\]

\[
\begin{align*}
D & \leftarrow 0 \\
F & \leftarrow C \\
R & \leftarrow R \land (F \leq C)
\]

Figure 4.428: Automaton for the INCREASING_SURF.Strictly_DECREASINGSEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STRICTLY_DECREASINGSEQUENCE pattern.
INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE

DESCRIPTION

INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE

Origin
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

Constraint
INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES.var)

Succeeds if the values denoting the surface of each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

Purpose
An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression '<+'.

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

Example
((6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 2, 3, 4, 5, 5))

Figure 4.429 provides an example where the INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE ((6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 2, 3, 4, 5, 5)) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES.var) > 1
Figure 4.429: Illustrating the increasing strictly increasing sequence constraint of the Example slot
Figure 4.430 depicts the automaton associated with the constraint \texttt{INCREASING\_SURF\_STRICTLY\_INCREASING\_SEQUENCE}.

\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow 0 \\
F & \leftarrow -\infty \\
R & \leftarrow 1
\end{align*}

\begin{align*}
\{ 
D & \leftarrow 0 \\
F & \leftarrow C \\
R & \leftarrow R \land (F \leq C)
\}
\end{align*}

\begin{align*}
\begin{cases}
C & \leftarrow D + \text{VAR} + \text{VAR}_{i+1} \\
D & \leftarrow 0
\end{cases}
\end{align*}

\begin{align*}
\begin{cases}
C & \leftarrow C + D + \text{VAR}_{i+1} \\
D & \leftarrow 0
\end{cases}
\end{align*}

---

\textbf{Automaton}

Figure 4.430: Automaton for the \texttt{INCREASING\_SURF\_STRICTLY\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.38 to the seed transducer of the \texttt{STRICTLY\_INCREASING\_SEQUENCE} pattern.
INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE
INCREASING_SURF_SUMMIT

**Condition**

**Feature**

**Pattern**

**DESCRIPTION**

**AUTOMATON**

$(< | < (|=|) <)(| > (|=|) >)$

**Origin**

Based on the SUMMIT pattern.

**Constraint**

INCREASING_SURF_SUMMIT(VARIABLES)

**Argument**

VARIABLES : collection(var–dvar)

**Restriction**

required(VARIABLES.var)

Succeeds if the values denoting the surface of each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection are increasing. An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression ‘$(< | < (|=|) <)(| > (|=|) >)$’.

**Purpose**

Assume that the occurrence of the pattern SUMMIT starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Example**

$((1, 5, 2, 1, 6, 6, 2, 3, 5, 4, 1, 4, 6, 4, 3, 2))$

Figure 4.431 provides an example where the INCREASING_SURF_SUMMIT ($[(1, 5, 2, 1, 6, 6, 2, 3, 5, 4, 1, 4, 6, 4, 3, 2)]$) constraint holds.

**Typical**

$|\text{VARIABLES}| > 2$

range(VARIABLES.var) > 1
Figure 4.431: Illustrating the INCREASING_SURF_SUMMIT constraint of the Example slot
Automaton

Figure 4.432 depicts the automaton associated with the constraint INCREASING_SURF_SUMMIT.

![Automaton Diagram](image)

Figure 4.432: Automaton for the INCREASING_SURF_SUMMIT constraint obtained by applying decoration Table 3.38 to the seed transducer of the SUMMIT pattern (transition \( u \rightarrow r \) has the same register update as transition \( r \rightarrow u \)).
### INCREASING_SURF_VALLEY

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>INCREASING_SURF_VALLEY</strong></td>
<td>▶ ▶ ▶ ▶</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

#### Origin
Based on the VALLEY pattern.

#### Constraint
INCREASING_SURF_VALLEY(VARIABLES)

#### Argument
VARIABLES : collection(var−dvar)

#### Restriction
required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression '>(=|>)*(<|=)*<'. Assume that the occurrence of the pattern VALLEY starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i+1$ to index $j$.

#### Example

```
((7, 2, 2, 6, 3, 4, 5, 6, 4, 6, 4, 6, 7, 3, 3, 1))
```

Figure 4.433 provides an example where the INCREASING_SURF_VALLEY
((7, 2, 2, 6, 3, 4, 5, 6, 4, 6, 4, 6, 7, 3, 3, 1)) constraint holds.

#### Typical

- $|\text{VARIABLES}| > 2$
- $\text{range}(\text{VARIABLES, var}) > 1$
Figure 4.433: Illustrating the INCREASING_SURF_VALLEY constraint of the Example slot
Automaton Figure 4.434 depicts the automaton associated with the constraint INCREASING_SURF_VALLEY.

Figure 4.434: Automaton for the INCREASING_SURF_VALLEY constraint obtained by applying decoration Table 3.38 to the seed transducer of the VALLEY pattern.
INCREASING_SURF_ZIGZAG

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ ▶ ▶ ▶</td>
<td>DESCRIPTION AUTOMATON</td>
<td>▶ ▶ ▶ ▶</td>
</tr>
</tbody>
</table>

**Origin**
Based on the ZIGZAG pattern.

**Constraint**
INCREASING_SURF_ZIGZAG(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the values denoting the surface of each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern ZIGZAG is the *maximal* subsequence which matches the regular expression `'(<<)>+|<<>)(<<)>+|<>>'`. Assume that the occurrence of the pattern ZIGZAG starts at position `i` and ends at position `j`. The feature SURF computes the sum of the values from index `i + 1` to index `j`.

**Example**

Example: `((7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6))`

Figure 4.435 provides an example where the INCREASING_SURF_ZIGZAG (`[7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6]`) constraint holds.

**Typical**

- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 1`
Figure 4.435: Illustrating the INCREASING_SURF_ZIGZAG constraint of the Example slot.
Automaton  Figure 4.436 depicts the automaton associated with the constraint INCREASING_SURF_ZIGZAG.
Figure 4.436: Automaton for the INCREASING_SURF_ZIGZAG constraint obtained by applying decoration Table 3.38 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $\neq$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the register $F$ is reset to $C$, and the register $R$ is updated wrt $C$ and $F$. 

\[
\begin{align*}
\{ C \leftarrow \infty \} & \quad \{ D \leftarrow 0 \} \\
F & \leftarrow \infty \\
\{ R \leftarrow 1 \} \\
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + \text{VAR} \} & \quad \{ D \leftarrow 0 \} \\
\{ C \leftarrow C + D + \text{VAR} \} & \quad \{ D \leftarrow 0 \} \\
\{ C \leftarrow \infty \} & \quad \{ D \leftarrow 0 \} \\
\{ C \leftarrow \infty \} & \quad \{ D \leftarrow 0 \} \\
\{ C \leftarrow D + \text{VAR} \} & \quad \{ D \leftarrow 0 \} \\
\{ C \leftarrow C + D + \text{VAR} \} & \quad \{ D \leftarrow 0 \} \\
\end{align*}
\]
INCREASING_WIDTH_DECREASING_SEQUENCE

**Description**

Based on the **DECREASING_SEQUENCE** pattern.

**Constraint**

INCREASING_WIDTH_DECREASING_SEQUENCE(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES.var)

Succeeds if the values denoting the width of each occurrence of the **DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern **DECREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression ‘\(> (> | =)^* > | >\)’.

Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i + 2\).

**Example**

\(((1,6,6,4,5,6,6,5,2,3,3,2,2,1,1))\)

Figure 4.437 provides an example where the **INCREASING_WIDTH_DECREASING_SEQUENCE** constraint holds.

**Typical**

\(|\text{VARIABLES}| > 1\)

\(|\text{range(VARIABLES}.var)| > 1\)
Figure 4.437: Illustrating the INCREASING_WIDTH_DECREASING_SEQUENCE constraint of the Example slot
Figure 4.438: Automaton for the INCREASING_WIDTH_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING_SEQUENCE pattern.
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING WIDTH</td>
<td>DECREASING TERRACE</td>
<td>INCREASING WIDTH DECREASING TERRACE</td>
</tr>
</tbody>
</table>

**Description**

**Origin**

Based on the **DECREASING TERRACE** pattern.

**Constraint**

\[
\text{INCREASING WIDTH}\_\text{DECREASING TERRACE}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the values denoting the width of each occurrence of the **DECREASING TERRACE** pattern in the time-series given by the **VARIABLES** collection are increasing.

**Purpose**

An occurrence of the pattern **DECREASING TERRACE** is the *maximal* subsequence which matches the regular expression ‘\( > = + > \)’. Assume that the occurrence of the pattern **DECREASING TERRACE** starts at position \( i \) and ends at position \( j \). The feature **WIDTH** computes the value \( j - i \).

**Example**

\[
\langle 5, 2, 2, 1, 5, 4, 3, 3, 2, 4, 4, 6, 5, 5, 5, 4 \rangle
\]

Figure 4.439 provides an example where the **INCREASING WIDTH DECREASING TERRACE** \( \langle [5, 2, 2, 1, 5, 4, 3, 3, 2, 4, 4, 6, 5, 5, 4] \rangle \) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 3
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]
Figure 4.439: Illustrating the **INCREASING_WIDTH_DECREASING_TERRACE** constraint of the **Example** slot.
Figure 4.440 depicts the automaton associated with the constraint INCREASING_WIDTH_DECREASING_TERRACE.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow 0 \\
F & \leftarrow -\infty \\
R & \leftarrow 1 \\
\{D \leftarrow D + 1\} & \quad \{D \leftarrow D + 1\} \\
\end{align*}
\]

\[
\begin{align*}
\{ \leq s \} & \quad \{ \leq s \} \\
\{ \leq s \} & \quad \{ \leq s \} \\
\{ \leq s \} & \quad \{ \leq s \} \\
\{ \leq s \} & \quad \{ \leq s \} \\
\end{align*}
\]

Figure 4.440: Automaton for the INCREASING_WIDTH_DECREASING_TERRACE constraint obtained by applying decoration Table 3.38 to the seed transducer of the DECREASING_TERRACE pattern
### INCREASING_WIDTH_GORGE

**Description**

**Origin**

Based on the GORGE pattern.

**Constraint**

INCREASING_WIDTH_GORGE(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES.var)

**Purpose**

Succeeds if the values denoting the width of each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘(>|>|=(|>)*)<(|<(<(|<)*)<)’.

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j-i\).

**Example**

\((6,2,5,1,1,5,4,2,3,6,5,4,4,3,5,5)\)

Figure 4.441 provides an example where the INCREASING_WIDTH_GORGE ((|[6,2,5,1,1,5,4,2,3,6,5,4,4,3,5,5]|)) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 2\\n\text{range}(\text{VARIABLES}.var) > 1\)
Figure 4.441: Illustrating the INCREASING_WIDTH_GORGE constraint of the **Example** slot
**Automaton**

Figure 4.442 depicts the automaton associated with the constraint INCREASING_WIDTH_GORGE.

\[
\begin{align*}
& \left\{ \begin{array}{l}
C \leftarrow -\infty \\
D \leftarrow 0 \\
F \leftarrow -\infty \\
R \leftarrow 1 
\end{array} \right\} \quad \leq \quad \leq 8 \\
& \left\{ \begin{array}{l}
C \leftarrow D + 1 \\
D \leftarrow 0 
\end{array} \right\} \quad > \quad \{D \leftarrow D + 1\} \\
& \left\{ \begin{array}{l}
C \leftarrow D + 1 \\
D \leftarrow 0 
\end{array} \right\} \quad \leq \quad \leq t \\
& \left\{ \begin{array}{l}
D \leftarrow 0 
\end{array} \right\} \quad > \quad \{D \leftarrow D + 1\} \\
& \left\{ \begin{array}{l}
D \leftarrow C + D + 1 \\
D \leftarrow 0 
\end{array} \right\} \quad = \quad = \{D \leftarrow D + 1\} \\
& \left\{ \begin{array}{l}
D \leftarrow D + 1 
\end{array} \right\} \\
& \left\{ \begin{array}{l}
D \leftarrow D + 1 \\
D \leftarrow 0 
\end{array} \right\} \\
& \left\{ \begin{array}{l}
D \leftarrow D + 1 
\end{array} \right\} \\
& \left\{ \begin{array}{l}
D \leftarrow D + 1 
\end{array} \right\} \\
& \left\{ \begin{array}{l}
D \leftarrow D + 1 
\end{array} \right\} \\
\end{align*}
\]

**Figure 4.442**: Automaton for the INCREASING_WIDTH_GORGE constraint obtained by applying decoration Table 3.38 to the seed transducer of the GORGE pattern (transition \(u \rightarrow r\) has the same register update as transition \(r \rightarrow u\))
**CONDITION**  
**FEATURE**  
**PATTERN**  

**INCREASING_WIDTH_INCREASING_SEQUENCE**

<table>
<thead>
<tr>
<th>Description Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Description Automaton" /></td>
</tr>
</tbody>
</table>

**Origin**  
Based on the **INCREASING_SEQUENCE** pattern.

**Constraint**  
**INCREASING_WIDTH_INCREASING_SEQUENCE**(VARIABLES)

**Argument**  
VARIABLES : **collection**(var−dvar)

**Restriction**  
**required**(VARIABLES.var)

Succeeds if the values denoting the width of each occurrence of the **INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**  
An occurrence of the pattern **INCREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression ‘’**< (| =)* < |’’.
Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position *i* and ends at position *j*. The feature **WIDTH** computes the value *j* − *i* + 2.

**Example**  
((6, 2, 3, 6, 5, 5, 5, 4, 5, 6, 1, 3, 3, 4, 5, 5))

Figure 4.443 provides an example where the **INCREASING_WIDTH_INCREASING_SEQUENCE** constraint holds.

**Typical**  
| VARIABLES | > 1  
**range**(VARIABLES.var) | > 1  

| VARIABLES | > 1  
range(VARIABLES.var) | > 1  

Figure 4.443 provides an example where the **INCREASING_WIDTH_INCREASING_SEQUENCE** constraint holds.
Figure 4.443: Illustrating the **INCREASING_WIDTH_INCREASING_SEQUENCE** constraint of the **Example** slot
Automaton Figure 4.444 depicts the automaton associated with the constraint INCREASING_WIDTH_INCREASING_SEQUENCE.

\[
\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow 0 \\
F &\leftarrow -\infty \\
R &\leftarrow 1 \\
\end{align*}
\]

Figure 4.444: Automaton for the INCREASING_WIDTH_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING_SEQUENCE pattern.
INCREASING_WIDTH_INCREASING_TERRACE

**Description**

Origin: Based on the INCREASING_TERRACE pattern.

Constraint: INCREASING_WIDTH_INCREASING_TERRACE(VARIABLES)

Argument: VARIABLES : collection(var−dvar)

Restriction: required(VARIABLES, var)

**Purpose**

Succeeds if the values denoting the width of each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression '<=+<'. Assume that the occurrence of the pattern INCREASING_TERRACE starts at position i and ends at position j. The feature WIDTH computes the value j − i.

**Example**

\(((1, 2, 2, 4, 5, 5, 6, 6, 4, 4, 2, 5, 5, 7, 7))\)

Figure 4.445 provides an example where the INCREASING_WIDTH_INCREASING_TERRACE \(((1, 2, 2, 4, 5, 5, 6, 6, 4, 4, 2, 5, 5, 7, 7))\) constraint holds.

**Typical**

|VARIABLES| > 3
range(VARIABLES.var) > 2
Figure 4.445: Illustrating the INCREASING_WIDTH_INCREASING_TERRACE constraint of the Example slot
Automaton Figure 4.446 depicts the automaton associated with the constraint INCREASING_WIDTH_INCREASING_TERRACE.

Figure 4.446: Automaton for the INCREASING_WIDTH_INCREASING_TERRACE constraint obtained by applying decoration Table 3.38 to the seed transducer of the INCREASING_TERRACE pattern
INCREASING_WIDTH_INFLEXION

.origin

Based on the INFLEXION pattern.

.constraint

INCREASING_WIDTH_INFLEXION(VARIABLES)

.argument

VARIABLES : collection(var−dvar)

.restriction

required(VARIABLES.var)

.purpose

Succeeds if the values denoting the width of each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression '< (< | |=)* > | > (>| |=)* <'.

Assume that the occurrence of the pattern INFLEXION starts at position \textit{i} and ends at position \textit{j}. The feature \texttt{WIDTH} computes the value \textit{j} − \textit{i}.

.example

\{(1,1,2,1,3,6,4,3,3,2,3,4,5,5,6)\}

Figure 4.447 provides an example where the INCREASEING_WIDTH_INFLEXION (\{1,1,2,1,3,6,4,3,3,2,3,4,5,5,6\}) constraint holds.

.typical

\[|\text{VARIABLES}| > 2\]

\[\text{range}(\text{VARIABLES.var}) > 1\]
Figure 4.447: Illustrating the INCREASING_WIDTH_INFLEXION constraint of the Example slot
Automaton

Figure 4.448 depicts the automaton associated with the constraint `INCREASING_WIDTH_INFLEXION`.

\[
\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow 0 \\
F &\leftarrow -\infty \\
R &\leftarrow 1
\end{align*}
\]

Figure 4.448: Automaton for the `INCREASING_WIDTH_INFLEXION` constraint obtained by applying decoration Table 3.38 to the seed transducer of the INFLEXION pattern (transition \(r \rightarrow t\) has the same registers updates as transition \(t \rightarrow r\))
INCREASING_WIDTH_PEAK

Description:

Origin: Based on the PEAK pattern.

Constraint: INCREASING_WIDTH_PEAK(VARIABLES)

Argument: VARIABLES : collection(var–dvar)

Restriction: required(VARIABLES, var)

Purpose: Succeeds if the values denoting the width of each occurrence of the PEAK pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression '< (|=|<)*(>||=)* >'. Assume that the occurrence of the pattern PEAK starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example: ((7,5,5,1,2,3,2,3,4,5,2,6,6,1))

Figure 4.449 provides an example where the INCREASING_WIDTH_PEAK ([(7,5,5,1,2,3,2,3,4,5,2,6,6,1)]) constraint holds.

Typical: |VARIABLES| > 2

range(VARIABLES,var) > 1
Figure 4.449: Illustrating the INCREASING_WIDTH_PEEK constraint of the Example slot
Automaton  Figure 4.450 depicts the automaton associated with the constraint INCREASING_WIDTH_PEAK.

Figure 4.450: Automaton for the INCREASING_WIDTH_PEAK constraint obtained by applying decoration Table 3.38 to the seed transducer of the PEAK pattern.
### INCREASING_WIDTHPLAIN

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_WIDTHPLAIN</td>
<td>▶</td>
<td>△</td>
</tr>
</tbody>
</table>

**Origin**
Based on the PLAIN pattern.

**Constraint**
INCREASING_WIDTHPLAIN(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the values denoting the width of each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘> =<’. Assume that the occurrence of the pattern PLAIN starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$.

**Example**

```
((3, 6, 6, 3, 4, 5, 5, 4, 6, 6, 7, 5, 6, 3, 2))
```

Figure 4.451 provides an example where the INCREASING_WIDTHPLAIN ((3, 6, 6, 3, 4, 5, 5, 4, 6, 6, 7, 5, 6, 3, 2)) constraint holds.

**Typical**

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>
Figure 4.451: Illustrating the INCREASING_WIDTHPLAIN constraint of the Example slot
Automaton

Figure 4.452 depicts the automaton associated with the constraint INCREASING_WIDTH.PLAIN.

Figure 4.452: Automaton for the INCREASING_WIDTH.PLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLAIN pattern.
<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_WIDTH_PLATEAU</td>
<td>▶ ▶ ▶ ▶</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
<tr>
<td>▶ ▶ ▶ ▶</td>
<td>▶ ▶ ▶ ▶</td>
<td>▶ ▶ ▶ ▶</td>
</tr>
</tbody>
</table>

**Origin**
Based on the PLATEAU pattern.

**Constraint**
\[
\text{INCREASING\_WIDTH\_PLATEAU}(\text{VARIABLES})
\]

**Argument**
\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**
\[
\text{required}(\text{VARIABLES}.\text{var})
\]

Succeeds if the values denoting the width of each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern PLATEAU is the \textit{maximal} subsequence which matches the regular expression \textquoteleft \textless \textequal \textgreater \textquoteleft .
Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature \textit{WIDTH} computes the value \(j - i\).

**Example**
\[
(\langle 7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5 \rangle)
\]

Figure 4.453 provides an example where the \textit{INCREASING\_WIDTH\_PLATEAU} \((\langle 7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5 \rangle)\) constraint holds.

**Typical**
\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]
Figure 4.453: Illustrating the INCREASING_WIDTH_PLATEAU constraint of the Example slot
Automaton

Figure 4.454 depicts the automaton associated with the constraint INCREASING_WIDTH_PLATEAU.

R ∧ (F ≤ C)

<

Figure 4.454: Automaton for the INCREASING_WIDTH_PLATEAU constraint obtained by applying decoration Table 3.38 to the seed transducer of the PLATEAU pattern
**INCREASING_WIDTH_PROPER_PLAIN**

**Origin**
Based on the PROPERPLAIN pattern.

**Constraint**
\( \text{INCREASING\_WIDTH\_PROPER\_PLAIN}(\text{VARIABLES}) \)

**Argument**
\( \text{VARIABLES} : \text{collection}(\text{var} \to \text{dvar}) \)

**Restriction**
\( \text{required}(\text{VARIABLES}.\text{var}) \)

**Purpose**
Succeeds if the values denoting the width of each occurrence of the PROPERPLAIN pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression ‘\( > =^+ < \)’.

Assume that the occurrence of the pattern PROPERPLAIN starts at position \( i \) and ends at position \( j \). The feature \( \text{WIDTH} \) computes the value \( j - i \).

**Example**
\( ((5, \text{3, } 3, 5, 6, 5, 4, 4, 7, 3, 6, 5, 5, 7, 2)) \)

Figure 4.455 provides an example where the INCREASING_WIDTH_PROPERPLAIN\(([[5, \text{3, } 3, 5, 6, 5, 4, 4, 7, 3, 6, 5, 5, 7, 2]])\) constraint holds.

**Typical**
\( |\text{VARIABLES}| > 3 \)
\( \text{range}(\text{VARIABLES}.$\text{var}) > 1 \)
Figure 4.455: Illustrating the \texttt{INCREASING\_WIDTH\_PROPER\_PLAIN} constraint of the \texttt{Example} slot
Figure 4.456 depicts the automaton associated with the constraint INCREASING_WIDTH_PROPERPLAIN.

Automaton

Figure 4.456: Automaton for the INCREASING_WIDTH_PROPER.PLAIN constraint obtained by applying decoration Table 3.38 to the seed transducer of the PROPER.PLAIN pattern.
INCREASING_WIDTH_PROPER_PLATEAU

**Origin**
Based on the PROPER_PLATEAU pattern.

**Constraint**
INCREASING_WIDTH_PROPER_PLATEAU(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the values denoting the width of each occurrence of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression '<= + >'.

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position i and ends at position j. The feature WIDTH computes the value j − i.

**Example**

\((7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))\)

Figure 4.457 provides an example where the INCREASING_WIDTH_PROPER_PLATEAU ((7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3)) constraint holds.

**Typical**

|VARIABLES| > 3
range(VARIABLES.var) > 1
Figure 4.457: Illustrating the INCREASING_WIDTH_PROPER_PLATEAU constraint of the Example slot
Automaton

Figure 4.458 depicts the automaton associated with the constraint INCREASING_WIDTH_PROPER_PLATEAU.

Figure 4.458: Automaton for the INCREASING_WIDTH_PROPER_PLATEAU constraint obtained by applying decoration Table 3.38 to the seed transducer of the PROPER_PLATEAU pattern
INCREASING_WIDTH_STEADY_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_WIDTH_STEADY_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DESCRIPTION**

**Origin**
Based on the STEADY_SEQUENCE pattern.

**Constraint**
INCREASING_WIDTH_STEADY_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the values denoting the width of each occurrence of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression ’:=+‘.

Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position i and ends at position j. The feature WIDTH computes the value j – i + 2.

**Example**

```plaintext
((6, 2, 2, 3, 5, 6, 4, 4, 2, 3, 5, 5, 2, 3, 4))
```

Figure 4.459 provides an example where the INCREASING_WIDTH_STEADY_SEQUENCE (|[6, 2, 2, 3, 5, 6, 4, 4, 2, 3, 5, 5, 2, 3, 4]|) constraint holds.

**Typical**

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 1</th>
</tr>
</thead>
</table>
Figure 4.459: Illustrating the INCREASING_WIDTH_STEADY_SEQUENCE constraint of the Example slot.
Automaton

Figure 4.460 depicts the automaton associated with the constraint INCREASING_WIDTH_STEADY_SEQUENCE.

Figure 4.460: Automaton for the INCREASING_WIDTH_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STEADY_SEQUENCE pattern.
INCREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING</td>
<td>WIDTH</td>
<td>STRICTLY DECREASING SEQUENCE</td>
</tr>
</tbody>
</table>

**Origin**
Based on the STRICTLY_DECREASING_SEQUENCE pattern.

**Constraint**
INCREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the values denoting the width of each occurrence of the STRICTLY_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression '>+'. Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at position i and ends at position j. The feature WIDTH computes the value j − i + 2.

**Example**
((5, 7, 4, 4, 2, 3, 4, 3, 2, 2, 4, 5, 6, 3, 2))

Figure 4.461 provides an example where the INCREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE ((5, 7, 4, 4, 2, 3, 4, 3, 2, 2, 4, 5, 6, 3, 2)) constraint holds.

**Typical**
<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>range(VARIABLES.var) &gt; 1</th>
</tr>
</thead>
</table>
Figure 4.461: Illustrating the INCREASING_WIDTH.Strictly_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 4.462 depicts the automaton associated with the constraint INCREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE.

\[
\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow 0 \\
F &\leftarrow -\infty \\
R &\leftarrow 1
\end{align*}
\]

Figure 4.462: Automaton for the INCREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.38 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern.
INCREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE
### INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE</td>
<td>↑</td>
<td>↑</td>
</tr>
</tbody>
</table>

**Origin**
Based on the `STRICTLY_INCREASING_SEQUENCE` pattern.

**Constraint**

\[
\text{INCREASING_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}, \text{var})
\]

Succeeds if the values denoting the width of each occurrence of the `STRICTLY_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing.

**Purpose**
An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the maximal sub-sequence which matches the regular expression `'<+>'`. Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `WIDTH` computes the value \(j - i + 2\).

**Example**

\[
((6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 2, 3, 4, 5, 5))
\]

Figure 4.463 provides an example where the `INCREASING_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE` ((6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 2, 3, 4, 5, 5)) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.463: Illustrating the `INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint of the **Example** slot
Figure 4.464 depicts the automaton associated with the constraint `INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE`.

Figure 4.464: Automaton for the `INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern.
INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE
**INCREASING_WIDTH_SUMMIT**

<table>
<thead>
<tr>
<th>CONDITION</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>INCREASING_WIDTH_SUMMIT</strong></td>
<td>▶ ◄ ◄ ◄</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the SUMMIT pattern.

**Constraint**

\[
\text{INCREASING\_WIDTH\_SUMMIT(VARIABLES)}
\]

**Argument**

\[
\text{VARIABLES : collection(var - dvar)}
\]

**Restriction**

\[
\text{required(VARIABLES.var)}
\]

Succeeds if the values denoting the width of each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern SUMMIT is the *maximal* subsequence which matches the regular expression \((< | < (= | <) <) (> | > (= | >) >)\). Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\[
((1,5,2,1,6,2,3,5,4,1,4,6,4,3,2))
\]

Figure 4.465 provides an example where the INCREASING_WIDTH_SUMMIT \(((1,5,2,1,6,2,3,5,4,1,4,6,4,3,2))\) constraint holds.

**Typical**

\[
\text{|VARIABLES| > 2} \\
\text{range(VARIABLES.var) > 1}
\]
Figure 4.465: Illustrating the INCREASING_WIDTH_SUMMIT constraint of the Example slot
Automaton Figure 4.466 depicts the automaton associated with the constraint INCREASING_WIDTH_SUMMIT.

![Automaton Diagram]

Figure 4.466: Automaton for the INCREASING_WIDTH_SUMMIT constraint obtained by applying decoration Table 3.38 to the seed transducer of the SUMMIT pattern (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
**Origin**
Based on the VALLEY pattern.

**Constraint**
\[ \text{INCREASING_WIDTH_VALLEY} \text{(VARIABLES)} \]

**Argument**
\[ \text{VARIABLES} : \text{collection(var-dvar)} \]

**Restriction**
\[ \text{required} \text{(VARIABLES.var)} \]

Succeeds if the values denoting the width of each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression '\( > (= | >)^* (< | =)^* < \)'. Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**
\[ (7, 2, 2, 6, 3, 4, 5, 6, 4, 6, 4, 6, 7, 3, 3, 1) \]

Figure 4.467 provides an example where the \text{INCREASING_WIDTH_VALLEY} \((7, 2, 2, 6, 3, 4, 5, 6, 4, 6, 4, 6, 7, 3, 3, 1)\) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 2 \]
\[ \text{range} \text{(VARIABLES.var)} > 1 \]
Figure 4.467: Illustrating the INCREASING_WIDTH_VALLEY constraint of the Example slot
Figure 4.468 depicts the automaton associated with the constraint `INCREASING_WIDTH_VALLEY`.

\[
\begin{align*}
\{ C \leftarrow -\infty \} & \quad \leq 8 \quad \leq \quad \{ D \leftarrow D + 1 \} \\
\{ D \leftarrow 0 \} & \quad > \quad \{ C \leftarrow D + 1 \} \\
\{ F \leftarrow -\infty \} & \quad ( \quad \{ R \leftarrow 1 \} \\
\{ R \leftarrow -\infty \} & \quad \{ F \leftarrow -\infty \}
\end{align*}
\]

Figure 4.468: Automaton for the `INCREASING_WIDTH_VALLEY` constraint obtained by applying decoration Table 3.38 to the seed transducer of the `VALLEY` pattern.
**INCREASING_WIDTH_ZIGZAG**

### Description

**Pattern**

- **Origin**: Based on the ZIGZAG pattern.
- **Constraint**: INCREASING_WIDTH_ZIGZAG(VARIABLES)
- **Argument**: VARIABLES : collection(var–dvar)
- **Restriction**: required(VARIABLES.var)

**Purpose**

Succeeds if the values denoting the width of each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression `\((<>)^+\)\((< | <>)(<>)^+(>|<)\)'.

Assume that the occurrence of the pattern ZIGZAG starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$.

**Example**

```
((7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6))
```

Figure 4.469 provides an example where the INCREASING_WIDTH_ZIGZAG ([7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6]) constraint holds.

**Typical**

- $|\text{VARIABLES}| > 3$
- $\text{range(VARIABLES.var)} > 1$
Figure 4.469: Illustrating the INCREASING_WIDTH_ZIGZAG constraint of the Example slot
Automaton

Figure 4.470 depicts the automaton associated with the constraint INCREASING_WIDTH_ZIGZAG.
INCREASING_WIDTH_ZIGZAG

Figure 4.470: Automaton for the INCREASING_WIDTH_ZIGZAG constraint obtained by applying decoration Table 3.38 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the register $F$ is reset to $C$, and the register $R$ is updated wrt $C$ and $F$
### INDEX_BUMP_ON_DECREASING_SEQUENCE

**Description**

Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**

\[
\text{INDEX\_BUMP\_ON\_DECREASING\_SEQUENCE}(\text{VARIABLES, FEATURES, DEFAULT})
\]

**Arguments**

- **VARIABLES**: \(\text{collection}(\text{var\_dvar})\)
- **FEATURES**: \(\text{collection}(\text{var\_dvar})\)
- **DEFAULT**: \(\text{int}\)

**Restrictions**

- \(\text{required(\text{VARIABLES}.\text{var})}\)
- \(\text{required(\text{FEATURES}.\text{var})}\)
- \(|\text{VARIABLES}| = |\text{FEATURES}|\)
- \(\text{FEATURES}.\text{var} \geq 0\)
- \(\text{FEATURES}.\text{var} \leq 1\)
- \(\text{DEFAULT} = 0\)

Consider the time-series given by the \(\text{VARIABLES}\) collection. If position \(i\) does not correspond to the first position in \(\text{VARIABLES}\) where an occurrence of BUMP_ON_DECREASING_SEQUENCE is identified then \(\text{FEATURES}[i]\) is the default value DEFAULT; otherwise \(\text{FEATURES}[i]\) gives the feature value of the corresponding occurrence of BUMP_ON_DECREASING_SEQUENCE.

**Purpose**

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression ‘\(>>\ll\)’. The feature ONE, called INDEX in the name of the constraint returns 1.

**Example**

Figure 4.471 provides an example where the INDEX_BUMP_ON_DECREASING_SEQUENCE constraint holds.

- \(|\text{VARIABLES}| > 5\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 2\)

**Functional dependency**

FEATURES determined by \(\text{VARIABLES}\) and DEFAULT.
Figure 4.471: Illustrating the INDEX_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
| Automaton | Use the decoration table 3.32 to synthesise the corresponding register automaton. |
Origin

Based on the DECREASING pattern.

Constraint

INDEX\_DECREASING(VARIABLES, FEATURES, DEFAULT)

Arguments

VARIABLES : collection(var+dvar)
FEATURES : collection(var+dvar)
DEFAULT : int

Restrictions

\[
\text{required}(\text{VARIABLES}.\text{var}) \\
\text{required}(\text{FEATURES}.\text{var}) \\
|\text{VARIABLES}| = |\text{FEATURES}| \\
\text{FEATURES}.\text{var} \geq 0 \\
\text{FEATURES}.\text{var} \leq 1 \\
\text{DEFAULT} = 0
\]

Purpose

Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of DECREASING is identified then FEATURES\[i\] is the default value DEFAULT; otherwise FEATURES\[i\] gives the feature value of the corresponding occurrence of DECREASING.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression '>'.

The feature ONE, called INDEX in the name of the constraint returns 1.

Example

Figure 4.472 provides an example where the INDEX\_DECREASING ([3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4], [0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0]) constraint holds.

Typical

|VARIABLES| > 1
range(VARIABLES.var) > 1

Arg. properties

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.472: Illustrating the INDEX_DECREASING constraint of the Example slot
Use the decoration table 3.32 to synthesise the corresponding register automaton.
DECREASING_SEQUENCE

DESCRIPTION AUTOMATON

Origin
Based on the DECREASING_SEQUENCE pattern.

Constraint
INDEX_DECREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)

Arguments
<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>collection(var−dvar)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FEATURES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>int</td>
</tr>
</tbody>
</table>

Restrictions
required(VARIABLES.var)
required(FEATURES.var)
|VARIABLES| = |FEATURES|
FEATURES.var ≥ 0
FEATURES.var ≤ 1
DEFAULT = 0

Purpose
Consider the time-series given by the VARIABLES collection. If position i does not correspond to the first position in VARIABLES where an occurrence of DECREASING_SEQUENCE is identified (even if this occurrence of pattern is not complete) then FEATURES[i] is the default value DEFAULT; otherwise FEATURES[i] gives the feature value of the corresponding occurrence of DECREASING_SEQUENCE.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘(>||=)∗>|’.

The feature ONE, called INDEX in the name of the constraint returns 1.

Example
Figure 4.473 provides an example where the INDEX_DECREASING_SEQUENCE ([3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 4, 6, 4, 4], [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES.var) > 1

Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.473: Illustrating the INDEX_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
INDEX_DECREASING_SEQUENCE
### INDEX DECREASING_TERRACE

**Description**

Based on the **DECREASING_TERRACE** pattern.

**Constraint**

```
INDEX_DECREASING_TERRACE(VARIABLES, FEATURES, DEFAULT)
```

**Arguments**

- **VARIABLES**: `collection(var−dvar)`
- **FEATURES**: `collection(var−dvar)`
- **DEFAULT**: `int`

**Restrictions**

- `required(VARIABLES.var)`
- `required(FEATURES.var)`
- `|VARIABLES| = |FEATURES|`
- `FEATURES.var ≥ 0`
- `FEATURES.var ≤ 1`
- `DEFAULT = 0`

Consider the time-series given by the **VARIABLES** collection. If position \( i \) does not correspond to the first position in **VARIABLES** where an occurrence of **DECREASING_TERRACE** is identified then \( \text{FEATURES}[i] \) is the default value **DEFAULT**; otherwise \( \text{FEATURES}[i] \) gives the feature value of the corresponding occurrence of **DECREASING_TERRACE**.

**Purpose**

An occurrence of the pattern **DECREASING_TERRACE** is the **maximal** subsequence which matches the regular expression ‘\( > = + > \)’.

The feature **ONE**, called **INDEX** in the name of the constraint returns 1.

**Example**

Figure 4.474 provides an example where the **INDEX_DECREASING_TERRACE** constraint holds.

**Typical**

```
|VARIABLES| > 3
range(VARIABLES.var) > 2
```

**Arg. properties**

Functional dependency: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.474: Illustrating the INDEX_DECREASING_TERRACE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
INDEX_DECREASING_TERRACE
INDEX_DIP_ON_INCREASING_SEQUENCE

Origin
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

Constraint
INDEX_DIP_ON_INCREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)

Arguments
VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int

Restrictions
required(VARIABLES.var)
required(FEATURES.var)
|VARIABLES| = |FEATURES|
FEATURES.var ≥ 0
FEATURES.var ≤ 1
DEFAULT = 0

Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of DIP_ON_INCREASING_SEQUENCE is identified then FEATURES$[i]$ is the default value DEFAULT; otherwise FEATURES$[i]$ gives the feature value of the corresponding occurrence of DIP_ON_INCREASING_SEQUENCE.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression ‘<<><<’.

The feature ONE, called INDEX in the name of the constraint returns 1.

Purpose
Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of DIP_ON_INCREASING_SEQUENCE is identified then FEATURES$[i]$ is the default value DEFAULT; otherwise FEATURES$[i]$ gives the feature value of the corresponding occurrence of DIP_ON_INCREASING_SEQUENCE.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression ‘<<><<’.

The feature ONE, called INDEX in the name of the constraint returns 1.

Example
Figure 4.475 provides an example where the INDEX_DIP_ON_INCREASING_SEQUENCE ([1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical
|VARIABLES| > 5
range(VARIABLES.var) > 2

Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.475: Illustrating the INDEX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
INDEX_DIP_ON_INCREASING_SEQUENCE
Origin
Based on the GORGE pattern.

Constraint
INDEX_GORGE(VARIABLES, FEATURES, DEFAULT)

Arguments
VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int

Restrictions
required(VARIABLES.var)
required(FEATURES.var)
|VARIABLES| = |FEATURES|
FEATURES.var ≥ 0
FEATURES.var ≤ 1
DEFAULT = 0

Purpose
Consider the time-series given by the VARIABLES collection. If position i does not correspond to the first position in VARIABLES where an occurrence of GORGE is identified (even if this occurrence of pattern is not complete) then FEATURES[i] is the default value DEFAULT; otherwise FEATURES[i] gives the feature value of the corresponding occurrence of GORGE.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘(>|=(>|)*>|(<|=(|)<>*|<)’.

The feature ONE, called INDEX in the name of the constraint returns 1.

Example
Figure 4.476 provides an example where the INDEX_GORGE([1, 7, 3, 4, 4, 5, 5, 4, 2, 6, 5, 4, 6, 5, 7], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0), constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1

Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.476: Illustrating the INDEX_GORGE constraint of the Example slot
Automaton  Use the decoration table 3.32 to synthesise the corresponding register automaton.
INDEX_INCREASING

**Description**

**Origin**
Based on the INCREASING pattern.

**Constraint**
INDEX_INCREASING(VARIABLES, FEATURES, DEFAULT)

**Arguments**
- VARIABLES: collection(var, dvar)
- FEATURES: collection(var, dvar)
- DEFAULT: int

**Restrictions**
- required(VARIABLES.var)
- required(FEATURES.var)
- |VARIABLES| = |FEATURES|
- FEATURES.var ≥ 0
- FEATURES.var ≤ 1
- DEFAULT = 0

Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of INCREASING is identified then FEATURES[$i$] is the default value DEFAULT; otherwise FEATURES[$i$] gives the feature value of the corresponding occurrence of INCREASING.

**Purpose**

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression ‘<’.

The feature ONE, called INDEX in the name of the constraint returns 1.

**Example**

Figure 4.477 provides an example where the INDEX_INCREASING ([4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0]) constraint holds.

**Typical**

- |VARIABLES| > 1
- range(VARIABLES.var) > 1

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.477: Illustrating the INDEX_INCREASING constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
INDEX_INCREASING_SEQUENCE

Origin

Based on the INCREASING_SEQUENCE pattern.

Constraint

INDEX_INCREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)

Arguments

| VARIABLES : collection(var−dvar) |
| FEATURES : collection(var−dvar) |
| DEFAULT : int |

Restrictions

required(VARIABLES.var)  
required(FEATURES.var)  
|VARIABLES| = |FEATURES|  
FEATURES.var ≥ 0  
FEATURES.var ≤ 1  
DEFAULT = 0

Purpose

Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of INCREASING_SEQUENCE is identified (even if this occurrence of pattern is not complete) then FEATURES[\(i\)] is the default value DEFAULT; otherwise FEATURES[\(i\)] gives the feature value of the corresponding occurrence of INCREASING_SEQUENCE.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\(<(<|=)^*<|<\)’.

The feature ONE, called INDEX in the name of the constraint returns 1.

Example

Figure 4.478 provides an example where the INDEX_INCREASING_SEQUENCE ([4, 3, 5, 2, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]) constraint holds.

Typical

|VARIABLES| > 1  
range(VARIABLES.var) > 1

Arg. properties

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.478: Illustrating the `INDEX_INCREASING_SEQUENCE` constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
### INDEX_INCREMENTING_TERRACE

#### Origin
Based on the `INCREMENTING_TERRACE` pattern.

#### Constraint
`INDEX_INCREMENTING_TERRACE(VARIABLES, FEATURES, DEFAULT)`

#### Arguments
- **VARIABLES**: `collection(var−dvar)`
- **FEATURES**: `collection(var−dvar)`
- **DEFAULT**: `int`

#### Restrictions
- `required(VARIABLES.var)`
- `required(FEATURES.var)`
- `|VARIABLES| = |FEATURES|`
- `FEATURES.var ≥ 0`
- `FEATURES.var ≤ 1`
- `DEFAULT = 0`

Consider the time-series given by the `VARIABLES` collection. If position `i` does not correspond to the first position in `VARIABLES` where an occurrence of `INCREMENTING_TERRACE` is identified then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `INCREMENTING_TERRACE`.

An occurrence of the pattern `INCREMENTING_TERRACE` is the **maximal** subsequence which matches the regular expression `’<=+<’`.

The feature `ONE`, called `INDEX` in the name of the constraint returns 1.

#### Purpose
- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 2`

**Functional dependency**: `FEATURES` determined by `VARIABLES` and `DEFAULT`.

---

**Example**

Figure 4.479 provides an example where the `INDEX_INCREMENTING_TERRACE` constraint holds. It uses the example: `V = [1, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 4, 4], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0]`.
Figure 4.479: Illustrating the `INDEX_INCREASING_TERRACE` constraint of the Example slot
Automaton  
Use the decoration table 3.32 to synthesise the corresponding register automaton.
INDEX_INFLEXION

DESCRIPTION

Automaton

< (< | =)∗ > | > (> | =)∗ <

Origin
Based on the INFLEXION pattern.

Constraint
INDEX_INFLEXION(VARIABLES, FEATURES, DEFAULT)

Arguments
VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int

Restrictions
required(VARIABLES.var)
required(FEATURES.var)
|VARIABLES| = |FEATURES|
FEATURES.var ≥ 0
FEATURES.var ≤ 1
DEFAULT = 0

Consider the time-series given by the VARIABLES collection. If position \( i \) does not correspond to the first position in VARIABLES where an occurrence of INFLEXION is identified then FEATURES[\( i \)] is the default value DEFAULT; otherwise FEATURES[\( i \)] gives the feature value of the corresponding occurrence of INFLEXION.

Purpose
An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression ‘< (< | =)∗ > | > (> | =)∗ <’. The feature \( \text{ONE} \), called INDEX in the name of the constraint returns 1.

Example
Figure 4.480 provides an example where the INDEX_INFLEXION
\([1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4], [0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0], 0\)
constraint holds.

Typical
\(|\text{VARIABLES}| > 2
range(\text{VARIABLES}.var) > 1

Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.480: Illustrating the INDEX_INFLEXION constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
INDEX PEAK

DESCRIPTION

Automaton

Origin
Based on the PEAK pattern.

Constraint
INDEX PEAK (VARIABLES, FEATURES, DEFAULT)

Arguments
VARIABLES : collection(var-dvar)
FEATURES : collection(var-dvar)
DEFAULT : int

Restrictions
required(VARIABLES.var)
required(FEATURES.var)
|VARIABLES| = |FEATURES|
FEATURES.var ≥ 0
FEATURES.var ≤ 1
DEFAULT = 0

Consider the time-series given by the VARIABLES collection. If position \( i \) does not correspond to the first position in VARIABLES where an occurrence of PEAK is identified (even if this occurrence of pattern is not complete) then FEATURES[\( i \)] is the default value DEFAULT; otherwise FEATURES[\( i \)] gives the feature value of the corresponding occurrence of PEAK.

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression ‘\(< (= | <) (> | =) >\)’.

The feature ONE, called INDEX in the name of the constraint returns 1.

Purpose

Example
Figure 4.481 provides an example where the INDEX PEAK ([7, 5, 5, 1, 4, 5, 2, 3, 5, 6, 2, 3, 3, 1], [0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0], 0) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1

Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.481: Illustrating the INDEX_PEAK constraint of the Example slot
Automaton: Use the decoration table 3.32 to synthesise the corresponding register automaton.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Pattern</th>
<th>INDEX/plain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Automaton</td>
<td></td>
</tr>
<tr>
<td>Origin</td>
<td>Based on the <code>PLAIN</code> pattern.</td>
<td></td>
</tr>
<tr>
<td>Constraint</td>
<td><code>INDEX/plain(VARIABLES, FEATURES, DEFAULT)</code></td>
<td></td>
</tr>
<tr>
<td>Arguments</td>
<td><code>VARIABLES : collection(var−dvar)</code></td>
<td><code>FEATURES : collection(var−dvar)</code></td>
</tr>
<tr>
<td></td>
<td><code>DEFAULT : int</code></td>
<td></td>
</tr>
<tr>
<td>Restrictions</td>
<td><code>required(VARIABLES, var)</code></td>
<td><code>required(FEATURES, var)</code></td>
</tr>
<tr>
<td></td>
<td>`</td>
<td>VARIABLES</td>
</tr>
<tr>
<td></td>
<td><code>FEATURES.var ≥ 0</code></td>
<td><code>FEATURES.var ≤ 1</code></td>
</tr>
<tr>
<td></td>
<td><code>DEFAULT = 0</code></td>
<td></td>
</tr>
<tr>
<td>Purpose</td>
<td>Consider the time-series given by the <code>VARIABLES</code> collection. If position <code>i</code> does not correspond to the first position in <code>VARIABLES</code> where an occurrence of <code>PLAIN</code> is identified then <code>FEATURES[i]</code> is the default value <code>DEFAULT</code>; otherwise <code>FEATURES[i]</code> gives the feature value of the corresponding occurrence of <code>PLAIN</code>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>An occurrence of the pattern <code>PLAIN</code> is the <code>maximal</code> subsequence which matches the regular expression <code>'&gt;=' '&lt;'</code>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The feature <code>ONE</code>, called <code>INDEX</code> in the name of the constraint returns 1.</td>
<td></td>
</tr>
<tr>
<td>Example</td>
<td>Figure 4.482 provides an example where the <code>INDEX/plain</code> ([2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 6, 6, 3], [0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0], 0) constraint holds.</td>
<td></td>
</tr>
<tr>
<td>Typical</td>
<td>`</td>
<td>VARIABLES</td>
</tr>
<tr>
<td>Arg. properties</td>
<td>Functional dependency: <code>FEATURES</code> determined by <code>VARIABLES</code> and <code>DEFAULT</code>.</td>
<td></td>
</tr>
</tbody>
</table>
Figure 4.482: Illustrating the INDEX_PLAIN constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
### INDEX_PLATEAU

**Origin**

Based on the PLATEAU pattern.

**Constraint**

\[
\text{INDEX_PLATEAU} \left( \text{VARIABLES}, \text{FEATURES}, \text{DEFAULT} \right)
\]

**Arguments**

- VARIABLES : \(\text{collection(var – dvar)}\)
- FEATURES : \(\text{collection(var – dvar)}\)
- DEFAULT : \(\text{int}\)

**Restrictions**

- required(VARIABLES.var)
- required(FEATURES.var)
- \(|\text{VARIABLES}| = |\text{FEATURES}|\)
- FEATURES.var \(\geq 0\)
- FEATURES.var \(\leq 1\)
- DEFAULT = 0

**Purpose**

Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of PLATEAU is identified then FEATURES\([i]\) is the default value DEFAULT; otherwise FEATURES\([i]\) gives the feature value of the corresponding occurrence of PLATEAU.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression ‘\(<=^*^>\)’. The feature ONE, called INDEX in the name of the constraint returns 1.

**Example**

Figure 4.483 provides an example where the INDEX_PLATEAU\((\{7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5\}, \{0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0\}, 0)\) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 2\)
- range(VARIABLES.var) > 1

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.483: Illustrating the INDEX_PLATEAU constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
Origin

Based on the PROPERPLAIN pattern.

Constraint

INDEX_PROPER_PLAIN(VARIABLES, FEATURES, DEFAULT)

Arguments

VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int

Restrictions

required(VARIABLES.var)
required(FEATURES.var)
|VARIABLES| = |FEATURES|
FEATURES.var ≥ 0
FEATURES.var ≤ 1
DEFAULT = 0

Purpose

Consider the time-series given by the VARIABLES collection. If position i does not correspond to the first position in VARIABLES where an occurrence of PROPERPLAIN is identified then FEATURES[i] is the default value DEFAULT; otherwise FEATURES[i] gives the feature value of the corresponding occurrence of PROPERPLAIN.

An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression ‘> =+ <’.

The feature ONE, called INDEX in the name of the constraint returns 1.

Example

Figure 4.484 provides an example where the INDEX_PROPERPLAIN ([2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5], [0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0], 0) constraint holds.

Typical

|VARIABLES| > 3
range(VARIABLES.var) > 1

Arg. properties

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.484: Illustrating the INDEX_PROPER_PLAIN constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
## INDEX_PROPER_PLATEAU

### Origin
Based on the PROPER_PLATEAU pattern.

### Constraint
INDEX_PROPER_PLATEAU(VARIABLES, FEATURES, DEFAULT)

### Arguments
- **VARIABLES**: `collection(var−dvar)`
- **FEATURES**: `collection(var−dvar)`
- **DEFAULT**: `int`

### Restrictions
- `required(VARIABLES.var)`
- `required(FEATURES.var)`
- `|VARIABLES| = |FEATURES|`
- `FEATURES.var ≥ 0`
- `FEATURES.var ≤ 1`
- `DEFAULT = 0`

### Purpose
Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of PROPER_PLATEAU is identified then FEATURES\([i]\) is the default value DEFAULT; otherwise FEATURES\([i]\) gives the feature value of the corresponding occurrence of PROPER_PLATEAU.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression '<\(=+\)('>.

The feature ONE, called INDEX in the name of the constraint returns 1.

### Example
Figure 4.485 provides an example where the INDEX_PROPER_PLATEAU ([7, 1, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3], [0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0], 0) constraint holds.

### Typical
- \(|VARIABLES| > 3\)
- `range(VARIABLES.var) > 1`

### Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.485: Illustrating the INDEX_PROPER_PLATEAU constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
### INDEX_STEADY

**Origin**  
Based on the `STEADY` pattern.

**Constraint**  
\[
\text{INDEX\_STEADY}(\text{VARIABLES}, \text{FEATURES}, \text{DEFAULT})
\]

**Arguments**  
- VARIABLES: `collection(var-dvar)`
- FEATURES: `collection(var-dvar)`
- DEFAULT: `int`

**Restrictions**  
- `required(\text{VARIABLES}.var)`
- `required(\text{FEATURES}.var)`
- `|\text{VARIABLES}| = |\text{FEATURES}|`
- `\text{FEATURES}.var \geq 0`
- `\text{FEATURES}.var \leq 1`
- `\text{DEFAULT} = 0`

**Purpose**  
Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of `STEADY` is identified then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `STEADY`.

An occurrence of the pattern `STEADY` is the subsequence which matches the regular expression `\'\=\'`. The feature `ONE`, called `INDEX` in the name of the constraint returns 1.

**Example**  
Figure 4.486 provides an example where the `INDEX\_STEADY([1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6], [1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0])` constraint holds.

**Typical**  
\[|\text{VARIABLES}| > 1\]

**Arg. properties**  
Functional dependency: `FEATURES` determined by `VARIABLES` and `DEFAULT`. 
Figure 4.486: Illustrating the INDEX_STEADY constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
### Description

**Origin**  
Based on the `STEADY_SEQUENCE` pattern.

**Constraint**  
`INDEX_STEADY_SEQUENCE(VARIABLES, FEATURES, DEFAULT)`

**Arguments**  
- `VARIABLES`: `collection(var−dvar)`
- `FEATURES`: `collection(var−dvar)`
- `DEFAULT`: `int`

**Restrictions**  
- `required(VARIABLES.var)`
- `required(FEATURES.var)`
- `|VARIABLES| = |FEATURES|`
- `FEATURES.var ≥ 0`
- `FEATURES.var ≤ 1`
- `DEFAULT = 0`

Consider the time-series given by the `VARIABLES` collection. If position `i` does not correspond to the first position in `VARIABLES` where an occurrence of `STEADY_SEQUENCE` is identified (even if this occurrence of pattern is not complete) then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `STEADY_SEQUENCE`.

An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression `'=+ '`.  
The feature `ONE`, called `INDEX` in the name of the constraint returns 1.

**Example**  
Figure 4.487 provides an example where the `INDEX_STEADY_SEQUENCE` constraint holds.

**Purpose**  

**Typical**  

**Arg. properties**  

Functional dependency: `FEATURES` determined by `VARIABLES` and `DEFAULT`.
Figure 4.487: Illustrating the INDEX_STEADY_SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
**INDEX\_STRICTLY\_DECREASING\_SEQUENCE**

<table>
<thead>
<tr>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDEX_STRICTLY_DECREASING_SEQUENCE</td>
<td>▼ ▼ ▼</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
</table>

**Origin**  
Based on the **STRICTLY\_DECREASING\_SEQUENCE** pattern.

**Constraint**  
**INDEX\_STRICTLY\_DECREASING\_SEQUENCE**(VARIABLES, FEATURES, DEFAULT)

**Arguments**  
<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>FEATURES</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>: collection(var_dvar)</td>
<td>: collection(var_dvar)</td>
<td>: int</td>
</tr>
</tbody>
</table>

**Restrictions**  
<table>
<thead>
<tr>
<th>VARIABLES.var</th>
<th>FEATURES.var</th>
</tr>
</thead>
<tbody>
<tr>
<td>required(VARIABLES.var)</td>
<td>required(FEATURES.var)</td>
</tr>
<tr>
<td></td>
<td>[VARIABLES] =</td>
</tr>
<tr>
<td>FEATURES.var ( \geq 0 )</td>
<td></td>
</tr>
<tr>
<td>FEATURES.var ( \leq 1 )</td>
<td></td>
</tr>
<tr>
<td>DEFAULT = 0</td>
<td></td>
</tr>
</tbody>
</table>

**Purpose**  
Consider the time-series given by the VARIABLES collection. If position \( i \) does not correspond to the first position in VARIABLES where an occurrence of **STRICTLY\_DECREASING\_SEQUENCE** is identified (even if this occurrence of pattern is not complete) then FEATURES[\( i \)] is the default value DEFAULT; otherwise FEATURES[\( i \)] gives the feature value of the corresponding occurrence of **STRICTLY\_DECREASING\_SEQUENCE**.  
An occurrence of the pattern **STRICTLY\_DECREASING\_SEQUENCE** is the **maximal** subsequence which matches the regular expression \( ^\text{>}^+ \).  
The feature **ONE**, called INDEX in the name of the constraint returns 1.

**Example**  
Figure 4.488 provides an example where the **INDEX\_STRICTLY\_DECREASING\_SEQUENCE** ([4, 4, 6, 1, 1, 3, 4, 6, 6, 5, 2, 4, 3], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**  
<table>
<thead>
<tr>
<th>VARIABLES</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>\text{VARIABLES}</td>
</tr>
<tr>
<td>range(VARIABLES.var) &gt; 1</td>
<td></td>
</tr>
</tbody>
</table>

**Arg. properties**  
**Functional dependency**: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.488: Illustrating the INDEX\_STRICTLY\_DECREASING\_SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
INDEX STRICTLY INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDEX STRICTLY INCREASING_SEQUENCE</td>
<td>↑</td>
</tr>
</tbody>
</table>

### Description

**Origin**

Based on the `STRICTLY_INCREASING_SEQUENCE` pattern.

**Constraint**

```
INDEX STRICTLY INCREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)
```

**Arguments**

- `VARIABLES`: collection(var-dvar)
- `FEATURES`: collection(var-dvar)
- `DEFAULT`: int

**Restrictions**

- `required(VARIABLES.var)`
- `required(FEATURES.var)`
- `|VARIABLES| = |FEATURES|`
- `FEATURES.var ≥ 0`
- `FEATURES.var ≤ 1`
- `DEFAULT = 0`

Consider the time-series given by the `VARIABLES` collection. If position `i` does not correspond to the first position in `VARIABLES` where an occurrence of `STRICTLY_INCREASING_SEQUENCE` is identified (even if this occurrence of pattern is not complete) then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `STRICTLY_INCREASING_SEQUENCE`.

An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the maximal sub-sequence which matches the regular expression `'<.'`.

The feature `ONE`, called `INDEX` in the name of the constraint returns 1.

**Purpose**

**Example**

Figure 4.489 provides an example where the `INDEX STRICTLY INCREASING_SEQUENCE` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Arg. properties**

Functional dependency: `FEATURES` determined by `VARIABLES` and `DEFAULT`.
Figure 4.489: Illustrating the INDEX\_STRICTLY\_INCREASING\_SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
INDEX_SUMMIT

Origin
Based on the SUMMIT pattern.

Constraint
INDEX_SUMMIT(VARIABLES, FEATURES, DEFAULT)

Arguments
VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int

Restrictions
required(VARIABLES.var)
required(FEATURES.var)
|VARIABLES| = |FEATURES|
FEATURES.var ≥ 0
FEATURES.var ≤ 1
DEFAULT = 0

Consider the time-series given by the VARIABLES collection. If position \( i \) does not correspond to the first position in VARIABLES where an occurrence of SUMMIT is identified (even if this occurrence of pattern is not complete) then FEATURES[\( i \)] is the default value DEFAULT; otherwise FEATURES[\( i \)] gives the feature value of the corresponding occurrence of SUMMIT.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression \( (< | < (= | <)^* <)(> | > (= | >)^* >) \).

The feature ONE, called INDEX in the name of the constraint returns 1.

Purpose

Example
Figure 4.490 provides an example where the INDEX_SUMMIT ([7, 1, 5, 4, 4, 3, 4, 6, 2, 3, 4, 2, 3, 1], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0]) constraint holds.

Typical

|VARIABLES| > 2
range(VARIABLES.var) > 1

Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.490: Illustrating the INDEX_SUMMIT constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
Origin Based on the VALLEY pattern.

Constraint

INDEX_VALLEY(VARIABLES, FEATURES, DEFAULT)

Arguments

VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int

Restrictions

required(VARIABLES.var)
required(FEATURES.var)
|VARIABLES| = |FEATURES|
FEATURES.var ≥ 0
FEATURES.var ≤ 1
DEFAULT = 0

Purpose

Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of VALLEY is identified (even if this occurrence of pattern is not complete) then FEATURES[\(i\)] is the default value DEFAULT; otherwise FEATURES[\(i\)] gives the feature value of the corresponding occurrence of VALLEY.

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \(\geq \left(\geq | > \right)^* (\leq | =)^* \leq \).

The feature ONE, called INDEX in the name of the constraint returns 1.

Example

Figure 4.491 provides an example where the INDEX_VALLEY ([1, 3, 7, 4, 3, 6, 5, 3, 2, 6, 5, 5, 7], [0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0], 0) constraint holds.

Typical

|VARIABLES| > 2
range(VARIABLES.var) > 1

Arg. properties

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.491: Illustrating the INDEX_VALLEY constraint of the Example slot
Use the decoration table 3.32 to synthesise the corresponding register automaton.
### INDEX_ZIGZAG

#### Origin
Based on the ZIGZAG pattern.

#### Constraint

<table>
<thead>
<tr>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDEX_ZIGZAG(VARIABLES, FEATURES, DEFAULT)</td>
<td></td>
</tr>
</tbody>
</table>

#### Arguments

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>FEATURES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>int</td>
</tr>
</tbody>
</table>

#### Restrictions

- required(VARIABLES.var)
- required(FEATURES.var)
- |VARIABLES| = |FEATURES|
- FEATURES.var ≥ 0
- FEATURES.var ≤ 1
- DEFAULT = 0

#### Purpose

Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of ZIGZAG is identified (even if this occurrence of pattern is not complete) then FEATURES[$i$] is the default value DEFAULT; otherwise FEATURES[$i$] gives the feature value of the corresponding occurrence of ZIGZAG.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression '(<>)*(< | <>) | (<>)*(> | <<)'. The feature ONE, called INDEX in the name of the constraint returns 1.

#### Example

Figure 4.492 provides an example where the INDEX_ZIGZAG ([4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0], 0) constraint holds.

#### Typical

- |VARIABLES| > 3
- range(VARIABLES.var) > 1

#### Arg. properties

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.492: Illustrating the INDEX_ZIGZAG constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
**MAX_BUMP_ON_Decreasing_Sequence**

**Pattern**

\[ > > < > > \]

**Description Automaton**

**Origin**

Based on the *BUMP_ON_Decreasing_Sequence* pattern.

**Constraint**

\[ \text{MAX\_BUMP\_ON\_DECREASING\_SEQUENCE}\left(\text{VARIABLES, FEATURES, DEFAULT}\right) \]

**Arguments**

- **VARIABLES**: \( \text{collection}(\text{var} - \text{dvar}) \)
- **FEATURES**: \( \text{collection}(\text{var} - \text{dvar}) \)
- **DEFAULT**: \( \text{int} \)

**Restrictions**

- \( \text{required}(\text{VARIABLES}.\text{var}) \)
- \( \text{required}(\text{FEATURES}.\text{var}) \)
- \( |\text{VARIABLES}| = |\text{FEATURES}| \)
- \( \text{sv} \leq 5 \lor \text{rv} \leq 2 \Rightarrow \text{FEATURES}.\text{var} = \text{DEFAULT} \)
- \( \text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \geq \text{minv} + 2 \)
- \( \text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \leq \text{maxv} \)
- \( \text{DEFAULT} < \text{minv} + 2 \lor \text{DEFAULT} > \text{maxv} \)

**Purpose**

Consider the time-series given by the **VARIABLES** collection. If position \( i \) does not correspond to the first position in **VARIABLES** where an occurrence of *BUMP_ON_Decreasing_Sequence* is identified then \( \text{FEATURES}[i] \) is the default value **DEFAULT**; otherwise \( \text{FEATURES}[i] \) gives the feature value of the corresponding occurrence of *BUMP_ON_Decreasing_Sequence*.

An occurrence of the pattern *BUMP_ON_Decreasing_Sequence* is the subsequence which matches the regular expression ‘>><<’.

Assume that the occurrence of the pattern *BUMP_ON_Decreasing_Sequence* starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i + 2 \) to index \( j \).

**Example**

Figure 4.493 provides an example where the MAX_BUMP_ON_Decreasing_Sequence \([7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3], [0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 0, 0], 0)\) constraint holds.

**Typical**

- \( |\text{VARIABLES}| > 5 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 2 \)

**Arg. properties**

Functional dependency: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.493: Illustrating the MAX_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
MAX_BUMP_ON_DECREASING_SEQUENCE
**MAX_DECREASING**

**DESCRIPTION**

**Origin**
Based on the **DECREASING** pattern.

**Constraint**

\[
\text{MAX\_DECREASING}(\text{VARIABLES}, \text{FEATURES}, \text{DEFAULT})
\]

**Arguments**

- **VARIABLES**: \(\text{collection}(\text{var}−\text{dvar})\)
- **FEATURES**: \(\text{collection}(\text{var}−\text{dvar})\)
- **DEFAULT**: \(\text{int}\)

**Restrictions**

\[
\begin{align*}
\text{required}(\text{VARIABLES}.\text{var}) \\
\text{required}(\text{FEATURES}.\text{var}) \\
|\text{VARIABLES}| = |\text{FEATURES}| \\
\text{sv} \leq 1 \lor \text{rv} \leq 1 \implies \text{FEATURES}.\text{var} = \text{DEFAULT} \\
\text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \geq \text{minv} + 1 \\
\text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \leq \text{maxv} \\
\text{DEFAULT} < \text{minv} + 1 \lor \text{DEFAULT} > \text{maxv}
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

Consider the time-series given by the **VARIABLES** collection. If position \(i\) does not correspond to the first position in **VARIABLES** where an occurrence of **DECREASING** is identified then **FEATURES**\([i]\) is the default value **DEFAULT**; otherwise **FEATURES**\([i]\) gives the feature value of the corresponding occurrence of **DECREASING**.

**Example**

Figure 4.494 provides an example where the **MAX\_DECREASING** ([3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4], [0, 4, 0, 0, 0, 0, 6, 0, 4, 3, 0, 0, 0, 6, 0, 4]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

Functional dependency: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.494: Illustrating the MAX\_DECREASING constraint of the \textbf{Example} slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
**MAX_DECREASING_SEQUENCE**

<table>
<thead>
<tr>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAX_DECREASING_SEQUENCE</td>
<td></td>
</tr>
</tbody>
</table>

### Description

**Origin**

Based on the **DECREASING_SEQUENCE** pattern.

**Constraint**

\[
\text{MAX\_DECREASING\_SEQUENCE}(\text{VARIABLES}, \text{FEATURES}, \text{DEFAULT})
\]

**Arguments**

- **VARIABLES**: collection(var–dvar)
- **FEATURES**: collection(var–dvar)
- **DEFAULT**: int

**Restrictions**

\[
\begin{align*}
\text{required}(\text{VARIABLES}.\text{var}) \\
\text{required}(\text{FEATURES}.\text{var}) \\
|\text{VARIABLES}| = |\text{FEATURES}| \\
\text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{FEATURES}.\text{var} = \text{DEFAULT} \\
\text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \geq \text{minv} + 1 \\
\text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \leq \text{maxv} \\
\text{DEFAULT} < \text{minv} + 1 \lor \text{DEFAULT} > \text{maxv}
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

Consider the time-series given by the **VARIABLES** collection. If position \(i\) does not correspond to the first position in **VARIABLES** where an occurrence of **DECREASING\_SEQUENCE** is identified (even if this occurrence of pattern is not complete) then **FEATURES**\([i]\) is the default value **DEFAULT**; otherwise **FEATURES**\([i]\) gives the feature value of the corresponding occurrence of **DECREASING\_SEQUENCE**.

An occurrence of the pattern **DECREASING\_SEQUENCE** is the **maximal** subsequence which matches the regular expression ‘\(> (> | =) ^* > |\)’.

Assume that the occurrence of the pattern **DECREASING\_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

Figure 4.495 provides an example where the **MAX\_DECREASING\_SEQUENCE** ([3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 4, 6, 4, 4], [0, 4, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0], 0) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| &> 1 \\
\text{range}(\text{VARIABLES}.\text{var}) &> 1
\end{align*}
\]

**Arg. properties**

**Functional dependency**: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.495: Illustrating the MAX_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
MAX_DECREASING_SEQUENCE

1213
<table>
<thead>
<tr>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAX_DIP_ON_INCREASING_SEQUENCE</td>
<td></td>
</tr>
</tbody>
</table>

**Description Automaton**

**Origin**
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**
```latex
MAX_DIP_ON_INCREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)
```

**Arguments**

```
VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int
```

**Restrictions**

```latex
\text{required}(\text{VARIABLES}.\text{var})
\text{required}(\text{FEATURES}.\text{var})
|\text{VARIABLES}| = |\text{FEATURES}|
sv \leq 5 \lor rv \leq 2 \Rightarrow \text{FEATURES}.\text{var} = \text{DEFAULT}
\text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \geq \text{minv} + 2
\text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \leq \text{maxv}
\text{DEFAULT} < \text{minv} + 2 \lor \text{DEFAULT} > \text{maxv}
```

where

```
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})
sv = |\text{VARIABLES}|
rv = \text{range}(\text{VARIABLES}.\text{var})
```

**Purpose**
Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position where an occurrence of DIP_ON_INCREASING_SEQUENCE is identified then FEATURES[\(i\)] is the default value DEFAULT; otherwise FEATURES[\(i\)] gives the feature value of the corresponding occurrence of DIP_ON_INCREASING_SEQUENCE.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression ‘<<><<’.

Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 2\) to index \(j\).

**Example**

Figure 4.496 provides an example where the MAX_DIP_ON_INCREASING_SEQUENCE ([1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4], [0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 5
\text{range}(\text{VARIABLES}.\text{var}) > 2
```

**Argument properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.496: Illustrating the MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
MAX_DIP_ON_INCREASING_SEQUENCE
Origin
Based on the \textsc{DECREASING\_TERRACE} pattern.

Constraint
\textsc{MAX\_HEIGHT\_DECREASING\_TERRACE}(\textsc{VALUE}, \textsc{VARIABLES})

Arguments
\begin{itemize}
  \item \textsc{VALUE} : \texttt{dvar}
  \item \textsc{VARIABLES} : \texttt{collection(var-dvar)}
\end{itemize}

Restrictions
\begin{align*}
\textsc{sv} \leq 3 & \lor \textsc{rv} \leq 2 \Rightarrow \textsc{VALUE} = -\infty \\
\textsc{VALUE} = -\infty & \lor \textsc{VALUE} \geq \textsc{minv} + 1 \\
\textsc{VALUE} \leq \textsc{maxv} - 1 & \Rightarrow \text{\texttt{required(VARIABLES,var)}}
\end{align*}

where
\begin{itemize}
  \item \textsc{maxv} = \texttt{maxval(VARIABLES.var)}
  \item \textsc{minv} = \texttt{minval(VARIABLES.var)}
  \item \textsc{sv} = |\textsc{VARIABLES}|
  \item \textsc{rv} = \texttt{range(VARIABLES.var)}
\end{itemize}

\textsc{VALUE} is the maximum of all minimum values in each occurrence of the \textsc{DECREASING\_TERRACE} pattern in the time-series given by the \textsc{VARIABLES} collection. If the pattern does not occur, \textsc{VALUE} takes the default value $-\infty$.

An occurrence of the pattern \textsc{DECREASING\_TERRACE} is the maximal subsequence which matches the regular expression ‘$> + >$’.

Assume that the occurrence of the pattern \textsc{DECREASING\_TERRACE} starts at position $i$ and ends at position $j$. The feature \textsc{MIN}, called \textsc{HEIGHT} in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i + 1$ to index $j$.

Example
(4, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3))

Figure 4.497 provides an example where the \textsc{MAX\_HEIGHT\_DECREASING\_TERRACE} (4, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3]) constraint holds.

Typical
|\textsc{VARIABLES}| > 3
\texttt{range(VARIABLES.var)} > 2

Arg. properties
\textbf{Functional dependency}: \textsc{VALUE} determined by \textsc{VARIABLES}. 
Figure 4.497: Illustrating the MAX\_HEIGHT\_DECREASING\_TERRACE constraint of the Example slot

```
MAX\_HEIGHT\_DECREASING\_TERRACE
```

feature values (HEIGHT ≡ MIN)
Figures 4.498 and 4.499 respectively depict the automaton associated with the constraint MAX_HEIGHT_DECREASING_TERRACE and its simplified form.

Figure 4.498: Automaton for the MAX_HEIGHT_DECREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING_TERRACE pattern where default is $-\infty$.

Figure 4.499: Simplified automaton for the MAX_HEIGHT_DECREASING_TERRACE constraint obtained by applying decoration Table 3.39 to the seed transducer of the DECREASING_TERRACE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.1: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the \textsc{Max-Height-Decreasing-Terrace} constraint defined as the composition of the \textsc{Decreasing-Terrace} pattern, the feature \textsc{Min}, and the aggregator \textsc{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\max(C, C))</td>
<td>(\max(C, C))</td>
<td>(\max(C, C))</td>
</tr>
<tr>
<td>(r)</td>
<td>(\max(C, C))</td>
<td>(\max(C, C))</td>
<td>(\min(D, D, \text{VAR}_{i+1}))</td>
</tr>
<tr>
<td>(t)</td>
<td>(\max(C, C))</td>
<td>(\min(D, D, \text{VAR}_{i+1}))</td>
<td>(\min(D, D, \text{VAR}_{i+1}))</td>
</tr>
</tbody>
</table>

Table 4.2: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the simplified automaton of the \textsc{Max-Height-Decreasing-Terrace} constraint defined as the composition of the \textsc{Decreasing-Terrace} pattern, the feature \textsc{Min}, and the aggregator \textsc{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
</tr>
<tr>
<td>(r)</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
<td>(\text{VAR}_{i+1})</td>
</tr>
<tr>
<td>(t)</td>
<td>(-\infty)</td>
<td>(\text{VAR}_{i+1})</td>
<td>(\text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>
MAX_HEIGHT_INCREASING_TERRACE

Origin
Based on the INCREASING_TERRACE pattern.

Constraint
MAX_HEIGHT_INCREASING_TERRACE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
\[sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = -\infty\]
\[VALUE = -\infty \lor VALUE \geq \text{minv} + 1\]
\[VALUE \leq \text{maxv} - 1 \land \text{required}(\text{VARIABLES}, \text{var})\]
where
\[
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
\]
\[
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})
\]
\[
sv = |\text{VARIABLES}|
\]
\[
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

VALUE is the maximum of all minimum values in each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $-\infty$.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression ‘$\langle \leq \rangle$’.

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i + 1$ to index $j$.

Purpose

Example
(5, (1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4))

Figure 4.500 provides an example where the MAX_HEIGHT_INCREASING_TERRACE (5, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4]) constraint holds.

Typical
|VARIABLES| > 3
range(VARIABLES.var) > 2

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.500: Illustrating the MAX_HEIGHT_INCREASING_TERRACE constraint of the Example slot
Figures 4.501 and 4.502 respectively depict the automaton associated with the constraint MAX_HEIGHT_INCREASING_TERRACE and its simplified form.

Figure 4.501: Automaton for the MAX_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_TERRACE pattern where default is \(-\infty\).

Figure 4.502: Simplified automaton for the MAX_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 3.39 to the seed transducer of the INCREASING_TERRACE pattern where default is \(-\infty\); \(R_i - R_{i-1} \geq 0\) is a linear invariant.
Table 4.3: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the MAX_HEIGHT_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\max(C, C)$</td>
<td>$\max(C, C)$</td>
<td>$\max(C, C)$</td>
</tr>
<tr>
<td>r</td>
<td>$\max(C, C)$</td>
<td>$\max(C, C)$</td>
<td>$\min(D, D, \text{VAR}_{i+1})$</td>
</tr>
<tr>
<td>t</td>
<td>$\max(C, C)$</td>
<td>$\min(D, D, \text{VAR}_{i+1})$</td>
<td>$\min(D, D, \text{VAR}_{i+1})$</td>
</tr>
</tbody>
</table>

Table 4.4: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the simplified automaton of the MAX_HEIGHT_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
</tr>
<tr>
<td>r</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
<td>$\text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>t</td>
<td>$-\infty$</td>
<td>$\text{VAR}_{i+1}$</td>
<td>$\text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>
Origin  Based on the PLAIN pattern.

Constraint  MAX_HEIGHTPLAIN(VALUE, VARIABLES)

Arguments  
VALUE : dvar  
VARIABLES : collection(var−dvar)

Restrictions  
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞  
VALUE = −∞ ∨ VALUE ≥ minv  
VALUE ≤ maxv − 1  
required(VARIABLES, var)  
where  
maxv = maxval(VARIABLES.var)  
minv = minval(VARIABLES.var)  
sv = |VARIABLES|  
rv = range(VARIABLES.var)

VALUE is the maximum of all minimum values in each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

Purpose  
An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘> = ∗ <’.  
Assume that the occurrence of the pattern PLAIN starts at position i and ends at position j. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index i + 1 to index j.

Example  
(5, ⟨2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3⟩)

Figure 4.503 provides an example where the MAX_HEIGHTPLAIN(5, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3]) constraint holds.

Typical  
|VARIABLES| > 2  
range(VARIABLES.var) > 1

Symmetry  
Items of VARIABLES can be reversed.

Arg. properties  
Functional dependency: VALUE determined by VARIABLES.
Figure 4.503: Illustrating the MAX_HEIGHTPLAIN constraint of the **Example** slot
Automaton

Figures 4.504 and 4.505 respectively depict the automaton associated with the constraint MAX_HEIGHTPLAIN and its simplified form.

Figure 4.504: Automaton for the MAX_HEIGHTPLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLAIN pattern where default is $-\infty$.

Figure 4.505: Simplified automaton for the MAX_HEIGHTPLAIN constraint obtained by applying decoration Table 3.39 to the seed transducer of the PLAIN pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.5: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the \textsc{MAX} \_\textsc{HEIGHT} \_\textsc{PLAIN} constraint defined as the composition of the \textsc{PLAIN} pattern, the feature \textsc{MIN}, and the aggregator \textsc{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\max(C, C)$</td>
<td>$\max(C, C)$</td>
<td>$\max(C, C)$</td>
</tr>
<tr>
<td>r</td>
<td>$\max(C, C)$</td>
<td>$\min(D, D, \text{VAR}_{i+1})$</td>
<td>$\min(D, D, \text{VAR}_{i+1})$</td>
</tr>
<tr>
<td>t</td>
<td>$\max(C, C)$</td>
<td>$\min(D, D, \text{VAR}_{i+1})$</td>
<td>$\min(D, D, \text{VAR}_{i+1})$</td>
</tr>
</tbody>
</table>

Table 4.6: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the simplified automaton of the \textsc{MAX} \_\textsc{HEIGHT} \_\textsc{PLAIN} constraint defined as the composition of the \textsc{PLAIN} pattern, the feature \textsc{MIN}, and the aggregator \textsc{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
</tr>
<tr>
<td>r</td>
<td>$-\infty$</td>
<td>$\text{VAR}_{i+1}$</td>
<td>$\text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>t</td>
<td>$-\infty$</td>
<td>$\text{VAR}_{i+1}$</td>
<td>$\text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>
### MAX_HEIGHT_PLATEAU

#### Description

**Origin**
Based on the `PLATEAU` pattern.

**Constraint**

\[
\text{MAX\_HEIGHT\_PLATEAU}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`

**Restrictions**

\[
\text{sv} \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1 \\
\text{VALUE} \leq \text{maxv}
\]

where

- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

VALUE is the maximum of all minimum values in each occurrence of the `PLATEAU` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value \(-\infty\). An occurrence of the pattern `PLATEAU` is the maximal subsequence which matches the regular expression `'<=' '>'`. Assume that the occurrence of the pattern `PLATEAU` starts at position \(i\) and ends at position \(j\). The feature `MIN`, called `HEIGHT` in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i+1\) to index \(j\).

**Example**

\[(5, 7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5)\]

Figure 4.506 provides an example where the `MAX\_HEIGHT\_PLATEAU` \((5, 7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5)\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Symmetry**

Items of `VARIABLES` can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.506: Illustrating the MAX_HEIGHT_PLATEAU constraint of the Example slot
Figures 4.507 and 4.508 respectively depict the automaton associated with the constraint MAX_HEIGHT_PLATEAU and its simplified form.  

Figure 4.507: Automaton for the MAX_HEIGHT_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLATEAU pattern where default is $-\infty$.

Figure 4.508: Simplified automaton for the MAX_HEIGHT_PLATEAU constraint obtained by applying decoration Table 3.39 to the seed transducer of the PLATEAU pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.7: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the MAX_HEIGHT_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$r$</td>
<td>$t$</td>
</tr>
<tr>
<td>$s$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1})$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1})$</td>
</tr>
</tbody>
</table>

Table 4.8: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the simplified automaton of the MAX_HEIGHT_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$r$</td>
<td>$t$</td>
</tr>
<tr>
<td>$s$</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
</tr>
<tr>
<td>$r$</td>
<td>$-\infty$</td>
<td>$\text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>$t$</td>
<td>$-\infty$</td>
<td>$\text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>
Origin

Based on the \texttt{PROPER\_PLAIN} pattern.

Constraint

\begin{equation}
\text{MAX\_HEIGHT\_PROPER\_PLAIN}(\text{VALUE}, \text{VARIABLES})
\end{equation}

Arguments

\begin{itemize}
\item \texttt{VALUE} : dvar
\item \texttt{VARIABLES} : collection(var\_dvar)
\end{itemize}

Restrictions

\begin{enumerate}
\item \(sv \leq 3 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty\)
\item \(\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv}\)
\item \(\text{VALUE} \leq \text{maxv} - 1\)
\item \(\text{required}(\text{VARIABLES}, \text{var})\)
\end{enumerate}

where

\begin{itemize}
\item \(\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})\)
\item \(\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})\)
\item \(sv = |\text{VARIABLES}|\)
\item \(rv = \text{range}(\text{VARIABLES}.\text{var})\)
\end{itemize}

Purpose

VALUE is the maximum of all minimum values in each occurrence of the \texttt{PROPER\_PLAIN} pattern in the time-series given by the \texttt{VARIABLES} collection. If the pattern does not occur, \texttt{VALUE} takes the default value \(-\infty\).

An occurrence of the pattern \texttt{PROPER\_PLAIN} is the maximal subsequence which matches the regular expression \(\textquote{\textgreater=+<}\).

Assume that the occurrence of the pattern \texttt{PROPER\_PLAIN} starts at position \(i\) and ends at position \(j\). The feature \texttt{MIN}, called \texttt{HEIGHT} in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

Example

\begin{itemize}
\item \((5, (2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5))\)
\end{itemize}

Figure 4.509 provides an example where the \texttt{MAX\_HEIGHT\_PROPER\_PLAIN} constraint holds.

Typical

\begin{itemize}
\item \(|\text{VARIABLES}| > 3\)
\item \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)
\end{itemize}

Symmetry

Items of \texttt{VARIABLES} can be reversed.

Arg. properties

Functional dependency: \texttt{VALUE} determined by \texttt{VARIABLES}. 
Figure 4.509: Illustrating the MAX_HEIGHT_PROPER_PLAIN constraint of the Example slot
Automaton

Figures 4.510 and 4.511 respectively depict the automaton associated with the constraint MAX_HEIGHT_PROPER_PLAIN and its simplified form.

Figure 4.510: Automaton for the MAX_HEIGHT_PROPER_PLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER_PLAIN pattern where default is $-\infty$.

Figure 4.511: Simplified automaton for the MAX_HEIGHT_PROPER_PLAIN constraint obtained by applying decoration Table 3.39 to the seed transducer of the PROPER_PLAIN pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.9: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the MAX_HEIGHT_PROPER_PLAIN constraint defined as the composition of the PROPER_PLAIN pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

Table 4.10: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the simplified automaton of the MAX_HEIGHT_PROPER_PLAIN constraint defined as the composition of the PROPER_PLAIN pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MAX_HEIGHT_PROPER_PLATEAU**

### Description

**Origin**
Based on the `PROPER_PLATEAU` pattern.

**Constraint**

```
MAX_HEIGHT_PROPER_PLATEAU(VALUE, VARIABLES)
```

**Arguments**

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var−dvar)`

**Restrictions**

- `sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ minv + 1`
- `VALUE ≤ maxv`

where

- `maxv = maxval(VARIABLES.var)`
- `minv = minval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**

VALUE is the maximum of all minimum values in each occurrence of the `PROPER_PLATEAU` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value $−∞$.

An occurrence of the pattern `PROPER_PLATEAU` is the maximal subsequence which matches the regular expression `'<=+=>'`.

Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position $i$ and ends at position $j$. The feature `MIN`, called `HEIGHT` in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i + 1$ to index $j$.

**Example**

```
(5, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))
```

Figure 4.512 provides an example where the `MAX_HEIGHT_PROPER_PLATEAU` (5, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3]) constraint holds.

**Typical**

- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 1`

**Symmetry**

Items of `VARIABLES` can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by `VARIABLES`. 

---

**AGGREGATOR**

**DESCRIPTION**

**AUTOMATON**

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the <code>PROPER_PLATEAU</code> pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td><code>MAX_HEIGHT_PROPER_PLATEAU(VALUE, VARIABLES)</code></td>
</tr>
<tr>
<td>Arguments</td>
<td><code>VALUE</code> : <code>dvar</code>, <code>VARIABLES</code> : <code>collection(var−dvar)</code></td>
</tr>
<tr>
<td>Restrictions</td>
<td><code>sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = −∞</code>, <code>VALUE = −∞ ∨ VALUE ≥ minv + 1</code>, <code>VALUE ≤ maxv</code></td>
</tr>
<tr>
<td>Purpose</td>
<td>VALUE is the maximum of all minimum values in each occurrence of the <code>PROPER_PLATEAU</code> pattern in the time-series given by the <code>VARIABLES</code> collection. If the pattern does not occur, VALUE takes the default value $−∞$. An occurrence of the pattern <code>PROPER_PLATEAU</code> is the maximal subsequence which matches the regular expression <code>'&lt;=+=&gt;'</code>. Assume that the occurrence of the pattern <code>PROPER_PLATEAU</code> starts at position $i$ and ends at position $j$. The feature <code>MIN</code>, called <code>HEIGHT</code> in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i + 1$ to index $j$.</td>
</tr>
<tr>
<td>Example</td>
<td><code>(5, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))</code></td>
</tr>
<tr>
<td>Typical</td>
<td>`</td>
</tr>
<tr>
<td>Symmetry</td>
<td>Items of <code>VARIABLES</code> can be reversed.</td>
</tr>
<tr>
<td>Arg. properties</td>
<td>Functional dependency: VALUE determined by <code>VARIABLES</code>.</td>
</tr>
</tbody>
</table>
Figure 4.512: Illustrating the MAX_HEIGHT_PROPER_PLATEAU constraint of the Example slot
Figures 4.513 and 4.514 respectively depict the automaton associated with the constraint MAX_HEIGHT_PROPER_PLATEAU and its simplified form.

Figure 4.513: Automaton for the MAX_HEIGHT_PROPER_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER_PLATEAU pattern where default is $-\infty$.

Figure 4.514: Simplified automaton for the MAX_HEIGHT_PROPER_PLATEAU constraint obtained by applying decoration Table 3.39 to the seed transducer of the PROPER_PLATEAU pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.11: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the \texttt{MAX_HEIGHT_PROPER_PLATEAU} constraint defined as the composition of the \texttt{PROPER_PLATEAU} pattern, the feature \texttt{MIN}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>\texttt{s}</th>
<th>\texttt{r}</th>
<th>\texttt{t}</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{s}</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>\texttt{r}</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{D}, \overrightarrow{D}, \texttt{VAR}_{i+1}))</td>
</tr>
<tr>
<td>\texttt{t}</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{D}, \overrightarrow{D}, \texttt{VAR}_{i+1}))</td>
<td>(\min(\overrightarrow{D}, \overrightarrow{D}, \texttt{VAR}_{i+1}))</td>
</tr>
</tbody>
</table>

Table 4.12: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the simplified automaton of the \texttt{MAX_HEIGHT_PROPER_PLATEAU} constraint defined as the composition of the \texttt{PROPER_PLATEAU} pattern, the feature \texttt{MIN}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>\texttt{s}</th>
<th>\texttt{r}</th>
<th>\texttt{t}</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{s}</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
</tr>
<tr>
<td>\texttt{r}</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
<td>(\texttt{VAR}_{i+1})</td>
</tr>
<tr>
<td>\texttt{t}</td>
<td>(-\infty)</td>
<td>(\texttt{VAR}_{i+1})</td>
<td>(\texttt{VAR}_{i+1})</td>
</tr>
</tbody>
</table>
MAX_HEIGHT_STEADY

**Origin**
Based on the **STEADY** pattern.

**Constraint**

\[
\text{MAX_HEIGHT_STEADY}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE** : \text{dvar}
- **VARIABLES** : \text{collection(var−dvar)}

**Restrictions**

- \(sv \leq 1 \Rightarrow \text{VALUE} = -\infty\)
- \(\text{VALUE} = -\infty \vee \text{VALUE} \geq \text{minv}\)
- \(\text{VALUE} \leq \text{maxv}\)

where

\[
\text{maxv} = \maxval(\text{VARIABLES.var})
\]
\[
\text{minv} = \minval(\text{VARIABLES.var})
\]
\[
sv = |\text{VARIABLES}|
\]

**Purpose**

\(\text{VALUE}\) is the maximum of all minimum values in each occurrence of the **STEADY** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, \(\text{VALUE}\) takes the default value \(-\infty\).

An occurrence of the pattern **STEADY** is the subsequence which matches the regular expression ‘\(=\)’. Assume that the occurrence of the pattern **STEADY** starts at position \(i\) and ends at position \(j\). The feature **MIN**, called **HEIGHT** in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[(6, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6))\]

Figure 4.515 provides an example where the **MAX_HEIGHT_STEADY** \((6, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6])\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 1\)

**Symmetry**

Items of **VARIABLES** can be reversed.

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.515: Illustrating the MAX_HEIGHT_STEADY constraint of the Example slot
Figures 4.516 and 4.517 respectively depict the automaton associated with the constraint \( \text{MAX\_HEIGHT\_STEADY} \) and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow +\infty \\
& R \leftarrow \text{default} \} \\
\end{align*}
\]

\[
\begin{align*}
\{ & D \leftarrow +\infty \\
& R \leftarrow \max(R, \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1})) \} \\
\end{align*}
\]

Figure 4.516: Automaton for the \( \text{MAX\_HEIGHT\_STEADY} \) constraint obtained by applying decoration Table 3.37 to the seed transducer of the \( \text{STEADY} \) pattern where default is \(-\infty\).

\[
\begin{align*}
\{ & R \leftarrow \text{default} \} \\
\end{align*}
\]

\[
\begin{align*}
\{ & R \leftarrow \max(R, \text{VAR}_i) \} \\
\end{align*}
\]

Figure 4.517: Simplified automaton for the \( \text{MAX\_HEIGHT\_STEADY} \) constraint obtained by applying decoration Table 3.39 to the seed transducer of the \( \text{STEADY} \) pattern where default is \(-\infty\); \( R_i - R_{i-1} \geq 0 \) is a linear invariant.

Table 4.13: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the \( \text{MAX\_HEIGHT\_STEADY} \) constraint defined as the composition of the \( \text{STEADY} \) pattern, the feature \( \text{MIN} \), and the aggregator \( \text{max} \); cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.14: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the simplified automaton of the MAX_HEIGHT_STEADY constraint defined as the composition of the STEADY pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin
Based on the STEADY_SEQUENCE pattern.

Constraint
MAX_HEIGHT_STEADY_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var-dvar)

Restrictions
sv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv
required(VARIABLES, var)
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|

VALUE is the maximum of all minimum values in each occurrence of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression ‘=+’.

Purpose
Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i$ to index $j + 1$.

Example
(5, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1])

Figure 4.518 provides an example where the MAX_HEIGHT_STEADY_SEQUENCE (5, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1]) constraint holds.

Typical
|VARIABLES| > 1

Symmetry
Items of VARIABLES can be reversed.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.518: Illustrating the MAX_HEIGHT_STEADY_SEQUENCE constraint of the Example slot.
Figures 4.519 and 4.520 respectively depict the automaton associated with the constraint \textsc{Max\_Height\_Steady\_Sequence} and its simplified form.

**Automaton**

\[
\begin{align*}
    \{ & C \leftarrow \text{default} \\
    & D \leftarrow +\infty \\
    & R \leftarrow \text{default} \}
\end{align*}
\]

Figure 4.519: Automaton for the \textsc{Max\_Height\_Steady\_Sequence} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsc{Steady\_Sequence} pattern where \texttt{default} is \(-\infty\).

\[
\begin{align*}
    \{ & C \leftarrow \text{default} \\
    & D \leftarrow +\infty \\
    & R \leftarrow \max(R, C) \}
\end{align*}
\]

\[
\begin{align*}
    \{ & C \leftarrow \min\left(\min(D, \texttt{VAR}_i), \texttt{VAR}_{i+1}\right) \\
    & D \leftarrow +\infty \}
\end{align*}
\]

\[
\begin{align*}
    \{ & C \leftarrow \min(C, \min(D, \texttt{VAR}_{i+1})) \\
    & D \leftarrow +\infty \}
\end{align*}
\]

Figure 4.520: Simplified automaton for the \textsc{Max\_Height\_Steady\_Sequence} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \textsc{Steady\_Sequence} pattern where \texttt{default} is \(-\infty\); \(R_i - R_{i-1} \geq 0\) is a linear invariant.
Table 4.15: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the MAX_HEIGHT_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
</tr>
<tr>
<td>r</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
<td>min(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D})</td>
</tr>
</tbody>
</table>

Table 4.16: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the simplified automaton of the MAX_HEIGHT_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
</tr>
<tr>
<td>r</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
</tr>
</tbody>
</table>
### Description

**Origin**
Based on the `INCREASING` pattern.

**Constraint**

$$\text{MAX\_INCREMENTING}(\text{VARIABLES, FEATURES, DEFAULT})$$

**Arguments**

- **VARIABLES**: `collection(var-dvar)`
- **FEATURES**: `collection(var-dvar)`
- **DEFAULT**: `int`

**Restrictions**

- `required(\text{VARIABLES}.\text{var})`
- `required(\text{FEATURES}.\text{var})`
- \(|\text{VARIABLES}| = |\text{FEATURES}|
- \(sv \leq 1 \vee rv \leq 1 \Rightarrow \text{FEATURES}.\text{var} = \text{DEFAULT} \)
- \(\text{FEATURES}.\text{var} = \text{DEFAULT} \vee \text{FEATURES}.\text{var} \geq \text{minv} + 1 \)
- \(\text{FEATURES}.\text{var} = \text{DEFAULT} \vee \text{FEATURES}.\text{var} \leq \text{maxv} \)
- \(\text{DEFAULT} < \text{minv} + 1 \vee \text{DEFAULT} > \text{maxv} \)

Where

- \(\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})\)
- \(\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})\)
- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range}(\text{VARIABLES}.\text{var})\)

### Purpose

Consider the time-series given by the `VARIABLES` collection. If position \(i\) does not correspond to the first position in `VARIABLES` where an occurrence of `INCREASING` is identified then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `INCREASING`.

### Example

Figure 4.521 provides an example where the `MAX\_INCREMENTING` constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- Functional dependency: `FEATURES` determined by `VARIABLES` and `DEFAULT`. 
Figure 4.521: Illustrating the MAX_INCREASING constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
Origin
Based on the INCREASING_SEQUENCE pattern.

Constraint
MAX_INCREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)

Arguments
VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int

Restrictions
required(VARIABLES, var)
required(FEATURES, var)
|VARIABLES| = |FEATURES|
sv ≤ 1 ∨ rv ≤ 1 ⇒ FEATURES.var = DEFAULT
FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv + 1
FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv
DEFAULT < minv + 1 ∨ DEFAULT > maxv
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
Consider the time-series given by the VARIABLES collection. If position i does not correspond to the first position in VARIABLES where an occurrence of INCREASING_SEQUENCE is identified (even if this occurrence of pattern is not complete) then FEATURES[i] is the default value DEFAULT; otherwise FEATURES[i] gives the feature value of the corresponding occurrence of INCREASING_SEQUENCE.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘<( < | =)* < |’.

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i to index j + 1.

Example
Figure 4.522 provides an example where the MAX_INCREASING_SEQUENCE ([4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 3, 1, 3, 3], [0, 5, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 3, 0, 0], 0) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES.var) > 1

Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.522: Illustrating the MAX_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
MAX_INFLEXION

**DESCRIPTION**

**Origin**
Based on the INFLEXION pattern.

**Constraint**
MAX_INFLEXION(VARIABLES, FEATURES, DEFAULT)

**Arguments**
- VARIABLES: collection(var–dvar)
- FEATURES: collection(var–dvar)
- DEFAULT: int

**Restrictions**
- required(VARIABLES, var)
- required(FEATURES, var)
- |VARIABLES| = |FEATURES|
- sv ≤ 2 ∨ rv ≤ 1 ⇒ FEATURES.var = DEFAULT
- FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv
- FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv
- DEFAULT < minv ∨ DEFAULT > maxv

where
- maxv = maxval(VARIABLES.var)
- minv = minval(VARIABLES.var)
- sv = |VARIABLES|
- rv = range(VARIABLES.var)

Consider the time-series given by the VARIABLES collection. If position i does not correspond to the first position in VARIABLES where an occurrence of INFLEXION is identified then FEATURES[i] is the default value DEFAULT; otherwise FEATURES[i] gives the feature value of the corresponding occurrence of INFLEXION.

**Purpose**
An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression '< (| | =)* > | > (> | | =)* '<.

Assume that the occurrence of the pattern INFLEXION starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 1 to index j.

**Example**
Figure 4.523 provides an example where the MAX_INFLEXION ([1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 4, 4], [0, 0, 0, 6, 0, 0, 4, 5, 2, 5, 1, 5, 0, 3, 0, 0], 0) constraint holds.

**Typical**
- |VARIABLES| > 2
- range(VARIABLES.var) > 1

**Arg. properties**
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.523: Illustrating the MAX_INFLEXION constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
MAX_MAX_BUMP_ON_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the `BUMP_ON_DECREASING_SEQUENCE` pattern.

**Constraint**

\[
\text{MAX\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var\_dvar)`

**Restrictions**

\[sv \leq 5 \lor rv \leq 2 \Rightarrow \text{VALUE} = -\infty\]
\[\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 2\]
\[\text{VALUE} \leq \text{maxv} \text{required}(\text{VARIABLES}.\text{var})\]

where

\[\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})\]
\[\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})\]
\[sv = |\text{VARIABLES}|\]
\[rv = \text{range}(\text{VARIABLES}.\text{var})\]

**Purpose**

\[\text{VALUE}\] is the maximum of all maximum values in each occurrence of the `BUMP_ON_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, \[\text{VALUE}\] takes the default value \[-\infty\].

An occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` is the subsequence which matches the regular expression `’>><>>’`. Assume that the occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature \text{MAX} computes the maximum of the values from index \(i + 2\) to index \(j\).

**Example**

\[(6, (7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3))\]

Figure 4.524 provides an example where the \text{MAX\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE} (6, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3]) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 5\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 2\]

**Arg. properties**

Functional dependency: \[\text{VALUE}\] determined by \text{VARIABLES}.
Figure 4.524: Illustrating the MAX_MAX_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.525 and 4.526 respectively depict the automaton associated with the constraint MAX_MAX_BUMP_ON_DECREASING_SEQUENCE and its simplified form.

Figure 4.525: Automaton for the MAX_MAX_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $-\infty$. 

\[
\begin{aligned}
\{ C \leftarrow \text{default} \\ D \leftarrow -\infty \\ R \leftarrow \text{default} \} & \xrightarrow{\leq} \equiv & \{ D \leftarrow \max(D, \text{VAR}_i) \} \leq \{ D \leftarrow -\infty \} \\
\{ D \leftarrow -\infty \} & \xrightarrow{\leq} \{ D \leftarrow \max(D, \text{VAR}_i) \} \leq \{ D \leftarrow -\infty \} \\
\end{aligned}
\]
Figure 4.526: Simplified automaton for the MAX_MAX_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.28 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
MAX_MAX_DECREASING

DESCRIPTION

Origin
Based on the DECREASING pattern.

Constraint
MAX_MAX_DECREASING(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv + 1
VALUE ≤ maxv
required(VARIABLES, var)
where
maxv = maxval(VARIABLES, var)
minv = minval(VARIABLES, var)
sv = |VARIABLES|
r = range(VARIABLES, var)

VALUE is the maximum of all maximum values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

Purpose
An occurrence of the pattern DECREASING is the subsequence which matches the regular expression ‘>’. Assume that the occurrence of the pattern DECREASING starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i to index j + 1.

Example
(6, (3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))

Figure 4.527 provides an example where the MAX_MAX_DECREASING (6, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES, var) > 1

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.527: Illustrating the MAX_MAX_DECREASING constraint of the \textbf{Example} slot
Automaton

Figures 4.528 and 4.529 respectively depict the automaton associated with the constraint \texttt{MAX\_MAX\_DECREASING} and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow -\infty \\
& R \leftarrow \text{default} \} \\
\{ & D \leftarrow -\infty \\
& R \leftarrow \max(R, \max(\max(D, \text{VAR}_i), \text{VAR}_i+1)) \}
\end{align*}
\]

\[
\begin{array}{c}
\downarrow \\
\preceq \ s \\
\max(R, C)
\end{array}
\]

\[
\begin{align*}
\{ & R \leftarrow \text{default} \\
& \} \\
\{ & R \leftarrow \max(R, \max(\text{VAR}_i)) \}
\end{align*}
\]

\[
\begin{array}{c}
\downarrow \\
\preceq \ s \\
R
\end{array}
\]

Figure 4.528: Automaton for the \texttt{MAX\_MAX\_DECREASING} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{DECREASING} pattern where default is \(-\infty\).

Figure 4.529: Simplified automaton for the \texttt{MAX\_MAX\_DECREASING} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \texttt{DECREASING} pattern where default is \(-\infty\); \(R_i - R_{i-1} \geq 0\) is a linear invariant.

Table 4.17: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the \texttt{MAX\_MAX\_DECREASING} constraint defined as the composition of the \texttt{DECREASING} pattern, the feature \texttt{MAX}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.18: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the simplified automaton of the MAX_MAX_DECREASING constraint defined as the composition of the DECREASING pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_MAX_DECREASING_SEQUENCE

**Origin**
Based on the **DECREASING_SEQUENCE** pattern.

**Constraint**
MAX_MAX_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var - dvar)

**Restrictions**
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv + 1
VALUE ≤ maxv
required(VARIABLES, var)
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

**Purpose**
VALUE is the maximum of all maximum values in each occurrence of the **DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

An occurrence of the pattern **DECREASING_SEQUENCE** is the maximal subsequence which matches the regular expression ‘>(> | =)∗ > |’.
Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i to index j + 1.

**Example**
(6, (3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))

Figure 4.530 provides an example where the MAX_MAX_DECREASING_SEQUENCE (6, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

**Typical**
|VARIABLES| > 1
range(VARIABLES.var) > 1

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.530: Illustrating the MAX_MAX_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.531 and 4.532 respectively depict the automaton associated with the constraint MAX_MAX_DECREASING_SEQUENCE and its simplified form.

Figure 4.531: Automaton for the MAX_MAX_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING_SEQUENCE pattern where default is $-\infty$.

Figure 4.532: Simplified automaton for the MAX_MAX_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.39 to the seed transducer of the DECREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.19: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the MAX_MAX_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>(t)</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}))</td>
</tr>
</tbody>
</table>

Table 4.20: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the simplified automaton of the MAX_MAX_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
</tr>
<tr>
<td>(t)</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
</tr>
</tbody>
</table>
MAX_MAX_DIP_ON_INCREASING_SEQUENCE

**Description**

Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**

\[
\text{MAX\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv \leq 5 \lor rv \leq 2 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \minv + 2 \\
\text{VALUE} \leq \maxv & \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{where} \\
\maxv &= \text{maxval}(\text{VARIABLES}, \text{var}) \\
\minv &= \text{minval}(\text{VARIABLES}, \text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}, \text{var})
\end{align*}
\]

**Purpose**

VALUE is the maximum of all maximum values in each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $-\infty$.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression ‘<<><<’.

Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i+2$ to index $j$.

**Example**

\[(6, 1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4)\]

Figure 4.533 provides an example where the MAX_MAX_DIP_ON_INCREASING_SEQUENCE (6, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 5 \\
\text{range}(\text{VARIABLES}, \text{var}) > 2
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.533: Illustrating the MAX_MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.534 and 4.535 respectively depict the automaton associated with the constraint MAX_MAX_DIP_ON_INCREASING_SEQUENCE and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow -\infty \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\geq s \quad < R \quad < t
\]

Figure 4.534: Automaton for the MAX_MAX_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is $-\infty$. 
Figure 4.535: Simplified automaton for the MAX_MAX_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.26 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Origin
Based on the INCREASING pattern.

Constraint
\[ \text{MAX}\_\text{MAX}\_\text{INCREASING}(\text{VALUE}, \text{VARIABLES}) \]

Arguments
\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} \rightarrow \text{dvar})
\end{align*}
\]

Restrictions
\[
\begin{align*}
sv & \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq \text{minv} + 1 \\
\text{VALUE} & \leq \text{maxv} \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where
\[
\begin{align*}
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Purpose
VALUE is the maximum of all maximum values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression ‘\(<\)’.

Assume that the occurrence of the pattern INCREASING starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j+1\).

Example
\((6, 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)\)

Typical
\[
\begin{align*}
|\text{VARIABLES}| & > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.536: Illustrating the MAX_MAX_INCREASING constraint of the Example slot
Figures 4.537 and 4.538 respectively depict the automaton associated with the constraint MAX_MAX_INCREASING and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow -\infty \\
& R \leftarrow \text{default} \} \\
\{ & D \leftarrow -\infty \\
& R \leftarrow \max(R, \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1})) \}
\end{align*}
\]

Figure 4.537: Automaton for the MAX_MAX_INCREASING constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING pattern where default is $-\infty$.

\[
\begin{align*}
\{ & R \leftarrow \text{default} \\
& \{ R \leftarrow \max(R, \max(\text{VAR}_i, \text{VAR}_{i+1})) \} \\
\}
\end{align*}
\]

Figure 4.538: Simplified automaton for the MAX_MAX_INCREASING constraint obtained by applying decoration Table 3.40 to the seed transducer of the INCREASING pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 4.21: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the MAX_MAX_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.22: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the simplified automaton of the MAX_MAX_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_MAX_INCREASING_SEQUENCE

**Origin**
Based on the INCREASING_SEQUENCE pattern.

**Constraint**
MAX_MAX_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \]
\[ VALUE = -\infty \lor VALUE \geq minv + 1 \]
\[ VALUE \leq maxv \]
required(VARIABLES, var)

where
- \( maxv = \maxval(VARIABLES, var) \)
- \( minv = \minval(VARIABLES, var) \)
- \( sv = |VARIABLES| \)
- \( rv = \range(VARIABLES, var) \)

**Purpose**
\( VALUE \) is the maximum of all maximum values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \( VALUE \) takes the default value \(-\infty\).

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \'\(< (< | =)\)^\ast | < \' .
Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i \) to index \( j + 1 \).

**Example**
\( (6, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)) \)

Figure 4.539 provides an example where the MAX_MAX_INCREASING_SEQUENCE (6, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

**Typical**
\( |VARIABLES| > 1 \)
\( \range(VARIABLES, var) > 1 \)

**Arg. properties**
Functional dependency: \( VALUE \) determined by VARIABLES.
Figure 4.539: Illustrating the MAX_MAX_INCREASING_SEQUENCE constraint of the Example slot
Figures 4.540 and 4.541 respectively depict the automaton associated with the constraint MAX_MAX_INCREASING_SEQUENCE and its simplified form.

Figure 4.540: Automaton for the MAX_MAX_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $-\infty$.

Figure 4.541: Simplified automaton for the MAX_MAX_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.40 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.23: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the MAX_MAX_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max(\text{C}, \text{C})</td>
<td>max(\text{C}, \text{C})</td>
</tr>
<tr>
<td>t</td>
<td>max(\text{C}, \text{C})</td>
<td>max(\text{C}, \text{C}, \text{D}, \text{D})</td>
</tr>
</tbody>
</table>

Table 4.24: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the simplified automaton of the MAX_MAX_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
</tr>
<tr>
<td>t</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
</tr>
</tbody>
</table>
Origin | Based on the INFLEXION pattern.
---|---
Constraint | MAX_MAX_INFLEXION(VALUE, VARIABLES)

Arguments

| VALUE | dvar |
| VARIABLES | collection(var−dvar) |

Restrictions

\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
VALUE = -\infty \lor VALUE \geq \text{minv} \\
VALUE \leq \text{maxv}\textbf{b} \\
\text{required}(\text{VARIABLES}.\text{var})
\]

where

\[
\text{maxv} = \maxval(\text{VARIABLES}.\text{var}) \\
\text{minv} = \minval(\text{VARIABLES}.\text{var}) \\
sv = |\text{VARIABLES}| \\
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

Purpose

VALUE is the maximum of all maximum values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (< | =) > | > (>| =) )* <\)'.

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

Example

\[(6, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))\]

Figure 4.542 provides an example where the MAX_MAX_INFLEXION \((6, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])\) constraint holds.

Typical

\[|\text{VARIABLES}| > 2 \lor \text{range}(\text{VARIABLES}.\text{var}) > 1\]

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.542: Illustrating the MAX_MAX_INFLEXION constraint of the Example slot
Figures 4.543 and 4.544 respectively depict the automaton associated with the constraint MAX_MAX_INFLEXION and its simplified form.

Figure 4.543: Automaton for the MAX_MAX_INFLEXION constraint obtained by applying decoration Table 3.37 to the seed transducer of the INFLEXION pattern where default is $-\infty$ (transition $r \rightarrow t$ has the same registers updates as transition $t \rightarrow r$).

Figure 4.544: Simplified automaton for the MAX_MAX_INFLEXION constraint obtained by applying decoration Table 3.26 to the seed transducer of the INFLEXION pattern where default is $-\infty$ (transition $r \rightarrow t$ has the same registers updates as transition $t \rightarrow r$); $R_i - R_{i-1} \geq 0$ is a linear invariant.
Aggregator | Feature | Pattern
---|---|---
**MAX_MAX_PEAK**

**DESCRIPTION**

**Origin**
Based on the **PEAK** pattern.

**Constraint**

\[
\text{MAX_MAX_PEAK}(\text{VALUE, VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} &: \text{dvar} \\
\text{VARIABLES} &: \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} &\leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} &\geq -\infty \lor \text{VALUE} \geq \text{minv} + 1 \\
\text{VALUE} &\leq \text{maxv} \\
\text{required}(\text{VARIABLES}.\text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern **PEAK** is the maximal subsequence which matches the regular expression `\(< (= | <) (>| =)* >\)`. Assume that the occurrence of the pattern **PEAK** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\((6, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1])\)

Figure 4.545 provides an example where the \textbf{MAX_MAX_PEAK} \((6, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1])\) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| &> 2 \\
\text{range}(\text{VARIABLES}.\text{var}) &> 1
\end{align*}
\]

**Symmetry**
Items of \text{VARIABLES} can be reversed.

**Arg. properties**

Functional dependency: \text{VALUE} determined by \text{VARIABLES}. 
Figure 4.545: Illustrating the MAX_MAX_PEAK constraint of the Example slot
Figures 4.546 and 4.547 respectively depict the automaton associated with the constraint MAX_MAX_PEAK and its simplified form.

Figure 4.546: Automaton for the MAX_MAX_PEAK constraint obtained by applying decoration Table 3.37 to the seed transducer of the PEAK pattern where default is $-\infty$.

Figure 4.547: Simplified automaton for the MAX_MAX_PEAK constraint obtained by applying decoration Table 3.39 to the seed transducer of the PEAK pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.25: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the MAX_MAX_PEAK constraint defined as the composition of the PEAK pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{D}, \vec{D}, \text{VAR}_{i+1})$</td>
<td>$\max(\vec{C}, \vec{D}, \vec{VAR}_{i+1})$</td>
</tr>
<tr>
<td>t</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{C}, \vec{D}, \vec{VAR}_{i+1})$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
</tr>
</tbody>
</table>

Table 4.26: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the simplified automaton of the MAX_MAX_PEAK constraint defined as the composition of the PEAK pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
</tr>
<tr>
<td>r</td>
<td>$-\infty$</td>
<td>$\text{VAR}_{i+1}$</td>
<td>$-\infty$</td>
</tr>
<tr>
<td>t</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
</tr>
</tbody>
</table>
MAX_MAX_STRICTLY_DECREASING_SEQUENCE

**Description**

- **Origin**: Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

- **Constraint**: `MAX_MAX_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)`

- **Arguments**:
  - `VALUE`: `dvar`
  - `VARIABLES`: `collection(var−dvar)`

- **Restrictions**:

  - `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = −∞`
  - `VALUE = −∞ ∨ VALUE ≥ minv + 1`
  - `VALUE ≤ maxv`
  - `required(VARIABLES, var)`

  Where:
  - `maxv = maxval(VARIABLES, var)`
  - `minv = minval(VARIABLES, var)`
  - `sv = |VARIABLES|`
  - `rv = range(VARIABLES, var)`

**Purpose**

VALUE is the maximum of all maximum values in each occurrence of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value $-\infty$.

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the *maximal* sub-sequence which matches the regular expression `'>+'`.

Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `MAX` computes the maximum of the values from index $i$ to index $j + 1$.

**Example**

`(6, [4, 4, 6, 4, 1, 3, 4, 6, 6, 5, 2, 2, 4, 3])`

Figure 4.548 provides an example where the `MAX_MAX_STRICTLY_DECREASING_SEQUENCE` `(6, [4, 4, 6, 4, 1, 3, 4, 6, 6, 5, 2, 2, 4, 3])` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES, var) > 1`

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.548: Illustrating the MAX_MAX_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.549 and 4.550 respectively depict the automaton associated with the constraint \texttt{MAX\_\texttt{MAX\_\texttt{STRICTLY\_DECREASING\_SEQUENCE}} and its simplified form.

\[
\begin{aligned}
\{ & C \leftarrow \text{default} \\ & D \leftarrow -\infty \\ & R \leftarrow \text{default} \} \\
\end{aligned}
\]

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure4.549}
\caption{Automaton for the \texttt{MAX\_\texttt{MAX\_\texttt{STRICTLY\_DECREASING\_SEQUENCE}} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is $-\infty$}
\end{figure}

\[
\begin{aligned}
\{ & C \leftarrow \text{default} \\ & D \leftarrow -\infty \\ & R \leftarrow \text{max}(R, C) \} \\
\end{aligned}
\]

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure4.549}
\caption{Simplified automaton for the \texttt{MAX\_\texttt{MAX\_\texttt{STRICTLY\_DECREASING\_SEQUENCE}} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.}
\end{figure}
Table 4.27: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the MAX_MAX_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}))</td>
</tr>
</tbody>
</table>

Table 4.28: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the MAX_MAX_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
</tr>
<tr>
<td>r</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
</tr>
</tbody>
</table>
### MAX_MAX_STRICTLY_INCREASING_SEQUENCE

#### Description

**Origin**
Based on the `STRICTLY_INCREASING_SEQUENCE` pattern.

**Constraint**

\[
\text{MAX_MAX_STRICTLY_INCREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

**Restrictions**

\[
sv \leq 1 \vee rv \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \vee \text{VALUE} \geq \text{minv} + 1 \\
\text{VALUE} \leq \text{maxv} \Rightarrow \text{required}(\text{VARIABLES}.\text{var})
\]

where

- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

VALUE is the maximum of all maximum values in each occurrence of the `STRICTLY_INCREASING_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the maximal sub-sequence which matches the regular expression ‘\(<\)’.

Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[ (6, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) \]

Figure 4.551 provides an example where the `MAX_MAX_STRICTLY_INCREASING_SEQUENCE` constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.551: Illustrating the `MAX_MAX_STRICTLY_INCREASING_SEQUENCE` constraint of the Example slot
Automaton

Figures 4.552 and 4.553 respectively depict the automaton associated with the constraint MAX_MAX_STRICTLY_INCREASING_SEQUENCE and its simplified form.

Figure 4.552: Automaton for the MAX_MAX_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is $-\infty$.

Figure 4.553: Simplified automaton for the MAX_MAX_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.40 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.29: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the MAX_MAX_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
</tbody>
</table>

Table 4.30: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the simplified automaton of the MAX_MAX_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$-\infty$</td>
</tr>
<tr>
<td>r</td>
<td>$-\infty$</td>
</tr>
</tbody>
</table>
### MAX_MAX_SUMMIT

**Origin**
Based on the SUMMIT pattern.

**Constraint**
\[
\text{MAX\_MAX\_SUMMIT}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var - dvar)`

**Restrictions**
- \(sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty\)
- \(\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1\)
- \(\text{VALUE} \leq \text{maxv}\)
- \(\text{required(\text{VARIABLES}.var)}\)

where
- \(\text{maxv} = \text{maxval(\text{VARIABLES}.var)}\)
- \(\text{minv} = \text{minval(\text{VARIABLES}.var)}\)
- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range(\text{VARIABLES}.var)}\)

**Purpose**
VALUE is the maximum of all maximum values in each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \(\text{VALUE}\) takes the default value \(-\infty\).

An occurrence of the pattern `SUMMIT` is the maximal subsequence which matches the regular expression `\((< | < (= | <)^* <) (> | > (= | >)^* >)\)`.

Assume that the occurrence of the pattern `SUMMIT` starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**
\((5, (7, 1, 5, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1))\)

Figure 4.554 provides an example where the `MAX\_MAX\_SUMMIT` \((5, [7, 1, 5, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1])\) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2\)
- \(\text{range(\text{VARIABLES}.var)} > 1\)

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: \(\text{VALUE}\) determined by VARIABLES.
Figure 4.554: Illustrating the MAX_MAX_SUMMIT constraint of the Example slot
Automaton

Figures 4.555 and 4.556 respectively depict the automaton associated with the constraint MAX_MAX_SUMMIT and its simplified form.

Figure 4.555: Automaton for the MAX_MAX_SUMMIT constraint obtained by applying decoration Table 3.37 to the seed transducer of the SUMMIT pattern where default is $-\infty$ (transition $r \to u$ has the same register update as transition $r \to u$)

Table 4.31: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the MAX_MAX_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.556: Simplified automaton for the MAX_MAX_SUMMIT constraint obtained by applying decoration Table 3.39 to the seed transducer of the SUMMIT pattern where default is $-\infty$ (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$); $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 4.32: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the simplified automaton of the MAX_MAX_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin Based on the ZIGZAG pattern.

Constraint \( \text{MAX\_MAX\_ZIGZAG}(\text{VALUE}, \text{VARIABLES}) \)

Arguments

- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection}(\text{var} \mid \text{dvar}) \)

Restrictions

- \( \text{sv} \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \)
- \( \text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1 \)
- \( \text{VALUE} \leq \text{maxv} \)
- \( \text{required}(\text{VARIABLES}.\text{var}) \)

where

- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \)

Purpose

\( \text{VALUE} \) is the maximum of all maximum values in each occurrence of the ZIGZAG pattern in the time-series given by the \( \text{VARIABLES} \) collection. If the pattern does not occur, \( \text{VALUE} \) takes the default value \(-\infty\).

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \( (<>)^*(< | <>)(<>)^*(> | >>) \).

Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i + 1 \) to index \( j \).

Example

\[ (7, \langle 4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1 \rangle) \]

Figure 4.557 provides an example where the \( \text{MAX\_MAX\_ZIGZAG} \) \( (7, \langle 4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1 \rangle) \) constraint holds.

Typical

- \( |\text{VARIABLES}| > 3 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

Symmetry Items of \( \text{VARIABLES} \) can be reversed.

Arg. properties Functional dependency: \( \text{VALUE} \) determined by \( \text{VARIABLES} \).
Figure 4.557: Illustrating the MAX_MAX_ZIGZAG constraint of the Example slot
Figures 4.558 and 4.559 respectively depict the automaton associated with the constraint MAX_MAX_ZIGZAG and its simplified form.
Figure 4.558: Automaton for the MAX_MAX_ZIGZAG constraint obtained by applying decoration Table 3.37 to the seed transducer of the ZIGZAG pattern where default is $-\infty$: (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.
Figure 4.559: Simplified automaton for the MAX_MAX_ZIGZAG constraint obtained by applying decoration Table 3.29 to the seed transducer of the ZIGZAG pattern where default is $-\infty$; (1) missing transitions from a, b, c, d, e, f to s are labelled by $=$; (2) on transitions from b, c, e, f to s the register D is reset to its initial value; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.33: Concrete glue matrix, derived from the parameterised glue matrix 3.22, for the MAX-MAX-ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
</tr>
<tr>
<td>a</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
</tr>
<tr>
<td>b</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
</tr>
<tr>
<td>c</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
</tr>
<tr>
<td>d</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
</tr>
<tr>
<td>e</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
</tr>
<tr>
<td>f</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
<td>max((L, C))</td>
</tr>
</tbody>
</table>
Table 4.34: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the simplified automaton of the MAX\_MAX\_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_MIN_BUMP_ON_DECREASING_SEQUENCE

Description

Origin
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

Constraint
MAX_MIN_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
SV ≤ 5 ∨ RV ≤ 2 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv − 2
required(VARIABLES, var)

where
maxv = maxval(VARIABLES, var)
minv = minval(VARIABLES, var)
SV = |VARIABLES|
RV = range(VARIABLES, var)

Purpose
VALUE is the maximum of all minimum values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞. An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence ‘><><>’.

Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 2 to index j.

Example
(5, ⟨7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3⟩)

Figure 4.560 provides an example where the MAX_MIN_BUMP_ON_DECREASING_SEQUENCE (5, ⟨7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3⟩) constraint holds.

Typical
|VARIABLES| > 5
range(VARIABLES, var) > 2

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.560: Illustrating the MAX_MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Figures 4.561 and 4.562 respectively depict the automaton associated with the constraint MAX_MIN_BUMP_ON_DECREASING_SEQUENCE and its simplified form.

Figure 4.561: Automaton for the MAX_MIN_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $-\infty$. 

\[ \begin{align*} 
&\{ \begin{array}{c} C \leftarrow \text{default} \\ D \leftarrow +\infty \\ R \leftarrow \text{default} \end{array} \} \rightarrow \leq s \\
&\{ D \leftarrow \max(R, \min(D, \text{VAR}_i)) \} \rightarrow \leq v \\
&\{ D \leftarrow \min(D, \text{VAR}_i) \} \rightarrow > r \\
&\{ D \leftarrow +\infty \} \rightarrow \geq \ u \\
&\{ D \leftarrow \min(D, \text{VAR}_i) \} \rightarrow \leq \ max(R, C) \\
&\{ D \leftarrow +\infty \} \rightarrow > t \\
&\{ D \leftarrow \min(D, \text{VAR}_i) \} \rightarrow \leq \ R \leftarrow \max(R, \min(D, \text{VAR}_i)) \\
&\end{align*} \]
Figure 4.562: Simplified automaton for the \texttt{MAX\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.26 to the seed transducer of the \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} pattern where $\texttt{default}$ is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
**MAX_MIN_DECREASING**

**Origin**
Based on the **DECREASING** pattern.

**Constraint**
```
MAX_MIN_DECREASING(VALUE, VARIABLES)
```

**Arguments**
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

**Restrictions**
- \(sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = −\infty\)
- \(VALUE = −\infty \lor VALUE \geq minv\)
- \(VALUE \leq maxv − 1\)
- \(required(VARIABLES, var)\)

where
- \(maxv = maxval(VARIABLES.var)\)
- \(minv = minval(VARIABLES.var)\)
- \(sv = |VARIABLES|\)
- \(rv = range(VARIABLES.var)\)

**Purpose**
VALUE is the maximum of all minimum values in each occurrence of the **DECREASING** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression ‘\(>\)’.
Assume that the occurrence of the pattern **DECREASING** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i\) to index \(j+1\).

**Example**
\((4, (3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\)

Figure 4.563 provides an example where the **MAX_MIN_DECREASING** \((4, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4])\) constraint holds.

**Typical**
- \(|VARIABLES| > 1\)
- \(range(VARIABLES.var) > 1\)

**Arg. properties**
- **Functional dependency**: VALUE determined by **VARIABLES**.
Figure 4.563: Illustrating the MAX_MIN_DECREASING constraint of the Example slot
Figures 4.564 and 4.565 respectively depict the automaton associated with the constraint `MAX_MIN_DECREASING` and its simplified form.

**Automaton**

Figures 4.564: Automaton for the `MAX_MIN_DECREASING` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `DECREASING` pattern where default is $-\infty$.

Figures 4.565: Simplified automaton for the `MAX_MIN_DECREASING` constraint obtained by applying decoration Table 3.40 to the seed transducer of the `DECREASING` pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 4.35: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the `MAX_MIN_DECREASING` constraint defined as the composition of the `DECREASING` pattern, the feature `MIN`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.36: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the simplified automaton of the MAX_MIN_DECREASING constraint defined as the composition of the DECREASING pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_MIN_DECREASING_SEQUENCE

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**
MAX_MIN_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**
\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \]
\[ VALUE = -\infty \lor VALUE \geq minv \]
\[ VALUE \leq maxv−1\]
required(VARIABLES, var)

where
\[ maxv = \text{maxval}(VARIABLES, \text{var}) \]
\[ minv = \text{minval}(VARIABLES, \text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES, var}) \]

**Purpose**
VALUE is the maximum of all minimum values in each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $-\infty$.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘$>$ ($>$|$|=)|>$’.

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i$ to index $j + 1$.

**Example**
\[(4, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\]

Figure 4.566 provides an example where the MAX_MIN_DECREASING_SEQUENCE (4, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

**Typical**
\[ |VARIABLES| > 1 \]
\[ \text{range}(\text{VARIABLES, var}) > 1 \]

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
MAX_MIN_DECREASING_SEQUENCE constraint of the Example slot

Figure 4.566: Illustrating the MAX_MIN_DECREASING_SEQUENCE constraint of the Example slot
Figures 4.567 and 4.568 respectively depict the automaton associated with the constraint MAX_MIN_DECREASING_SEQUENCE and its simplified form.

Figures 4.567: Automaton for the MAX_MIN_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING_SEQUENCE pattern where default is $-\infty$

Figures 4.568: Simplified automaton for the MAX_MIN_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.25 to the seed transducer of the DECREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.37: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the MAX_MIN_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max(C, C)</td>
</tr>
<tr>
<td>t</td>
<td>max(C, C)</td>
</tr>
</tbody>
</table>

Table 4.38: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the simplified automaton of the MAX_MIN_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>−∞</td>
</tr>
<tr>
<td>t</td>
<td>−∞</td>
</tr>
</tbody>
</table>
### MAX_MIN_DIP_ON_INCREASING_SEQUENCE

![Diagram](image)

**Description**

Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**

\[
\text{MAX_MIN_DIP_ON_INCREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: \(\text{dvar}\)
- **VARIABLES**: \(\text{collection}(\text{var} - \text{dvar})\)

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 5 & \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty & \lor \text{VALUE} \geq \text{minv} \\
\text{VALUE} \leq \text{maxv} - 2 & \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

- \(\text{maxv} = \text{maxval}(\text{VARIABLES}, \text{var})\)
- \(\text{minv} = \text{minval}(\text{VARIABLES}, \text{var})\)
- \(\text{sv} = |\text{VARIABLES}|\)
- \(\text{rv} = \text{range}(\text{VARIABLES}, \text{var})\)

**Purpose**

\(\text{VALUE}\) is the maximum of all minimum values in each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \(\text{VALUE}\) takes the default value \(-\infty\).

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression ‘\(<\langle\rangle\rangle\rangle\)’.

Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 2\) to index \(j\).

**Example**

\((2, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))\)

Figure 4.569 provides an example where the MAX_MIN_DIP_ON_INCREASING_SEQUENCE \((2, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4])\) constraint holds.

**Typical**

\begin{align*}
|\text{VARIABLES}| & > 5 \\
\text{range}(\text{VARIABLES}, \text{var}) & > 2
\end{align*}

**Arg. properties**

Functional dependency: \(\text{VALUE}\) determined by \(\text{VARIABLES}\).
Figure 4.569: Illustrating the MAX_MIN_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.570 and 4.571 respectively depict the automaton associated with the constraint MAX_MIN_DIP_ON_INCREASING_SEQUENCE and its simplified form.

Figure 4.570: Automaton for the MAX_MIN_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is $-\infty$. 
Figure 4.571: Simplified automaton for the MAX_MIN_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.28 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
**MAX_MIN_GORGE**

**Origin**
Based on the GORGE pattern.

**Constraint**
MAX_MIN_GORGE(VALUE, VARIABLES)

**Arguments**
VALUE : dvar  
VARIABLES : collection(var−dvar)

**Restrictions**
sv ≤ 2 ∧ rv ≤ 1 ⇒ VALUE = −∞  
VALUE = −∞ ∨ VALUE ≥ minv  
VALUE ≤ maxv − 1⇒
  required(VARIABLES.var)
where
  maxv = maxval(VARIABLES.var)  
  minv = minval(VARIABLES.var)  
  sv = |VARIABLES|  
  rv = range(VARIABLES.var)

**Purpose**
VALUE is the maximum of all minimum values in each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘(>| > (|= )*>)(< | < (|= )*<)’.
Assume that the occurrence of the pattern GORGE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

**Example**
(5, (1, 7, 3, 4, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7))

Figure 4.572 provides an example where the MAX_MIN_GORGE (5,[1, 7, 3, 4, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7]) constraint holds.

**Typical**
|VARIABLES| > 2  
range(VARIABLES.var) > 1

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.572: Illustrating the MAX_MIN_GORGE constraint of the Example slot
Automaton

Figures 4.573 and 4.574 respectively depict the automaton associated with the constraint **MAX_MIN_GORGE** and its simplified form.

Figure 4.573: Automaton for the **MAX_MIN_GORGE** constraint obtained by applying decoration Table 3.37 to the seed transducer of the **GORGE** pattern where default is $-\infty$ (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
Figure 4.574: Simplified automaton for the MAX_MIN_GORGE constraint obtained by applying decoration Table 3.39 to the seed transducer of the GORGE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 4.39: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the MAX_MIN_GORGE constraint defined as the composition of the GORGE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.40: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the simplified automaton of the MAX_MIN_GORGE constraint defined as the composition of the GORGE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_MIN_INCREASING

**Origin**
Based on the INCREASING pattern.

**Constraint**
MAX_MIN_INCREASING(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv − 1 ⇒
required(VARIABLES.var)
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

**Purpose**
VALUE is the maximum of all minimum values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression ‘<’.
Assume that the occurrence of the pattern INCREASING starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

**Example**
(4, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))

Figure 4.575 provides an example where the MAX_MIN_INCREASING (4, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

**Typical**
|VARIABLES| > 1
range(VARIABLES.var) > 1

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.575: Illustrating the MAX_MIN_INCREASING constraint of the Example slot
Figures 4.576 and 4.577 respectively depict the automaton associated with the constraint MAX_MIN_INCREASING and its simplified form.

**Automaton**

Figures 4.576: Automaton for the MAX_MIN_INCREASING constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING pattern where default is $-\infty$.

Figures 4.577: Simplified automaton for the MAX_MIN_INCREASING constraint obtained by applying decoration Table 3.39 to the seed transducer of the INCREASING pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 4.41: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the MAX_MIN_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.42: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the simplified automaton of the MAX_MIN_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_MIN_INCREASING_SEQUENCE

**Origin** Based on the INCREASING_SEQUENCE pattern.

**Constraint**

\[
\text{MAX_MIN_INCREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(}\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} \\
\text{VALUE} \leq \text{maxv} - 1 \Rightarrow \text{required}(\text{VARIABLES}.\text{var})
\]

where

\[
\begin{align*}
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the maximum of all minimum values in each occurrence of the increasing_sequence pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern increasing_sequence is the maximal subsequence which matches the regular expression ‘\(< (< | =)^* < | <\)’. Assume that the occurrence of the pattern increasing_sequence starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
(3, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))
\]

Figure 4.578 provides an example where the MAX_MIN_INCREASING_SEQUENCE (3, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.578: Illustrating the MAX_MIN_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.579 and 4.580 respectively depict the automaton associated with the constraint MAX_MIN_INCREASING_SEQUENCE and its simplified form.

$$\geq s
\left\{ \begin{array}{l}
C \leftarrow \text{default} \\
D \leftarrow +\infty \\
R \leftarrow \text{default}
\end{array} \right. 
\leq t \max(R, C)$$

$$\geq$$

$$\left\{ \begin{array}{l}
C \leftarrow \text{default} \\
D \leftarrow +\infty \\
R \leftarrow \max(R, C)
\end{array} \right. \left\{ \begin{array}{l}
C \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_i+1) \\
D \leftarrow +\infty
\end{array} \right. <$$

$$\geq$$

$$\left\{ \begin{array}{l}
C \leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \\
D \leftarrow +\infty
\end{array} \right. <$$

$$\leq t \max(R, C)$$

Figure 4.579: Automaton for the MAX_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $-\infty$.

$$\{R \leftarrow \text{default}\}$$

$$\geq s$$

$$\left\{ \begin{array}{l}
C \leftarrow \text{default} \\
D \leftarrow +\infty \\
\{R \leftarrow \max(R, \text{VAR}_i)\}
\end{array} \right. <$$

$$\leq t$$

Figure 4.580: Simplified automaton for the MAX_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.39 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.43: Concrete glue matrix, derived from the parametrised glue matrix \ref{table:3.9}, for the \textsc{max\_min\_increasing\_sequence} constraint defined as the composition of the \textsc{increasing\_sequence} pattern, the feature \textsc{min}, and the aggregator \textsc{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{center}
\begin{tabular}{|c|c|}
\hline
s & t \\
\hline
s & $\max(\overrightarrow{C}, \overrightarrow{C})$ & $\max(\overrightarrow{C}, \overrightarrow{C})$ \\
\hline
\end{tabular}
\end{center}

Table 4.44: Concrete glue matrix, derived from the parametrised glue matrix \ref{table:3.9}, for the simplified automaton of the \textsc{max\_min\_increasing\_sequence} constraint defined as the composition of the \textsc{increasing\_sequence} pattern, the feature \textsc{min}, and the aggregator \textsc{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{center}
\begin{tabular}{|c|c|}
\hline
s & t \\
\hline
s & $-\infty$ & $\overrightarrow{C}$ \\
\hline
\end{tabular}
\end{center}
**MAX_MIN_INFLEXION**

**Description**

Based on the **INFLEXION** pattern.

**Constraint**

```
MAX_MIN_INFLEXION(VALUE, VARIABLES)
```

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

**Restrictions**

```
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv
required(VARIABLES, var)
where
maxv =maxval(VARIABLES.var)
minv =minval(VARIABLES.var)
sv = |VARIABLES|
rv =range(VARIABLES.var)
```

**Purpose**

VALUE is the maximum of all minimum values in each occurrence of the **INFLEXION** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

An occurrence of the pattern **INFLEXION** is the maximal subsequence which matches the regular expression ‘< (< | =)∗ | > (| > |)∗ | <’.

Assume that the occurrence of the pattern **INFLEXION** starts at position i and ends at position j. The feature **MIN** computes the minimum of the values from index i + 1 to index j.

**Example**

```
(5, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))
```

Figure 4.581 provides an example where the MAX_MIN_INFLEXION (5, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4]) constraint holds.

**Typical**

```
|VARIABLES| > 2
range(VARIABLES.var) > 1
```

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.581: Illustrating the MAX_MIN_INFLEXION constraint of the Example slot
Figures 4.582 and 4.583 respectively depict the automaton associated with the constraint MAX_MIN_INFLEXION and its simplified form.

Figure 4.582: Automaton for the MAX_MIN_INFLEXION constraint obtained by applying decoration Table 3.37 to the seed transducer of the INFLEXION pattern where default is $-\infty$ (transition $r \rightarrow t$ has the same registers updates as transition $t \rightarrow r$).

Figure 4.583: Simplified automaton for the MAX_MIN_INFLEXION constraint obtained by applying decoration Table 3.26 to the seed transducer of the INFLEXION pattern where default is $-\infty$ (transition $r \rightarrow t$ has the same registers updates as transition $t \rightarrow r$); $R_i - R_{i-1} \geq 0$ is a linear invariant.
**MAX_MIN_STRICTLY_DECREASING_SEQUENCE**

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th><strong>Automaton</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
<td>Based on the <strong>STRICTLY_DECREASING_SEQUENCE</strong> pattern.</td>
</tr>
<tr>
<td>Constraint</td>
<td><strong>MAX_MIN_STRICTLY_DECREASING_SEQUENCE</strong>(VALUE, VARIABLES)</td>
</tr>
</tbody>
</table>
| Arguments       | VALUE : dvar  
|                 | VARIABLES : collection(var−dvar) |
| Restrictions    | $$sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty$$  
|                 | VALUE = -\infty \lor VALUE \geq minv  
|                 | VALUE $\leq$ maxv − 1
|                 | where  
|                 | maxv =maxval(VARIABLES.var)  
|                 | minv =minval(VARIABLES.var)  
|                 | sv = |VARIABLES|  
|                 | rv =range(VARIABLES.var)  |
| Purpose         | VALUE is the maximum of all minimum values in each occurrence of the **STRICTLY_DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $-\infty$.  
|                 | An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the maximal subsequence which matches the regular expression ‘$>$’.  
|                 | Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position $i$ and ends at position $j$. The feature **MIN** computes the minimum of the values from index $i$ to index $j + 1$. |
| Example         | (3, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 5, 2, 2, 4, 3)) |
| Typical         | $|VARIABLES| > 1$  
|                 | range(VARIABLES.var) > 1 |
| Arg. properties | Functional dependency: VALUE determined by VARIABLES. |

Figure 4.584 provides an example where the **MAX_MIN_STRICTLY_DECREASING_SEQUENCE** (3, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 5, 2, 2, 4, 3]) constraint holds.
Figure 4.584: Illustrating the MAX_MIN_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.585 and 4.586 respectively depict the automaton associated with the constraint `MAX_MIN_STRICTLY_DECREASING_SEQUENCE` and its simplified form.

![Automaton Diagram](image)

Figure 4.585: Automaton for the `MAX_MIN_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern where `default` is $-\infty$.

![Simplified Automaton Diagram](image)

Figure 4.586: Simplified automaton for the `MAX_MIN_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.25 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern where `default` is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.45: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the `MAX_MIN_STRICTLY_DECREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_DECREASING_SEQUENCE` pattern, the feature `MIN`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>r</td>
</tr>
<tr>
<td>s</td>
<td><code>max(C, C)</code></td>
</tr>
<tr>
<td>r</td>
<td><code>max(C, C)</code></td>
</tr>
</tbody>
</table>

Table 4.46: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the `MAX_MIN_STRICTLY_DECREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_DECREASING_SEQUENCE` pattern, the feature `MIN`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>r</td>
</tr>
<tr>
<td>s</td>
<td><code>-∞</code></td>
</tr>
<tr>
<td>r</td>
<td><code>C</code></td>
</tr>
</tbody>
</table>
**MAX_MIN STRICTLY INCREASINGSEQUENCE**

**Origin**
Based on the **STRICTLY_INCREASING_SEQUENCE** pattern.

**Constraint**
```
MAX_MIN STRICTLY INCREASINGSEQUENCE(VALUE, VARIABLES)
```

**Arguments**
```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

**Restrictions**
```
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv − 1 ⇒
required(VARIABLES, var)
```

where
```
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

**Purpose**
\(\text{VALUE}\) is the maximum of all minimum values in each occurrence of the **STRICTLY_INCREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, \(\text{VALUE}\) takes the default value \(-\infty\).

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the maximal sub-sequence which matches the regular expression ‘\(<\)’.

Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature \(\text{MIN}\) computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**
```
(3, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))
```

Figure 4.587 provides an example where the **MAX_MIN STRICTLY INCREASING_SEQUENCE** \((3, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3])\) constraint holds.

**Typical**
```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```

**Arg. properties**
Functional dependency: \(\text{VALUE}\) determined by **VARIABLES**.
Figure 4.587: Illustrating the `MAX_MIN_STRICTLY_INCREASING_SEQUENCE` constraint of the `Example` slot
Automaton

Figures 4.588 and 4.589 respectively depict the automaton associated with the constraint MAX_MIN.Strictly.Increasing.Sequence and its simplified form.

Figure 4.588: Automaton for the MAX.MIN.Strictly.Increasing.Sequence constraint obtained by applying decoration Table 3.37 to the seed transducer of the Strictly.Increasing.Sequence pattern where default is $-\infty$.

Figure 4.589: Simplified automaton for the MAX_MIN.Strictly.Increasing.Sequence constraint obtained by applying decoration Table 3.39 to the seed transducer of the Strictly.Increasing.Sequence pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.47: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the MAX_MIN_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
</tr>
<tr>
<td>r</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
<td>min(\overrightarrow{C}, \overrightarrow{C}, \overleftarrow{D}, \overleftarrow{D})</td>
</tr>
</tbody>
</table>

Table 4.48: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the simplified automaton of the MAX_MIN_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>-\infty</td>
<td>\overrightarrow{C}</td>
</tr>
<tr>
<td>r</td>
<td>-\infty</td>
<td>-\infty</td>
</tr>
</tbody>
</table>

\[s_{\text{rs}} \rightarrow \text{max}(\overrightarrow{C}, \overrightarrow{C})\]

\[r_{\text{sr}} \rightarrow \text{max}(\overrightarrow{C}, \overrightarrow{C}) \text{ and } \text{min}(\overrightarrow{C}, \overrightarrow{C}, \overleftarrow{D}, \overleftarrow{D})\]
**Origin**  Based on the VALLEY pattern.

**Constraint**  
MAX_MIN_VALLEY(VALUE, VARIABLES)

**Arguments**

<table>
<thead>
<tr>
<th>VALUE</th>
<th>dvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

<table>
<thead>
<tr>
<th>sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE = −∞ ∨ VALUE ≥ minv</td>
</tr>
<tr>
<td>VALUE ≤ maxv − 1* required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

where

| maxv = maxval(VARIABLES, var) |
| minv = minval(VARIABLES, var) |
| sv = |VARIABLES| |
| rv = range(VARIABLES, var) |

**Purpose**

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression ‘> (= | >)∗ (< | =)∗ <’. Assume that the occurrence of the pattern VALLEY starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i+1 to index j.

**Example**

(5, (1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7))

Figure 4.590 provides an example where the MAX_MIN_VALLEY (5, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7]) constraint holds.

**Typical**

| | VARIABLES | > 2 |
|----------------|----------|
| range(VARIABLES, var) > 1 |

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.590: Illustrating the MAX_MIN_VALLEY constraint of the Example slot
Automaton  

Figures 4.591 and 4.592 respectively depict the automaton associated with the constraint \textsc{MAX\_MIN\_VALLEY} and its simplified form.

![Automaton Diagram](image)

Figure 4.591: Automaton for the \textsc{MAX\_MIN\_VALLEY} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsc{VALLEY} pattern where default is \(-\infty\).

![Simplified Automaton Diagram](image)

Figure 4.592: Simplified automaton for the \textsc{MAX\_MIN\_VALLEY} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \textsc{VALLEY} pattern where default is \(-\infty\); \(R_i - R_{i-1} \geq 0\) is a linear invariant.
Table 4.49: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the MAX_MIN_VALLEY constraint defined as the composition of the VALLEY pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}))</td>
</tr>
<tr>
<td>t</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
</tbody>
</table>

Table 4.50: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the simplified automaton of the MAX_MIN_VALLEY constraint defined as the composition of the VALLEY pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
</tr>
<tr>
<td>r</td>
<td>(-\infty)</td>
<td>\text{VAR}_{i+1}</td>
<td>(-\infty)</td>
</tr>
<tr>
<td>t</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
<td>(-\infty)</td>
</tr>
</tbody>
</table>
**MAX_MIN_ZIGZAG**

**Origin**
Based on the ZIGZAG pattern.

**Constraint**
\[
\text{MAX_MIN_ZIGZAG}(\text{VALUE, VARIABLES})
\]

**Arguments**

<table>
<thead>
<tr>
<th>VALUE</th>
<th>: dvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>collection(var–dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
sv \leq 3 \lor rv \leq 1 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} &= -\infty \lor \text{VALUE} \geq \text{minv} \\
\text{VALUE} &\leq \text{maxv} - 1 \lor \\
\text{required} &\left(\text{VARIABLES}, \text{var}\right)
\end{align*}
\]

Where

\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern ZIGZAG is the *maximal* subsequence which matches the regular expression \(’(<>)^+(< | <> | (<>)^+(> | <>))’\).

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i+1\) to index \(j\).

**Example**

\[(1, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\]

Figure 4.593 provides an example where the \text{MAX_MIN_ZIGZAG} (\(1, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1]\)) constraint holds.

**Typical**

\(|\text{VARIABLES}| \geq 3\)

\(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.593: Illustrating the MAX_MIN_ZIGZAG constraint of the Example slot
<table>
<thead>
<tr>
<th>Figures</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.594, 4.595</td>
<td>Figures respectively depict the automaton associated with the constraint <code>MAX_MIN_ZIGZAG</code> and its simplified form.</td>
</tr>
</tbody>
</table>
Figure 4.594: Automaton for the MAX_MIN_ZIGZAG constraint obtained by applying decoration Table 3.37 to the seed transducer of the ZIGZAG pattern where \texttt{default} is $-\infty$: (1) missing transitions from $a$, $b$, $c$, $d$, $e$, $f$ to $s$ are labelled by $=$; (2) on transitions from $b$, $c$, $e$, $f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c$, $f$ to $s$ the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.
Figure 4.595: Simplified automaton for the MAX_MIN_ZIGZAG constraint obtained by applying decoration Table 3.24 to the seed transducer of the ZIGZAG pattern where default is $-\infty$; missing transitions from a, b, c, d, e, f to s are labelled by $=;$ (2) on transitions from b, c, e, f to s the register $D$ is reset to its initial value; (3) on transitions from c, f to s the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.51: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the MAX_MIN_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.52: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the simplified automaton of the \textsc{Max-Min-Zigzag} constraint defined as the composition of the \textsc{Zigzag} pattern, the feature \textsc{Min}, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>a</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>b</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{min}(\overline{B}, \overline{B}, \text{VAR}_{+1}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>c</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>d</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{min}(\overline{B}, \overline{B}, \text{VAR}_{+1}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>e</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>f</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{min}(\overline{B}, \overline{B}, \text{VAR}_{+1}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
<td>(\text{max}(\overline{C}, \overline{C}))</td>
</tr>
</tbody>
</table>
MAX_PEAk

<table>
<thead>
<tr>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAX_PEAk</td>
<td>⌋ ▾ ™ ⌂</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

- **Origin**: Based on the PEAK pattern.
- **Constraint**: MAX_PEAk(VARIABLES, FEATURES, DEFAULT)
- **Arguments**:
  - VARIABLES : collection(var−dvar)
  - FEATURES : collection(var−dvar)
  - DEFAULT : int
- **Restrictions**:
  - required(VARIABLES.var)
  - required(FEATURES.var)
  - |VARIABLES| = |FEATURES|
  - sv ≤ 2 ∧ rv ≤ 1 ⇒ FEATURES.var = DEFAULT
  - FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv + 1
  - FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv
  - DEFAULT < minv + 1 ∨ DEFAULT > maxv
  - where
    - maxv = maxval(VARIABLES.var)
    - minv = minval(VARIABLES.var)
    - sv = |VARIABLES|
    - rv = range(VARIABLES.var)

Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of PEAK is identified (even if this occurrence of pattern is not complete) then FEATURES[$i$] is the default value DEFAULT; otherwise FEATURES[$i$] gives the feature value of the corresponding occurrence of PEAK.

**Purpose**

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression '< (< | <=) * (> | =>) * >'.
Assume that the occurrence of the pattern PEAK starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i + 1$ to index $j$.

**Example**

Figure 4.596 provides an example where the MAX_PEAk ([7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1], [0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 6, 0, 0, 0, 3, 0], 0) constraint holds.

**Typical**

- |VARIABLES| > 2
- range(VARIABLES.var) > 1

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.596: Illustrating the MAX_peak constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
MAX_PEAK  1371
MAX_RANGE_DECREASING

**Origin**
Based on the **DECREASING** pattern.

**Constraint**
MAX_RANGE_DECREASING(VALUE, VARIABLES)

**Arguments**
- **VALUE** : dvar
- **VARIABLES** : collection(var−dvar)

**Restrictions**
- \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE \geq 0 \)
- \( VALUE \leq rv - 1 \)
  - required(VARIABLES, var)
where
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)

**Purpose**
VALUE is the maximum value of the differences between the largest and smallest value in each occurrence of the **DECREASING** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression ‘\(>\)’.
Assume that the occurrence of the pattern **DECREASING** starts at position \(i\) and ends at position \(j\). The feature RANGE computes the range of the values from index \(i\) to index \(j + 1\).

**Example**
(2, (3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))

Figure 4.597 provides an example where the **MAX_RANGE_DECREASING** (2, [3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**
One and the same constant can be added to the var attribute of all items of **VARIABLES**.

**Arg. properties**
Functional dependency: VALUE determined by **VARIABLES**.
Figure 4.597: Illustrating the MAX_RANGE_DECREASING constraint of the Example slot
Figures 4.598 and 4.599 respectively depict the automaton associated with the constraint MAX\_RANGE\_DECREASING and its simplified form.

Figure 4.598: Automaton for the MAX\_RANGE\_DECREASING constraint obtained by applying decoration Table 3.48 to the seed transducer of the DECREASING pattern where default is 0.

Figure 4.599: Simplified automaton for the MAX\_RANGE\_DECREASING constraint obtained by applying decoration Table 3.46 to the seed transducer of the DECREASING pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $R_i + \text{VAR}_{i-1} - \text{VAR}_{i-2} \geq 0$ are linear invariants.
### MAX_RANGE_DECREASING_SEQUENCE

#### Origin
Based on the `DECREASING_SEQUENCE` pattern.

#### Constraint
```
MAX_RANGE_DECREASING_SEQUENCE(VALUE, VARIABLES)
```

#### Arguments
- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`

#### Restrictions
- `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0`
- `VALUE ≥ 0`
- `VALUE ≤ rv − 1 ∧ required(VARIABLES, var)`

where
- `rv = range(VARIABLES.var)`
- `sv = |VARIABLES|`

#### Purpose
An occurrence of the pattern `DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `‘> (> | =)’ > | >’`. Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `RANGE` computes the range of the values from index `i` to index `j + 1`.

#### Example
```
(5, ⟨3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4⟩)
```

Figure 4.600 provides an example where the `MAX_RANGE_DECREASING_SEQUENCE` (5, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

#### Typical
- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

#### Symmetry
One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

#### Arg. properties
- Functional dependency: `VALUE` determined by `VARIABLES`.

---

A description of the automaton for the `MAX_RANGE_DECREASING_SEQUENCE` feature, showing the transitions and states for input values and conditions.
Figure 4.600: Illustrating the MAX_RANGE_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.601 and 4.602 respectively depict the automaton associated with the constraint `MAX_RANGE_DECREASING_SEQUENCE` and its simplified form.

Figure 4.601: Automaton for the `MAX_RANGE_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.48 to the seed transducer of the `DECREASING_SEQUENCE` pattern where `default` is 0.

Figure 4.602: Simplified automaton for the `MAX_RANGE_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.44 to the seed transducer of the `DECREASING_SEQUENCE` pattern where `default` is 0; \( R_i - R_{i-1} \geq 0 \) is a linear invariant.
MAX_RANGE_INCREASING

<table>
<thead>
<tr>
<th>Aggregator</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAX_RANGE_INCREASING</td>
<td>▶ ▶ &lt; ◀</td>
<td>DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the **INCREASING** pattern.

**Constraint**

\[
\text{MAX\_\_RANGE\_\_INCREASING}(\text{VALUE, VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} &: \text{dvar} \\
\text{VARIABLES} &: \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \text{rv} - 1 \\
\text{required}(\text{VARIABLES, var}) & \text{required}(\text{VARIABLES, var})
\end{align*}
\]

where

\[
\begin{align*}
\text{rv} &= \text{range}(\text{VARIABLES, var}) \\
\text{sv} &= |\text{VARIABLES}|
\end{align*}
\]

**Purpose**

\[\text{VALUE}\] is the maximum value of the differences between the largest and smallest value in each occurrence of the **INCREASING** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, \[\text{VALUE}\] takes the default value 0.

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression ‘\(<\)’.

Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature **RANGE** computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\[(2, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\]

Figure 4.603 provides an example where the **MAX\_\_RANGE\_\_INCREASING**\((2, [4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES, var}) > 1
\]

**Symmetry**

One and the same constant can be added to the \text{var} attribute of all items of \text{VARIABLES}.

**Arg. properties**

Functional dependency: \[\text{VALUE}\] determined by \text{VARIABLES}. 
Figure 4.603: Illustrating the **MAX_RANGE_INCREASING** constraint of the **Example** slot
Automaton Figures 4.604 and 4.605 respectively depict the automaton associated with the constraint MAX_RANGE_INCREASING and its simplified form.

\[
\begin{align*}
\{ C \leftarrow \text{default}, \\
H \leftarrow \text{VAR}, \\
R \leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
\{ H \leftarrow \text{VAR}_i + 1, \\
R \leftarrow \max(R, |H - \text{VAR}_i + 1|) \}
\end{align*}
\]

Figure 4.604: Automaton for the MAX_RANGE_INCREASING constraint obtained by applying decoration Table 3.48 to the seed transducer of the INCREASING pattern where default is 0.

\[
\{ R \leftarrow \text{default} \}
\]

\[
\{ R \leftarrow \max(R, \max(0, \text{VAR}_i + 1 - \text{VAR}_i)) \}
\]

Figure 4.605: Simplified automaton for the MAX_RANGE_INCREASING constraint obtained by applying decoration Table 3.47 to the seed transducer of the INCREASING pattern where default is 0; \(R_i - R_{i-1} \geq 0\) and \(R_i - \text{VAR}_{i-1} + \text{VAR}_{i-2} \geq 0\) are linear invariants.
### Description

**Origin**
Based on the `INCREASING_SEQUENCE` pattern.

**Constraint**

\[
\text{MAX\_RANGE\_INCREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(\text{var}−\text{dvar})}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq rv - 1 \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
rv & = \text{range}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}|
\end{align*}
\]

**Purpose**

An occurrence of the pattern `INCREASING_SEQUENCE` is the \textit{maximal} subsequence which matches the regular expression `(\lt | =\ast ) \lt | \lt`.

Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature \textit{RANGE} computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\((5, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\)

Figure 4.606 provides an example where the \textit{MAX\_RANGE\_INCREASING\_SEQUENCE} \((5, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**

One and the same constant can be \textit{added} to the \textit{var} attribute of all items of \textit{VARIABLES}.

**Arg. properties**

\textit{Functional dependency}: \textit{VALUE} determined by \textit{VARIABLES}.
Figure 4.606: Illustrating the MAX_RANGE_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.607 and 4.608 respectively depict the automaton associated with the constraint MAX_RANGE_INCREASING_SEQUENCE and its simplified form.

Figure 4.607: Automaton for the MAX_RANGE_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.48 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0

Figure 4.608: Simplified automaton for the MAX_RANGE_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.45 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) is a linear invariant.
**MAX_RANGE_STRICTLY_DECREASING_SEQUENCE**

**Origin**
Based on the **STRICTLY_DECREASING_SEQUENCE** pattern.

**Constraint**

\[
\text{MAX\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`

**Restrictions**

\[
\begin{align*}
sv & \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \\
VALUE & \geq 0 \\
VALUE & \leq rv - 1 \\
\text{required}(\text{VARIABLES}, \text{var}) & \text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
rv &= \text{range}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}|
\end{align*}
\]

**Purpose**
An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the \textit{maximal} sub-sequence which matches the regular expression `'>\+'.

Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature \text{RANGE} computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\[(5, \langle 4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3 \rangle)\]

Figure 4.609 provides an example where the \text{MAX\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE} \((5, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**
One and the same constant can be added to the \text{var} attribute of all items of \text{VARIABLES}.

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.609: Illustrating the MAX_RANGE_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Figures 4.610 and 4.611 respectively depict the automaton associated with the constraint \texttt{MAX\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE} and its simplified form.

**Figure 4.610:** Automaton for the \texttt{MAX\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.48 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0.

**Figure 4.611:** Simplified automaton for the \texttt{MAX\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.44 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0; $R_i - R_{i-1} \geq 0$ is a linear invariant.
MAX_RANGE STRICTLY DECREASING SEQUENCE
## MAX_RANGE.Strictly.Increasing.Sequence

### Description

**Origin**

Based on the `Strictly.Increasing.Sequence` pattern.

**Constraint**

\[
\text{MAX\_RANGE\_STRICLY\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var\_dvar)`

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq rv - 1 \\
\text{required}(\text{VARIABLES}, \text{var}) & \text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{rv} & = \text{range}(\text{VARIABLES}, \text{var}) \\
sv & = |\text{VARIABLES}|
\end{align*}
\]

**Purpose**

An occurrence of the pattern `Strictly.Increasing.Sequence` is the maximal subsequence which matches the regular expression ‘<’.

Assume that the occurrence of the pattern `Strictly.Increasing.Sequence` starts at position \( i \) and ends at position \( j \). The feature `range` computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
(5, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))
\]

Figure 4.612 provides an example where the `MAX\_RANGE\_STRICLY\_INCREASING\_SEQUENCE` constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}, \text{var}) > 1
\]

**Symmetry**

One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**

- **Functional dependency**: `VALUE` determined by `VARIABLES`.
Figure 4.612: Illustrating the \texttt{MAX\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE} constraint of the \texttt{Example} slot.
Figures 4.613 and 4.614 respectively depict the automaton associated with the constraint MAX_RANGE_STRICTLY_INCREASING_SEQUENCE and its simplified form.

Figure 4.613: Automaton for the MAX_RANGE_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.48 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0.

Figure 4.614: Simplified automaton for the MAX_RANGE_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.45 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0; $R_i - R_{i-1} \geq 0$ is a linear invariant.
MAX_RANGE_STRICTLY_INCREASING_SEQUENCE
Origin: Based on the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern.

Constraint: \texttt{MAX\_STRICTLY\_DECREASING\_SEQUENCE(VARIABLES, FEATURES, DEFAULT)}

Arguments:
- \texttt{VARIABLES}: \texttt{collection(var\_dvar)}
- \texttt{FEATURES}: \texttt{collection(var\_dvar)}
- \texttt{DEFAULT}: \texttt{int}

Restrictions:
- \texttt{required(VARIABLES.var)}
- \texttt{required(FEATURES.var)}
- \texttt{|VARIABLES| = |FEATURES|}
- \texttt{sv \leq 1 \lor rv \leq 1 \Rightarrow FEATURES.var = DEFAULT}
- \texttt{FEATURES.var = DEFAULT \lor FEATURES.var \geq minv + 1}
- \texttt{FEATURES.var = DEFAULT \lor FEATURES.var \leq maxv}
- \texttt{DEFAULT < minv + 1 \lor DEFAULT > maxv}

where
- \texttt{maxv = maxval(VARIABLES.var)}
- \texttt{minv = minval(VARIABLES.var)}
- \texttt{sv = |VARIABLES|}
- \texttt{rv = range(VARIABLES.var)}

Purpose:
Consider the time-series given by the \texttt{VARIABLES} collection. If position \(i\) does not correspond to the first position in \texttt{VARIABLES} where an occurrence of \texttt{STRICTLY\_DECREASING\_SEQUENCE} is identified (even if this occurrence of pattern is not complete) then \texttt{FEATURES}[i] is the default value \texttt{DEFAULT}; otherwise \texttt{FEATURES}[i] gives the feature value of the corresponding occurrence of \texttt{STRICTLY\_DECREASING\_SEQUENCE}.

An occurrence of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression ‘>\+’.

Assume that the occurrence of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{MAX} computes the maximum of the values from index \(i\) to index \(j+1\).

Example:
Figure 4.615 provides an example where the \texttt{MAX\_STRICTLY\_DECREASING\_SEQUENCE ([4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3], [0, 0, 6, 0, 0, 0, 0, 0, 0, 6, 0, 0, 0, 4, 0], 0)} constraint holds.

Typical:
- \texttt{|VARIABLES| > 1}
- \texttt{range(VARIABLES.var) > 1}
Figure 4.615: Illustrating the MAX\_STRICTLY\_DECREASING\_SEQUENCE constraint of the Example slot

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
**Automaton**

Use the decoration table 3.32 to synthesise the corresponding register automaton.
MAX_STRICTLY_DECREASING_SEQUENCE
MAX\_STRICTLY\_INCREASING\_SEQUENCE

Origin

Based on the \texttt{STRICTLY\_INCREASING\_SEQUENCE} pattern.

Constraint

\texttt{MAX\_STRICTLY\_INCREASING\_SEQUENCE}(\texttt{VARIABLES, FEATURES, DEFAULT})

Arguments

\begin{itemize}
  \item \texttt{VARIABLES} : \texttt{collection(var\_dvar)}
  \item \texttt{FEATURES} : \texttt{collection(var\_dvar)}
  \item \texttt{DEFAULT} : \texttt{int}
\end{itemize}

Restrictions

\begin{itemize}
  \item \texttt{required(VARIABLES.var)}
  \item \texttt{required(FEATURES.var)}
  \item $|\texttt{VARIABLES}| = |\texttt{FEATURES}|$
  \item $sv \leq 1 \lor rv \leq 1 \Rightarrow \texttt{FEATURES.var} = \texttt{DEFAULT}$
  \item $\texttt{FEATURES.var} = \texttt{DEFAULT} \lor \texttt{FEATURES.var} \geq minv + 1$
  \item $\texttt{FEATURES.var} = \texttt{DEFAULT} \lor \texttt{FEATURES.var} \leq maxv$
  \item $\texttt{DEFAULT} < minv + 1 \lor \texttt{DEFAULT} > maxv$
\end{itemize}

where

\begin{itemize}
  \item $maxv = \texttt{maxval(VARIABLES.var)}$
  \item $minv = \texttt{minval(VARIABLES.var)}$
  \item $sv = |\texttt{VARIABLES}|$
  \item $rv = \texttt{range(VARIABLES.var)}$
\end{itemize}

Purpose

Consider the time-series given by the \texttt{VARIABLES} collection. If position $i$ does not correspond to the first position in \texttt{VARIABLES} where an occurrence of \texttt{STRICTLY\_INCREASING\_SEQUENCE} is identified (even if this occurrence of pattern is not complete) then \texttt{FEATURES[i]} is the default value \texttt{DEFAULT}; otherwise \texttt{FEATURES[i]} gives the feature value of the corresponding occurrence of \texttt{STRICTLY\_INCREASING\_SEQUENCE}.

An occurrence of the pattern \texttt{STRICTLY\_INCREASING\_SEQUENCE} is the \textit{maximal} sub-sequence which matches the regular expression '\textless'\textsuperscript{+}'

Assume that the occurrence of the pattern \texttt{STRICTLY\_INCREASING\_SEQUENCE} starts at position $i$ and ends at position $j$. The feature \texttt{MAX} computes the maximum of the values from index $i$ to index $j + 1$.

Example

Figure \ref{fig:example} provides an example where the \texttt{MAX\_STRICTLY\_INCREASING\_SEQUENCE} ([4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 3, 1, 2, 3], [0, 5, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] 0) constraint holds.

Typical

\begin{itemize}
  \item $|\texttt{VARIABLES}| > 1$
  \item $\texttt{range(VARIABLES.var)} > 1$
\end{itemize}
Figure 4.616: Illustrating the MAX_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot

Arg. properties

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
Based on the SUMMIT pattern.

\[
\text{MAX}_\text{SUMMIT}(\text{VARIABLES}, \text{FEATURES}, \text{DEFAULT})
\]

**Arguments**

- **VARIABLES**: collection(var−dvar)
- **FEATURES**: collection(var−dvar)
- **DEFAULT**: int

**Restrictions**

\[
\begin{align*}
\text{required} & (\text{VARIABLES}.\text{var}) \\
\text{required} & (\text{FEATURES}.\text{var}) \\
|\text{VARIABLES}| & = |\text{FEATURES}| \\
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{FEATURES}.\text{var} = \text{DEFAULT} \\
\text{FEATURES}.\text{var} & = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \geq \text{minv} + 1 \\
\text{FEATURES}.\text{var} & = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \leq \text{maxv} \\
\text{DEFAULT} & < \text{minv} + 1 \lor \text{DEFAULT} > \text{maxv}
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Consider the time-series given by the \text{VARIABLES} collection. If position \(i\) does not correspond to the first position in \text{VARIABLES} where an occurrence of \text{SUMMIT} is identified (even if this occurrence of pattern is not complete) then \text{FEATURES}[i] is the default value \text{DEFAULT}; otherwise \text{FEATURES}[i] gives the feature value of the corresponding occurrence of \text{SUMMIT}.

An occurrence of the pattern \text{SUMMIT} is the maximal subsequence which matches the regular expression \(’(\prec | \prec (= | \prec)^* \prec (\succ | \prec (= | \succ)^* \succ)^*’\).

Assume that the occurrence of the pattern \text{SUMMIT} starts at position \(i\) and ends at position \(j\). The feature \text{MAX} computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

Figure 4.617 provides an example where the \text{MAX}_\text{SUMMIT} \(([7, 1, 5, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1], [0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 3, 0, 0])\) constraint holds.

Typical

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

Functional dependency: \text{FEATURES} determined by \text{VARIABLES} and \text{DEFAULT}.
Figure 4.617: Illustrating the MAX_SUMMIT constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
MAX_SURF_BUMP_ON_DECREASING_SEQUENCE

**Origin**
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**
MAX_SURF_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
\[ sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = -\infty \]
VALUE = -\infty \lor VALUE \geq 3 * minv + 3
VALUE \leq 3 * maxv - 3 \times
among(n1,VARIABLES[3,sv - 1],(maxv - 2,maxv - 1,maxv))
n1 \geq VALUE - 3 - max(maxv - 3,0)
among(n2,VARIABLES[3,sv - 1],(minv,minv + 1,minv + 2))
n2 \geq min(minv + 3,0) - 3 - VALUE
required(VARIABLES,var)

where
maxv =maxval(VARIABLES.var)
minv =minval(VARIABLES.var)
sv = |VARIABLES|
rv =range(VARIABLES.var)

**Purpose**
An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression ‘>>><’.
Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 2\) to index \(j\).

**Example**

\[(16, \langle 7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3 \rangle)\]

Figure 4.618 provides an example where the MAX_SURF_BUMP_ON_DECREASING_SEQUENCE (16, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3]) constraint holds.

**Typical**
|VARIABLES| > 5
range(VARIABLES.var) > 2
Figure 4.618: Illustrating the MAX_SURF_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.619 and 4.620 respectively depict the automaton associated with the constraint `MAX_SURF_BUMP_ON_DECREASING_SEQUENCE` and its simplified form.

Figure 4.619: Automaton for the `MAX_SURF_BUMP_ON_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `BUMP_ON_DECREASING_SEQUENCE` pattern where `default` is $-\infty$. 
Figure 4.620: Simplified automaton for the MAX_SURF_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.26 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
### MAX_SURF_DECREASING

**Origin**
Based on the **DECREASING** pattern.

**Constraint**

```
MAX_SURF_DECREASING(VALUE, VARIABLES)
```

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var − dvar)
```

**Restrictions**

```
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ 2 * minv + 1
VALUE ≤ 2 * maxv − 1
required(VARIABLES, var)
```

where

```
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

**Purpose**

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression ‘>’. Assume that the occurrence of the pattern **DECREASING** starts at position $i$ and ends at position $j$. The feature **SURF** computes the sum of the values from index $i$ to index $j + 1$.

**Example**

```
(10, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4])
```

Figure 4.621 provides an example where the **MAX_SURF_DECREASING** (10, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4]) constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.621: Illustrating the MAX_SURF_DECREASING constraint of the **Example** slot
Figures 4.622 and 4.623 respectively depict the automaton associated with the constraint MAX_SURF_DECREASING and its simplified form.

Figure 4.622: Automaton for the MAX_SURF_DECREASING constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING pattern where default is $-\infty$.

Figure 4.623: Simplified automaton for the MAX_SURF_DECREASING constraint obtained by applying decoration Table 3.40 to the seed transducer of the DECREASING pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 4.53: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the MAX_SURF_DECREASING constraint defined as the composition of the DECREASING pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.54: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the simplified automaton of the MAX_SURF_DECREASING constraint defined as the composition of the DECREASING pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Based on the DECENDING_SEQUENCE pattern.

\[ \text{MAX\_SURF\_DECREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(\text{var} - \text{dvar})}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{rv} & = 2 \Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq 2 \cdot \text{minval} + 1 \\
\text{rv} & \geq 3 \Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq \min\{2 \cdot \text{minval} + 1, \text{sv} \cdot (\text{minval} + 1)\} \\
\text{rv} & = 2 \Rightarrow \text{VALUE} \leq 2 \cdot \text{maxval} - 1 \\
\text{rv} & \geq 3 \Rightarrow \text{VALUE} \leq \max\{2 \cdot \text{maxval} - 1, \cdot \text{sv} \cdot (\text{maxval} - 1)\} \\
\text{among}\{\text{n}1, \text{VARIABLES}[\text{l}, \text{sv}], (\text{maxval} - 1, \text{maxval})\} \\
\text{rv} & = 2 \land \text{maxval} = 1 \Rightarrow \text{n}1 \geq \text{VALUE} - \max(0, 2 \cdot \text{maxval} - 3) \\
\text{rv} & = 2 \land \text{minval} \geq 1 \Rightarrow \text{n}1 \geq \text{VALUE} - \text{sv} \cdot (\text{maxval} - 2) - 1 \\
\text{among}\{\text{n}2, \text{VARIABLES}[\text{l}, \text{sv}], (\text{minval}, \text{minval} + 1)\} \\
\text{rv} & = 2 \land \text{minval} = -1 \Rightarrow \text{n}2 \geq \min(0, 2 \cdot \text{minval} + 3) - \text{VALUE} \\
\text{rv} & > 2 \land \text{minval} \geq -1 \Rightarrow \text{n}2 \geq \text{sv} \cdot (\text{minval} + 2) - 1 - \text{VALUE} \\
\text{required}(\text{VARIABLES}.\text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{maxval} & = \mathsf{maxval}(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \mathsf{range}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = [\text{VARIABLES}] \\
\text{minval} & = \mathsf{minval}(\text{VARIABLES}.\text{var})
\end{align*}
\]

VALUE is the maximal surface of occurrences of the DECENDING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

**Purpose**

An occurrence of the pattern DECENDING_SEQUENCE is the maximal subsequence which matches the regular expression \(\text{\textgreater \textgreater \textless \textless} \textgreater \textgreater \textgreater\).

Assume that the occurrence of the pattern DECENDING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[(18, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 4.624 provides an example where the MAX\_SURF\_DECREASING\_SEQUENCE (18, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.
Figure 4.624: Illustrating the MAX_SURF_DECREASING_SEQUENCE constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\var) > 1 \]

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.625 depicts the automaton associated with the constraint MAX_SURF_DECREASING_SEQUENCE.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow 0 \\
& R \leftarrow \text{default} \} \\
\有条件 \leq s \quad \有条件 \leq \leq \\
\有条件 < \{ & C \leftarrow \text{default} \\
& D \leftarrow 0 \\
& R \leftarrow \max(R, C) \} \\
\有条件 \geq t \{ & C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
& D \leftarrow 0 \} \\
\有条件 = \{ & D \leftarrow D + \text{VAR}_{i+1} \}
\end{align*}
\]

Figure 4.625: Automaton for the MAX_SURF_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

<table>
<thead>
<tr>
<th>$s$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\max(C, \overline{C})$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\max(C, \overline{C})$</td>
</tr>
</tbody>
</table>

Table 4.55: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the MAX_SURF_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin

Based on the DECREASING_TERRACE pattern.

Constraint

\[
\text{MAX}\_\text{SURF}\_\text{DECREASING}\_\text{TERRACE}(\text{VALUE, VARIABLES})
\]

Arguments

\[
\text{VALUE} : \text{dvar} \\
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

Restrictions

\[
\begin{align*}
sv \leq 3 & \lor rv \leq 2 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty & \lor \text{VALUE} \geq \min(2 \ast (\text{minv} + 1), (sv - 2) \ast (\text{minv} + 1)) \\
\text{VALUE} \leq \max(2 \ast (\text{maxv} - 1) \ast (sv - 2) \ast (\text{maxv} - 1) \ast x) & \text{among}[n1, \text{VARIABLES}[2, sv - 1], (\text{maxv} - 1)] \\
n1 \geq \text{VALUE} - \max(0, (sv - 2) \ast (\text{maxv} - 2)) & \text{among}[n2, \text{VARIABLES}[2, sv - 1], (\text{minv} + 1)] \\
n2 \geq \min(0, (sv - 2) \ast (\text{minv} + 2)) & -\text{VALUE} \text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

\[
\text{VALUE} \text{ is the maximal surface of occurrences of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value } -\infty.
\]

Purpose

An occurrence of the pattern \text{DECREASING_TERRACE} is the \textit{maximal} subsequence which matches the regular expression ‘\texttt{> = + >}’.

Assume that the occurrence of the pattern \text{DECREASING_TERRACE} starts at position \(i\) and ends at position \(j\). The feature \text{SURF} computes the sum of the values from index \(i + 1\) to index \(j\).

Example

\[
(8, [6, 4, 4, 5, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3])
\]

Figure 4.626 provides an example where the \text{MAX}\_\text{SURF}\_\text{DECREASING}\_\text{TERRACE} \((8, [6, 4, 4, 5, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3])\) constraint holds.

Typical

\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]
Figure 4.626: Illustrating the MAX_SURF_DECREASING_TERRACE constraint of the Example slot

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figures 4.627 and 4.628 respectively depict the automaton associated with the constraint MAX_SURF_DECREASING_TERRACE and its simplified form.

Figure 4.627: Automaton for the MAX_SURF_DECREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING_TERRACE pattern where \( \text{default} \) is \(-\infty\)

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \max(\overline{C}, \overline{C}) )</td>
<td>( \max(\overline{C}, \overline{C}) )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \max(\overline{C}, \overline{C}) )</td>
<td>( \overline{D} + \overline{D} + \text{VAR}_{i+1} )</td>
</tr>
<tr>
<td>( t )</td>
<td>( \max(\overline{C}, \overline{C}) )</td>
<td>( \overline{D} + \overline{D} + \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>

Table 4.56: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the MAX_SURF_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature SURF, and the aggregator \( \text{max} \); cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.628: Simplified automaton for the MAX_SURF_DECREASING_TERRACE constraint obtained by applying decoration Table 3.26 to the seed transducer of the DECREASING_TERRACE pattern where default is \(-\infty\); \(R_i - R_{i-1} \geq 0\) is a linear invariant.

Table 4.57: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the simplified automaton of the MAX_SURF_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**Origin**
Based on the `DIP_ON_INCREASING_SEQUENCE` pattern.

**Constraint**
`MAX_SURF_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES)`

**Arguments**
- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`

**Restrictions**
- `sv ≤ 5 ∨ rv ≤ 2 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ 3 * minv + 3`
- `VALUE ≤ 3 * maxv − 3 * VALUE`
- `among(n1, VARIABLES[3, sv − 1], {maxv − 2, maxv − 1, maxv})`
- `n1 ≥ VALUE − 3 − max(maxv − 3, 0)`
- `among(n2, VARIABLES[3, sv − 1], {minv, minv + 1, minv + 2})`
- `n2 ≥ min(minv + 3, 0) − 3 − VALUE`
- `required(VARIABLES, var)`

Where
- `maxv = maxval(VARIABLES.var)`
- `minv = minval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**
`VALUE` is the maximal surface of occurrences of the `DIP_ON_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value −∞.

An occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` is the subsequence which matches the regular expression ‘<<><<’. Assume that the occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 2` to index `j`.

**Example**

```
(10, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))
```

Figure 4.629 provides an example where the `MAX_SURF_DIP_ON_INCREASING_SEQUENCE (10, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4])` constraint holds.

**Typical**
- `|VARIABLES| > 5`
- `range(VARIABLES.var) > 2`
Figure 4.629: Illustrating the MAX_SURF_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot

Arg. properties

**Functional dependency:** VALUE determined by VARIABLES.
Figures 4.630 and 4.631 respectively depict the automaton associated with the constraint \texttt{MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE} and its simplified form.

Figure 4.630: Automaton for the \texttt{MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern where default is $-\infty$. 
Figure 4.631: Simplified automaton for the \texttt{MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.26 to the seed transducer of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern where \texttt{default} is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Based on the GORGE pattern.

\[
\text{MAX\_SURF\_GORGE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- \text{VALUE} : \text{dvar}
- \text{VARIABLES} : \text{collection(var\_dvar)}

**Restricions**

\[
\begin{align*}
sv \leq 2 & \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \\
rv = 2 & \Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} \\
rv \geq 3 & \Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq \min(\text{minv}, (sv - 2) \times (minv + 1) - 1) \\
rv = 2 & \Rightarrow \text{VALUE} \leq \text{maxv} - 1 \\
rv \geq 3 & \Rightarrow \text{VALUE} \leq \max(\text{maxv} - 12, (sv - 2) \times (\text{maxv} - 1) - 13) \\
\text{among}(n1, \text{VARIABLES}[2, sv - 1], (\text{maxv} - 1)) & \\
rv > 2 \land \text{maxv} > 1 & \Rightarrow n1 \geq \text{VALUE} - (sv - 2) \times (\text{maxv} - 2) \\
rv = 2 \lor \text{maxv} = 1 & \Rightarrow n1 \geq \text{VALUE} - \max(0, \text{maxv} - 2) \\
\text{among}(n2, \text{VARIABLES}[2, sv - 1], (\text{minv}, \text{minv} + 1)) & \\
rv > 2 \land \text{minv} < -1 & \Rightarrow n2 \geq (sv - 2) \times (\text{minv} + 2) - 1 - \text{VALUE} \\
rv = 2 \lor \text{minv} = -1 & \Rightarrow n2 \geq \min(0, \text{minv} + 1) - \text{VALUE} \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
\text{maxv} = \text{maxval}(\text{VARIABLES.\text{var}}) & \\
\text{rv} = \text{range}(\text{VARIABLES.\text{var}}) & \\
sv = |\text{VARIABLES}| & \\
\text{minv} = \text{minval}(\text{VARIABLES.\text{var}}) &
\end{align*}
\]

\text{VALUE} is the maximal surface of occurrences of the GORGE pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value \(-\infty\).

**Purpose**

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘\((> | > (> | >)\ast >) (< | (< | <)\ast <)\)’. Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) + 1 to index \(j\).

**Example**

\[(11, (1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7))\]

Figure 4.632 provides an example where the MAX\_SURF\_GORGE \((11, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7])\) constraint holds.
Figure 4.632: Illustrating the `MAX_SURF_GORGE` constraint of the _Example_ slot

**Typical**

\[|\text{VARIABLES}| > 2\]
\[\text{range(VARIABLES.var)} > 1\]

**Symmetry**

Items of `VARIABLES` can be reversed.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`.
Automaton

Figure 4.633 depicts the automaton associated with the constraint MAX_SURF_GORGE.

![Automaton Diagram]

Figure 4.633: Automaton for the MAX_SURF_GORGE constraint obtained by applying decoration Table 3.37 to the seed transducer of the GORGE pattern where default is $-\infty$ (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$); $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.58: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the MAX_SURF_GORGE constraint defined as the composition of the GORGE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin: Based on the INCREASING pattern.

Constraint: \( \text{MAX\_SURF\_INCREASING}(\text{VALUE}, \text{VARIABLES}) \)

Arguments:
- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection}(\text{var}-\text{dvar}) \)

Restrictions:
- \( sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \)
- \( \text{VALUE} = -\infty \lor \text{VALUE} \geq 2 \times \text{minv} + 1 \)
- \( \text{VALUE} \leq 2 \times \text{maxv} - 1 \)
- \( \text{required} (\text{VARIABLES}.\text{var}) \)

Where:
- \( \text{maxv} = \text{maxval} (\text{VARIABLES}.\text{var}) \)
- \( \text{minv} = \text{minval} (\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range} (\text{VARIABLES}.\text{var}) \)

Purpose:
An occurrence of the pattern INCREASING is the subsequence which matches the regular expression ‘<’. Assume that the occurrence of the pattern INCREASING starts at position \( i \) and ends at position \( j \). The feature \( \text{SURF} \) computes the sum of the values from index \( i \) to index \( j + 1 \).

Example:
\( (10, \{4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3\}) \)

Figure 4.634 provides an example where the \( \text{MAX\_SURF\_INCREASING} (10, \{4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3\}) \) constraint holds.

Typical:
- \( |\text{VARIABLES}| > 1 \)
- \( \text{range} (\text{VARIABLES}.\text{var}) > 1 \)

Arg. properties:
Functional dependency: \( \text{VALUE} \) determined by \( \text{VARIABLES} \).
Figure 4.634: Illustrating the MAX_SURF_INCREASING constraint of the Example slot
Automaton

Figures 4.635 and 4.636 respectively depict the automaton associated with the constraint MAX_SURF_INCREASING and its simplified form.

\[
\begin{align*}
\text{max} & (R, \text{default}) \\
& \geq \\
\{ D \leftarrow 0 \} \\
& \{ R \leftarrow \text{max}(R, D + \text{VAR}_i + \text{VAR}_{i+1}) \} \\
\end{align*}
\]

Figure 4.635: Automaton for the MAX_SURF_INCREASING constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING pattern where default is $-\infty$.

\[
\begin{align*}
\{ R \leftarrow \text{default} \} \\
& \leq \\
\{ R \leftarrow \text{max}(R, \text{VAR}_i + \text{VAR}_{i+1}) \} \\
\end{align*}
\]

Figure 4.636: Simplified automaton for the MAX_SURF_INCREASING constraint obtained by applying decoration Table 3.40 to the seed transducer of the INCREASING pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 4.59: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the MAX_SURF_INCREASING constraint defined as the composition of the INCREASING pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.60: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the simplified automaton of the *MAX_SURF_INCREASING* constraint defined as the composition of the *INCREASING* pattern, the feature *SURF*, and the aggregator *max*. Cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MAX_SURF_INCREASING_SEQUENCE**

**Description**

- **Origin**: Based on the **INCREASING_SEQUENCE** pattern.
- **Constraint**: `MAX_SURF_INCREASING_SEQUENCE(VALUE, VARIABLES)`
- **Arguments**:
  - `VALUE : dvar`
  - `VARIABLES : collection(var−dvar)`

**Restrictions**

\[
\begin{align*}
sv &\leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
rv &> 2 \Rightarrow VALUE = -\infty \lor VALUE \geq 2 \times \text{minv} + 1 \\
rv &\geq 3 \Rightarrow VALUE = -\infty \lor VALUE \geq \min(2 \times \text{minv} + 1, sv \times (\text{minv} + 1)) \\
rv &> 2 \Rightarrow VALUE \leq 2 \times \text{maxv} - 1 \\
sv &\geq 3 \Rightarrow VALUE \leq \max(2 \times \text{maxv} - 1, sv \times (\text{maxv} - 1)) \\
\text{among}(n1, \text{VARIABLES}[1, sv], (\text{maxv} - 1, \text{maxv})) \\
rv &> 2 \Rightarrow \text{maxv} = 1 \Rightarrow n1 \geq VALUE = \max(0, 2 \times \text{maxv} - 3) \\
rv &> 2 \Rightarrow \text{maxv} > 1 \Rightarrow n1 \geq VALUE = sv \times (\text{maxv} - 2) - 1 \\
\text{among}(n2, \text{VARIABLES}[1, sv], (\text{minv}, \text{minv} + 1)) \\
rv &> 2 \Rightarrow \text{minv} = -1 \Rightarrow n2 \geq \min(0, 2 \times \text{minv} + 3) - VALUE \\
rv &> 2 \Rightarrow \text{minv} < -1 \Rightarrow n2 \geq sv \times (\text{minv} + 2) - 1 - VALUE \\
\text{required}(\text{VARIABLES}.var)
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} &= \maxval(\text{VARIABLES}.var) \\
\text{rv} &= \text{range}(\text{VARIABLES}.var) \\
sv &= |\text{VARIABLES}| \\
\text{minv} &= \minval(\text{VARIABLES}.var)
\end{align*}
\]

**Purpose**

- **VALUE** is the maximal surface of occurrences of the **INCREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value \(-\infty\).
- **An occurrence of the pattern **INCREASING_SEQUENCE** is the maximal subsequence which matches the regular expression \('< (< | =) < | <'\).
- **Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position i and ends at position j.** The feature **SURF** computes the sum of the values from index i to index j + 1.

**Example**

\[(17, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\]

Figure 4.637 provides an example where the **MAX_SURF_INCREASING_SEQUENCE** constraint holds.
**Typical**  
$|\text{VARIABLES}| > 1$  
$\text{range}(\text{VARIABLES}.) > 1$

**Arg. properties**  
Functional dependency: VALUE determined by VARIABLES.
Figure 4.638 depicts the automaton associated with the constraint `MAX_SURF_INCREASING_SEQUENCE`.

\[
\begin{align*}
\{ C &\leftarrow \text{default} \\ D &\leftarrow 0 \\ R &\leftarrow \text{default} \} \\
\{ C &\leftarrow \text{default} \\ D &\leftarrow 0 \\ R &\leftarrow \max(R, C) \} \\
\{ D &\leftarrow D + \text{VAR}_{i+1} \}
\end{align*}
\]

Figure 4.638: Automaton for the `MAX_SURF_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `INCREASING_SEQUENCE` pattern where `default` is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{C} + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} - \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

Table 4.61: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the `MAX_SURF_INCREASING_SEQUENCE` constraint defined as the composition of the `INCREASING_SEQUENCE` pattern, the feature `SURF`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_SURF_INCREASING_TERRACE

DESCRIPTION

Origin
Based on the INCREASING_TERRACE pattern.

Constraint

\[
\text{MAX\_SURF\_INCREASING\_TERRACE}(\text{VALUE}, \text{VARIABLES})
\]

Arguments

\[
\begin{align*}
\text{VALUE} &: \text{dvar} \\
\text{VARIABLES} &: \text{collection(var} - \text{dvar)}
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} & \leq 3 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} &= -\infty \lor \text{VALUE} \geq \min(2 \ast (\text{minv} + 1), (\text{sv} - 2) \ast (\text{minv} + 1)) \\
\text{VALUE} &\leq \max(2 \ast (\text{maxv} - 1) \ast \text{sv}, (\text{sv} - 2) \ast (\text{maxv} - 1) \ast \text{sv}) \\
\text{among}(n1, \text{VARIABLES}[2, \text{sv} - 1], (\text{maxv} - 1)) &\geq \text{VALUE} \\
\text{among}(n2, \text{VARIABLES}[2, \text{sv} - 1], (\text{minv} + 1)) &\geq \min(0, (\text{sv} - 2) \ast (\text{maxv} - 2)) - \text{VALUE} \\
\text{required}(\text{VARIABLES}, \text{var}) &\geq \text{VALUE} - \max(0, (\text{sv} - 2) \ast (\text{maxv} - 2)) - \text{VALUE}
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

VALUE is the maximal surface of occurrences of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $-\infty$.

Purpose

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression ‘< = + <’. Assume that the occurrence of the pattern INCREASING_TERRACE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

Example

\[(10, (1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4))\]

Figure 4.639 provides an example where the MAX_SURF_INCREASING_TERRACE \[(10, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4])] constraint holds.

Typical

\[
\begin{align*}
|\text{VARIABLES}| &> 3 \\
\text{range}(\text{VARIABLES}.\text{var}) &> 2
\end{align*}
\]
Figure 4.639: Illustrating the MAX_SURF_INCREASING_TERRACE constraint of the Example slot

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.640 and 4.641 respectively depict the automaton associated with the constraint MAX_SURF_INCREASING_TERRACE and its simplified form.

Figure 4.640: Automaton for the MAX_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_TERRACE pattern where default is $-\infty$

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$r$</td>
<td>$t$</td>
</tr>
<tr>
<td>$s$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

Table 4.62: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the MAX_SURF_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.641: Simplified automaton for the MAX_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 3.26 to the seed transducer of the INCREASING_TERRACE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 4.63: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the simplified automaton of the MAX_SURF_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**Origin**
Based on the **INFLEXION** pattern.

**Constraint**
MAX_SURF_INFLEXION(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty \]
\[ VALUE = -\infty \lor VALUE \geq \min(\text{minv}, (sv - 2) \ast \text{minv}) \]
\[ VALUE \leq \max(\text{maxv}, (sv - 2) \ast \text{maxv}) \]
\[ \text{among}(n1, \text{VARIABLES}[2, sv - 1], [\text{maxv}]) \]
\[ n1 \geq VALUE - \max(0, (sv - 2) \ast (\text{maxv} - 1)) \]
\[ \text{among}(n2, \text{VARIABLES}[2, sv - 1], [\text{minv}]) \]
\[ n2 \geq \min(0, (sv - 2) \ast (\text{minv} + 1)) - VALUE \]
\[ \text{required}(\text{VARIABLES}.\text{var}) \]

where
\[ \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \]
\[ \text{sv} = |\text{VARIABLES}| \]
\[ \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \]
\[ \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \]

**Purpose**
VALUE is the maximal surface of occurrences of the **INFLEXION** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern **INFLEXION** is the *maximal* subsequence which matches the regular expression ‘< (< | =)** > | > (>| =)**’. Assume that the occurrence of the pattern **INFLEXION** starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

**Example**
(14, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))

Figure 4.642 provides an example where the MAX_SURF_INFLEXION (14, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4]) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 4.642: Illustrating the MAX_SURF_INFLEXION constraint of the Example slot

**Symmetry**
- Items of VARIABLES can be reversed.

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
Figures 4.643 and 4.644 respectively depict the automaton associated with the constraint \texttt{MAX\_SURF\_INFLEXION} and its simplified form.

Figure 4.643: Automaton for the \texttt{MAX\_SURF\_INFLEXION} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{INFLEXION} pattern where \texttt{default} is $-\infty$ (transition $r \rightarrow t$ has the same registers updates as transition $t \rightarrow r$).

Figure 4.644: Simplified automaton for the \texttt{MAX\_SURF\_INFLEXION} constraint obtained by applying decoration Table 3.26 to the seed transducer of the \texttt{INFLEXION} pattern where \texttt{default} is $-\infty$ (transition $r \rightarrow t$ has the same registers updates as transition $t \rightarrow r$); $R_i - R_{i-1} \geq 0$ is a linear invariant.
MAX_SURF_PEAK

**Origin**
Based on the **PEAK** pattern.

**Constraint**

\[ \text{MAX\_SURF\_PEAK}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(\text{var} - \text{dvar})}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 2 \vee \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} & = -\infty \vee \text{VALUE} \geq \min(\minv + 1, (\text{sv} - 2) \ast (\minv + 1)) \\
\text{VALUE} & \leq \max(\maxv\text{\_v}, (\text{sv} - 2) + \maxv\text{\_v}) \\
\text{among}(\text{n1}, \text{VARIABLES}[2, \text{sv} - 1], [\maxv]) & \\
\text{n1} & > \text{VALUE} - \max(0, (\text{sv} - 2) \ast (\maxv - 1)) \\
\text{among}(\text{n2}, \text{VARIABLES}[2, \text{sv} - 1], [\minv + 1]) & \\
\text{n2} & > \min(0, (\text{sv} - 2) \ast (\minv + 2)) - \text{VALUE} \\
\text{required}(\text{VARIABLES}.\text{var}) & \end{align*}
\]

where

\[
\begin{align*}
\text{minv} & = \text{minval}([\text{VARIABLES}}.\text{var}) \\
\text{maxv} & = \text{maxval}([\text{VARIABLES}}.\text{var}) \\
\text{sv} & = |[\text{VARIABLES}]| \\
\text{rv} & = \text{range}([\text{VARIABLES}.\text{var}])
\end{align*}
\]

**Purpose**

An occurrence of the pattern **PEAK** is the **maximal** subsequence which matches the regular expression \'(\leq | <)\*(\geq | =)\'*.

Assume that the occurrence of the pattern **PEAK** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(14, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1])\]

Figure 4.645 provides an example where the **MAX\_SURF\_PEAK** \((14, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1])\) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]
Figure 4.645: Illustrating the MAX_SURF_PEEK constraint of the Example slot

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figure 4.646 depicts the automaton associated with the constraint MAX_SURF_PEAK.

\[
\begin{align*}
\{ C \leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default} \\
\}
\end{align*}
\geq s
\begin{align*}
\begin{cases}
C &\leftarrow D + \text{VAR}_i \\
D &\leftarrow 0 \\
\end{cases}
\leq \max(R, C)
\end{align*}
\geq t
\begin{align*}
\begin{cases}
C &\leftarrow C + D + \text{VAR}_i \\
D &\leftarrow 0 \\
\end{cases}
\end{align*}
\}
\}
\]

Figure 4.646: Automaton for the MAX_SURF_PEAK constraint obtained by applying decoration Table 3.37 to the seed transducer of the PEAK pattern where \texttt{default} is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\max(C, C)$</td>
<td>$\max(C, C)$</td>
<td>$\max(C, C)$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\max(C, C)$</td>
<td>$D + D + \text{VAR}_{i+1}$</td>
<td>$C$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\max(C, C)$</td>
<td>$C + D + D + \text{VAR}_{i+1}$</td>
<td>$L$</td>
</tr>
</tbody>
</table>

Table 4.64: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the MAX_SURF_PEAK constraint defined as the composition of the PEAK pattern, the feature \texttt{SURF}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin

Based on the PLAIN pattern.

Constraint

\[ \text{MAX\_SURF\_PLAIN}(\text{VALUE}, \text{VARIABLES}) \]

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} \leq 2 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \min(\text{minv}, (\text{sv} - 2) \ast \text{minv}) & \\
\text{VALUE} \leq \max(\text{maxv} - 1, (\text{sv} - 2) \ast (\text{maxv} - 1)) & \\
\text{among}([\text{n1}, \text{VARIABLES}[2, \text{sv} - 1], (\text{maxv} - 1)] & \\
\text{n1} \geq \text{VALUE} - \max(0, (\text{sv} - 2) \ast (\text{maxv} - 2)) & \\
\text{among}([\text{n2}, \text{VARIABLES}[2, \text{sv} - 1], \text{minv}] & \\
\text{n2} \geq \min(0, (\text{sv} - 2) \ast (\text{minv} + 1)) - \text{VALUE} & \\
\text{required}(\text{VARIABLES}, \text{var}) &
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) & \\
\text{sv} = |\text{VARIABLES}| & \\
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) & \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var}) &
\end{align*}
\]

\[\text{VALUE} \leq 8 = (10 - 2) \ast (2 - 1)\]

VALUE is the maximal surface of occurrences of the PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

Purpose

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘\(>=' '<\)’.

Assume that the occurrence of the pattern PLAIN starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example

\[(6, (2, 3, 6, 5, 7, 6, 6, 4, 5, 4, 3, 3, 6, 6, 3))\]

Figure 4.647 provides an example where the MAX\_SURF\_PLAIN \((6, [2, 3, 6, 5, 7, 6, 6, 4, 5, 4, 3, 3, 6, 6, 3])\) constraint holds.

Typical

\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]
Figure 4.647: Illustrating the MAX_SURF.PLAIN constraint of the Example slot

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.648 and 4.649 respectively depict the automaton associated with the constraint MAX_SURF_PLAIN and its simplified form.

Figure 4.648: Automaton for the MAX_SURF_PLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLAIN pattern where default is $-\infty$.

Figure 4.649: Simplified automaton for the MAX_SURF_PLAIN constraint obtained by applying decoration Table 3.26 to the seed transducer of the PLAIN pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.65: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the MAX_SURF_PLAIN constraint defined as the composition of the PLAIN pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\max(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overleftarrow{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\max(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1}$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>t</td>
<td>$\max(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1}$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

Table 4.66: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the simplified automaton of the MAX_SURF_PLAIN constraint defined as the composition of the PLAIN pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
</tr>
<tr>
<td>r</td>
<td>$-\infty$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1}$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>t</td>
<td>$-\infty$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1}$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>
Origin Based on the PLATEAU pattern.

Constraint \[
\text{MAX\_SURF\_PLATEAU} \left( \text{VALUE}, \text{VARIABLES} \right)
\]

Arguments

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

Restrictions

- \[ sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \]
- \[ \text{VALUE} = -\infty \lor \text{VALUE} \geq \min(\minv + 1, (sv - 2) + (\minv + 1)) \]
- \[ \text{VALUE} \leq \max(\maxv, (sv - 2) + \maxv) \]
- \[ \text{among}(n1, \text{VARIABLES}[2, sv - 1], \langle \maxv \rangle) \]
- \[ n1 \geq \text{VALUE} - \max(0, (sv - 2) + (\maxv - 1)) \]
- \[ \text{among}(n2, \text{VARIABLES}[2, sv - 1], \langle \minv + 1 \rangle) \]
- \[ n2 \geq \min(0, (sv - 2) + (\minv + 2)) - \text{VALUE} \]

where

- \[ \maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \]
- \[ sv = |\text{VARIABLES}| \]
- \[ \minv = \text{minval}(\text{VARIABLES}.\text{var}) \]
- \[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

VALUE is the maximal surface of occurrences of the PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

Purpose

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression '\(<\,=\,*\,>\)'.
Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example

\[
(10, (7, 5, 2, 3, 1, 2, 2, 4, 3, 4, 5, 5, 2, 2, 5))
\]

Figure 4.650 provides an example where the \text{MAX\_SURF\_PLATEAU} \((10, [7, 5, 2, 3, 1, 2, 2, 4, 3, 4, 5, 5, 2, 2, 5])\) constraint holds.

Typical

- \[ |\text{VARIABLES}| > 2 \]
- \[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 4.650: Illustrating the MAX_SURF_PLATEAU constraint of the Example slot.

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.651 and 4.652 respectively depict the automaton associated with the constraint MAX_SURF_PLATEAU and its simplified form.

Figure 4.651: Automaton for the MAX_SURF_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLATEAU pattern where default is $-\infty$.

Figure 4.652: Simplified automaton for the MAX_SURF_PLATEAU constraint obtained by applying decoration Table 3.26 to the seed transducer of the PLATEAU pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
### Table 4.67: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the MAX_SURF_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
<th>( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \max(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \max(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \max(\overrightarrow{C}, \overrightarrow{C}) )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \max(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
</tr>
<tr>
<td>( t )</td>
<td>( \max(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>

### Table 4.68: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the simplified automaton of the MAX_SURF_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
<th>( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( -\infty )</td>
<td>( -\infty )</td>
<td>( -\infty )</td>
</tr>
<tr>
<td>( r )</td>
<td>( -\infty )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
</tr>
<tr>
<td>( t )</td>
<td>( -\infty )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>
Origin
Based on the PROPERPLAIN pattern.

Constraint
MAX_SURF_PROPER_PLAIN(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ min(2 + minv, (sv − 2) + minv)
VALUE ≤ max(2 + (maxv − 1)x, (sv − 2) + (maxv − 1)y)
   among(n1, VARIABLES[2, sv − 1], (maxv − 1))
n1 ≥ VALUE − max(0, (sv − 2) + (maxv − 2))
   among(n2, VARIABLES[2, sv − 1], (minv))
n2 ≥ min(0, (sv − 2) + (minv + 1)) − VALUE
required(VARIABLES, var)
where
maxv =maxval(VARIABLES.var)
sv = |VARIABLES|
minv =minval(VARIABLES.var)
rv =range(VARIABLES.var)

VALUE is the maximal surface of occurrences of the PROPER_PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

Purpose
An occurrence of the pattern PROPER_PLAIN is the maximal subsequence which matches the regular expression ’> = + <’.
Assume that the occurrence of the pattern PROPER_PLAIN starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
(10, ⟨2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3⟩)

Figure 4.653 provides an example where the MAX_SURF_PROPER_PLAIN (10, ⟨2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3⟩) constraint holds.

Typical
|VARIABLES| > 3
range(VARIABLES.var) > 1
Figure 4.653: Illustrating the MAX_SURF_PROPER_plain constraint of the **Example** slot

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.654 and 4.655 respectively depict the automaton associated with the constraint MAX_SURF_PROPER PLAIN and its simplified form.

Figure 4.654: Automaton for the MAX_SURF_PROPER_PLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER_PLAIN pattern where default is $-\infty$.

Figure 4.655: Simplified automaton for the MAX_SURF_PROPER_PLAIN constraint obtained by applying decoration Table 3.26 to the seed transducer of the PROPER_PLAIN pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
### Table 4.69: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the **MAX_SURF_PROPER.PLAIN** constraint defined as the composition of the **PROPER.PLAIN** pattern, the feature **SURF**, and the aggregator **max**; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th><strong>s</strong></th>
<th><strong>r</strong></th>
<th><strong>t</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>s</strong></td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
</tr>
<tr>
<td><strong>r</strong></td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
<tr>
<td><strong>t</strong></td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

### Table 4.70: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the simplified automaton of the **MAX_SURF_PROPER.PLAIN** constraint defined as the composition of the **PROPER.PLAIN** pattern, the feature **SURF**, and the aggregator **max**; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th><strong>s</strong></th>
<th><strong>r</strong></th>
<th><strong>t</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>s</strong></td>
<td>$-\infty$</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
</tr>
<tr>
<td><strong>r</strong></td>
<td>$-\infty$</td>
<td>$-\infty$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
<tr>
<td><strong>t</strong></td>
<td>$-\infty$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>
**Origin**
Based on the \texttt{PROPER\_PLATEAU} pattern.

**Constraint**
\texttt{MAX\_SURF\_PROPER\_PLATEAU}(\texttt{VALUE, VARIABLES})

**Arguments**
\begin{itemize}
  \item \texttt{VALUE} : \texttt{dvar}
  \item \texttt{VARIABLES} : \texttt{collection(var–dvar)}
\end{itemize}

**Restrictions**
\begin{align*}
  sv \leq 3 \land rv \leq 1 \Rightarrow \text{VALUE} &= -\infty \\
  \text{VALUE} &= -\infty \lor \text{VALUE} \geq \min(2 * (\text{minv} + 1), (sv - 2) * (\text{minv} + 1)) \\
  \text{VALUE} &\leq \max(2 * \text{maxv}, (sv - 2) * \text{maxv}) \\
  \text{among}(n1, \text{VARIABLES}[sv - 1], \langle \text{maxv} \rangle) \\
  n1 &\geq \text{VALUE} - \max(0, (sv - 2) * (\text{maxv} - 1)) \\
  \text{among}(n2, \text{VARIABLES}[sv - 1], \langle \text{minv} + 1 \rangle) \\
  n2 &\geq \min(0, (sv - 2) * (\text{minv} + 2)) - \text{VALUE} \\
  \text{required}(\text{VARIABLES}, \text{var})
\end{align*}

where
\begin{align*}
  \text{maxv} &= \maxval(\text{VARIABLES}.\text{var}) \\
  \text{sv} &= |\text{VARIABLES}| \\
  \text{minv} &= \minval(\text{VARIABLES}.\text{var}) \\
  \text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}

**Purpose**
\begin{itemize}
  \item \text{VALUE} is the maximal surface of occurrences of the \texttt{PROPER\_PLATEAU} pattern in the time-series given by the \texttt{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value \text{\(-\infty\)}. \\
  \item An occurrence of the pattern \texttt{PROPER\_PLATEAU} is the \textit{maximal} subsequence which matches the regular expression `\texttt{< =+ >}`. Assume that the occurrence of the pattern \texttt{PROPER\_PLATEAU} starts at position \(i\) and ends at position \(j\). The feature \texttt{SURF} computes the sum of the values from index \(i+1\) to index \(j\).
\end{itemize}

**Example**
\begin{itemize}
  \item \((15, \langle 7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3 \rangle)\)
\end{itemize}

\begin{figure} \centering
\includegraphics[width=\textwidth]{example}
\caption{Example of \texttt{MAX\_SURF\_PROPER\_PLATEAU} with \((15, \langle 7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3 \rangle)\) constraint holds.}
\end{figure}

**Typical**
\begin{itemize}
  \item \(|\text{VARIABLES}| > 3\)
  \item \text{range}(\text{VARIABLES}.\text{var}) > 1
\end{itemize}
Figure 4.656: Illustrating the MAX_SURF_PROPER_PLATEAU constraint of the Example slot

Symmetry: Items of VARIABLES can be reversed.

Arg. properties: Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.657 and 4.658 respectively depict the automaton associated with the constraint `MAX_SURF_PROPER_PLATEAU` and its simplified form.

![Automaton Diagram](image)

Figure 4.657: Automaton for the `MAX_SURF_PROPER_PLATEAU` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `PROPER_PLATEAU` pattern where `default` is $-\infty$.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$r$</td>
<td>$t$</td>
</tr>
<tr>
<td>$s$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \overrightarrow{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

Table 4.71: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the `MAX_SURF_PROPER_PLATEAU` constraint defined as the composition of the `PROPER_PLATEAU` pattern, the feature `SURF`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.658: Simplified automaton for the MAX_SURF_PROPER_PLATEAU constraint obtained by applying decoration Table 3.26 to the seed transducer of the PROPER_PLATEAU pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
</tr>
<tr>
<td>$r$</td>
<td>$-\infty$</td>
<td>$-\infty$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>$t$</td>
<td>$-\infty$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

Table 4.72: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the simplified automaton of the MAX_SURF_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### Description

**Origin**
Based on the `STEADY` pattern.

**Constraint**

\[
\text{MAX}_\text{SURF}_\text{STEADY}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty & \lor \text{VALUE} \geq 2 + \text{minv} \\
\text{VALUE} \leq 2 & \times \text{maxv} \\
\text{among}([n1, \text{VARIABLES}[1, \text{sv}], (\text{maxv})]) \\
\text{n1} \geq \text{VALUE} - 2 & \times (\text{maxv} - 1) \\
\text{among}([n2, \text{VARIABLES}[1, \text{sv}], (\text{minv})]) \\
\text{n2} \geq 2 & \times (\text{minv} + 1) - \text{VALUE} \\
\text{required}(&\text{VARIABLES}.\text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}|
\end{align*}
\]

**Purpose**

An occurrence of the pattern `STEADY` is the subsequence which matches the regular expression `'=.'`. Assume that the occurrence of the pattern `STEADY` starts at position \(i\) and ends at position \(j\). The feature \(\text{SURF}\) computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
(12, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6))
\]

Figure 4.659 provides an example where the `MAX_SURF_STEADY` \((12, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6])\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]

**Symmetry**

Items of \(\text{VARIABLES}\) can be reversed.

**Arg. properties**

Functional dependency: \(\text{VALUE}\) determined by \(\text{VARIABLES}\).
Figure 4.659: Illustrating the MAX_SURF_STEADY constraint of the Example slot
Automaton

Figures 4.660 and 4.661 respectively depict the automaton associated with the constraint MAX_SURF_STEADY and its simplified form.

\[ \begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*} \]

\[ R \leftarrow \max(R, C) \]

\[ D = \{ \begin{align*}
R &\leftarrow 0 \\
R &\leftarrow \max(R, D + \text{VAR}_i + \text{VAR}_{i+1})
\end{align*} \]

Figure 4.660: Automaton for the MAX_SURF_STEADY constraint obtained by applying decoration Table 3.37 to the seed transducer of the STEADY pattern where default is \( -\infty \).

\[ \{ R \leftarrow \text{default} \} \]

\[ \{ R \leftarrow \max(R, \text{VAR}_i + \text{VAR}_{i+1}) \} \]

Figure 4.661: Simplified automaton for the MAX_SURF_STEADY constraint obtained by applying decoration Table 3.40 to the seed transducer of the STEADY pattern where default is \( -\infty \); \( R_i - R_{i-1} \geq 0 \) is a linear invariant.

Table 4.73: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the MAX_SURF_STEADY constraint defined as the composition of the STEADY pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.74: Concrete glue matrix, derived from the parametrised glue matrix \(3.16\), for the simplified automaton of the MAX_SURF_STEADY constraint defined as the composition of the \textit{STEADY} pattern, the feature \textit{SURF}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Based on the `STEADY_SEQUENCE` pattern.

**Constraint**

\[
\text{MAX}_\text{SURF}_\text{STEADY}_\text{SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- \text{VALUE} : dvar
- \text{VARIABLES} : \text{collection}(\var\text{−dvar})

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{rv} = 1 \Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq \text{sv} \times \text{minv} \\
\text{rv} \geq 2 \Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq \min(2 \times \text{minv}, \text{sv} \times \text{minv}) \\
\text{rv} = 1 \Rightarrow \text{VALUE} \leq \text{sv} \times \text{maxv} \\
\text{rv} \geq 2 \Rightarrow \text{VALUE} \leq \max(2 \times \text{maxv}, \text{sv} \times \text{maxv}) \\
\text{among}(n1, \text{VARIABLES}[1, \text{sv}], (\text{maxv})) \\
\text{n1} > \text{VALUE} - \max(0, \text{sv} \times (\text{maxv} - 1)) \\
\text{among}(n2, \text{VARIABLES}[2, \text{sv}], (\text{minv})) \\
\text{n2} \geq \min(0, \text{sv} \times (\text{minv} + 1)) - \text{VALUE} \\
\text{required}(\text{VARIABLES, var})
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES, var}) \\
\text{rv} &= \text{range}(\text{VARIABLES, var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{minv} &= \text{minval}(\text{VARIABLES, var})
\end{align*}
\]

**Example**

\[
(15, [3, 1, 1, 4, 5, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1])
\]

Figure 4.662 provides an example where the `MAX_SURF_STEADY_SEQUENCE` \((15, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1])\) constraint holds.

**Purpose**

\text{VALUE} is the maximal surface of occurrences of the `STEADY_SEQUENCE` pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value \(-\infty\).

An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression `\(=+\)`.

Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature \text{SURF} computes the sum of the values from index \(i\) to index \(j+1\).

**Typical**

\(|\text{VARIABLES}| > 1\)
Figure 4.662: Illustrating the MAX_SURF_STEADY_SEQUENCE constraint of the Example slot

**Symmetry**: Items of VARIABLES can be reversed.

**Arg. properties**: Functional dependency: VALUE determined by VARIABLES.
Figures 4.663 and 4.664 respectively depict the automaton associated with the constraint `MAX_SURF_STEADY_SEQUENCE` and its simplified form.

Figure 4.663: Automaton for the `MAX_SURF_STEADY_SEQUENCE` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `STEADY_SEQUENCE` pattern where `default` is $-\infty$.

Figure 4.664: Simplified automaton for the `MAX_SURF_STEADY_SEQUENCE` constraint obtained by applying decoration Table 3.25 to the seed transducer of the `STEADY_SEQUENCE` pattern where `default` is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.75: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the \text{MAX} \_ \text{SURF} \_ \text{STEADY} \_ \text{SEQUENCE} constraint defined as the composition of the \text{STEADY} \_ \text{SEQUENCE} pattern, the feature \text{SURF}, and the aggregator \text{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>\textit{s}</th>
<th>\textit{r}</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{s}</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
</tr>
<tr>
<td>\textit{r}</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\vec{C} - \vec{C} + \vec{D} + \vec{D} - \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

Table 4.76: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the simplified automaton of the \text{MAX} \_ \text{SURF} \_ \text{STEADY} \_ \text{SEQUENCE} constraint defined as the composition of the \text{STEADY} \_ \text{SEQUENCE} pattern, the feature \text{SURF}, and the aggregator \text{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>\textit{s}</th>
<th>\textit{r}</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{s}</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
</tr>
<tr>
<td>\textit{r}</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\vec{C} - \vec{C} - \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>
**MAX_SURF_STRICTLY_DECREASING_SEQUENCE**

**Description**
Based on the **STRICTLY_DECREASING_SEQUENCE** pattern.

**Arguments**
- `VALUE : dvar`
- `VARIABLES : collection(var−dvar)`

**Restrictions**
- `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = −∞`
- `minv < 0 ⇒ VALUE = −∞ ∨ VALUE ≥ ℓ₁ * minv + ⌊ℓ₁ * (ℓₑ − 1)/2⌋`
- `minv ≥ 0 ⇒ VALUE = −∞ ∨ VALUE ≥ 2 * minv + 1`
- `maxv ≥ 0 ⇒ VALUE ≤ ℓ₂ * maxv − ⌊ℓ₂ * (ℓ₂ − 1)/2⌋`.
- `maxv < 0 ⇒ VALUE ≤ 2 * maxv − 12` among
  - `n₁ ≥ VALUE = max(0, min(sv,rv) * (maxv − 2)) − 1` among
    - `n₂ ≥ min(0, min(sv,rv) * (minv + 2)) − 1 − VALUE`

where
- `maxv = maxval(VARIABLES.var)`
- `rv = range(VARIABLES.var)`
- `sv = |VARIABLES|`
- `ℓ₁ = min(min(sv,rv), |minv|)`
- `ℓ₂ = min(min(sv,rv), |maxv|)`
- `minv = minval(VARIABLES.var)`

**Purpose**
An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the *maximal* sub-sequence which matches the regular expression ‘‘’. Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position `i` and ends at position `j`. The feature **SURF** computes the sum of the values from index `i` to index `j + 1`.

**Example**
- `(13, 4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))`

Figure 4.665 provides an example where the `MAX_SURF_STRICTLY_DECREASING_SEQUENCE`
- `(13, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3])` constraint holds.
Figure 4.665: Illustrating the \texttt{MAX\_SURF\_STRICTLY\_DECREASING\_SEQUENCE} constraint of the \textbf{Example} slot

\begin{itemize}
  \item **Typical**
    \begin{itemize}
      \item $|\text{VARIABLES}| > 1$
      \item $\text{range(}\text{VARIABLES.var}\text{)} > 1$
    \end{itemize}
  \item **Arg. properties**
    \begin{itemize}
      \item Functional dependency: VALUE determined by VARIABLES.
    \end{itemize}
\end{itemize}
Figures 4.666 and 4.667 respectively depict the automaton associated with the constraint \text{MAX\_SURF\_STRICTLY\_DECREASING\_SEQUENCE} and its simplified form.

Figure 4.666: Automaton for the \text{MAX\_SURF\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \text{STRICTLY\_DECREASING\_SEQUENCE} pattern where \text{default} is $-\infty$.

Figure 4.667: Simplified automaton for the \text{MAX\_SURF\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.25 to the seed transducer of the \text{STRICTLY\_DECREASING\_SEQUENCE} pattern where \text{default} is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.77: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the MAX_SURF_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\max(\vec{C}, \vec{C}))</td>
<td>(\max(\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\max(\vec{C}, \vec{C}))</td>
<td>(\vec{C} + \vec{C} + \vec{D} + \vec{D} - \text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>

Table 4.78: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the MAX_SURF_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\max(\vec{C}, \vec{C}))</td>
<td>(\max(\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\max(\vec{C}, \vec{C}))</td>
<td>(\vec{C} + \vec{C} - \text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>
Based on the `STRICTLY_INCREASING_SEQUENCE` pattern.

**Constraints**

```
MAX_SURF_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

**Restrictions**

```
sv ≤ 1 \lor \ n1 ≤ 1 \Rightarrow VALUE = −∞
minv < 0 \Rightarrow VALUE = −∞ \lor VALUE ≥ ℓ1 * minv + ⌊ℓ1 * (ℓ1 − 1)/2⌋
minv ≥ 0 \Rightarrow VALUE = −∞ \lor VALUE ≥ 2 * minv + 1
maxv > 0 \Rightarrow VALUE ≤ ℓ2 * maxv − ⌊ℓ2 * (ℓ2 − 1)/2⌋
maxv ≤ 0 \Rightarrow VALUE ≤ 2 * maxv − 1
\forall n1, VARIABLES[1, sv], (maxv − 1, maxv)
\forall n1 ≥ VALUE − max(0, min(sv, rv) * (maxv − 2)) − 1
\forall n2 ≥ min(0, min(sv, rv) * (minv + 2)) − 1 − VALUE
```

**Purpose**

Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression ‘<’.

Figure 4.668 provides an example where the `MAX_SURF_STRICTLY_INCREASING_SEQUENCE` constraint holds.

```
(16, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))
```

**Example**

VALUE is the maximal surface of occurrences of the `STRICTLY_INCREASING_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $-\infty$. An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `SURF` computes the sum of the values from index $i$ to index $j+1$. 

```
MAX_SURF_STRICTLY_INCREASING_SEQUENCE
```

**Description**

```
<table>
<thead>
<tr>
<th>Aggregator</th>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>MAX</code></td>
<td><code>VALUE</code></td>
<td><code>STRICTLY_INCREASING_SEQUENCE</code></td>
</tr>
</tbody>
</table>
```

**Automaton**

![Automaton Diagram](image-url)
Figure 4.668: Illustrating the MAX_SURF_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot

Typical

$|\text{VARIABLES}| > 1$

$\text{range} (\text{VARIABLES}\.\text{var}) > 1$

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figures 4.669 and 4.670 respectively depict the automaton associated with the constraint MAX_SURF_STRICTLY_INCREASING_SEQUENCE and its simplified form.

Figure 4.669: Automaton for the MAX_SURF_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is $-\infty$.

Figure 4.670: Simplified automaton for the MAX_SURF_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.25 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.79: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the MAX_SURF_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASINGSEQUENCE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\[
\begin{array}{ccc}
\text{s} & \text{r} \\
\text{s} & \max(\overrightarrow{C}, \overrightarrow{C}) & \max(\overrightarrow{C}, \overrightarrow{C}) \\
\text{r} & \max(\overrightarrow{C}, \overrightarrow{C}) & \overrightarrow{C} + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} - \text{VAR}_{i+1} \text{ M}
\end{array}
\]

Table 4.80: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the simplified automaton of the MAX_SURF_STRICTLY_INCREASINGSEQUENCE constraint defined as the composition of the STRICTLY_INCREASINGSEQUENCE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\[
\begin{array}{ccc}
\text{s} & \text{r} \\
\text{s} & \max(\overrightarrow{C}, \overrightarrow{C}) & \max(\overrightarrow{C}, \overrightarrow{C}) \\
\text{r} & \max(\overrightarrow{C}, \overrightarrow{C}) & \overrightarrow{C} + \overrightarrow{C} - \text{VAR}_{i+1} \text{ M}
\end{array}
\]
MAX_SURF_SUMMIT

**Description Automaton**

**Origin**
Based on the SUMMIT pattern.

**Constraint**
\[ \text{MAX\_SURF\_SUMMIT}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**
- \text{VALUE} : \text{dvar}
- \text{VARIABLES} : \text{collection}([\text{var} \ldots \text{dvar}])

**Restrictions**
\[
\begin{align*}
\text{sv} \leq 2 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{rv} = 2 & \Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1 \\
\text{rv} \geq 3 & \Rightarrow \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq \min(\text{minv} + 1, (\text{sv} - 2) \cdot (\text{minv} + 1) + 1) \\
\text{rv} = 2 & \Rightarrow \text{VALUE} \leq \text{maxv} \\
\text{rv} \geq 3 & \Rightarrow \text{VALUE} \leq \max(\text{maxv}, (\text{sv} - 2) \cdot (\text{maxv} - 1) + 1) \\
\text{among} & ([n1, \text{VARIABLES}][2, \text{sv} - 1], [\text{maxv} - 1, \text{maxv}]) \\
\text{rv} = 2 & \lor \text{maxv} = 1 \Rightarrow n1 \geq \text{VALUE} - \max(0, \text{maxv} - 1) \\
\text{rv} > 2 & \land \text{maxv} > 1 \Rightarrow n1 \geq \text{VALUE} - (\text{sv} - 2) \cdot (\text{maxv} - 2) - 1 \\
\text{among} & ([n2, \text{VARIABLES}][2, \text{sv} - 1], [\text{minv} + 1]) \\
\text{rv} = 2 & \lor \text{minv} = -1 \Rightarrow n2 \geq \min(0, \text{minv} + 2) - \text{VALUE} \\
\text{rv} > 2 & \land \text{minv} < -1 \Rightarrow n2 \geq (\text{sv} - 2) \cdot (\text{minv} + 2) - \text{VALUE} \\
\text{required} & (\text{VARIABLES}, \text{var})
\end{align*}
\]

where
- \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
- \text{rv} = \text{range}(\text{VARIABLES}.\text{var})
- \text{sv} = |\text{VARIABLES}|
- \text{minv} = \text{minval}(\text{VARIABLES}.\text{var})

**Purpose**
VALUE is the maximal surface of occurrences of the SUMMIT pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression (\(< | < (= | <)^* (< | > (= | >)^*)>\)).

Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+1\) to index \(j\).

**Example**
\[(13, \langle 7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1 \rangle)\]

Figure 4.671 provides an example where the MAX_SURF_SUMMIT \((13, \langle 7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1 \rangle)\) constraint holds.
Figure 4.671: Illustrating the MAX_SURF_SUMMIT constraint of the Example slot

Typical
\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

Symmetry
Items of VARIABLES can be reversed.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Automaton

Figure 4.672 depicts the automaton associated with the constraint MAX_SURF_SUMMIT.

\[
\begin{align*}
C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
\geq \quad \max(R, C) \\
\geq s \\
\geq r \\
\geq t \\
\geq \langle \{D \leftarrow D + \text{VAR}_i\}\rangle
\end{align*}
\]

\[
\begin{align*}
\langle \{D \leftarrow D + \text{VAR}_i\}\rangle \\
\langle \{C \leftarrow D + \text{VAR}_i\}, D \leftarrow 0\rangle
\end{align*}
\]

\[
\begin{align*}
\geq \langle \{D \leftarrow D + \text{VAR}_i\}\rangle \\
\geq \langle \{C \leftarrow D + \text{VAR}_i\}, D \leftarrow 0\rangle
\end{align*}
\]

\[
\begin{align*}
\langle \{D \leftarrow D + \text{VAR}_i\}\rangle \\
\langle \{C \leftarrow D + \text{VAR}_i\}, D \leftarrow 0\rangle
\end{align*}
\]

\[
\begin{align*}
\langle \{D \leftarrow D + \text{VAR}_i\}\rangle \\
\langle \{C \leftarrow D + \text{VAR}_i\}, D \leftarrow 0\rangle
\end{align*}
\]

\[
\begin{align*}
\langle \{D \leftarrow D + \text{VAR}_i\}\rangle \\
\langle \{C \leftarrow D + \text{VAR}_i\}, D \leftarrow 0\rangle
\end{align*}
\]

\[
\begin{align*}
\langle \{D \leftarrow D + \text{VAR}_i\}\rangle \\
\langle \{C \leftarrow D + \text{VAR}_i\}, D \leftarrow 0\rangle
\end{align*}
\]

Figure 4.672: Automaton for the MAX_SURF_SUMMIT constraint obtained by applying decoration Table 3.37 to the seed transducer of the SUMMIT pattern where default is $-\infty$ (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$); $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.81: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the \texttt{MAX\_SURF\_SUMMIT} constraint defined as the composition of the \texttt{SUMMIT} pattern, the feature \texttt{SURF}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max((\vec{C}, \vec{C}))</td>
<td>max((\vec{C}, \vec{C}))</td>
<td>max((\vec{C}, \vec{C}))</td>
<td>max((\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\vec{C} + \vec{D} + \text{VAR}_{i+1})</td>
<td>(\vec{C} + \vec{D} + \text{VAR}_{i+1})</td>
<td>(\vec{C} + \vec{D} + \text{VAR}_{i+1})</td>
<td>max((\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>t</td>
<td>(\vec{C} + \vec{D} + \text{VAR}_{i+1})</td>
<td>max((\vec{C}, \vec{C}))</td>
<td>(\vec{C} + \vec{D} + \text{VAR}_{i+1})</td>
<td>max((\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>u</td>
<td>max((\vec{C}, \vec{C}))</td>
<td>max((\vec{C}, \vec{C}))</td>
<td>(\vec{C} + \vec{D} + \text{VAR}_{i+1})</td>
<td>max((\vec{C}, \vec{C}))</td>
</tr>
</tbody>
</table>
**Origin**
Based on the VALLEY pattern.

**Constraint**
\[
\text{MAX\_SURF\_VALLEY}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**
\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**
\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq \min(\text{minv}, (\text{sv} - 2) \times \text{minv}) \\
\text{VALUE} & \leq \max(\text{maxv} - 1, (\text{sv} - 2) \times (\text{maxv} - 1)) \\
& \quad \text{among}(\text{n1}, \text{VARIABLES}[2, \text{sv} - 1], (\text{maxv} - 1)) \\
\text{n1} \geq \text{VALUE} & \geq \max(0, (\text{sv} - 2) \times (\text{maxv} - 2)) \\
& \quad \text{among}(\text{n2}, \text{VARIABLES}[2, \text{sv} - 1], (\text{minv})) \\
\text{n2} \geq \min(0, (\text{sv} - 2) \times (\text{minv} + 1)) & - \text{VALUE} \\
\text{required}(\text{VARIABLES}, \text{var}) & \text{where} \\
\text{maxv} & = \maxval(\text{VARIABLES}, \text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{minv} & = \minval(\text{VARIABLES}, \text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}, \text{var})
\end{align*}
\]

\text{VALUE} is the maximal surface of occurrences of the VALLEY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \text{VALUE} takes the default value \(-\infty\).

**Purpose**
An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \( > (= | >)(< | =)^* < \).
Assume that the occurrence of the pattern VALLEY starts at position \(i\) and ends at position \(j\). The feature \text{SURF} computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**
\[
(15, (1, 3, 7, 4, 3, 6, 5, 3, 2, 6, 5, 5, 7))
\]

Figure 4.673 provides an example where the \text{MAX\_SURF\_VALLEY} \((15, [1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7])\) constraint holds.

**Typical**
\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}, \text{var}) > 1
\]
Figure 4.673: Illustrating the MAX_SURF_VALLEY constraint of the Example slot

Symmetry: Items of VARIABLES can be reversed.

Arg. properties: Functional dependency: VALUE determined by VARIABLES.
Automaton

Figure 4.674 depicts the automaton associated with the constraint MAX_SURF_VALLEY.

Figure 4.674: Automaton for the MAX_SURF_VALLEY constraint obtained by applying decoration Table 3.37 to the seed transducer of the VALLEY pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 4.82: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the MAX_SURF_VALLEY constraint defined as the composition of the VALLEY pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_SURF_ZIGZAG

MAX_SURF_ZIGZAG

DESCRIPTION AUTOMATON

Origin

Based on the ZIGZAG pattern.

Constraint

MAX_SURF_ZIGZAG(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var − dvar)

Restrictions

\[
sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = -\infty
\]

\[
VALUE \geq \min \left(2 \cdot \text{minv} + 1, \left[\frac{sv - 1}{2}\right] \cdot \text{minv} + \left[\frac{sv - 2}{2}\right] \cdot (\text{minv} + 1)\right)
\]

\[
VALUE \leq \max \left(2 \cdot \text{maxv} - 1 \cdot \text{minv}, \left[\frac{sv - 1}{2}\right] \cdot \text{maxv} + \left[\frac{sv - 2}{2}\right] \cdot (\text{maxv} - 1)\right)
\]

among(n1, VARIABLES[2, sv - 1], [maxv - 1, maxv])

\[
n1 \geq VALUE - \left[\frac{sv - 1}{2}\right] - \max(0, (sv - 2) \cdot (\text{maxv} - 2))
\]

among(n2, VARIABLES[2, sv - 1], [minv, minv + 1])

\[
n2 \geq \min(0, (sv - 2) \cdot (\text{minv} + 2)) - \left[\frac{sv - 1}{2}\right] - VALUE
\]

required(VARIABLES, var)

where

\[
\text{minv} = \text{minval}(\text{VARIABLES} \cdot \text{var})
\]

\[
\text{maxv} = \text{maxval}(\text{VARIABLES} \cdot \text{var})
\]

\[
sv = |\text{VARIABLES}|
\]

\[
rv = \text{range}(\text{VARIABLES} \cdot \text{var})
\]

VALUE is the maximal surface of occurrences of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

Purpose

An occurrence of the pattern ZIGZAG is the \textit{maximal} subsequence which matches the regular expression \((<<)\left((<<)\right)\). Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example

\((21, \langle 4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1 \rangle)\)

Figure 4.675 provides an example where the MAX_SURF_ZIGZAG (21,[4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1]) constraint holds.
Figure 4.675: Illustrating the MAX_SURF_ZIGZAG constraint of the Example slot

Typical

\[
|\text{VARIABLES}| > 3 \\
\text{range(\text{VARIABLES}\.var)} > 1
\]

Symmetry

Items of \text{VARIABLES} can be reversed.

Arg. properties

Functional dependency: VALUE determined by \text{VARIABLES}. 
Figures 4.676 and 4.677 respectively depict the automaton associated with the constraint MAX_SURF_ZIGZAG and its simplified form.
Figure 4.676: Automaton for the MAX_SURF_ZIGZAG constraint obtained by applying decoration Table 3.37 to the seed transducer of the ZIGZAG pattern where default is $-\infty$: (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.
Figure 4.677: Simplified automaton for the MAX_SURF_ZIGZAG constraint obtained by applying decoration Table 3.24 to the seed transducer of the ZIGZAG pattern where default is $-\infty$; missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $\equiv$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 4.83: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the MAX_SURF_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.84: Concrete glue matrix derived from the parametrised glue matrix 3.22 for the simplified automaton of the MAX-SURF-ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
</tr>
<tr>
<td>a</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
</tr>
<tr>
<td>b</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>( \mathcal{B} + \mathcal{B} + \text{VAR}_{i+1} )</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
</tr>
<tr>
<td>c</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>( \mathcal{C} + \mathcal{C} + \mathcal{B} + \mathcal{B} + \text{VAR}_{i+1} )</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
</tr>
<tr>
<td>d</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>( \mathcal{C} + \mathcal{C} + \mathcal{B} + \mathcal{B} + \text{VAR}_{i+1} )</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
</tr>
<tr>
<td>e</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>( \mathcal{C} + \mathcal{B} + \text{VAR}_{i+1} )</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
</tr>
<tr>
<td>f</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
<td>max(( \mathcal{C}, \mathcal{C} ))</td>
</tr>
</tbody>
</table>
**MAX_WIDTH_DECREASING_SEQUENCE**

**Origin**
Based on the **DECREASING_SEQUENCE** pattern.

**Constraint**

\[
\text{MAX_WIDTH_DECREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

| VALUE     | : dvar |
| VARIABLES | : collection(var−dvar) |

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & = 0 \lor \text{VALUE} \geq 2 \\
\text{rv} & = 2 \Rightarrow \text{VALUE} \leq 2 \\
\text{rv} & \geq 3 \Rightarrow \text{VALUE} \leq \text{sv} \\
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range(} \text{VARIABLES.var}) \\
\end{align*}
\]

**Purpose**
An occurrence of the pattern **DECREASING_SEQUENCE** is the maximal subsequence which matches the regular expression ‘\( > (\geq | =) > | > \)’.

Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **WIDTH** computes the value \( j - i + 2 \).

**Example**

\((5, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\)

Figure 4.678 provides an example where the **MAX_WIDTH_DECREASING_SEQUENCE** 
\((5, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 1 \\
\text{range(} \text{VARIABLES.var}) > 1 \\
\]

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.678: Illustrating the MAX_WIDTH_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.679 and 4.680 respectively depict the automaton associated with the constraint MAX_WIDTH_DECREASING_SEQUENCE and its simplified form.

\[
\begin{align*}
C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \max(R, C)
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow D + 2 \\
D & \leftarrow 0
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow C + D + 1 \\
D & \leftarrow 0
\end{align*}
\]

\[
\begin{align*}
D & \leftarrow D + 1
\end{align*}
\]

Figure 4.679: Automaton for the MAX_WIDTH_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASINGSEQUENCE pattern where default is 0

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\max(\overline{C}, \overline{C}))</td>
<td>(\max(\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>t</td>
<td>(\max(\overline{C}, \overline{C}))</td>
<td>(\overline{C} + \overline{C} + D + D - 1)</td>
</tr>
</tbody>
</table>

Table 4.85: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the MAX_WIDTH_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.680: Simplified automaton for the \textsc{MAX\_WIDTH\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.31 to the seed transducer of the \textsc{DECREASING\_SEQUENCE} pattern where \texttt{default} is 0; \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-1} + D_i \geq 0\) are linear invariants.

Table 4.86: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the simplified automaton of the \textsc{MAX\_WIDTH\_DECREASING\_SEQUENCE} constraint defined as the composition of the \textsc{DECREASING\_SEQUENCE} pattern, the feature \texttt{WIDTH}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MAX_WIDTH_DECREASING_TERRACE**

**Description**
Based on the DECREASING_TERRACE pattern.

**Constraint**
```plaintext
MAX_WIDTH_DECREASING_TERRACE(VALUE, VARIABLES)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[ sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = 0 \]
\[ VALUE = 0 \lor VALUE \geq 2 \]
\[ VALUE \leq \max(0, sv - 2k) \]

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

where

\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range} (\text{VARIABLES}.\text{var}) \]

**Purpose**

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression ‘\( \geq + > \)’. Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\( (2, (6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3)) \)

Figure 4.681 provides an example where the MAX_WIDTH_DECREASING_TERRACE \((2, [6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3])\) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 2 \]

**Symmetry**

One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.681: Illustrating the MAX_WIDTH_DECREASING_TERRACE constraint of the Example slot
Automaton

Figures 4.682 and 4.683 respectively depict the automaton associated with the constraint MAX_WIDTH_DECREASING_TERRACE and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{cases}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{cases}
\leq s
\begin{cases}
D &\leftarrow D + 1 \\
R &\leftarrow \max(R, D + 1)
\end{cases}
\leq
\begin{cases}
D &\leftarrow 0 \\
R &\leftarrow \max(R, D + 1)
\end{cases}
> r
max(R, C)
\leq
> =
\begin{cases}
D &\leftarrow D + 1 \\
R &\leftarrow \max(R, D + 1)
\end{cases}
<
\begin{cases}
D &\leftarrow 0 \\
R &\leftarrow \max(R, D + 1)
\end{cases}

Figure 4.682: Automaton for the MAX_WIDTH_DECREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING_TERRACE pattern where \texttt{default} is 0

\[
\begin{array}{ccc}
s & r & t \\
\text{max}(\overline{C}, \overline{C}) & \text{max}(\overline{C}, \overline{C}) & \text{max}(\overline{C}, \overline{C}) \\
\text{max}(\overline{C}, \overline{C}) & \text{max}(\overline{C}, \overline{C}) & \overline{D} + \overline{D} + 1 \\
\text{max}(\overline{C}, \overline{C}) & \overline{D} + \overline{D} + 1 & \overline{D} + \overline{D} + 1
\end{array}
\]

Table 4.87: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the MAX_WIDTH_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.683: Simplified automaton for the MAX_WIDTH_DECREASING_TERRACE constraint obtained by applying decoration Table 3.26 to the seed transducer of the DECREASING_TERRACE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_i \geq 0 \) are linear invariants.

Table 4.88: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the simplified automaton of the MAX_WIDTH_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin

Based on the GORGE pattern.

Constraint

MAX_WIDTH_GORGE(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 1
rv = 2 ⇒ VALUE ≤ 1
rv ≥ 3 ⇒ VALUE ≤ max(0, sv − 2)
required(VARIABLES, var)
where
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘(>| > (|= | >)* )(< | < (|= | <)*)’.
Assume that the occurrence of the pattern GORGE starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example

(3, (1, 7, 3, 4, 5, 5, 4, 2, 6, 5, 4, 6, 5, 7))

Figure 4.684 provides an example where the MAX_WIDTH_GORGE (3,[1, 7, 3, 4, 5, 5, 4, 2, 6, 5, 4, 6, 5, 7]) constraint holds.

Typical

|VARIABLES| > 2
range(VARIABLES.var) > 1

Symmetries

• Items of VARIABLES can be reversed.
• One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.684: Illustrating the MAX_WIDTH_GORGE constraint of the Example slot
Figures 4.685 and 4.686 respectively depict the automaton associated with the constraint \texttt{MAX\_WIDTH\_GORGE} and its simplified form.

Figure 4.685: Automaton for the \texttt{MAX\_WIDTH\_GORGE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{GORGE} pattern where default is 0 (transition \textit{u} \rightarrow \textit{r} has the same register update as transition \textit{r} \rightarrow \textit{u})
Figure 4.686: Simplified automaton for the MAX_WIDTH_GORGE constraint obtained by applying decoration Table 3.27 to the seed transducer of the GORGE pattern where default is 0 (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$); $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
Table 4.89: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the MAX_WIDTH_GORGE constraint defined as the composition of the GORGE pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\text{max}(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\text{max}(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\text{max}(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\text{max}(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>$\text{max}(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\text{max}(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>u</td>
<td>$\text{max}(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\text{max}(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>

Table 4.90: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the simplified automaton of the MAX_WIDTH_GORGE constraint defined as the composition of the GORGE pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>0</td>
</tr>
<tr>
<td>u</td>
<td>0</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>
Origin: Based on the INCREASING_SEQUENCE pattern.

Constraint: 
\[ \text{MAX\_WIDTH\_INCREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES}) \]

Arguments:
- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \)

Restrictions:
- \( sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \)
- \( \text{VALUE} = 0 \lor \text{VALUE} \geq 2 \)
- \( rv = 2 \Rightarrow \text{VALUE} \leq 2v \)
- \( rv \geq 3 \Rightarrow \text{VALUE} \leq \text{sv}^2 \)

where
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

VALUE is the maximal width of occurrences of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose: An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\(< (< | =)^* (< | =) < | < \)’. Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i + 2 \).

Example: 
\[ (5, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)) \]

Figure 4.687 provides an example where the MAX\_WIDTH\_INCREASING\_SEQUENCE (5, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

Typical:
- \( |\text{VARIABLES}| > 1 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

Symmetry: One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties: Functional dependency: VALUE determined by VARIABLES.
Figure 4.687: Illustrating the MAX_WIDTH_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.688 and 4.689 respectively depict the automaton associated with the constraint `MAX_WIDTH_INCREMENTAL_SEQUENCE` and its simplified form.

\[
\begin{align*}
& s \quad t \\
\begin{array}{|c|c|}
\hline
s & t \\
\hline
\text{max}(\overline{C}, \overline{C}) & \text{max}(\overline{C}, \overline{C}) \\
\max(\overline{C}, \overline{C}) & \overline{C} + \overline{C} + \overline{D} + \overline{D} - 1 \\
\hline
\end{array}
\end{align*}
\]

Table 4.91: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the `MAX_WIDTH_INCREMENTAL_SEQUENCE` constraint defined as the composition of the `INCREMENTAL_SEQUENCE` pattern, the feature `WIDTH`, and the aggregator `max`. Cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.689: Simplified automaton for the MAX_WIDTH_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.31 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_i \geq 0 \) are linear invariants.

Table 4.92: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the simplified automaton of the MAX_WIDTH_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MAX_WIDTH_INCREASING_TERRACE**

**Description**: Based on the `INCREASING_TERRACE` pattern.

**Constraint**: `MAX_WIDTH_INCREASING_TERRACE(VALUE, VARIABLES)`

**Arguments**
- `VALUE` : `dvar`
- `VARIABLES` : `collection(var−dvar)`

**Restrictions**
- `sv ≤ 3 ∨ rv ≤ 2 ⇒ VALUE = 0`
- `VALUE = 0 ∨ VALUE ≥ 2`
- `VALUE ≤ max(0, sv − 2v)`
- `required(VARIABLES, var)`

where
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**: `VALUE` is the maximal width of occurrences of the `INCREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value 0.

An occurrence of the pattern `INCREASING_TERRACE` is the maximal subsequence which matches the regular expression `'< + '<`.

Assume that the occurrence of the pattern `INCREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

**Example**: `(3,(1,3,3,3,2,5,5,6,4,4,2,3,3,3,4,4))`

Figure 4.690 provides an example where the `MAX_WIDTH_INCREASING_TERRACE` `(3,[1,3,3,3,2,5,5,6,4,4,2,3,3,3,4,4])` constraint holds.

**Typical**
- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 2`

**Symmetry**: One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**: Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 4.690: Illustrating the MAX_WIDTH_INCREASING_TERRACE constraint of the Example slot
Automaton

Figures 4.691 and 4.692 respectively depict the automaton associated with the constraint MAX_WIDTH_INCREASING_TERRACE and its simplified form.

Figure 4.691: Automaton for the MAX_WIDTH_INCREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_TERRACE pattern where default is 0.

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s max((\overline{C}, \overline{C}))</td>
<td>max((\overline{C}, \overline{C}))</td>
<td>max((\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>r max((\overline{C}, \overline{C}))</td>
<td>max((\overline{C}, \overline{C}))</td>
<td>(\overline{D} + \overline{D} + 1) C</td>
</tr>
<tr>
<td>t max((\overline{C}, \overline{C}))</td>
<td>(\overline{D} + \overline{D} + 1) C</td>
<td>(\overline{D} + \overline{D} + 1) C</td>
</tr>
</tbody>
</table>

Table 4.93: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the MAX_WIDTH_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_WIDTH_INCREASING_TERRACE

Figure 4.692: Simplified automaton for the MAX_WIDTH_INCREASING_TERRACE constraint obtained by applying decoration Table 3.26 to the seed transducer of the INCREASING_TERRACE pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.

Table 4.94: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the simplified automaton of the MAX_WIDTH_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature WIDTH, and the aggregator max: cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin: Based on the `INFLEXION` pattern.

Constraint: `MAX_WIDTH_INFLEXION(VALUE, VARIABLES)`

Arguments:
- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`

Restrictions:
- `sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0`
- `VALUE ≥ 0`
- `VALUE ≤ max(0, sv − 2)`
- `required(VARIABLES, var)`
- Where:
  - `sv = |VARIABLES|`
  - `rv = range(VARIABLES.var)`

Purpose:
An occurrence of the pattern `INFLEXION` is the maximal subsequence which matches the regular expression `'< (< | =)† > | > (> | =)† <'`. Assume that the occurrence of the pattern `INFLEXION` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

Example:
```
(3, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))
```

Figure 4.693 provides an example where the `MAX_WIDTH_INFLEXION (3, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4])` constraint holds.

Typical:
```
|VARIABLES| > 2
range(VARIABLES.var) > 1
```

Symmetries:
- Items of `VARIABLES` can be reversed.
- One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

Arg. properties:
- Functional dependency: `VALUE` determined by `VARIABLES`.
Figure 4.693: Illustrating the MAX_WIDTH_INFLEXION constraint of the Example slot
Automaton

Figures 4.694 and 4.695 respectively depict the automaton associated with the constraint MAX_WIDTH_INFLEXION and its simplified form.

Figure 4.694: Automaton for the MAX_WIDTH_INFLEXION constraint obtained by applying decoration Table 3.37 to the seed transducer of the INFLEXION pattern where default is 0 (transition \(r \to t\) has the same registers updates as transition \(t \to r\)).

Figure 4.695: Simplified automaton for the MAX_WIDTH_INFLEXION constraint obtained by applying decoration Table 3.26 to the seed transducer of the INFLEXION pattern where default is 0 (transition \(r \to t\) has the same registers updates as transition \(t \to r\); \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-1} + D_i \geq 0\) are linear invariants.)
MAX_WIDTH_INFLEXION

1525
**MAX_WIDTH_PEAK**

**DESCRIPTION**

**Origin**
Based on the **PEAK** pattern.

**Constraint**
MAX_WIDTH_PEAK(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
- sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
- VALUE ≥ 0
- VALUE ≤ max(0, sv − 2v)
- required(VARIABLES, var)

where
- sv = |VARIABLES|
- rv = range(VARIABLES.var)

**Purpose**
An occurrence of the pattern **PEAK** is the **maximal** subsequence which matches the regular expression '<( = | <)' (> | =)'>.
Assume that the occurrence of the pattern **PEAK** starts at position i and ends at position j. The feature WIDTH computes the value j − i.

**Example**

(3, ⟨7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1⟩)

Figure 4.696 provides an example where the MAX_WIDTH_PEAK (3, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1]) constraint holds.

**Typical**
- |VARIABLES| > 2
- range([VARIABLES.var]) > 1

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
- **Functional dependency**: VALUE determined by VARIABLES.
Figure 4.696: Illustrating the MAX_WIDTH_PEAK constraint of the Example slot
Figures 4.697 and 4.698 respectively depict the automaton associated with the constraint MAX_WIDTH_PEAK and its simplified form.

Figure 4.697: Automaton for the MAX_WIDTH_PEAK constraint obtained by applying decoration Table 3.37 to the seed transducer of the PEAK pattern where default is 0.

Figure 4.698: Simplified automaton for the MAX_WIDTH_PEAK constraint obtained by applying decoration Table 3.27 to the seed transducer of the PEAK pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_i \geq 0 \) are linear invariants.
Table 4.95: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the MAX_WIDTH_PEAK constraint defined as the composition of the PEAK pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

Table 4.96: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the simplified automaton of the MAX_WIDTH_PEAK constraint defined as the composition of the PEAK pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### Origin
Based on the **PLAIN** pattern.

### Constraint
```
MAX_WIDTH_PLAIN(VALUE, VARIABLES)
```

### Arguments
```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

### Restrictions
```
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE ≥ 0
VALUE ≤ \max(0, sv − 2x)
required(VARIABLES, var)
```

where
```
sv = |VARIABLES|
rv = range(VARIABLES, var)
```

### Purpose
An occurrence of the pattern **PLAIN** is the maximal subsequence which matches the regular expression ‘\(>^*<\)’.

Assume that the occurrence of the pattern **PLAIN** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j − i\).

### Example
```
(2, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3))
```

Figure 4.699 provides an example where the **MAX_WIDTH_PLAIN** (2, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3]) constraint holds.

### Typical
```
|VARIABLES| > 2
range(VARIABLES, var) > 1
```

### Symmetries
- Items of **VARIABLES** can be reversed.
- One and the same constant can be added to the **var** attribute of all items of **VARIABLES**.

### Arg. properties
- Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.699: Illustrating the MAX_WIDTH_PLAIN constraint of the Example slot
Automaton

Figures 4.700 and 4.701 respectively depict the automaton associated with the constraint MAX_WIDTH_PLAIN and its simplified form.

Figure 4.700: Automaton for the MAX_WIDTH_PLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLAIN pattern where default is 0.

Figure 4.701: Simplified automaton for the MAX_WIDTH_PLAIN constraint obtained by applying decoration Table 3.26 to the seed transducer of the PLAIN pattern where default is 0; \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-1} + D_i \geq 0\) are linear invariants.
### Table 4.97: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the MAX_WIDTH_PLAIN constraint defined as the composition of the PLAIN pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\max(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overleftarrow{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\max(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>$\max(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + 1$</td>
</tr>
</tbody>
</table>

### Table 4.98: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the simplified automaton of the MAX_WIDTH_PLAIN constraint defined as the composition of the PLAIN pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overleftarrow{D} + 1$</td>
</tr>
</tbody>
</table>
**MAX_WIDTH_PLATEAU**

**Origin**
Based on the **PLATEAU** pattern.

**Constraint**

\[
\text{MAX_WIDTH_PLATEAU}(\text{VALUE, VARIABLES})
\]

**Arguments**

- **VALUE** : dvar
- **VARIABLES** : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \max(0, sv - 2k) \\
\text{required}(\text{VARIABLES, var}) & \text{required}
\end{align*}
\]

where

\[
\begin{align*}
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES.var})
\end{align*}
\]

**Purpose**

VALUE is the maximal width of occurrences of the **PLATEAU** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern **PLATEAU** is the maximal subsequence which matches the regular expression '<<=' '>''.

Assume that the occurrence of the pattern **PLATEAU** starts at position i and ends at position j. The feature **WIDTH** computes the value \(j - i\).

**Example**

\[(4, (1, 3, 3, 5, 5, 5, 2, 4, 4, 3, 3, 1, 5, 5))\]

Figure 4.702 provides an example where the **MAX_WIDTH_PLATEAU** (4, [1, 3, 3, 5, 5, 5, 2, 4, 4, 3, 3, 1, 5, 5]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES.var}) & > 1
\end{align*}
\]

**Symmetries**

- Items of **VARIABLES** can be reversed.
- One and the same constant can be added to the **var** attribute of all items of **VARIABLES**.

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.702: Illustrating the MAX_WIDTH_PLATEAU constraint of the Example slot
Automaton

Figures 4.703 and 4.704 respectively depict the automaton associated with the constraint `MAX_WIDTH_PLATEAU` and its simplified form.

Figure 4.703: Automaton for the `MAX_WIDTH_PLATEAU` constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLATEAU pattern where `default` is 0.

Figure 4.704: Simplified automaton for the `MAX_WIDTH_PLATEAU` constraint obtained by applying decoration Table 3.26 to the seed transducer of the PLATEAU pattern where `default` is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
MAX_WIDTH_PLATEAU

Table 4.99: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the MAX_WIDTH_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\text{max}(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\text{max}(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\text{max}(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\text{max}(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
<tr>
<td>t</td>
<td>(\text{max}(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
</tbody>
</table>

Table 4.100: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the simplified automaton of the MAX_WIDTH_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
</tbody>
</table>
Origin

Based on the PROPER_plain pattern.

Constraint

\( \text{MAX_WIDTH_PROPERPlain}(\text{VALUE}, \text{VARIABLES}) \)

Arguments

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

Restrictions

\[ \text{sv} \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \]
\[ \text{VALUE} = 0 \lor \text{VALUE} \geq 2 \]
\[ \text{VALUE} \leq \max(0, \text{sv} - 2) \]

\[ \text{required} (\text{VARIABLES}, \text{var}) \]

where

\[ \text{sv} = |\text{VARIABLES}| \]
\[ \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \]

VALUE is the maximal width of occurrences of the PROPER_plain pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose

An occurrence of the pattern **PROPER_plain** is the *maximal* subsequence which matches the regular expression ‘\(> = + <\)’.
Assume that the occurrence of the pattern **PROPER_plain** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i\).

Example

\( (3, (2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5)) \)

Figure 4.705 provides an example where the **MAX_WIDTH_PROPERPLAIN** (3, [2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5]) constraint holds.

Typical

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Symmetries

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties

**Functional dependency**: VALUE determined by VARIABLES.
Figure 4.705: Illustrating the MAX_WIDTH_PROPER_PLAIN constraint of the Example slot
Figures 4.706 and 4.707 respectively depict the automaton associated with the constraint MAX_WIDTH_PROPERPLAIN and its simplified form.

**Figure 4.706:** Automaton for the MAX_WIDTH_PROPERPLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPERPLAIN pattern where default is 0.

**Figure 4.707:** Simplified automaton for the MAX_WIDTH_PROPERPLAIN constraint obtained by applying decoration Table 3.26 to the seed transducer of the PROPERPLAIN pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
Table 4.101: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the \texttt{MAX\_WIDTH\_PROPER\_PLAIN} constraint defined as the composition of the \texttt{PROPER\_PLAIN} pattern, the feature \texttt{WIDTH}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>(r)</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
<tr>
<td>(t)</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
</tbody>
</table>

Table 4.102: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the simplified automaton of the \texttt{MAX\_WIDTH\_PROPER\_PLAIN} constraint defined as the composition of the \texttt{PROPER\_PLAIN} pattern, the feature \texttt{WIDTH}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(r)</td>
<td>0</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
<tr>
<td>(t)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
</tbody>
</table>
**MAX_WIDTH_PROPER_PLATEAU**

**Description**

Based on the PROPER_PLATEAU pattern.

**Constraint**

\[
\text{MAX_WIDTH_PROPER_PLATEAU}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- \(\text{VALUE} : \text{dvar}\)
- \(\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})\)

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 3 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 & \lor \text{VALUE} \geq 2 \\
\text{VALUE} \leq \max(0, \text{sv} - 2) & \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the maximal width of occurrences of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

**Example**

\[ (3, \langle 7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3 \rangle) \]

Figure 4.708 provides an example where the MAX_WIDTH_PROPER_PLATEAU \((3, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3])\) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\end{align*}
\]

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.708: Illustrating the MAX_WIDTH_PROPER_PLATEAU constraint of the Example slot
Automaton

Figures 4.709 and 4.710 respectively depict the automaton associated with the constraint \texttt{MAX\_WIDTH\_PROPER\_PLATEAU} and its simplified form.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{automaton.png}
\caption{Automaton for the \texttt{MAX\_WIDTH\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{PROPER\_PLATEAU} pattern where \texttt{default} is 0.}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{simplified_automaton.png}
\caption{Simplified automaton for the \texttt{MAX\_WIDTH\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 3.26 to the seed transducer of the \texttt{PROPER\_PLATEAU} pattern where \texttt{default} is 0; \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-1} + D_i \geq 0\) are linear invariants.}
\end{figure}
Table 4.103: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the MAX_WIDTH_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1) (\mathbf{C})</td>
</tr>
<tr>
<td>t</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1) (\mathbf{C})</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1) (\mathbf{C})</td>
</tr>
</tbody>
</table>

Table 4.104: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the simplified automaton of the MAX_WIDTH_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1) (\mathbf{C})</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1) (\mathbf{C})</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1) (\mathbf{C})</td>
</tr>
</tbody>
</table>
### Origin
Based on the `STEADY_SEQUENCE` pattern.

### Constraint
\[
\text{MAX\_WIDTH\_STEADY\_SEQUENCE}(\text{VALUE, VARIABLES})
\]

### Arguments
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

### Restrictions
- \(sv \leq 1 \Rightarrow \text{VALUE} = 0\)
- \(\text{VALUE} = 0 \lor \text{VALUE} \geq 2\)
- \(rv = 1 \land av \geq 2 \Rightarrow \text{VALUE} \geq av\)
- \(rv \geq 2 \land av \geq 2 \Rightarrow \text{VALUE} \geq 0\)
- \(\text{VALUE} \leq \text{sv}\)
- \(\text{required}(\text{VARIABLES, var})\)
  where
- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range}(\text{VARIABLES.var})\)

### Purpose
An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression `\('=+'\)`. Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `WIDTH` computes the value \(j - i + 2\).

### Example
(3, (3, 1, 1, 4, 5, 5, 5, 5, 6, 2, 4, 4, 3, 2, 1, 1))

Figure 4.711 provides an example where the `MAX\_WIDTH\_STEADY\_SEQUENCE` constraint holds.

### Typical
\(|\text{VARIABLES}| > 1\)

### Symmetries
- Items of `VARIABLES` can be reversed.
- One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

### Arg. properties
- **Functional dependency**: `VALUE` determined by `VARIABLES`. 
Figure 4.711: Illustrating the MAX_WIDTH_STEADY_SEQUENCE constraint of the Example slot.
Automaton

Figures 4.712 and 4.713 respectively depict the automaton associated with the constraint MAX_WIDTH_STEADY_SEQUENCE and its simplified form.

\[
\begin{cases}
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default}
\end{cases}
\]

Figure 4.712: Automaton for the MAX_WIDTH_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STEADY_SEQUENCE pattern where default is 0.

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\text{max}(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\text{max}(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
</tbody>
</table>

Table 4.105: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the MAX_WIDTH_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.713: Simplified automaton for the \texttt{MAX\_WIDTH\_STEADY\_SEQUENCE} constraint obtained by applying decoration Table 3.31 to the seed transducer of the \texttt{STEADY\_SEQUENCE} pattern where \texttt{default} is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.

Table 4.106: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the simplified automaton of the \texttt{MAX\_WIDTH\_STEADY\_SEQUENCE} constraint defined as the composition of the \texttt{STEADY\_SEQUENCE} pattern, the feature \texttt{WIDTH}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE

DESCRIPTION AUTOMATON

Origin
Based on the STRICTLY_DECREASING_SEQUENCE pattern.

Constraint
MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 2
VALUE ≤ \text{min}(sv, rv)
required(VARIABLES, var)
where
sv = |VARIABLES|
rv = \text{range}(VARIABLES.var)

Purpose
An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression ‘>‘.
Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at position i and ends at position j. The feature WIDTH computes the value j − i + 2.

Example
(3, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))

Figure 4.714 provides an example where the MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE (3, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3]) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES.var) > 1

Symmetry
One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.714: Illustrating the MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Figures 4.715 and 4.716 respectively depict the automaton associated with the constraint MAX\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE and its simplified form.

![Automaton diagram](image)

Figure 4.715: Automaton for the MAX\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICTLY\_DECREASING\_SEQUENCE pattern where default is 0

<table>
<thead>
<tr>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
</tbody>
</table>

Table 4.107: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the MAX\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE constraint defined as the composition of the STRICTLY\_DECREASING\_SEQUENCE pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.716: Simplified automaton for the `MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.31 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern where `default` is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + 2 \geq 0 \) are linear invariants.

Table 4.108: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the `MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_DECREASING_SEQUENCE` pattern, the feature `WIDTH`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Based on the `STRICTLY_INCREASING_SEQUENCE` pattern.

**MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE**

**Description**

**Origin**

Based on the `STRICTLY_INCREASING_SEQUENCE` pattern.

**Constraint**

`MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)`

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0
\]

\[
VALUE = 0 \lor VALUE \geq 2
\]

\[
VALUE \leq \min(sv, rv) \Rightarrow \text{required}(VARIABLES, var)
\]

where

\[
sv = |VARIABLES|
\]

\[
rv = \text{range}(VARIABLES.var)
\]

**Purpose**

An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the maximal sub-sequence which matches the regular expression `'<+'`.

Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

**Example**

\((5, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))\)

Figure 4.717 provides an example where the `MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Symmetry**

One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 4.717: Illustrating the **MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE** constraint of the **Example** slot
Figures 4.718 and 4.719 respectively depict the automaton associated with the constraint `MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE` and its simplified form.

```
\[
\begin{align*}
&\{ C \gets \text{default} \\
&\quad D \gets 0 \\
&\quad R \gets \text{default} \}
\]

\[
\begin{align*}
&\geq s \\
&\quad \geq \\
&\quad \geq \\
&\quad \geq \max(R, C)
\end{align*}
\]

\[
\begin{align*}
&\geq \\
&\quad \geq \\
&\quad \geq \\
&\quad \geq \max(R, C)
\end{align*}
\]

\[
\begin{align*}
&\{ C \gets \text{default} \\
&\quad D \gets 0 \\
&\quad R \gets \max(R, C) \}
\]

\[
\begin{align*}
&\{ C \gets D + 2 \\
&\quad D \gets 0 \}
\]

\[
\begin{align*}
&\{ C \gets C + D + 1 \\
&\quad D \gets 0 \}
\end{align*}
\]

\[
\begin{align*}
&\{ \{ C \gets \text{default} \\
&\quad \text{for } \text{decorations} \} \}
\]

\[
\begin{align*}
&\quad \{ \{ D \gets 0 \\
&\quad \text{for } \text{decorations} \} \}
\end{align*}
\]

\[
\begin{align*}
&\quad \{ \{ R \gets \max(R, C) \\
&\quad \text{for } \text{decorations} \} \}
\end{align*}
\]

Figure 4.718: Automaton for the `MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where `default` is 0.

Table 4.109: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the `MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the feature `WIDTH`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.719: Simplified automaton for the MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.31 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + 2 \geq 0$ are linear invariants.

Table 4.110: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the simplified automaton of the MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature width, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MAX_WIDTH_SUMMIT**

**Origin**  
Based on the SUMMIT pattern.

**Constraint**  
MAX_WIDTH_SUMMIT(VALUE, VARIABLES)

**Arguments**  
VALUE : dvar  
VARIABLES : collection(var−dvar)

**Restrictions**  
\[sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0\]  
VALUE \geq 0  
rv = 2 \Rightarrow VALUE \leq 1  
rv \geq 3 \Rightarrow VALUE \leq \max(0, sv−2)  
\text{required}(\text{VARIABLES}, \text{var})

where  
sv = |\text{VARIABLES}|  
rv = \text{range}(\text{VARIABLES}.\text{var})

**Purpose**  
An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression ‘\((< | < (= | <)^* <)(> | > (= | >)^* >)\)’.  
Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j − i\).

**Example**  
\((3, (7, 1, 5, 4, 4, 3, 4, 6, 4, 6, 2, 3, 4, 2, 3, 1))\)

Figure 4.720 provides an example where the MAX_WIDTH_SUMMIT (3,[7, 1, 5, 4, 4, 3, 4, 6, 4, 6, 2, 3, 4, 2, 3, 1]) constraint holds.

**Typical**  
|\text{VARIABLES}| > 2  
\text{range}(\text{VARIABLES}.\text{var}) > 1

**Symmetries**  
- Items of VARIABLES can be reversed.  
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**  
Functional dependency: VALUE determined by VARIABLES.
Figure 4.720: Illustrating the MAX_WIDTH_SUMMIT constraint of the Example slot.
Figures 4.721 and 4.722 respectively depict the automaton associated with the constraint MAX_WIDTH_SUMMIT and its simplified form.

Figure 4.721: Automaton for the MAX_WIDTH_SUMMIT constraint obtained by applying decoration Table 3.37 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
Figure 4.722: Simplified automaton for the MAX_WIDTH_SUMMIT constraint obtained by applying decoration Table 3.27 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$); $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
Table 4.111: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the MAX_WIDTH_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$s$</td>
<td>$r$</td>
<td>$t$</td>
</tr>
<tr>
<td></td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
</tr>
<tr>
<td>$s$</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>$r$</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
</tr>
<tr>
<td>$t$</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>$u$</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
<td>max($\overrightarrow{C}, \overrightarrow{C}$)</td>
</tr>
</tbody>
</table>

Table 4.112: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the simplified automaton of the MAX_WIDTH_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$s$</td>
<td>$r$</td>
<td>$t$</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s$</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>$r$</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>0</td>
</tr>
<tr>
<td>$t$</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>0</td>
</tr>
<tr>
<td>$u$</td>
<td>0</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>
MAX_WIDTH_SUMMIT  1563
MAX_WIDTH_VALLEY

**Origin**
Based on the VALLEY pattern.

**Constraint**
MAX_WIDTH_VALLEY(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var − dvar)

**Restrictions**
\[
sv \leq 2 \land rv \leq 1 \Rightarrow VALUE = 0 \\
VALUE \geq 0 \\
VALUE \leq \max(0, sv - 2k) \\
\text{required}(\text{VARIABLES}, \text{var})
\]
where
\[
sv = |\text{VARIABLES}| \\
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

**Purpose**
An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression ‘(= | >)*(< | =)*’.
Assume that the occurrence of the pattern VALLEY starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**
\[(4, (1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7))\]

Figure 4.723 provides an example where the MAX_WIDTH_VALLEY (4, [1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.723: Illustrating the MAX_WIDTH_VALLEY constraint of the Example slot
Automaton

Figures 4.724 and 4.725 respectively depict the automaton associated with the constraint MAX_WIDTH_VALLEY and its simplified form.

Figure 4.724: Automaton for the MAX_WIDTH_VALLEY constraint obtained by applying decoration Table 3.37 to the seed transducer of the VALLEY pattern where default is 0

Figure 4.725: Simplified automaton for the MAX_WIDTH_VALLEY constraint obtained by applying decoration Table 3.27 to the seed transducer of the VALLEY pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
### Table 4.113: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the `MAX_WIDTH_VALLEY` constraint defined as the composition of the `VALLEY` pattern, the feature `WIDTH`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>s</strong></td>
<td><strong>r</strong></td>
<td><strong>t</strong></td>
</tr>
<tr>
<td><strong>s</strong></td>
<td><code>max(\vec{C}, \vec{C})</code></td>
<td><code>max(\vec{C}, \vec{C})</code></td>
</tr>
<tr>
<td><strong>r</strong></td>
<td><code>max(\vec{C}, \vec{C})</code></td>
<td><code>\vec{D} + \vec{D} + 1</code> (\textcolor{blue}{C})</td>
</tr>
<tr>
<td><strong>t</strong></td>
<td><code>max(\vec{C}, \vec{C})</code></td>
<td><code>\vec{C} + \vec{D} + \vec{D} + 1</code> (\textcolor{blue}{L})</td>
</tr>
</tbody>
</table>

### Table 4.114: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the simplified automaton of the `MAX_WIDTH_VALLEY` constraint defined as the composition of the `VALLEY` pattern, the feature `WIDTH`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>s</strong></td>
<td><strong>r</strong></td>
<td><strong>t</strong></td>
</tr>
<tr>
<td><strong>s</strong></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>r</strong></td>
<td>0</td>
<td><code>\vec{D} + \vec{D} + 1</code> (\textcolor{blue}{C})</td>
</tr>
<tr>
<td><strong>t</strong></td>
<td>0</td>
<td><code>\vec{D} + \vec{D} + 1</code> (\textcolor{blue}{L})</td>
</tr>
</tbody>
</table>
**MAX_WIDTH_ZIGZAG**

**DESCRIPTION**

**Origin**
Based on the ZIGZAG pattern.

**Constraint**

\[
\text{MAX_WIDTH_ZIGZAG}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- \text{VALUE} : dvar
- \text{VARIABLES} : collection(var−dvar)

**Restrictions**

- \( sv \leq 3 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \)
- \( \text{VALUE} = 0 \lor \text{VALUE} \geq 2 \)
- \( \text{VALUE} \leq \max(0, sv - 2v) \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)

where

- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

VALUE is the maximal width of occurrences of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

A occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘(<>)(< | <>)(<>)(> | <>)’.

Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\((6, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\)

Figure 4.726 provides an example where the MAX_WIDTH_ZIGZAG\((6, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1])\) constraint holds.

**Typical**

- \( |\text{VARIABLES}| > 3 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

**Symmetries**

- Items of \text{VARIABLES} can be reversed.
- One and the same constant can be added to the \text{var} attribute of all items of \text{VARIABLES}.

**Arg. properties**

Functional dependency: VALUE determined by \text{VARIABLES}. 
Figure 4.726: Illustrating the MAX_WIDTH_ZIGZAG constraint of the Example slot
Automaton: Figures 4.727 and 4.728 respectively depict the automaton associated with the constraint MAX_WIDTH_ZIGZAG and its simplified form.
Figure 4.727: Automaton for the MAX_WIDTH_ZIGZAG constraint obtained by applying decoration Table 3.37 to the seed transducer of the ZIGZAG pattern where default is 0; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the register D is reset to its initial value; (3) on transitions from c, f to s the register R is updated wrt C and the register C is reset to its initial value
Figure 4.728: Simplified automaton for the MAX_WIDTH_ZIGZAG constraint obtained by applying decoration Table 3.27 to the seed transducer of the ZIGZAG pattern where default is 0; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the register D is reset to its initial value; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + 2 \geq 0 \) are linear invariants.
Table 4.115: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the MAX_WIDTH_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.116: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the simplified automaton of the MAX_WIDTH_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### Description

**Origin**
Based on the **ZIGZAG** pattern.

**Constraint**

\[
\text{MAX\_ZIGZAG} (\text{VARIABLES}, \text{FEATURES}, \text{DEFAULT})
\]

**Arguments**

- **VARIABLES** : \text{collection(\text{var}\text{--}\text{dvar})}
- **FEATURES** : \text{collection(\text{var}\text{--}\text{dvar})}
- **DEFAULT** : \text{int}

**Restrictions**

\[
\begin{align*}
\text{required(\text{VARIABLES}.\text{var})} \\
\text{required(\text{FEATURES}.\text{var})} \\
|\text{VARIABLES}| &= |\text{FEATURES}| \\
\text{sv} &\leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{FEATURES}.\text{var} = \text{DEFAULT} \\
\text{FEATURES}.\text{var} &= \text{DEFAULT} \lor \text{FEATURES}.\text{var} \geq \text{minv} + 1 \\
\text{FEATURES}.\text{var} &= \text{DEFAULT} \lor \text{FEATURES}.\text{var} \leq \text{maxv} \\
\text{DEFAULT} &< \text{minv} + 1 \lor \text{DEFAULT} > \text{maxv} \\
\text{where}
\end{align*}
\]

\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Consider the time-series given by the \text{VARIABLES} collection. If position \(i\) does not correspond to the first position in \text{VARIABLES} where an occurrence of **ZIGZAG** is identified (even if this occurrence of pattern is not complete) then \text{FEATURES}[i] is the default value \text{DEFAULT}; otherwise \text{FEATURES}[i] gives the feature value of the corresponding occurrence of **ZIGZAG**.

**Purpose**

An occurrence of the pattern **ZIGZAG** is the \textit{maximal} subsequence which matches the regular expression ‘(<>)^*(< | <>) | (<>)^*(>) | >>)’.

Assume that the occurrence of the pattern **ZIGZAG** starts at position \(i\) and ends at position \(j\). The feature \text{MAX} computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

Figure 4.729 provides an example where the \text{MAX\_ZIGZAG} ([4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 3, 0, 0, 0, 6, 0, 0, 7, 0, 0, 0, 0, 0]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

Functional dependency: \text{FEATURES} determined by \text{VARIABLES} and \text{DEFAULT}.
Figure 4.729: Illustrating the MAX_ZIGZAG constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
MIN_BUMP_ON_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIN_BUMP_ON_DECREASING_SEQUENCE</td>
<td></td>
</tr>
</tbody>
</table>

**Description**

Based on the **BUMP_ON_DECREASING_SEQUENCE** pattern.

**Constraint**

$$\text{MIN\_BUMP\_ON\_DECREASING\_SEQUENCE}(\text{VARIABLES}, \text{FEATURES}, \text{DEFAULT})$$

**Arguments**

- **VARIABLES**: collection(var−dvar)
- **FEATURES**: collection(var−dvar)
- **DEFAULT**: int

**Restrictions**

- $\text{required}(\text{VARIABLES}, \text{var})$
- $\text{required}(\text{FEATURES}, \text{var})$
- $$\text{sv} = |\text{FEATURES}|$$
- $$\text{sv} \leq 5 \lor \text{rv} \leq 2 \Rightarrow \text{FEATURES.var} = \text{DEFAULT}$$
- $$\text{FEATURES.var} = \text{DEFAULT} \lor \text{FEATURES.var} \geq \text{minv}$$
- $$\text{FEATURES.var} = \text{DEFAULT} \lor \text{FEATURES.var} \leq \text{maxv} - 2$$
- $$\text{DEFAULT} < \text{minv} \lor \text{DEFAULT} \geq \text{maxv} - 2$$

where

- $$\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})$$
- $$\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})$$
- $$\text{sv} = |\text{VARIABLES}|$$
- $$\text{rv} = \text{range}(\text{VARIABLES}.\text{var})$$

**Purpose**

Consider the time-series given by the **VARIABLES** collection. If position $i$ does not correspond to the first position in **VARIABLES** where an occurrence of **BUMP_ON_DECREASING_SEQUENCE** is identified then **FEATURES**[$i$] is the default value **DEFAULT**; otherwise **FEATURES**[$i$] gives the feature value of the corresponding occurrence of **BUMP_ON_DECREASING_SEQUENCE**.

An occurrence of the pattern **BUMP_ON_DECREASING_SEQUENCE** is the subsequence which matches the regular expression ‘$>><>$’.

Assume that the occurrence of the pattern **BUMP_ON_DECREASING_SEQUENCE** starts at position $i$ and ends at position $j$. The feature **MIN** computes the minimum of the values from index $i + 2$ to index $j$.

**Example**

Figure 4.730 provides an example where the **MIN_BUMP_ON_DECREASING_SEQUENCE** ([7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3], [0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0]) constraint holds.

**Typical**

- $$|\text{VARIABLES}| > 5$$
- $$\text{range}(\text{VARIABLES}.\text{var}) > 2$$

**Arg. properties**

- Functional dependency: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.730: Illustrating the MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot.
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
MIN_BUMP_ON_DECREASING_SEQUENCE 1583
### Origin
Based on the `DECREASING` pattern.

### Constraint
`MIN_DECORATING(VARIABLES, FEATURES, DEFAULT)`

### Arguments

<table>
<thead>
<tr>
<th>VARIABLE</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>FEATURES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>int</td>
</tr>
</tbody>
</table>

### Restrictions

```
required(VARIABLES, var)
required(FEATURES, var)
sv = |FEATURES|
sv ≤ 1 ∨ rv ≤ 1 ⇒ FEATURES.var = DEFAULT
FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv
FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv − 1
DEFAULT < minv ∨ DEFAULT > maxv − 1
```

where

```
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

Consider the time-series given by the VARIABLES collection. If position \( i \) does not correspond to the first position in VARIABLES where an occurrence of `DECREASING` is identified then FEATURES[\( i \)] is the default value DEFAULT; otherwise FEATURES[\( i \)] gives the feature value of the corresponding occurrence of `DECREASING`.

### Purpose
An occurrence of the pattern `DECREASING` is the subsequence which matches the regular expression `\(<>\)`. Assume that the occurrence of the pattern `DECREASING` starts at position \( i \) and ends at position \( j \). The feature `MIN` computes the minimum of the values from index \( i \) to index \( j + 1 \).

### Example

Figure 4.731 provides an example where the `MIN_DECORATING` ([3, 4, 2, 5, 6, 4, 4, 3, 1, 4, 6, 4, 4], [0, 2, 0, 0, 0, 4, 0, 3, 1, 0, 0, 4, 0, 0], 0) constraint holds.

### Typical

- \(|VARIABLES| > 1\)
- `range(VARIABLES.var) > 1`

### Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.731: Illustrating the MIN_DECREASING constraint of the Example slot
Use the decoration table 3.32 to synthesise the corresponding register automaton.
MIN_DECREASING

1587
**Origin** Based on the **DECREASING_SEQUENCE** pattern.

**Constraint** 

\[
\text{MIN\_DECREASING\_SEQUENCE}(:, :, :, \text{DEFAULT})
\]

**Arguments**

- **VARIABLES**: collection(var−dvar)
- **FEATURES**: collection(var−dvar)
- **DEFAULT**: int

**Restrictions**

- \(\text{required}(\text{VARIABLES}, \text{var})\)
- \(\text{required}(\text{FEATURES}, \text{var})\)
- \(\text{sv} = |\text{FEATURES}|\)
- \(\text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{FEATURES} \cdot \text{var} = \text{DEFAULT}\)
- \(\text{FEATURES} \cdot \text{var} = \text{DEFAULT} \lor \text{FEATURES} \cdot \text{var} \geq \text{minv}\)
- \(\text{FEATURES} \cdot \text{var} = \text{DEFAULT} \land \text{FEATURES} \cdot \text{var} \leq \text{maxv} - 1\)
- \(\text{FEATURES} \cdot \text{var} = \text{DEFAULT} \lor \text{FEATURES} \cdot \text{var} > \text{maxv} - 1\)

where

- \(\text{maxv} = \maxval(\text{VARIABLES} \cdot \text{var})\)
- \(\text{minv} = \minval(\text{VARIABLES} \cdot \text{var})\)
- \(\text{sv} = |\text{VARIABLES}|\)
- \(\text{rv} = \text{range}(\text{VARIABLES} \cdot \text{var})\)

**Purpose**

Consider the time-series given by the **VARIABLES** collection. If position \(i\) does not correspond to the first position in **VARIABLES** where an occurrence of **DECREASINGSEQUENCE** is identified (even if this occurrence of pattern is not complete) then **FEATURES**\([i]\) is the default value **DEFAULT**; otherwise **FEATURES**\([i]\) gives the feature value of the corresponding occurrence of **DECREASINGSEQUENCE**.

An occurrence of the pattern **DECREASINGSEQUENCE** is the maximal subsequence which matches the regular expression ‘\(> (> | =)^* > | >\)’.

Assume that the occurrence of the pattern **DECREASINGSEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

Figure 4.732 provides an example where the **MIN\_DECREASING\_SEQUENCE** ([3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4], [0, 2, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0], 0) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES} \cdot \text{var}) > 1\)

**Arg. properties**

Functional dependency: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.732: Illustrating the MIN\_DECREASING\_SEQUENCE constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
MIN_DIP_ON_INCREASING_SEQUENCE

**Origin**
Based on the **DIP_ON_INCREASING_SEQUENCE** pattern.

**Constraint**
```
MIN_DIP_ON_INCREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)
```

**Arguments**

<table>
<thead>
<tr>
<th>VARIABLE</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>FEATURES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>int</td>
</tr>
</tbody>
</table>

**Restrictions**

```
required(VARIABLES.var)
required(FEATURES.var)
sv = |FEATURES|
sv ≤ 5 ∨ rv ≤ 2 ⇒ FEATURES.var = DEFAULT
FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv
FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv − 2
DEFAULT < minv ∨ DEFAULT > maxv − 2
```

where

maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

**Purpose**
Consider the time-series given by the **VARIABLES** collection. If position \( i \) does not correspond to the first position in **VARIABLES** where an occurrence of **DIP_ON_INCREASING_SEQUENCE** is identified then **FEATURES**[\( i \)] is the default value **DEFAULT**; otherwise **FEATURES**[\( i \)] gives the feature value of the corresponding occurrence of **DIP_ON_INCREASING_SEQUENCE**.

An occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** is the subsequence which matches the regular expression ‘<<><<’.

Assume that the occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **MIN** computes the minimum of the values from index \( i + 2 \) to index \( j \).

**Example**

Figure 4.733 provides an example where the **MIN_DIP_ON_INCREASING_SEQUENCE** ([1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4], [0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**

```
|VARIABLES| > 5
range(VARIABLES.var) > 2
```

**Arg. properties**

Functional dependency: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.733: Illustrating the MIN_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
MIN_DIP_ON_INCREASING_SEQUENCE
MIN_GORGE

DESCRIPTION AUTOMATON

Origin
Based on the GORGE pattern.

Constraint
MIN_GORGE(VARIABLES, FEATURES, DEFAULT)

Arguments
VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int

Restrictions
required(VARIABLES.var)
required(FEATURES.var)
sv = |FEATURES|
sv ≤ 2 ∨ rv ≤ 1 ⇒ FEATURES.var = DEFAULT
FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv
FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv − 1
DEFAULT < minv ∨ DEFAULT > maxv − 1
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of GORGE is identified (even if this occurrence of pattern is not complete) then FEATURES[$i$] is the default value DEFAULT; otherwise FEATURES[$i$] gives the feature value of the corresponding occurrence of GORGE.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘$(> | > (= | >)* >)(< | < (= | <)* <)$’.
Assume that the occurrence of the pattern GORGE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

Example
Figure 4.734 provides an example where the MIN_GORGE constraint holds.

Typical

|VARIABLES| > 2
range(VARIABLES.var) > 1

Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.734: Illustrating the MIN_GORGE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
**MIN_HEIGHT_DECREASING_TERRACE**

**DESCRIPTION**

Based on the **DECREASING_TERRACE** pattern.

**Constraint**

\[
\text{MIN_HEIGHT_DECREASING_TERRACE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

<table>
<thead>
<tr>
<th>VALUE</th>
<th>dvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>collection(var_dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 3 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{minv} + 1 + \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \text{maxv} - 1 \\
\text{required}(\text{VARIABLES}.\text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

\(\text{VALUE}\) is the minimum of all minimum values in each occurrence of the **DECREASING_TERRACE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, \(\text{VALUE}\) takes the default value \(+\infty\).

An occurrence of the pattern **DECREASING_TERRACE** is the *maximal* subsequence which matches the regular expression ‘\(> = + >\)’.

Assume that the occurrence of the pattern **DECREASING_TERRACE** starts at position \(i\) and ends at position \(j\). The feature **MIN**, called **HEIGHT** in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(2, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3))\]

Figure 4.735 provides an example where the **MIN_HEIGHT_DECREASING_TERRACE** (2, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]

**Arg. properties**

Functional dependency: \(\text{VALUE}\) determined by \(\text{VARIABLES}\).
Figure 4.735: Illustrating the MIN_HEIGHT_DECREASING_TERRACE constraint of the Example slot
Figures 4.736 and 4.737 respectively depict the automaton associated with the constraint MIN_HEIGHT_DECREASING_TERRACE and its simplified form.

Figure 4.736: Automaton for the MIN_HEIGHT_DECREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING_TERRACE pattern where default is $+\infty$.

Figure 4.737: Simplified automaton for the MIN_HEIGHT_DECREASING_TERRACE constraint obtained by applying decoration Table 3.39 to the seed transducer of the DECREASING_TERRACE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.117: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the `MIN_HEIGHT_DECREASING_TERRACE` constraint defined as the composition of the `DECREASING_TERRACE` pattern, the feature `MIN`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\min(\overline{C}, \overline{C}))</td>
<td>(\min(\overline{C}, \overline{C}))</td>
<td>(\min(\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\min(\overline{C}, \overline{C}))</td>
<td>(\min(\overline{C}, \overline{C}))</td>
<td>(\min(\overline{D}, \overline{D}, \text{VAR}_{i+1}))</td>
</tr>
<tr>
<td>t</td>
<td>(\min(\overline{C}, \overline{C}))</td>
<td>(\min(\overline{D}, \overline{D}, \text{VAR}_{i+1}))</td>
<td>(\min(\overline{D}, \overline{D}, \text{VAR}_{i+1}))</td>
</tr>
</tbody>
</table>

Table 4.118: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the simplified automaton of the `MIN_HEIGHT_DECREASING_TERRACE` constraint defined as the composition of the `DECREASING_TERRACE` pattern, the feature `MIN`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(+\infty)</td>
<td>(+\infty)</td>
<td>(+\infty)</td>
</tr>
<tr>
<td>r</td>
<td>(+\infty)</td>
<td>(+\infty)</td>
<td>(\text{VAR}_{i+1})</td>
</tr>
<tr>
<td>t</td>
<td>(+\infty)</td>
<td>(\text{VAR}_{i+1})</td>
<td>(\text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>
**MIN_HEIGHT_INCREASING_TERRACE**

**DESCRIPTION**

**Origin**
Based on the INCREASING_TERRACE pattern.

**Constraint**

\[
\text{MIN_HEIGHT_INCREASING_TERRACE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

| \text{VALUE} | : dvar |
| \text{VARIABLES} | : collection(var−dvar) |

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 3 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{minv} + 1 \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \text{maxv} - 1 \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

\text{VALUE} is the minimum of all minimum values in each occurrence of the INCREASING_TERRACE pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value \(+\infty\). An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression ‘\(<=+<\)’. Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\((3, (1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4))\)

Figure 4.738 provides an example where the \text{MIN_HEIGHT_INCREASING_TERRACE} \((3, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4])\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 2\)

**Arg. properties**

Functional dependency: \text{VALUE} determined by \text{VARIABLES}. 
Figure 4.738: Illustrating the MIN_HEIGHT_INCREASING_TERRACE constraint of the Example slot
Automaton

Figures 4.739 and 4.740 respectively depict the automaton associated with the constraint MIN_HEIGHT_INCREASING_TERRACE and its simplified form.

Figure 4.739: Automaton for the MIN_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_TERRACE pattern where default is $+\infty$.

Figure 4.740: Simplified automaton for the MIN_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 3.39 to the seed transducer of the INCREASING_TERRACE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.119: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the MIN_HEIGHT_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>( \min(\overline{C}, \overline{C}) )</td>
<td>( \min(\overline{C}, \overline{C}) )</td>
<td>( \min(\overline{C}, \overline{C}) )</td>
</tr>
<tr>
<td>r</td>
<td>( \min(\overline{C}, \overline{C}) )</td>
<td>( \min(\overline{C}, \overline{C}) )</td>
<td>( \min(\overline{D}, \overline{D}, \text{VAR}_{i+1}) )</td>
</tr>
<tr>
<td>t</td>
<td>( \min(\overline{C}, \overline{C}) )</td>
<td>( \min(\overline{D}, \overline{D}, \text{VAR}_{i+1}) )</td>
<td>( \min(\overline{D}, \overline{D}, \text{VAR}_{i+1}) )</td>
</tr>
</tbody>
</table>

Table 4.120: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the simplified automaton of the MIN_HEIGHT_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>( +\infty )</td>
<td>( +\infty )</td>
<td>( +\infty )</td>
</tr>
<tr>
<td>r</td>
<td>( +\infty )</td>
<td>( +\infty )</td>
<td>( \text{VAR}_{i+1} )</td>
</tr>
<tr>
<td>t</td>
<td>( +\infty )</td>
<td>( \text{VAR}_{i+1} )</td>
<td>( \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>
MIN_HEIGHT_PLAIN

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
<td>Based on the PLAIN pattern.</td>
</tr>
<tr>
<td>Constraint</td>
<td>MIN_HEIGHT_PLAIN(VALUE, VARIABLES)</td>
</tr>
</tbody>
</table>
| Arguments   | VALUE : dvar  
VARIABLES : collection(var−dvar) |
| Restrictions| sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = +∞  
VALUE ≥ minv  
VALUE = +∞ ∨ VALUE ≤ maxv − 1  
required(VARIABLES, var)  
where  
minv = minval(VARIABLES.var)  
maxv = maxval(VARIABLES.var)  
sv = |VARIABLES|  
rv = range(VARIABLES.var) |

VALUE is the minimum of all minimum values in each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘>∗<’.

Assume that the occurrence of the pattern PLAIN starts at position i and ends at position j. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index i + 1 to index j.

Example

(3, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3))

Figure 4.741 provides an example where the MIN_HEIGHT_PLAIN (3, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3]) constraint holds.

Typical

| VARIABLES| > 2  
range(VARIABLES.var) > 1 |

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.741: Illustrating the MIN_HEIGHT_PLAIN constraint of the Example slot
Figures 4.742 and 4.743 respectively depict the automaton associated with the constraint MIN_HEIGHT_PLAIN and its simplified form.

Figure 4.742: Automaton for the MIN_HEIGHT_PLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLAIN pattern where default is $+\infty$.

Figure 4.743: Simplified automaton for the MIN_HEIGHT_PLAIN constraint obtained by applying decoration Table 3.39 to the seed transducer of the PLAIN pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.121: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the MIN_HEIGHTPLAIN constraint defined as the composition of the \texttt{PLAIN} pattern, the feature \texttt{MIN}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\min(\overrightarrow{C}, \overleftarrow{C}))</td>
<td>(\min(\overrightarrow{D}, \overleftarrow{D}, \texttt{VAR}_{i+1}))</td>
<td>(\min(\overrightarrow{D}, \overleftarrow{D}, \texttt{VAR}_{i+1}))</td>
</tr>
<tr>
<td>(r)</td>
<td>(\min(\overrightarrow{C}, \overleftarrow{C}))</td>
<td>(\min(\overrightarrow{D}, \overleftarrow{D}, \texttt{VAR}_{i+1}))</td>
<td>(\min(\overrightarrow{D}, \overleftarrow{D}, \texttt{VAR}_{i+1}))</td>
</tr>
<tr>
<td>(t)</td>
<td>(\min(\overrightarrow{C}, \overleftarrow{C}))</td>
<td>(\min(\overrightarrow{D}, \overleftarrow{D}, \texttt{VAR}_{i+1}))</td>
<td>(\min(\overrightarrow{D}, \overleftarrow{D}, \texttt{VAR}_{i+1}))</td>
</tr>
</tbody>
</table>

Table 4.122: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the simplified automaton of the MIN_HEIGHTPLAIN constraint defined as the composition of the \texttt{PLAIN} pattern, the feature \texttt{MIN}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(+\infty)</td>
<td>(+\infty)</td>
<td>(+\infty)</td>
</tr>
<tr>
<td>(r)</td>
<td>(+\infty)</td>
<td>(\texttt{VAR}_{i+1})</td>
<td>(\texttt{VAR}_{i+1})</td>
</tr>
<tr>
<td>(t)</td>
<td>(+\infty)</td>
<td>(\texttt{VAR}_{i+1})</td>
<td>(\texttt{VAR}_{i+1})</td>
</tr>
</tbody>
</table>
### MIN_HEIGHT_PLATEAU

**Description**

Based on the **PLATEAU** pattern.

**Constraint**

\[ \text{MIN\_HEIGHT\_PLATEAU}(\text{VALUE, VARIABLES}) \]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: \( \text{collection} (\text{var} - \text{dvar}) \)

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{minv} + 1 \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \text{maxv} \\
\text{required}(\text{VARIABLES, var}) & \\
\text{where} & \\
\text{minv} & = \text{minval}(\text{VARIABLES, var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES, var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES, var})
\end{align*}
\]

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the **PLATEAU** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern **PLATEAU** is the maximal subsequence which matches the regular expression \(\langle < = * > \rangle\).

Assume that the occurrence of the pattern **PLATEAU** starts at position \(i\) and ends at position \(j\). The feature **MIN**, called **HEIGHT** in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\((3, \langle 7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5 \rangle)\)

Figure 4.744 provides an example where the **MIN\_HEIGHT\_PLATEAU** \((3, [7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5])\) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES, var}) > 1 \]

**Symmetry**

Items of **VARIABLES** can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.744: Illustrating the MIN_HEIGHT_PLATEAU constraint of the Example slot
Figures 4.745 and 4.746 respectively depict the automaton associated with the constraint MIN_HEIGHT_PLATEAU and its simplified form.

Figure 4.745: Automaton for the MIN_HEIGHT_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLATEAU pattern where default is $+\infty$.

Figure 4.746: Simplified automaton for the MIN_HEIGHT_PLATEAU constraint obtained by applying decoration Table 3.39 to the seed transducer of the PLATEAU pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.123: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the \textsc{MIN}_{HEIGHT}_{PLATEAU} constraint defined as the composition of the \textsc{PLATEAU} pattern, the feature \textsc{MIN}, and the aggregator \textsc{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{center}
\begin{tabular}{ccc}
  \hline
  & \textit{s} & \textit{r} & \textit{t} \\
  \hline
  \textit{s} & \text{min}\bigl(\overrightarrow{C}, \overrightarrow{C}\bigr) & \text{min}\bigl(\overrightarrow{C}, \overrightarrow{C}\bigr) & \text{min}\bigl(\overrightarrow{C}, \overrightarrow{C}\bigr) \\
  \textit{r} & \text{min}\bigl(\overrightarrow{C}, \overrightarrow{C}\bigr) & \text{min}\bigl(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}\bigr) \text{ C} & \text{min}\bigl(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}\bigr) \text{ C} \\
  \textit{t} & \text{min}\bigl(\overrightarrow{C}, \overrightarrow{C}\bigr) & \text{min}\bigl(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}\bigr) \text{ C} & \text{min}\bigl(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}\bigr) \text{ C} \\
  \hline
\end{tabular}
\end{center}

Table 4.124: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the simplified automaton of the \textsc{MIN}_{HEIGHT}_{PLATEAU} constraint defined as the composition of the \textsc{PLATEAU} pattern, the feature \textsc{MIN}, and the aggregator \textsc{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{center}
\begin{tabular}{ccc}
  \hline
  & \textit{s} & \textit{r} & \textit{t} \\
  \hline
  \textit{s} & +\infty & +\infty & +\infty \\
  \textit{r} & +\infty & \text{VAR}_{i+1} \text{ C} & \text{VAR}_{i+1} \text{ C} \\
  \textit{t} & +\infty & \text{VAR}_{i+1} \text{ C} & \text{VAR}_{i+1} \text{ C} \\
  \hline
\end{tabular}
\end{center}
MIN_HEIGHT_PROPER/plain

### Origin
Based on the `PROPER/plain` pattern.

### Constraint

\[
\text{MIN\_HEIGHT\_PROPER\_PLAIN}(\text{VALUE, VARIABLES})
\]

### Arguments

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`

### Restrictions

- \(sv \leq 3 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty\)
- \(\text{VALUE} \geq \minv\)
- \(\text{VALUE} = +\infty \lor \text{VALUE} \leq \maxv - 1\)

**required**(`VARIABLES.var`)

where

- \(\text{minv} = \minval(\text{VARIABLES.var})\)
- \(\text{maxv} = \maxval(\text{VARIABLES.var})\)
- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range}(\text{VARIABLES}.\text{var})\)

### Purpose

**VALUE** is the minimum of all minimum values in each occurrence of the `PROPER/plain` pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value \(+\infty\).

An occurrence of the pattern `PROPER/plain` is the maximal subsequence which matches the regular expression `'>= + <'`.

Assume that the occurrence of the pattern `PROPER/plain` starts at position \(i\) and ends at position \(j\). The feature **MIN**, called **HEIGHT** in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

### Example

\((3, (2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5))\)

Figure 4.747 provides an example where the `MIN_HEIGHT_PROPER/plain` \((3, [2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5])\) constraint holds.

### Typical

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

### Symmetry

Items of **VARIABLES** can be reversed.

### Arg. properties

**Functional dependency**: **VALUE** determined by **VARIABLES**.
Figure 4.747: Illustrating the MIN\_HEIGHT\_PROPER\_PLAIN constraint of the Example slot
Figures 4.748 and 4.749 respectively depict the automaton associated with the constraint MIN\_HEIGHT\_PROPER\_PLAIN and its simplified form.

**Automaton**

\[
\begin{align*}
&\leq s \\
&\leq R \leftarrow \min(R, \min(D, VAR_i)) \\
&\{D \leftarrow \min(D, VAR_i)\} \\
&\{D \leftarrow +\infty\} \\
&\{R \leftarrow \min(R, VAR_i)\} \\
&\leq R \leftarrow \min(R, \min(D, VAR_i)) \\
&\{R \leftarrow \min(R, VAR_i)\} \\
&\leq \{R \leftarrow \text{default}\}
\end{align*}
\]

Figure 4.748: Automaton for the MIN\_HEIGHT\_PROPER\_PLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER\_PLAIN pattern where default is $+$\infty.

Figure 4.749: Simplified automaton for the MIN\_HEIGHT\_PROPER\_PLAIN constraint obtained by applying decoration Table 3.39 to the seed transducer of the PROPER\_PLAIN pattern where default is $+$\infty; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.125: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the MIN_HEIGHT_PROPER_PLAIN constraint defined as the composition of the PROPER_PLAIN pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

Table 4.126: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the simplified automaton of the MIN_HEIGHT_PROPER_PLAIN constraint defined as the composition of the PROPER_PLAIN pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MIN_HEIGHT_PROPER_PLATEAU**

<table>
<thead>
<tr>
<th>AGGREGATOR</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>PROPER_PLATEAU</td>
</tr>
</tbody>
</table>

**DESCRIPTION**

**Origin**

Based on the **PROPER_PLATEAU** pattern.

**Constraint**

\[
\text{MIN_HEIGHT_PROPER_PLATEAU}(\text{VALUE, VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var−dvar)}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
sv & \leq 3 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{minv} + 1 \lor \text{VALUE} = +\infty \\
\text{VALUE} & \leq \text{maxv} \\
\text{required}(\text{VARIABLES, var})
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} & = \text{minval}(\text{VARIABLES, var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES, var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES, var})
\end{align*}
\]

**Purpose**

\text{VALUE} is the minimum of all minimum values in each occurrence of the \text{PROPER_PLATEAU} pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value \(+\infty\).

An occurrence of the pattern \text{PROPER_PLATEAU} is the \textit{maximal} subsequence which matches the regular expression \'\text{< = }+\text{> }\'.

Assume that the occurrence of the pattern \text{PROPER_PLATEAU} starts at position \(i\) and ends at position \(j\). The feature \text{MIN}, called \text{HEIGHT} in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
(3, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))
\]

Figure 4.750 provides an example where the \text{MIN_HEIGHT_PROPER_PLATEAU} \((3, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES, var}) > 1
\]

**Symmetry**

Items of \text{VARIABLES} can be reversed.

**Arg. properties**

Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
Figure 4.750: Illustrating the MIN_HEIGHT_PROPER_PLATEAU constraint of the Example slot
Automaton

Figures 4.751 and 4.752 respectively depict the automaton associated with the constraint MIN_HEIGHT_PROPER_PLATEAU and its simplified form.

Figure 4.751: Automaton for the MIN_HEIGHT_PROPER_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER_PLATEAU pattern where default is $+\infty$.

Figure 4.752: Simplified automaton for the MIN_HEIGHT_PROPER_PLATEAU constraint obtained by applying decoration Table 3.39 to the seed transducer of the PROPER_PLATEAU pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.127: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the \texttt{MIN\_HEIGHT\_PROPER\_PLATEAU} constraint defined as the composition of the \texttt{PROPER\_PLATEAU} pattern, the feature \texttt{MIN}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
<th>( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \min(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \min(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \min(\overrightarrow{C}, \overrightarrow{C}) )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \min(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}) )</td>
<td>( \text{VAR}_{i+1} )</td>
</tr>
<tr>
<td>( t )</td>
<td>( \min(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}) )</td>
<td>( \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>

Table 4.128: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the simplified automaton of the \texttt{MIN\_HEIGHT\_PROPER\_PLATEAU} constraint defined as the composition of the \texttt{PROPER\_PLATEAU} pattern, the feature \texttt{MIN}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
<th>( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( +\infty )</td>
<td>( +\infty )</td>
<td>( +\infty )</td>
</tr>
<tr>
<td>( r )</td>
<td>( +\infty )</td>
<td>( +\infty )</td>
<td>( \text{VAR}_{i+1} )</td>
</tr>
<tr>
<td>( t )</td>
<td>( +\infty )</td>
<td>( \text{VAR}_{i+1} )</td>
<td>( \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>
**MIN_HEIGHT_STEADY**

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <strong>STEADY</strong> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>$\text{MIN_HEIGHT_STEADY}(\text{VALUE, VARIABLES})$</td>
</tr>
</tbody>
</table>
| **Arguments**   | \begin{align*}
  \text{VALUE} & : \text{dvar} \\
  \text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*} |
| **Restrictions**| \begin{align*}
  sv & \leq 1 \Rightarrow \text{VALUE} = +\infty \\
  \text{VALUE} & \geq \text{minv} \\
  
  \begin{cases}
    \text{VALUE} = +\infty \lor \\
    \text{VALUE} \leq \text{maxv}
  \end{cases} & \text{required}(\text{VARIABLES.var})
\end{align*} \\
where \begin{align*}
  \text{minv} & = \text{minval}(\text{VARIABLES.var}) \\
  \text{maxv} & = \text{maxval}(\text{VARIABLES.var}) \\
  sv & = |\text{VARIABLES}|
\end{align*} |

**Purpose**
VALUE is the minimum of all minimum values in each occurrence of the **STEADY** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $+\infty$.

An occurrence of the pattern **STEADY** is the subsequence which matches the regular expression '='.
Assume that the occurrence of the pattern **STEADY** starts at position $i$ and ends at position $j$. The feature **MIN**, called **HEIGHT** in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i$ to index $j+1$.

**Example**
(1, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6))

Figure 4.753 provides an example where the **MIN\_HEIGHT\_STEADY** (1, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6]) constraint holds.

**Typical**
$|\text{VARIABLES}| > 1$

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
MIN_HEIGHT_STEADY

Figure 4.753: Illustrating the MIN_HEIGHT_STEADY constraint of the Example slot
Figures 4.754 and 4.755 respectively depict the automaton associated with the constraint \( \text{MIN\_HEIGHT\_STEADY} \) and its simplified form.

\[
\begin{cases}
C \leftarrow \text{default} \\
D \leftarrow +\infty \\
R \leftarrow \text{default}
\end{cases}
\]

\[
\{ D \leftarrow +\infty \} \quad \geq
\}
\]

Figure 4.754: Automaton for the \( \text{MIN\_HEIGHT\_STEADY} \) constraint obtained by applying decoration Table 3.37 to the seed transducer of the \( \text{STEADY} \) pattern where \( \text{default} \) is \( +\infty \).

\[
\{ R \leftarrow \text{default} \}
\]

\[
\{ R \leftarrow \min(R, \text{VAR}_i) \} \quad \geq
\]

\[
\min(R, C)
\]

Figure 4.755: Simplified automaton for the \( \text{MIN\_HEIGHT\_STEADY} \) constraint obtained by applying decoration Table 3.39 to the seed transducer of the \( \text{STEADY} \) pattern where \( \text{default} \) is \( +\infty \); \( -R_i + R_{i-1} \geq 0 \) is a linear invariant.

\[
\begin{array}{c}
\text{s} \\
\text{min}((\frac{C_i}{C_i}, \frac{C}{C})
\end{array}
\]

Table 4.129: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the \( \text{MIN\_HEIGHT\_STEADY} \) constraint defined as the composition of the \( \text{STEADY} \) pattern, the feature \( \text{MIN} \), and the aggregator \( \text{min} \); cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.130: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the simplified automaton of the MIN_HEIGHT_STEADY constraint defined as the composition of the STEADY pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### MIN_HEIGHT_STEADY_SEQUENCE

**Description**
Based on the STEADY_SEQUENCE pattern.

**Constraint**
MIN_HEIGHT_STEADY_SEQUENCE(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
- $sv \leq 1 \Rightarrow VALUE = +\infty$
- $VALUE \geq minv$
- $VALUE = +\infty \lor VALUE \leq maxv$

where
- $minv = \text{minval}(\text{VARIABLES}.\text{var})$
- $maxv = \text{maxval}(\text{VARIABLES}.\text{var})$
- $sv = |\text{VARIABLES}|$

**Purpose**
During the occurrence of the pattern STEADY_SEQUENCE, the minimal subsequence which matches the regular expression ‘=+’ is identified. The feature MIN, called HEIGHT in the name of the constraint, computes the minimum of the values from index $i$ to index $j+1$.

**Example**
$\{1, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1)\}$

Figure 4.756 provides an example where the MIN_HEIGHT_STEADY_SEQUENCE constraint holds.

**Typical**
$|\text{VARIABLES}| > 1$

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.756: Illustrating the MIN_HEIGHT_STEADY_SEQUENCE constraint of the Example slot
Figures 4.757 and 4.758 respectively depict the automaton associated with the constraint MIN_HEIGHT_STEADY_SEQUENCE and its simplified form.

**Automaton**

The automaton is defined by the following transition rules:

- Initial state: $C \leftarrow \text{default}$, $D \leftarrow +\infty$, $R \leftarrow \text{default}$
- Transition function:

  - If $C \geq 8$, then $R \leftarrow \text{default}$
  - If $D \leftarrow +\infty$, then $R \leftarrow \text{default}$
  - If $R \leftarrow \min(R, C)$, then $C \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1})$
  - If $D \leftarrow +\infty$

The simplified automaton is obtained by applying decoration Table 3.39 to the seed transducer of the STEADY_SEQUENCE pattern where $\text{default} = +\infty$

**Figure 4.757**: Automaton for the MIN_HEIGHT_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STEADY_SEQUENCE pattern where $\text{default}$ is $+\infty$.

**Figure 4.758**: Simplified automaton for the MIN_HEIGHT_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.39 to the seed transducer of the STEADY_SEQUENCE pattern where $\text{default}$ is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.131: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the MIN_HEIGHT_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>r</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}, \overline{D}, \overline{D}))</td>
</tr>
</tbody>
</table>

Table 4.132: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the simplified automaton of the MIN_HEIGHT_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>+(\infty)</td>
<td>+(\infty)</td>
</tr>
<tr>
<td>r</td>
<td>+(\infty)</td>
<td>+(\infty)</td>
</tr>
</tbody>
</table>
### MIN_INCREASING

**Description**

Based on the INCREASING pattern.

**Constraint**

\[
MIN\_INCREASING(VARIABLES, FEATURES, DEFAULT)
\]

**Arguments**

- VARIABLES: \(\text{collection}(\text{var} - \text{dvar})\)
- FEATURES: \(\text{collection}(\text{var} - \text{dvar})\)
- DEFAULT: \(\text{int}\)

**Restrictions**

\[
\begin{align*}
\text{required}(\text{VARIABLES}.\text{var}) \\
\text{required}(\text{FEATURES}.\text{var}) \\
sv &= |\text{FEATURES}| \\
sv \leq 1 \lor rv \leq 1 &\Rightarrow \text{FEATURES}.\text{var} = \text{DEFAULT} \\
\text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \geq \text{minv} \\
\text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \leq \text{maxv} - 1 \\
\text{DEFAULT} < \text{minv} \lor \text{DEFAULT} > \text{maxv} - 1 \\
\text{where} \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of INCREASING is identified then FEATURES\([i]\) is the default value DEFAULT; otherwise FEATURES\([i]\) gives the feature value of the corresponding occurrence of INCREASING.

**Purpose**

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression ‘\(<\)’. Assume that the occurrence of the pattern INCREASING starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

Figure 4.759 provides an example where the MIN_INCREASING ([4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 3, 0, 0, 0, 0, 1, 0, 3, 4, 0, 0, 0, 1, 0, 0], 0) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.759: Illustrating the MIN_INCREASING constraint of the Example slot
Use the decoration table 3.32 to synthesise the corresponding register automaton.
MIN_INCREASING_SEQUENCE

**DESCRIPTION**

Origin

Based on the **INCREASING_SEQUENCE** pattern.

Constraint

\[ \text{MIN\_INCREASING\_SEQUENCE}(\text{VARIABLES}, \text{FEATURES}, \text{DEFAULT}) \]

Arguments

- **VARIABLES**: collection(var–dvar)
- **FEATURES**: collection(var–dvar)
- **DEFAULT**: int

Restrictions

- \( \text{required}(\text{VARIABLES}.\text{var}) \)
- \( \text{required}(\text{FEATURES}.\text{var}) \)
- \( \text{sv} = |\text{FEATURES}| \)
- \( \text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{FEATURES}.\text{var} = \text{DEFAULT} \)
- \( \text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \geq \text{minv} \)
- \( \text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \leq \text{maxv} - 1 \)
- \( \text{DEFAULT} < \text{minv} \lor \text{DEFAULT} > \text{maxv} - 1 \)

where

- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \)

Purpose

Consider the time-series given by the **VARIABLES** collection. If position \( i \) does not correspond to the first position in **VARIABLES** where an occurrence of **INCREASING_SEQUENCE** is identified (even if this occurrence of pattern is not complete) then **FEATURES\[i\]** is the default value **DEFAULT**; otherwise **FEATURES\[i\]** gives the feature value of the corresponding occurrence of **INCREASING_SEQUENCE**.

An occurrence of the pattern **INCREASING_SEQUENCE** is the maximal subsequence which matches the regular expression ‘\(< (< | =)^* < | <\)’.

Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **MIN** computes the minimum of the values from index \( i \) to index \( j + 1 \).

Example

Figure 4.760 provides an example where the **MIN_INCREASING_SEQUENCE** constraint holds.

Typical

- \( |\text{VARIABLES}| > 1 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

Arg. properties

Functional dependency: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.760: Illustrating the MIN_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
MIN_INFLEXION

**Origin**
Based on the INFLEXION pattern.

**Constraint**
MIN_INFLEXION(VARIABLES, FEATURES, DEFAULT)

**Arguments**
- VARIABLES : collection(var−dvar)
- FEATURES : collection(var−dvar)
- DEFAULT : int

**Restrictions**
- required(VARIABLES, var)
- required(FEATURES, var)
- sv = |FEATURES|
- sv ≤ 2 \(\lor\) rv ≤ 1 ⇒ FEATURES.var = DEFAULT
- FEATURES.var = DEFAULT \(\lor\) FEATURES.var ≥ minv
- FEATURES.var = DEFAULT \(\lor\) FEATURES.var ≤ maxv
- DEFAULT < minv \(\lor\) DEFAULT > maxv

   where
   - maxv = maxval(VARIABLES.var)
   - minv = minval(VARIABLES.var)
   - sv = |VARIABLES|
   - rv = range(VARIABLES.var)

**Purpose**
Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of INFLEXION is identified then FEATURES[\(i\)] is the default value DEFAULT; otherwise FEATURES[\(i\)] gives the feature value of the corresponding occurrence of INFLEXION.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (< | =)^* > | > (> | =)^* <\).

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i+1\) to index \(j\).

**Example**
Figure 4.761 provides an example where the MIN_INFLEXION ([1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 4, 4], [0, 0, 0, 2, 0, 0, 3, 5, 2, 5, 1, 5, 0, 3, 0, 0], 0) constraint holds.

**Typical**
- |VARIABLES| > 2
- range(VARIABLES.var) > 1

**Arg. properties**
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.761: Illustrating the MIN_INFLEXION constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
MIN_MAX_BUMP_ON_DECREASING_SEQUENCE

### Description

**Origin**
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**

\[
\text{MIN\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 5 \lor rv \leq 2 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \minv + 2 \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \maxv \\
\end{align*}
\]

\[
\text{required}(\text{VARIABLES, var})
\]

where

\[
\begin{align*}
\minv & = \minval(\text{VARIABLES, var}) \\
\maxv & = \maxval(\text{VARIABLES, var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES, var})
\end{align*}
\]

**Purpose**

**VALUE** is the minimum of all maximum values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, **VALUE** takes the default value \(+\infty\).

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression '>
\text{VALUE} \geq 2
\]

**Example**

\[(5, \langle 7, 6, 5, 6, 5, 4, 1, 7, 5, 4, 2, 5, 4, 3, 3 \rangle)\]

Figure 4.762 provides an example where the MIN_MAX_BUMP_ON_DECREASING_SEQUENCE constraint holds.

**Typical**

\[|\text{VARIABLES}| > 5 \]
\[\text{range}(\text{VARIABLES, var}) > 2\]

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.762: Illustrating the MIN_MAX_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Figures 4.763 and 4.764 respectively depict the automaton associated with the constraint MIN_MAX_BUMP_ON_DECREASING_SEQUENCE and its simplified form.

Figure 4.763: Automaton for the MIN_MAX_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $+\infty$. 
Figure 4.764: Simplified automaton for the MIN_MAX_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.28 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
## MIN_MAX_DECREASING

### Origin
Based on the \textsc{DECREASING} pattern.

### Constraint
\[
\text{MIN\_MAX\_DECREASING}(\text{VALUE}, \text{VARIABLES})
\]

### Arguments
\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

### Restrictions
\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \implies \text{VALUE} = +\infty \\
\text{VALUE} \geq \text{minv} + 1 & \\
\text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} & \text{required}((\text{VARIABLES}, \text{var})
\end{align*}
\]

where
\[
\begin{align*}
\text{minv} & = \text{minval}((\text{VARIABLES}, \text{var})) \\
\text{maxv} & = \text{maxval}((\text{VARIABLES}, \text{var})) \\
sv & = |(\text{VARIABLES})| \\
rv & = \text{range}((\text{VARIABLES}, \text{var}))
\end{align*}
\]

### Purpose
\[
\text{VALUE} \text{ is the minimum of all maximum values in each occurrence of the \textsc{DECREASING} pattern in the time-series given by the } \text{VARIABLES} \text{ collection. If the pattern does not occur, } \text{VALUE} \text{ takes the default value } +\infty.
\]

An occurrence of the pattern \textsc{DECREASING} is the subsequence which matches the regular expression ‘\textgreater\textgreater’.

Assume that the occurrence of the pattern \textsc{DECREASING} starts at position \(i\) and ends at position \(j\). The feature \textsc{MAX} computes the maximum of the values from index \(i\) to index \(j + 1\).

### Example
\[
(3, (3, 4, 2, 5, 6, 4, 3, 1, 1, 4, 6, 4, 4))
\]

Figure 4.765 provides an example where the \textsc{MIN\_MAX\_DECREASING} \((3, [3, 4, 2, 5, 6, 4, 3, 1, 1, 4, 6, 4])\) constraint holds.

### Typical
\[
\begin{align*}
|\text{VARIABLES}| > 1 \\
\text{range}((\text{VARIABLES}, \text{var})) > 1
\end{align*}
\]

### Arg. properties
Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
Figure 4.765: Illustrating the MIN_MAX_DECREASING constraint of the Example slot
Figures 4.766 and 4.767 respectively depict the automaton associated with the constraint MIN_MAX_DECRESHING and its simplified form.

Figure 4.766: Automaton for the MIN_MAX_DECRESHING constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASHING pattern where default is $+\infty$

Figure 4.767: Simplified automaton for the MIN_MAX_DECRESHING constraint obtained by applying decoration Table 3.39 to the seed transducer of the DECREASHING pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

Table 4.133: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the MIN_MAX_DECRESHING constraint defined as the composition of the DECREASHING pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.134: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the simplified automaton of the MIN_MAX_DECREASING constraint defined as the composition of the DECREASING pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin  Based on the DECREASING_SEQUENCE pattern.

Constraint  MIN_MAX_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments  VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions  \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \)
\( VALUE \geq minv + 1 \)
\( VALUE = +\infty \lor VALUE \leq maxv \)
required(VARIABLES, var)

where  
\( minv = minval(VARIABLES.var) \)
\( maxv = maxval(VARIABLES.var) \)
\( sv = |VARIABLES| \)
\( rv = range(VARIABLES.var) \)

Purpose  
VALUE is the minimum of all maximum values in each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\( > (> | =) \ast > | > \)’.

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i \) to index \( j + 1 \).

Example  
\( (4, (3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4)) \)

Figure 4.768 provides an example where the MIN_MAX_DECREASING_SEQUENCE \((4, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\) constraint holds.

Typical  
\( |VARIABLES| > 1 \)
\( range(VARIABLES.var) > 1 \)

Arg. properties  
Functional dependency: VALUE determined by VARIABLES.
Figure 4.768: Illustrating the MIN_MAX_DECREASING_SEQUENCE constraint of the Example slot
Figures 4.769 and 4.770 respectively depict the automaton associated with the constraint MIN\_MAX\_DECREASING\_SEQUENCE and its simplified form.

Figure 4.769: Automaton for the MIN\_MAX\_DECREASING\_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING\_SEQUENCE pattern where default is $+\infty$.

Figure 4.770: Simplified automaton for the MIN\_MAX\_DECREASING\_SEQUENCE constraint obtained by applying decoration Table 3.39 to the seed transducer of the DECREASING\_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.135: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the MIN_MAX_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

Table 4.136: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the simplified automaton of the MIN_MAX_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MIN_MAX_DIP_ON_INCREASING_SEQUENCE**

**DESCRIPTION**

**Origin**
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**

\[
\text{MIN_MAX_DIP_ON_INCREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var-dvar)

**Restrictions**

- \(sv \leq 5 \vee rv \leq 2 \Rightarrow \text{VALUE} = +\infty\)
- \(\text{VALUE} \geq \minv + 2\)
- \(\text{VALUE} = +\infty \vee \text{VALUE} \leq \maxv\)

**Purpose**

\(\text{VALUE}\) is the minimum of all maximum values in each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \(\text{VALUE}\) takes the default value \(+\infty\).

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression ‘<<><<<’.

Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature \(\text{MAX}\) computes the maximum of the values from index \(i + 2\) to index \(j\).

**Example**

- \((5, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))\)

Figure 4.771 provides an example where the MIN_MAX_DIP_ON_INCREASING_SEQUENCE \((5, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4])\) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 5\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 2\)

**Arg. properties**

- Functional dependency: \(\text{VALUE}\) determined by \(\text{VARIABLES}\).
Figure 4.771: Illustrating the MIN_MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Figures 4.772 and 4.773 respectively depict the automaton associated with the constraint \( \text{MIN\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE} \) and its simplified form.

Figure 4.772: Automaton for the \( \text{MIN\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE} \) constraint obtained by applying decoration Table 3.37 to the seed transducer of the \( \text{DIP\_ON\_INCREASING\_SEQUENCE} \) pattern where \text{default} is \(+\infty\)
Figure 4.773: Simplified automaton for the MIN\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE constraint obtained by applying decoration Table 3.26 to the seed transducer of the DIP\_ON\_INCREASING\_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
MIN_MAX_INCREASING

**Origin**
Based on the **INCREASING** pattern.

**Constraint**

\[
\text{MIN\_MAX\_INCREASING}(\text{VALUE, VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var - dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 \lor \text{rv} \leq 1 &\Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq \text{minv} + 1 &\Rightarrow \text{VALUE} \geq \text{minv} + 1 \\
\text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} &\Rightarrow \text{VALUE} \geq \text{minv} + 1
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the minimum of all maximum values in each occurrence of the **INCREASING** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression ‘<’.

Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\(3, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)\)

Figure 4.774 provides an example where the **MIN\_MAX\_INCREASING** \((3, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.774: Illustrating the MIN_MAX_INCREASING constraint of the Example slot
Automaton

Figures 4.775 and 4.776 respectively depict the automaton associated with the constraint MIN_MAX_INCREASING and its simplified form.

```
\begin{align*}
&\{ C \leftarrow \text{default} \\
&\quad D \leftarrow -\infty \\
&\quad R \leftarrow \text{default} \}
\end{align*}
```

Figure 4.775: Automaton for the MIN_MAX_INCREASING constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING pattern where default is $+\infty$.

```
\begin{align*}
&\{ R \leftarrow \text{default} \}
\end{align*}
```

Figure 4.776: Simplified automaton for the MIN_MAX_INCREASING constraint obtained by applying decoration Table 3.40 to the seed transducer of the INCREASING pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

Table 4.137: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the MIN_MAX_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.138: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the simplified automaton of the MIN_MAX_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MIN_MAX_INCREASING_SEQUENCE**

**Origin**: Based on the **INCREMENTING_SEQUENCE** pattern.

**Constraint**: 

\[
\text{MIN\_MAX\_INCREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**:

- \text{VALUE} : \text{dvar}
- \text{VARIABLES} : \text{collection(}\text{var} - \text{dvar}\text{)}

**Restrictions**:

\[
\begin{align*}
sv \leq 1 & \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq \text{minv} + 1 & \\
\text{VALUE} = +\infty & \lor \text{VALUE} \leq \maxv \\
\text{VALUE} \geq 1 & \\
\text{VALUE} \geq \text{minv} + 1 & \\
\text{VALUE} \geq \text{maxv} & \\
\text{VALUE} \geq 1 & \\
\end{align*}
\]

where

- \text{minv} = \text{minval}(\text{VARIABLES.}\text{var})
- \text{maxv} = \text{maxval}(\text{VARIABLES.}\text{var})
- \text{sv} = |\text{VARIABLES}|$
- \text{rv} = \text{range}(\text{VARIABLES.}\text{var})$

**Purpose**:

\text{VALUE} is the minimum of all maximum values in each occurrence of the **INCREMENTING_SEQUENCE** pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value \( +\infty \). An occurrence of the pattern **INCREMENTING_SEQUENCE** is the maximal subsequence which matches the regular expression ‘\(< (< | =)^* < | <\)’. Assume that the occurrence of the pattern **INCREMENTING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature \text{MAX} computes the maximum of the values from \( i \) to index \( j + 1 \).

**Example**:

\( (3, 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3) \)

Figure 4.777 provides an example where the **MIN\_MAX\_INCREASING\_SEQUENCE** \( (3, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) \) constraint holds.

**Typical**:

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES.}\text{var}) > 1
\]

**Arg. properties**:

Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
Figure 4.777: Illustrating the MIN_MAX_INCREASING_SEQUENCE constraint of the Example slot
Figures 4.778 and 4.779 respectively depict the automaton associated with the constraint MIN_MAX_INCREASING_SEQUENCE and its simplified form.

Figure 4.778: Automaton for the MIN_MAX_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $+\infty$.

Figure 4.779: Simplified automaton for the MIN_MAX_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.25 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.139: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the \textsc{MIN\_MAX\_INCREASING\_SEQUENCE} constraint defined as the composition of the \textsc{INCREASING\_SEQUENCE} pattern, the \textsc{feature MAX}, and the \textsc{aggregator min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{center}
\begin{tabular}{|c|c|}
\hline
s & t \\
\hline
s & \text{min}(\overline{C}, \overline{C}) & \text{min}(\overline{C}, \overline{C}) \\
\hline
\end{tabular}
\end{center}

Table 4.140: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the simplified automaton of the \textsc{MIN\_MAX\_INCREASING\_SEQUENCE} constraint defined as the composition of the \textsc{INCREASING\_SEQUENCE} pattern, the \textsc{feature MAX}, and the \textsc{aggregator min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{center}
\begin{tabular}{|c|c|}
\hline
s & t \\
\hline
s & +\infty & +\infty \\
\hline
\end{tabular}
\end{center}
MIN_MAX_INFLEXION

Origin  
Based on the INFLEXION pattern.

Constraint  
\[ \text{MIN\_MAX\_INFLEXION}(\text{VALUE}, \text{VARIABLES}) \]

Arguments  
\[ \text{VALUE} : \text{dvar} \]
\[ \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \]

Restrictions  
\[ \text{sv} \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \]
\[ \text{VALUE} \geq \text{minv} \]
\[ \text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} \]
\[ \text{required}(\text{VARIABLES}, \text{var}) \]

where  
\[ \text{minv} = \text{minval}(\text{VARIABLES} \cdot \text{var}) \]
\[ \text{maxv} = \text{maxval}(\text{VARIABLES} \cdot \text{var}) \]
\[ \text{sv} = |\text{VARIABLES}| \]
\[ \text{rv} = \text{range}(\text{VARIABLES} \cdot \text{var}) \]

Purpose  
VALUE is the minimum of all maximum values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \('<(\leq|\geq|\ast)>|>(\geq|\leq|\ast)\ast<'\).

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

Example  
\[(1, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))\]

Figure 4.780 provides an example where the MIN_MAX_INFLEXION \((1, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])\) constraint holds.

Typical  
\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES} \cdot \text{var}) > 1 \]

Symmetry  
Items of VARIABLES can be reversed.

Arg. properties  
Functional dependency: VALUE determined by VARIABLES.
Figure 4.780: Illustrating the MIN_MAX_INFLEXION constraint of the Example slot
Automaton

Figures 4.781 and 4.782 respectively depict the automaton associated with the constraint MIN_MAX_INFLEXION and its simplified form.

Figure 4.781: Automaton for the MIN_MAX_INFLEXION constraint obtained by applying decoration Table 3.37 to the seed transducer of the INFLEXION pattern where default is $+\infty$ (transition $r \rightarrow t$ has the same registers updates as transition $t \rightarrow r$).

Figure 4.782: Simplified automaton for the MIN_MAX_INFLEXION constraint obtained by applying decoration Table 3.26 to the seed transducer of the INFLEXION pattern where default is $+\infty$ (transition $r \rightarrow t$ has the same registers updates as transition $t \rightarrow r$); $-R_i + R_{i-1} \geq 0$ is a linear invariant.
MIN_MAX_INFLEXION
**MIN_MAX_PEAK**

**DESCRIPTION**

Origin

Based on the **PEAK** pattern.

Constraint

\[ \text{MIN\_MAX\_PEAK}(\text{VALUE}, \text{VARIABLES}) \]

Arguments

<table>
<thead>
<tr>
<th>VALUE</th>
<th>: dvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>: collection(var−dvar)</td>
</tr>
</tbody>
</table>

Restrictions

\[ sv \leq 2 \lor rv \leq 1 \rightarrow \text{VALUE} = +\infty \]
\[ \text{VALUE} \geq \text{minv} + 1 \]
\[ \text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} \]

\[ \text{required} (\text{VARIABLES}, \text{var}) \]

where

\[ \text{minv} = \text{minval} (\text{VARIABLES}.\text{var}) \]
\[ \text{maxv} = \text{maxval} (\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range} (\text{VARIABLES}.\text{var}) \]

**Purpose**

VALUE is the minimum of all maximum values in each occurrence of the PEAK pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

An occurrence of the pattern **PEAK** is the maximal subsequence which matches the regular expression '\(< (= | <)^(>| =)^* >\)'.

Assume that the occurrence of the pattern **PEAK** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i + 1\) to index \(j\).

Example

\((3, (7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1))\)

Figure 4.783 provides an example where the **MIN\_MAX\_PEAK** (3, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1]) constraint holds.

Typical

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range} (\text{VARIABLES}.\text{var}) > 1 \]

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.783: Illustrating the MIN_MAX_PEAK constraint of the Example slot
Automaton

Figures 4.784 and 4.785 respectively depict the automaton associated with the constraint MIN_MAX_PEAK and its simplified form.

Figure 4.784: Automaton for the MIN_MAX_PEAK constraint obtained by applying decoration Table 3.37 to the seed transducer of the PEAK pattern where default is $+\infty$.

Figure 4.785: Simplified automaton for the MIN_MAX_PEAK constraint obtained by applying decoration Table 3.39 to the seed transducer of the PEAK pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.141: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the MIN_MAX PEAK constraint defined as the composition of the PEAK pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min(( \overrightarrow{C}, \overrightarrow{C} ))</td>
<td>min(( \overrightarrow{C}, \overrightarrow{C} ))</td>
<td>min(( \overrightarrow{C}, \overrightarrow{C} ))</td>
</tr>
<tr>
<td>r</td>
<td>min(( \overrightarrow{C}, \overrightarrow{C} ))</td>
<td>max(( \overrightarrow{D}, \overrightarrow{D}, \text{VAR}<em>{i+1} &quot;, \text{VAR}</em>{i+1} )))</td>
<td>max(( \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1} ))</td>
</tr>
<tr>
<td>t</td>
<td>min(( \overrightarrow{C}, \overrightarrow{C} ))</td>
<td>max(( \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1} ))</td>
<td>min(( \overrightarrow{C}, \overrightarrow{C} ))</td>
</tr>
</tbody>
</table>

Table 4.142: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the simplified automaton of the MIN_MAX PEAK constraint defined as the composition of the PEAK pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>+( \infty )</td>
<td>+( \infty )</td>
<td>+( \infty )</td>
</tr>
<tr>
<td>r</td>
<td>+( \infty )</td>
<td>( \text{VAR}_{i+1} )</td>
<td>+( \infty )</td>
</tr>
<tr>
<td>t</td>
<td>+( \infty )</td>
<td>+( \infty )</td>
<td>+( \infty )</td>
</tr>
</tbody>
</table>
### MIN_MAX_STRICTLY_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the \textit{STRICTLY_DECREASING_SEQUENCE} pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>\texttt{MIN_MAX_STRICTLY_DECREASING_SEQUENCE}(\texttt{VALUE, VARIABLES})</td>
</tr>
<tr>
<td>Arguments</td>
<td>\texttt{VALUE} : \texttt{dvar}</td>
</tr>
<tr>
<td></td>
<td>\texttt{VARIABLES} : \texttt{collection(var_dvar)}</td>
</tr>
</tbody>
</table>
| Restrictions | \begin{align*} sv \leq 1 & \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE & \geq \text{minv} + 1 \\
VALUE & = +\infty \lor VALUE \leq \text{maxv} \\
\text{required}(\text{VARIABLES, var}) & \end{align*} |
|           | where \begin{align*} 
\text{minv} & = \text{minval}(\text{VARIABLES, var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES, var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES, var}) \end{align*} |
| Purpose | \texttt{VALUE} is the minimum of all maximum values in each occurrence of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection. If the pattern does not occur, \texttt{VALUE} takes the default value \texttt{+\infty}. An occurrence of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} is the maximal subsequence which matches the regular expression ‘\texttt{>}\texttt{}’.

Assume that the occurrence of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{MAX} computes the maximum of the values from index \texttt{i} to index \texttt{j} + 1. |
| Example | \[(4, 4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3)]\ |

Figure 4.786 provides an example where the \texttt{MIN\_MAX\_STRICTLY\_DECREASING\_SEQUENCE} \((4, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3])\) constraint holds. |
| Typical | \begin{align*} |\text{VARIABLES}| & > 1 \\
\text{range}(\text{VARIABLES, var}) & > 1 \end{align*} |
| Arg. properties | Functional dependency: \texttt{VALUE} determined by \texttt{VARIABLES}. |
Figure 4.786: Illustrating the MIN_MAX_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.787 and 4.788 respectively depict the automaton associated with the constraint MIN_MAX_STRICTLY_DECREASING_SEQUENCE and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow -\infty \\
& R \leftarrow \text{default} \\
\} \\
\begin{array}{c}
\leq S \\
\end{array} \\
\begin{array}{c}
\leq \\
\end{array} \\
\begin{array}{c}
\leq \\
\end{array} \\
\begin{array}{c}
\leq \\
\end{array} \\
\begin{array}{c}
\leq \\
\end{array} \\
\}\end{align*}
\]

Figure 4.787: Automaton for the MIN_MAX_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $+\infty$.

\[
\begin{align*}
\{ & R \leftarrow \text{default} \\
\} \\
\begin{array}{c}
\leq S \\
\end{array} \\
\begin{array}{c}
\leq \\
\end{array} \\
\begin{array}{c}
\leq \\
\end{array} \\
\begin{array}{c}
\leq \\
\end{array} \\
\}\end{align*}
\]

Figure 4.788: Simplified automaton for the MIN_MAX_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.39 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.143: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the `MIN_MAX_STRICTLY_DECREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_DECREASING_SEQUENCE` pattern, the feature `MAX`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\min(C, \bar{C}))</td>
<td>(\min(C, \bar{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\min(C, \bar{C}))</td>
<td>(\max(C, \bar{C}, \bar{D}, \bar{D}))</td>
</tr>
</tbody>
</table>

Table 4.144: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the `MIN_MAX_STRICTLY_DECREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_DECREASING_SEQUENCE` pattern, the feature `MAX`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>+(\infty)</td>
<td>(\bar{C})</td>
</tr>
<tr>
<td>r</td>
<td>+(\infty)</td>
<td>+(\infty)</td>
</tr>
</tbody>
</table>
**MIN_MAX STRICTLY_INCREASING_SEQUENCE**

**DESCRIPTION**

Origin

Based on the **STRICTLY_INCREASING_SEQUENCE** pattern.

Constraint

\[
\text{MIN\_MAX\_STRICTLY\_INCREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

Arguments

- \text{VALUE} : \text{dvar}
- \text{VARIABLES} : \text{collection(\text{var}\,\text{dvar})}

Restrictions

- \text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty
- \text{VALUE} \geq \text{minv} + 1
- \text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv}

**Purpose**

VALUE is the minimum of all maximum values in each occurrence of the **STRICTLY_INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \( +\infty \).

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the maximal sub-sequence which matches the regular expression ‘<+’.

Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature \text{MAX} computes the maximum of the values from index \( i \) to index \( j + 1 \).

Example

\[(3, 4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3)\]

Figure 4.789 provides an example where the **MIN_MAX STRICTLY_INCREASING_SEQUENCE** constraint holds.

Typical

- \(|\text{VARIABLES}| > 1\)
- \(\text{range(\text{VARIABLES}.\text{var})} > 1\)

Arg. properties

Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
Figure 4.789: Illustrating the MIN_MAX_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.790 and 4.791 respectively depict the automaton associated with the constraint MIN\_MAX\_STRICLTY\_INCREASING\_SEQUENCE and its simplified form.

Figure 4.790: Automaton for the MIN\_MAX\_STRICLTY\_INCREASING\_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICLTY\_INCREASING\_SEQUENCE pattern where default is $+\infty$.

Figure 4.791: Simplified automaton for the MIN\_MAX\_STRICLTY\_INCREASING\_SEQUENCE constraint obtained by applying decoration Table 3.25 to the seed transducer of the STRICLTY\_INCREASING\_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.145: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the MIN_MAX_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overline{C}, \overline{C})$</td>
<td>$\min(\overline{C}, \overline{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overline{C}, \overline{C})$</td>
<td>$\max(\overline{C}, \overline{C}, \overline{D}, \overline{D})$</td>
</tr>
</tbody>
</table>

Table 4.146: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the simplified automaton of the MIN_MAX_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$+\infty$</td>
<td>$+\infty$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\overline{C}$</td>
<td>$+\infty$</td>
</tr>
</tbody>
</table>
**MIN_MAX_SUMMIT**

**Origin**
Based on the SUMMIT pattern.

**Constraint**
MIN_MAX_SUMMIT(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \)
- \( VALUE \geq minv + 1 \)
- \( VALUE = +\infty \lor VALUE \leq maxv \)
- \( required(VARIABLES, var) \)

where
- \( minv = minval(VARIABLES.var) \)
- \( maxv = maxval(VARIABLES.var) \)
- \( sv = |VARIABLES| \)
- \( rv = range(VARIABLES.var) \)

**Purpose**
VALUE is the minimum of all maximum values in each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression \( '(< | < (= | <)*)>(> | > (= | >)*)>' \).

Assume that the occurrence of the pattern SUMMIT starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i + 1 \) to index \( j \).

**Example**
\[(3, 7, 1, 5, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1)\]

Figure 4.792 provides an example where the MIN_MAX_SUMMIT \((3, 7, 1, 5, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1)\) constraint holds.

**Typical**
- \(|VARIABLES| > 2\)
- \(range(VARIABLES.var) > 1\)

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.792: Illustrating the MIN_MAX_SUMMIT constraint of the Example slot
Automaton

Figures 4.793 and 4.794 respectively depict the automaton associated with the constraint MIN_MAX_SUMMIT and its simplified form.

Figure 4.793: Automaton for the MIN_MAX_SUMMIT constraint obtained by applying decoration Table 3.37 to the seed transducer of the SUMMIT pattern where default is $+\infty$ (transition $r \rightarrow u$ has the same register update as transition $r \rightarrow u$)

Table 4.147: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the MIN_MAX_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.794: Simplified automaton for the MIN_MAX_SUMMIT constraint obtained by applying decoration Table 3.39 to the seed transducer of the SUMMIT pattern where default is $+\infty$ (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$); $-R_i + R_{i-1} \geq 0$ is a linear invariant.

Table 4.148: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the simplified automaton of the MIN_MAX_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### Origin
Based on the ZIGZAG pattern.

### Constraint
\[
\text{MIN\_MAX\_ZIGZAG}(\text{VALUE, VARIABLES})
\]

### Arguments
- **VALUE**: dvar
- **VARIABLES**: collection(var\(\rightarrow\)dvar)

### Restrictions
- \(sv \leq 3 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty\)
- \(\text{VALUE} \geq \minv + 1\)
- \(\text{VALUE} = +\infty \lor \text{VALUE} \leq \maxv\)
- \(\text{VALUE} \geq 1\)

**Restrictions**

**where**
- \(\minv = \text{minval}(\text{VARIABLES}\cdot\text{var})\)
- \(\maxv = \text{maxval}(\text{VARIABLES}\cdot\text{var})\)
- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range}(\text{VARIABLES}\cdot\text{var})\)

### Purpose
**VALUE** is the minimum of all maximum values in each occurrence of the ZIGZAG pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value \(+\infty\).

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^*(< | <>)(<>)^*(> | <>))\).

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i + 1\) to index \(j\).

### Example
\[(3, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\]

Figure 4.795 provides an example where the **MIN\_MAX\_ZIGZAG** constraint holds.

### Typical
- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}\cdot\text{var}) > 1\)

### Symmetry
Items of **VARIABLES** can be reversed.

### Arg. properties
Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.795: Illustrating the MIN_MAX_ZIGZAG constraint of the Example slot
Automaton

Figures 4.796 and 4.797 respectively depict the automaton associated with the constraint MIN_MAX_ZIGZAG and its simplified form.
Figure 4.796: Automaton for the \textsc{MIN\_\textsc{MAX}\_\textsc{ZIGZAG}} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsc{ZIGZAG} pattern where default is $+\infty$; (1) missing transitions from $a$, $b$, $c$, $d$, $e$, $f$ to $s$ are labelled by $=$; (2) on transitions from $b$, $c$, $e$, $f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c$, $f$ to $s$ the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.
Figure 4.797: Simplified automaton for the MIN\_MAX\_ZIGZAG constraint obtained by applying decoration Table 3.24 to the seed transducer of the ZIGZAG pattern where default is $+\infty$; missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.; $-R_{i} + R_{i-1} \geq 0$ is a linear invariant.
Table 4.149: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the MIN_MAX_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.150: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the simplified automaton of the MIN_MAX_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_MIN_BUMP_ON_DECREASINGSEQUENCE

DESCRIPTION AUTOMATON

Origin Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

Constraint

\[
\text{MIN_MIN_BUMP_ON_DECREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

Arguments

- \text{VALUE} : \text{dvar}
- \text{VARIABLES} : \text{collection(\text{var}−\text{dvar})}

Restrictions

\[
\begin{align*}
\text{sv} & \leq 5 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{minV} \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \text{maxV} - 2 \\
\text{required}(\text{VARIABLES}..\text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{minV} & = \text{minval}(\text{VARIABLES}..\text{var}) \\
\text{maxV} & = \text{maxval}(\text{VARIABLES}..\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}..\text{var})
\end{align*}
\]

Purpose

\text{VALUE} is the minimum of all minimum values in each occurrence of the BUMP_ON_DECREASINGSEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \text{VALUE} takes the default value \(+\infty\).

An occurrence of the pattern BUMP_ON_DECREASINGSEQUENCE is the subsequence which matches the regular expression ‘\(\text{>}\text{<}\text{>}\text{>}\)’.

Assume that the occurrence of the pattern BUMP_ON_DECREASINGSEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 2\) to index \(j\).

Example

\[(2, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3])\]

Figure 4.798 provides an example where the MIN_MIN_BUMP_ON_DECREASINGSEQUENCE (2, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3]) constraint holds.

Typical

\[
|\text{VARIABLES}| > 5 \\
\text{range}(\text{VARIABLES}..\text{var}) > 2
\]

Arg. properties

Functional dependency: \text{VALUE} determined by \text{VARIABLES}. 
Figure 4.798: Illustrating the \texttt{MIN\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Automaton

Figures 4.799 and 4.800 respectively depict the automaton associated with the constraint MIN_MIN_BUMP_ON_DECREASING_SEQUENCE and its simplified form.

Figure 4.799: Automaton for the MIN_MIN_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $+\infty$. 
Figure 4.800: Simplified automaton for the \textit{MIN\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.26 to the seed transducer of the \textit{BUMP\_ON\_DECREASING\_SEQUENCE} pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
**MIN_MIN_DECREASING**

**DESCRIPTION**

**Origin**
Based on the `DECREASING` pattern.

**Constraint**

\[ \text{MIN}_\text{MIN}_\text{DECREASING}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var−dvar)}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 \vee \text{rv} \leq 1 & \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq \text{minv} & \\
\text{VALUE} = +\infty \vee \text{VALUE} \leq \text{maxv} - 1 & \Rightarrow \text{required}(\text{VARIABLES}, \text{var}) \\
\text{where} & \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the `DECREASING` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern `DECREASING` is the subsequence which matches the regular expression ‘\(>\)’.

Assume that the occurrence of the pattern `DECREASING` starts at position \(i\) and ends at position \(j\). The feature `MIN` computes the minimum of the values from index \(i\) to index \(j+1\).

**Example**

\[(1, (3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 4.801 provides an example where the `MIN_MIN_DECREASING` constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\end{align*}
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.801: Illustrating the MIN_MIN_DECREASING constraint of the Example slot
Figures 4.802 and 4.803 respectively depict the automaton associated with the constraint MIN_MIN_DECREASING and its simplified form.

\[
\begin{align*}
\{ C \leftarrow \text{default} \\
D & \leftarrow +\infty \\
R & \leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow +\infty \\
R & \leftarrow \min(R, \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1})) \}
\end{align*}
\]

Figure 4.802: Automaton for the MIN_MIN_DECREASING constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING pattern where default is $+\infty$.

\[
\begin{align*}
\{ R \leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
\{ R \leftarrow \min(R, \min(\text{VAR}_i, \text{VAR}_{i+1})) \}
\end{align*}
\]

Figure 4.803: Simplified automaton for the MIN_MIN_DECREASING constraint obtained by applying decoration Table 3.40 to the seed transducer of the DECREASING pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

Table 4.151: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the MIN_MIN_DECREASING constraint defined as the composition of the DECREASING pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.152: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the simplified automaton of the \texttt{MIN\_MIN\_DECREASING} constraint defined as the composition of the \texttt{DECREASING} pattern, the feature \texttt{MIN}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin: Based on the **DECREASING_SEQUENCE** pattern.

**Constraint**

\[
\text{MIN\_MIN\_DECREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{minv} \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \text{maxv} - 1 \\
\end{align*}
\]

\[
\text{where} \\
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{sv} = |\text{VARIABLES}| \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var})
\]

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the **DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $+\infty$.

An occurrence of the pattern **DECREASING_SEQUENCE** is the maximal subsequence which matches the regular expression ‘$>$ (>|=)’ $>$ |’.

Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position $i$ and ends at position $j$. The feature **MIN** computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

\[(1, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 4.804 provides an example where the **MIN\_MIN\_DECREASING\_SEQUENCE** constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.804: Illustrating the MIN_MIN_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.805 and 4.806 respectively depict the automaton associated with the constraint MIN_MIN_DECREASING_SEQUENCE and its simplified form.

Figure 4.805: Automaton for the MIN_MIN_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASINGSEQUENCE pattern where default is $+\infty$.

Figure 4.806: Simplified automaton for the MIN_MIN_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.40 to the seed transducer of the DECREASINGSEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.153: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the `MIN_MIN_DECREASING_SEQUENCE` constraint defined as the composition of the `DECREASING_SEQUENCE` pattern, the feature `MIN`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\[
\begin{array}{c|cc}
  & s & t \\
\hline
s & \min(\overrightarrow{C}, \overrightarrow{C}) & \min(\overrightarrow{C}, \overrightarrow{C}) \\
t & \min(\overrightarrow{C}, \overrightarrow{C}) & \min(\overrightarrow{C}, \overrightarrow{C}, \overleftarrow{D}, \overleftarrow{D})
\end{array}
\]

Table 4.154: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the simplified automaton of the `MIN_MIN_DECREASING_SEQUENCE` constraint defined as the composition of the `DECREASING_SEQUENCE` pattern, the feature `MIN`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\[
\begin{array}{c|cc}
  & s & t \\
\hline
s & +\infty & +\infty \\
t & +\infty & +\infty
\end{array}
\]
The feature **MIN_MIN_DIP_ON_INCREASING_SEQUENCE** is defined as follows:

**Origin**
Based on the **DIP_ON_INCREASING_SEQUENCE** pattern.

**Constraint**

\[
\text{MIN_MIN_DIP_ON_INCREASING_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 5 \lor rv \leq 2 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{minv} \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \text{maxv} - 2 \\
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the **DIP_ON_INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

An occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** is the subsequence which matches the regular expression ‘<<><’.

Assume that the occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 2\) to index \(j\).

**Example**

\[ (1, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4)) \]

Figure 4.807 provides an example where the **MIN_MIN_DIP_ON_INCREASING_SEQUENCE** constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 5 \\
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.807: Illustrating the MIN_MIN_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot.
Automaton

Figures 4.808 and 4.809 respectively depict the automaton associated with the constraint \texttt{MIN\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE} and its simplified form.

Figure 4.808: Automaton for the \texttt{MIN\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern where \texttt{default} is $+\infty$
Figure 4.809: Simplified automaton for the MIN_MIN_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.28 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
**MIN_MIN_GORGE**

**Origin**  
Based on the GORGE pattern.

**Constraint**  
MIN_MIN_GORGE(VALUE, VARIABLES)

**Arguments**  
VALUE : dvar  
VARIABLES : collection(var−dvar)

**Restrictions**  
\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq minv \\
VALUE = +\infty \lor VALUE \leq maxv - 1 \\
required(VARIABLES, var)
\]

where  
\[
minv = minval(VARIABLES, var) \\
maxv = maxval(VARIABLES, var) \\
sv = |VARIABLES| \\
rv = range(VARIABLES, var)
\]

**Purpose**  
VALUE is the minimum of all minimum values in each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \'(>|=(>|>)*)(<|=(<|>)<)\'. Assume that the occurrence of the pattern GORGE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

**Example**  
\[
(3, (1, 7, 3, 4, 5, 5, 4, 2, 6, 5, 4, 6, 5, 7))
\]

Figure 4.810 provides an example where the MIN_MIN_GORGE \((3,[1,7,3,4,5,5,4,2,6,5,4,6,5,7])\) constraint holds.

**Typical**  
\[|VARIABLES| > 2 \]
\[range(VARIABLES, var) > 1\]

**Symmetry**  
Items of VARIABLES can be reversed.

**Arg. properties**  
Functional dependency: VALUE determined by VARIABLES.
Figure 4.810: Illustrating the MIN_MIN_GORGE constraint of the Example slot
Automaton

Figures 4.811 and 4.812 respectively depict the automaton associated with the constraint MIN_MIN_GORGE and its simplified form.

Figure 4.811: Automaton for the MIN_MIN_GORGE constraint obtained by applying decoration Table 3.37 to the seed transducer of the GORGE pattern where default is $+\infty$ (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
Figure 4.812: Simplified automaton for the MIN_MIN_GORGE constraint obtained by applying decoration Table 3.39 to the seed transducer of the GORGE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

Table 4.155: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the MIN_MIN_GORGE constraint defined as the composition of the GORGE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.156: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the simplified automaton of the MIN\_MIN\_GORGE constraint defined as the composition of the GORGE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_MIN_INCREASING

**DESCRIPTION**

**Origin**
Based on the INCREASING pattern.

**Constraint**
MIN_MIN_INCREASING(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var-dvar)

**Restrictions**
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ minv ⇒ VALUE = +∞ ∨ VALUE ≤ maxv − 1
required(VARIABLES.var)

where
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

VALUE is the minimum of all minimum values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

**Purpose**
An occurrence of the pattern INCREASING is the subsequence which matches the regular expression '<'.
Assume that the occurrence of the pattern INCREASING starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

**Example**
(1, ⟨4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3⟩)

Figure 4.813 provides an example where the MIN_MIN_INCREASING (1, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

**Typical**
|VARIABLES| > 1
range(VARIABLES.var) > 1

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.813: Illustrating the MIN_MIN_INCREASING constraint of the Example slot
Automaton

Figures 4.814 and 4.815 respectively depict the automaton associated with the constraint MIN_MIN_INCREASING and its simplified form.

\[
\begin{align*}
C & \leftarrow \text{default} \\
D & \leftarrow +\infty \\
R & \leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
D & \leftarrow +\infty \\
R & \leftarrow \min(R, \min(D, \text{VAR}_i, \text{VAR}_{i+1}))
\end{align*}
\]

Figure 4.814: Automaton for the MIN_MIN_INCREASING constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING pattern where default is +\infty

\[
\begin{align*}
\{R & \leftarrow \text{default}\} \\
\{R & \leftarrow \min(R, \text{VAR}_i)\}
\end{align*}
\]

\[
\begin{align*}
\geq \quad \{R & \leftarrow \min(R, \text{VAR}_i)\}
\end{align*}
\]

Figure 4.815: Simplified automaton for the MIN_MIN_INCREASING constraint obtained by applying decoration Table 3.39 to the seed transducer of the INCREASING pattern where default is +\infty; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

Table 4.157: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the MIN_MIN_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.158: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the simplified automaton of the MIN_MIN_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MIN_MIN_INCREASING_SEQUENCE**

**Origin**
Based on the `INCREASING_SEQUENCE` pattern.

**Constraint**

\[
\text{MIN_MIN_INCREASING_SEQUENCE} \left( \text{VALUE}, \text{VARIABLES} \right)
\]

**Arguments**

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var−dvar)`

**Restrictions**

\[
\begin{align*}
sv \leq 1 \vee rv \leq 1 & \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq \text{minv} & \\
\text{VALUE} = +\infty \vee \text{VALUE} \leq \text{maxv} - 1 & \Rightarrow \text{required} \left( \text{VARIABLES}, \text{var} \right)
\end{align*}
\]

where

- `minv = minval \left( \text{VARIABLES}.\text{var} \right)`
- `maxv = maxval \left( \text{VARIABLES}.\text{var} \right)`
- `sv = \left| \text{VARIABLES} \right|`
- `rv = \text{range} \left( \text{VARIABLES}.\text{var} \right)`

**Purpose**

`VALUE` is the minimum of all minimum values in each occurrence of the `INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value `+\infty`.

An occurrence of the pattern `INCREASINGSEQUENCE` is the maximal subsequence which matches the regular expression ‘\(< (< | =)*) < | <\)’.

Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j+1`.

**Example**

\[(1, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\]

Figure 4.816 provides an example where the `MIN_MIN_INCREASING_SEQUENCE` \((1, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

- `\left| \text{VARIABLES} \right| > 1`
- `\text{range} \left( \text{VARIABLES}.\text{var} \right) > 1`

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 4.816: Illustrating the MIN_MIN_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.817 and 4.818 respectively depict the automaton associated with the constraint MIN_MIN_INCREASING_SEQUENCE and its simplified form.

Figure 4.817: Automaton for the MIN_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $+\infty$.

Figure 4.818: Simplified automaton for the MIN_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.39 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.159: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the `MIN_MIN_INCREMENTING_SEQUENCE` constraint defined as the composition of the `INCREMENTING_SEQUENCE` pattern, the feature `MIN`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\[
\begin{array}{c|c|c}
{s} & {s} & {t} \\
\hline
{s} & \min(\overline{C}, \overline{C}) & \min(\overline{C}, \overline{C}) \\
{t} & \min(\overline{C}, \overline{C}) & \min(\overline{C}, \overline{C}, \overline{D}, \overline{D})
\end{array}
\]

Table 4.160: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the simplified automaton of the `MIN_MIN_INCREMENTING_SEQUENCE` constraint defined as the composition of the `INCREMENTING_SEQUENCE` pattern, the feature `MIN`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\[
\begin{array}{c|c|c}
{s} & {s} & {t} \\
\hline
{s} & +\infty & +\infty \\
{t} & +\infty & +\infty
\end{array}
\]
**AGGREGATOR**  FEATURE  PATTERN

**MIN_MIN_INFLEXION**

**DESCRIPTION**  AUTOMATON

**Origin**  Based on the INFLEXION pattern.

**Constraint**  MIN_MIN_INFLEXION(VALUE, VARIABLES)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE & \geq minv \\
VALUE & = +\infty \lor VALUE \leq maxv \\
required & (VARIABLES, var) \\
\text{where} & \\
\text{minv} & = \text{minval}(VARIABLES.var) \\
\text{maxv} & = \text{maxval}(VARIABLES.var) \\
sv & = |VARIABLES| \\
rv & = \text{range}(VARIABLES.var)
\end{align*}
\]

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \'< (< | =)'> | > (>| =)'+'<'.

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(1, 1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4)\]

Figure 4.819 provides an example where the MIN_MIN_INFLEXION \((1, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])\) constraint holds.

**Typical**

- \(|VARIABLES| > 2\)
- \(\text{range}(VARIABLES.var) > 1\)

**Symmetry**  Items of VARIABLES can be reversed.

**Arg. properties**  Functional dependency: VALUE determined by VARIABLES.
Figure 4.819: Illustrating the MIN_MIN_INFLEXION constraint of the Example slot
Figures 4.820 and 4.821 respectively depict the automaton associated with the constraint MIN_MIN_INFLEXION and its simplified form.

Figure 4.820: Automaton for the MIN_MIN_INFLEXION constraint obtained by applying decoration Table 3.37 to the seed transducer of the INFLEXION pattern where default is $+\infty$ (transition $r \to t$ has the same registers updates as transition $t \to r$).

Figure 4.821: Simplified automaton for the MIN_MIN_INFLEXION constraint obtained by applying decoration Table 3.26 to the seed transducer of the INFLEXION pattern where default is $+\infty$ (transition $r \to t$ has the same registers updates as transition $t \to r$); $-R_i + R_{i-1} \geq 0$ is a linear invariant.
MIN_MIN_INFLEXION

1729
**MIN_MIN.StrictlyDecreasingSequence**

**Description**

Based on the **StrictlyDecreasingSequence** pattern.

**Constraint**

```
MIN_MIN.StrictlyDecreasingSequence(VALUE, VARIABLES)
```

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

```
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ minv
VALUE = +∞ ∨ VALUE ≤ maxv − 1
required(VARIABLES, var)
```

where

- minv = minval(VARIABLES.var)
- maxv = maxval(VARIABLES.var)
- sv = |VARIABLES|
- rv = range(VARIABLES.var)

**Purpose**

**VALUE** is the minimum of all minimum values in each occurrence of the **StrictlyDecreasingSequence** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value +∞.

An occurrence of the pattern **StrictlyDecreasingSequence** is the maximal sub-sequence which matches the regular expression '>>'.

Assume that the occurrence of the pattern **StrictlyDecreasingSequence** starts at position **i** and ends at position **j**. The feature **MIN** computes the minimum of the values from index **i** to index **j + 1**.

**Example**

```
(1, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3])
```

Figure 4.822 provides an example where the **MIN_MIN.StrictlyDecreasingSequence** (1, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3]) constraint holds.

**Typical**

- |VARIABLES| > 1
- range(VARIABLES.var) > 1

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.822: Illustrating the **MIN_MIN_STRICTLY_DECREASING_SEQUENCE** constraint of the **Example** slot
Automaton

Figures 4.823 and 4.824 respectively depict the automaton associated with the constraint \texttt{MIN\_MIN\_STRICTLY\_DECREASING\_SEQUENCE} and its simplified form.

\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow \text{default}
\end{align*}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure4_823.png}
\caption{Automaton for the \texttt{MIN\_MIN\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is $+\infty$}
\end{figure}

\begin{align*}
\{R &\leftarrow \text{default}\}
\end{align*}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure4_824.png}
\caption{Simplified automaton for the \texttt{MIN\_MIN\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.40 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.}
\end{figure}
### Table 4.161: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the MIN_MIN_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D})^M$</td>
</tr>
</tbody>
</table>

### Table 4.162: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the MIN_MIN_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>r</td>
<td>$\infty$</td>
<td>$\infty^M$</td>
</tr>
</tbody>
</table>
**MIN_MIN STRICTLY_INCREASING_SEQUENCE**

**Origin**
Based on the **STRICTLY_INCREASING_SEQUENCE** pattern.

**Constraint**

$$\text{MIN_MIN STRICTLY_INCREASING_SEQUENCE}(\text{VALUE, VARIABLES})$$

**Arguments**

<table>
<thead>
<tr>
<th>VALUE</th>
<th>dvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

$$\text{sv} \leq 1 \vee \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty$$

$$\text{VALUE} \geq \text{minv}$$

$$\text{VALUE} = +\infty \vee \text{VALUE} \leq \text{maxv} - 1$$

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the **STRICTLY_INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $$+\infty$$.

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the **maximal** sub-sequence which matches the regular expression ‘<+’.

Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position $$i$$ and ends at position $$j$$. The feature **MIN** computes the minimum of the values from index $$i$$ to index $$j + 1$$.

**Example**

$$(1, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))$$

Figure 4.825 provides an example where the **MIN_MIN STRICTLY_INCREASING_SEQUENCE** constraint holds.

**Typical**

$$|\text{VARIABLES}| > 1$$

$$\text{range}(\text{VARIABLES}.\text{var}) > 1$$

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.825: Illustrating the MIN_MIN_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot.
Automaton

Figures 4.826 and 4.827 respectively depict the automaton associated with the constraint MIN_MIN_STRICTLY_INCREASING_SEQUENCE and its simplified form.

Figure 4.826: Automaton for the MIN_MIN_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICTLY_INCREASINGSEQUENCE pattern where default is $+\infty$.

Figure 4.827: Simplified automaton for the MIN_MIN_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.39 to the seed transducer of the STRICTLY_INCREASINGSEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.163: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the MIN_MIN_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>min((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>min((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>min((\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}))</td>
</tr>
</tbody>
</table>

Table 4.164: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the simplified automaton of the MIN_MIN_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>+(\infty)</td>
<td>+(\infty)</td>
</tr>
<tr>
<td>r</td>
<td>+(\infty)</td>
<td>+(\infty)</td>
</tr>
</tbody>
</table>
**MIN_MIN_VALLEY**

**Description**

Based on the `VALLEY` pattern.

**Constraint**

`MIN_MIN_VALLEY(VALUE, VARIABLES)`

**Arguments**

- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`

**Restrictions**

- `sv ≤ 2` or `rv ≤ 1` ⇒ `VALUE = +∞`
- `VALUE ≥ minv`
- `VALUE = +∞` or `VALUE ≤ maxv − 1`

where
- `minv = minval(VARIABLES.var)`
- `maxv = maxval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the `VALLEY` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value `+∞`.

An occurrence of the pattern `VALLEY` is the maximal subsequence which matches the regular expression `'> (="|>|")* (<| |=)* '<`.

Assume that the occurrence of the pattern `VALLEY` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i + 1` to index `j`.

**Example**

`(2, (1, 3, 7, 4, 3, 6, 6, 6, 6, 3, 3, 2, 6, 5, 5, 5, 7))`

Figure 4.828 provides an example where the `MIN_MIN_VALLEY` `(2, [1, 3, 7, 4, 3, 6, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7])` constraint holds.

**Typical**

- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`

**Symmetry**

Items of `VARIABLES` can be reversed.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 4.828: Illustrating the MIN_MIN_VALLEY constraint of the Example slot
Automaton

Figures 4.829 and 4.830 respectively depict the automaton associated with the constraint MIN_MIN_VALLEY and its simplified form.

![Automaton Diagram]

Figure 4.829: Automaton for the MIN_MIN_VALLEY constraint obtained by applying decoration Table 3.37 to the seed transducer of the VALLEY pattern where default is +∞.

![Simplified Automaton Diagram]

Figure 4.830: Simplified automaton for the MIN_MIN_VALLEY constraint obtained by applying decoration Table 3.39 to the seed transducer of the VALLEY pattern where default is +∞; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
Table 4.165: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the MIN_MIN_VALLEY constraint defined as the composition of the VALLEY pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}))</td>
</tr>
<tr>
<td>t</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
</tbody>
</table>

Table 4.166: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the simplified automaton of the MIN_MIN_VALLEY constraint defined as the composition of the VALLEY pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>+(\infty)</td>
<td>+(\infty)</td>
<td>+(\infty)</td>
</tr>
<tr>
<td>r</td>
<td>+(\infty)</td>
<td>(\text{VAR}_{i+1})</td>
<td>+(\infty)</td>
</tr>
<tr>
<td>t</td>
<td>+(\infty)</td>
<td>+(\infty)</td>
<td>+(\infty)</td>
</tr>
</tbody>
</table>
Based on the ZIGZAG pattern.

**Constraint**

\[ \text{MIN}_\text{MIN}_\text{ZIGZAG}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection(\text{var}-\text{dvar})} \)

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{minv} \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \text{maxv} - 1 \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

- \( \text{minv} = \min\text{val}(\text{VARIABLES}, \text{var}) \)
- \( \text{maxv} = \max\text{val}(\text{VARIABLES}, \text{var}) \)
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{rv} = \text{range}(\text{VARIABLES}, \text{var}) \)

**Purpose**

\( \text{VALUE} \) is the minimum of all minimum values in each occurrence of the ZIGZAG pattern in the time-series given by the \( \text{VARIABLES} \) collection. If the pattern does not occur, \( \text{VALUE} \) takes the default value \( +\infty \).

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘\((<>)^+<(<>|<>)|(<>|>><)\)’. Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature \( \text{MIN} \) computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**

\[(1, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\]

Figure 4.831 provides an example where the \( \text{MIN}_\text{MIN}_\text{ZIGZAG} \) (1, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1]) constraint holds.

**Typical**

- \( |\text{VARIABLES}| > 3 \)
- \( \text{range}(\text{VARIABLES}, \text{var}) > 1 \)

**Symmetry**

Items of \( \text{VARIABLES} \) can be reversed.

**Arg. properties**

Functional dependency: \( \text{VALUE} \) determined by \( \text{VARIABLES} \).
Figure 4.831: Illustrating the MIN_MIN_ZIGZAG constraint of the Example slot
Automaton

Figures 4.832 and 4.833 respectively depict the automaton associated with the constraint MIN_MIN_ZIGZAG and its simplified form.
Figure 4.832: Automaton for the MIN_MIN_ZIGZAG constraint obtained by applying decoration Table 3.37 to the seed transducer of the ZIGZAG pattern where default is $+\infty$: (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.
Figure 4.833: Simplified automaton for the MIN_MIN_ZIGZAG constraint obtained by applying decoration Table 3.29 to the seed transducer of the ZIGZAG pattern where default is $+\infty$; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.167: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the MIN_MIN_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.168: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the simplified automaton of the \textsc{MIN\_MIN\_ZIGZAG} constraint defined as the composition of the \textsc{ZIGZAG} pattern, the feature \textsc{MIN}, and the aggregator \textsc{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_RANGE_DECENDING

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>[Diagram]</td>
</tr>
</tbody>
</table>

**Origin**
Based on the **DECREASING** pattern.

**Constraint**

\[ \text{MIN\_RANGE\_DECENDING} (\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>\text{dvar}</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>\text{collection(var-dvar)}</td>
</tr>
</tbody>
</table>

**Restrictions**

\[ \text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \]
\[ \text{VALUE} \geq 1 \]
\[ \text{required} (\text{VARIABLES}, \text{var}) \]

where

\[ \text{sv} = |\text{VARIABLES}| \]
\[ \text{rv} = \text{range} (\text{VARIABLES.var}) \]

**Purpose**

\text{VALUE} is the minimum value of the differences between the largest and smallest value in each occurrence of the **DECREASING** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, \text{VALUE} takes the default value \( +\infty \).

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression ‘>’. Assume that the occurrence of the pattern **DECREASING** starts at position \( i \) and ends at position \( j \). The feature **RANGE** computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**

\[ (1, (3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4)) \]

Figure 4.834 provides an example where the **MIN\_RANGE\_DECENDING** (1, [3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

**Typical**

<table>
<thead>
<tr>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{</td>
</tr>
<tr>
<td>\text{range} (\text{VARIABLES.var}) \times 1</td>
</tr>
</tbody>
</table>

**Symmetry**

One and the same constant can be added to the \text{var} attribute of all items of \text{VARIABLES}.

**Arg. properties**

Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
Figure 4.834: Illustrating the MIN_RANGE_DECREASING constraint of the Example slot
Figures 4.835 and 4.836 respectively depict the automaton associated with the constraint MIN_RANGE_DECREASING and its simplified form.

**Figure 4.835: Automaton for the MIN_RANGE_DECREASING constraint obtained by applying decoration Table 3.48 to the seed transducer of the DECREASING pattern where default is +∞**

**Figure 4.836: Simplified automaton for the MIN_RANGE_DECREASING constraint obtained by applying decoration Table 3.46 to the seed transducer of the DECREASING pattern where default is +∞; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.**
**MIN_RANGE_DECREASING_SEQUENCE**

**DESCRIPTION**

Origin

Based on the **DECREASING_SEQUENCE** pattern.

Constraint

\[ \text{MIN\_RANGE\_DECREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES}) \]

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(\text{var}−\text{dvar})}
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} \leq 1 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq 1 \Rightarrow \\
\text{required}(\text{VARIABLES}, \text{var}) & \text{ where } \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Purpose

\[ \text{VALUE} \] is the minimum value of the differences between the largest and smallest value in each occurrence of the **DECREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, \( \text{VALUE} \) takes the default value \(+\infty\).

An occurrence of the pattern **DECREASING_SEQUENCE** is the maximal subsequence which matches the regular expression ‘\( > (> | =) > | > \)’.

Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **RANGE** computes the range of the values from index \( i \) to index \( j + 1 \).

Example

\[
(2, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))
\]

Figure 4.837 provides an example where the **MIN\_RANGE\_DECREASING\_SEQUENCE** constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Symmetry**

One and the same constant can be added to the \text{var} attribute of all items of \text{VARIABLES}.

**Arg. properties**

**Functional dependency**: \( \text{VALUE} \) determined by \text{VARIABLES}. 

Figure 4.837: Illustrating the MIN_RANGE_DECREASING_SEQUENCE constraint of the Example slot
Figures 4.838 and 4.839 respectively depict the automaton associated with the constraint MIN_RANGE_DECREASING_SEQUENCE and its simplified form.

Figure 4.838: Automaton for the MIN_RANGE_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.48 to the seed transducer of the DECREASING_SEQUENCE pattern where default is $+\infty$. 
Figure 4.839: Simplified automaton for the \texttt{MIN\_RANGE\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.42 to the seed transducer of the \texttt{DECREASING\_SEQUENCE} pattern where \texttt{default} is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
**MIN_RANGE_INCREMENTING**

**DESCRIPTION**

**Origin**
Based on the **INCREASING** pattern.

**Constraint**
MIN_RANGE_INCREMENTING(VALUE, VARIABLES)

**Arguments**
VALUE : dvar  
VARIABLES : collection(var−dvar)

**Restrictions**
\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \]
VALUE \geq 1

required(VARIABLES, var)  
where  
\[ sv = |VARIABLES| \]
\[ rv = \text{range}(VARIABLES.var) \]

**Purpose**
VALUE is the minimum value of the differences between the largest and smallest value in each occurrence of the **INCREASING** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression ‘<’. Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature **RANGE** computes the range of the values from index \(i\) to index \(j+1\).

**Example**
\[(1, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\]

Figure 4.840 provides an example where the **MIN_RANGE_INCREMENTING** (1, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

**Typical**
\[ |VARIABLES| > 1 \]
\[ \text{range}(VARIABLES.var) > 1 \]

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.840: Illustrating the MIN_RANGE_INCREASING constraint of the Example slot
Figures 4.841 and 4.842 respectively depict the automaton associated with the constraint MIN_RANGE_INCREASING and its simplified form.

**Automaton**

Figures 4.841 and 4.842 respectively depict the automaton associated with the constraint MIN_RANGE_INCREASING and its simplified form.

\[
\begin{aligned}
&\{ C \leftarrow \text{default} \} \\
&\{ H \leftarrow \text{VAR}_i \} \\
&\{ R \leftarrow \text{default} \} \\
&\{ H \leftarrow \text{VAR}_{i+1} \} \\
&\{ R \leftarrow \min(R, |H - \text{VAR}_{i+1}|) \} \\
\end{aligned}
\]

Figure 4.841: Automaton for the MIN_RANGE_INCREASING constraint obtained by applying decoration Table 3.48 to the seed transducer of the INCREASING pattern where default is $+\infty$.

\[
\begin{aligned}
&\{ R \leftarrow \text{default} \} \\
&\{ R \leftarrow \min(R, \max(0, \text{VAR}_{i+1} - \text{VAR}_i)) \} \\
\end{aligned}
\]

Figure 4.842: Simplified automaton for the MIN_RANGE_INCREASING constraint obtained by applying decoration Table 3.47 to the seed transducer of the INCREASING pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
MIN_RANGE_INCREASING
**MIN_RANGE_INCREASING_SEQUENCE**

**Description**

**Origin**
Based on the **INCREASING_SEQUENCE** pattern.

**Constraint**

```
MIN_RANGE_INCREASING_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

```
sv ≤ 1 ∧ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ 1⇒
required(VARIABLES.var)
```

where

```
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

**Purpose**

VALUE is the minimum value of the differences between the largest and smallest value in each occurrence of the **INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

An occurrence of the pattern **INCREASING_SEQUENCE** is the maximal subsequence which matches the regular expression ‘\(< (< | =)\)* < | <\)’. Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **RANGE** computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\((2, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 3, 1, 3, 3))\)

Figure 4.843 provides an example where the **MIN_RANGE_INCREASING_SEQUENCE** constraint holds.

**Typical**

- \(|VARIABLES| > 1\)
- \(range(VARIABLES.var) > 1\)

**Symmetry**

One and the same constant can be added to the **var** attribute of all items of **VARIABLES**.

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.843: Illustrating the MIN_RANGE_INCREASING_SEQUENCE constraint of the Example slot
Automaton  

Figures 4.844 and 4.845 respectively depict the automaton associated with the constraint MIN_RANGE_INCREASING_SEQUENCE and its simplified form.

Figure 4.844: Automaton for the MIN_RANGE_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.48 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $+\infty$. 

\[\begin{align*}
C &\leftarrow \text{default} \\
H &\leftarrow \text{VAR}_i \\
R &\leftarrow \text{default} \\
\end{align*}\]
Figure 4.845: Simplified automaton for the **MIN\_RANGE\_INCREASING\_SEQUENCE** constraint obtained by applying decoration Table 3.43 to the seed transducer of the **INCREASING\_SEQUENCE** pattern where **default** is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
MIN_RANGE_STRICTLY_DECREASING_SEQUENCE

EXPLANATION

Origin
Based on the STRICLY_DECREASING_SEQUENCE pattern.

Constraint
MIN_RANGE_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var – dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ 1 ∧
required(VARIABLES, var)
where
sv = |VARIABLES|
rv = range(VARIABLES.var)

VALUE is the minimum value of the differences between the largest and smallest value in each occurrence of the STRICLY_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

Purpose
An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression ‘>’.
Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at position i and ends at position j. The feature RANGE computes the range of the values from index i to index j + 1.

Example
(1, ⟨4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 4, 3⟩)

Figure 4.846 provides an example where the MIN_RANGE_STRICTLY_DECREASING_SEQUENCE (1, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3]) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES,var) > 1

Symmetry
One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.846: Illustrating the MIN_RANGE.Strictly.DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.847 and 4.848 respectively depict the automaton associated with the constraint MIN_RANGE_STRICTLY_DECREASING_SEQUENCE and its simplified form.

Figure 4.847: Automaton for the MIN_RANGE_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.48 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $+\infty$. 
Figure 4.848: Simplified automaton for the **MIN\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE** constraint obtained by applying decoration Table 3.42 to the seed transducer of the **STRICTLY\_DECREASING\_SEQUENCE** pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
**MIN_RANGE.Strictly.Increasing.Sequence**

**Description**
Based on the `Strictly.Increasing.Sequence` pattern.

**Constraint**

\[ \text{MIN\_RANGE\_Strictly\_Increasing\_Sequence}(\text{VALUE, VARIABLES}) \]

**Arguments**
- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \)

**Restrictions**

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \]
\[ \text{VALUE} \geq 1 \]

\( sv = |\text{VARIABLES}| \)
\( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**
An occurrence of the pattern `Strictly.Increasing.Sequence` is the **maximal** sub-sequence which matches the regular expression `'<+'`. Assume that the occurrence of the pattern `Strictly.Increasing.Sequence` starts at position \( i \) and ends at position \( j \). The feature `RANGE` computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**

\( (2, 4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3) ) \)

Figure 4.849 provides an example where the `MIN\_RANGE\_Strictly\_Increasing\_Sequence` (\( 2, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3] \)) constraint holds.

**Typical**

\[ |\text{VARIABLES}| \geq 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) \geq 1 \]

**Symmetry**
One and the same constant can be added to the \( \text{var} \) attribute of all items of \( \text{VARIABLES} \).

**Arg. properties**
- **Functional dependency**: \( \text{VALUE} \) determined by \( \text{VARIABLES} \).
Figure 4.849: Illustrating the \texttt{MIN\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Figures 4.850 and 4.851 respectively depict the automaton associated with the constraint `MIN_RANGE_STRICTLY_INCREASING_SEQUENCE` and its simplified form.

![Automaton Diagram]

Figure 4.850: Automaton for the `MIN_RANGE_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.48 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where `default` is $+\infty$
Figure 4.851: Simplified automaton for the \texttt{MIN\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.43 to the seed transducer of the \texttt{STRICTLY\_INCREASING\_SEQUENCE} pattern where \texttt{default} is $\pm \infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
**MIN\_STRICTLY\_DECREASING\_SEQUENCE**

**Description**

Based on the `STRICTLY\_DECREASING\_SEQUENCE` pattern.

**Constraint**

`MIN\_STRICTLY\_DECREASING\_SEQUENCE(VARIABLES, FEATURES, DEFAULT)`

**Arguments**

- `VARIABLES` : `collection(var\_dvar)`
- `FEATURES` : `collection(var\_dvar)`
- `DEFAULT` : `int`

**Restrictions**

\[
\begin{align*}
\text{required}(\text{VARIABLES}.\text{var}) \\
\text{required}(\text{FEATURES}.\text{var}) \\
sv &= |\text{FEATURES}| \\
sv &\leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{FEATURES}.\text{var} = \text{DEFAULT} \\
\text{FEATURES}.\text{var} &= \text{DEFAULT} \lor \text{FEATURES}.\text{var} \geq \text{minv} \\
\text{FEATURES}.\text{var} &= \text{DEFAULT} \lor \text{FEATURES}.\text{var} \leq \text{maxv} - 1 \\
\text{DEFAULT} &< \text{minv} \lor \text{DEFAULT} > \text{maxv} - 1 \\
\text{where} & \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Consider the time-series given by the `VARIABLES` collection. If position \(i\) does not correspond to the first position in `VARIABLES` where an occurrence of `STRICTLY\_DECREASING\_SEQUENCE` is identified (even if this occurrence of pattern is not complete) then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `STRICTLY\_DECREASING\_SEQUENCE`.

**Purpose**

An occurrence of the pattern `STRICTLY\_DECREASING\_SEQUENCE` is the maximal sub-sequence which matches the regular expression `&gt;+`. Assume that the occurrence of the pattern `STRICTLY\_DECREASING\_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `MIN` computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

Figure 4.852 provides an example where the `MIN\_STRICTLY\_DECREASING\_SEQUENCE ([4, 4, 6, 4, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 3, 0], 0)` constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.852: Illustrating the MIN_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot

Arg. properties  
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
MIN_STRICTLY_DECREASING_SEQUENCE
1777
MIN\_STRICTLY\_INCREASING\_SEQUENCE

<table>
<thead>
<tr>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIN_STRICTLY_INCREASING_SEQUENCE</td>
<td></td>
</tr>
</tbody>
</table>

DESCRIPTION

- **Origin**: Based on the `STRICTLY\_INCREASING\_SEQUENCE` pattern.
- **Constraint**: `MIN\_STRICTLY\_INCREASING\_SEQUENCE(VARIABLES, FEATURES, DEFAULT)`
- **Arguments**:
  - `VARIABLES`: `collection(var\_dvar)`
  - `FEATURES`: `collection(var\_dvar)`
  - `DEFAULT`: `int`
- **Restrictions**:
  - `required(VARIABLES.var)`
  - `required(FEATURES.var)`
  - `sv = |FEATURES|`
  - `sv ≤ 1 ∨ rv ≤ 1 ⇒ FEATURES.var = DEFAULT`
  - `FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv`
  - `FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv - 1`
  - `DEFAULT < minv ∨ DEFAULT > maxv - 1`
  - Where
    - `maxv = maxval(VARIABLES.var)`
    - `minv = minval(VARIABLES.var)`
    - `sv = |VARIABLES|`
    - `rv = range(VARIABLES.var)`

**Purpose**

Consider the time-series given by the `VARIABLES` collection. If position `i` does not correspond to the first position in `VARIABLES` where an occurrence of `STRICTLY\_INCREASING\_SEQUENCE` is identified (even if this occurrence of pattern is not complete) then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `STRICTLY\_INCREASING\_SEQUENCE`.

An occurrence of the pattern `STRICTLY\_INCREASING\_SEQUENCE` is the maximal subsequence which matches the regular expression `'<+.'`.

Assume that the occurrence of the pattern `STRICTLY\_INCREASING\_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

Figure 4.853 provides an example where the `MIN\_STRICTLY\_INCREASING\_SEQUENCE ([4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 3, 1, 2, 3], [0, 3, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 4.853: Illustrating the MIN_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
MIN\_STRICLY\_INCREASING\_SEQUENCE
MIN_SURF_BUMP_ON_DECREASING_SEQUENCE

DESCRIPTION

Origin
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

Constraint
Min_SURF_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 5 ∨ rv ≤ 2 ⇒ VALUE = +∞
VALUE ≥ 3 * minv + 3
VALUE = +∞ ∨ VALUE ≤ 3 * maxv - 3
among(n1,VARIABLES[3,sv−1],{maxv−2,maxv−1,maxv})
VALUE < +∞ ⇒ n1 ≥ VALUE - 3 - max(maxv - 3, 0)
among(n2,VARIABLES[3,sv−1],{minv,minv+1,minv})
VALUE < +∞ ⇒ n2 ≥ min(minv+3,0) - 3 - VALUE
required(VARIABLES,var)
where
minv = minval(VARIABLES.var)
rv = range(VARIABLES.var)
sv = |VARIABLES|
maxv = maxval(VARIABLES.var)

VALUE is the minimal surface of occurrences of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

Purpose
An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression ‘>>>’.
Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 2 to index j.

Example
(11, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3])

Figure 4.854 provides an example where the MIN_SURF_BUMP_ON_DECREASING_SEQUENCE (11, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3]) constraint holds.

Typical
|VARIABLES| > 5
range(VARIABLES.var) > 2
Figure 4.854: Illustrating the MIN_SURF_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.855 and 4.856 respectively depict the automaton associated with the constraint MIN_SURF_BUMP_ON_DECREASING_SEQUENCE and its simplified form.

Figure 4.855: Automaton for the MIN_SURF_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $+\infty$.
Figure 4.856: Simplified automaton for the MIN_SURF_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.26 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
**Origin**

Based on the **DECREASING** pattern.

**Constraint**

\[ \text{MIN\_SURF\_DECREASING}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

\[
\begin{align*}
\text{VALUE} &: \ dvar \\
\text{VARIABLES} &: \ \text{collection}(\text{var}-\text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq 2 \cdot \minv + 1 & \\
\text{VALUE} = +\infty \lor \text{VALUE} \leq 2 \cdot \maxv - 1 & \Rightarrow \text{required}(\text{VARIABLES}.\text{var}) \\
\end{align*}
\]

where

\[
\begin{align*}
\minv &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\maxv &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression ‘>’.

Assume that the occurrence of the pattern **DECREASING** starts at position \( i \) and ends at position \( j \). The feature **SURF** computes the sum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[(4,\langle 3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4 \rangle)\]

Figure 4.857 provides an example where the **MIN\_SURF\_DECREASING** (4, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.857: Illustrating the MIN_SURF_DECREASING constraint of the Example slot
Automaton

Figures 4.858 and 4.859 respectively depict the automaton associated with the constraint MIN_SURF_DECREASING and its simplified form.

\[
\begin{align*}
&\{ C \leftarrow \text{default} \\
&D \leftarrow 0 \\
&R \leftarrow \text{default} \}
\end{align*}
\]

Figure 4.858: Automaton for the MIN_SURF_DECREASING constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING pattern where default is \(+\infty\).

\[
\begin{align*}
&\{ R \leftarrow \text{default} \}
\end{align*}
\]

Figure 4.859: Simplified automaton for the MIN_SURF_DECREASING constraint obtained by applying decoration Table 3.40 to the seed transducer of the DECREASING pattern where default is \(+\infty\); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

Table 4.169: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the MIN_SURF_DECREASING constraint defined as the composition of the DECREASING pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.170: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the simplified automaton of the MIN_SURF_DECREASING constraint defined as the composition of the DECREASING pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_SURF_DECREASING_SEQUENCE

Origin
Based on the DECREASING_SEQUENCE pattern.

Constraint
MIN_SURF_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
\[ \begin{align*}
sv & \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
v & = 2 \Rightarrow VALUE \geq 2 \times \minv + 1 \\
v & \geq 3 \Rightarrow VALUE \geq \min(2 \times \minv + 1, sv \times (\minv + 1)) \geq 2 \times \maxv - 1 \\
v & = 2 \Rightarrow VALUE = +\infty \lor VALUE \leq 2 \times \maxv - 1 \\
v & \geq 3 \Rightarrow VALUE = +\infty \lor VALUE \leq \max(2 \times \maxv - 1, sv \times (\maxv - 1)) \\
\text{among}(n1, \text{VARIABLES}[1, sv], (\maxv - 1, \maxv)) \\
VALUE & < +\infty \land (rv = 2 \lor \maxv = 1) \Rightarrow \\
n1 & \geq VALUE - \max(0, 2 \times \maxv - 3) \\
VALUE & < +\infty \land rv > 2 \land \maxv > 1 \Rightarrow n1 \geq VALUE - 1 - sv \times (\maxv - 2) \\
\text{among}(n2, \text{VARIABLES}[1, sv], (\minv, \minv + 1)) \\
VALUE & < +\infty \land (rv = 2 \land \minv = -1) \Rightarrow \\
n2 & \geq \min(0, 2 \times \minv + 3) - VALUE \\
VALUE & < +\infty \land rv > 2 \land \minv < -1 \Rightarrow n2 \geq sv \times (\minv + 2) - 1 - VALUE \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{where}
\minv & = \minval(\text{VARIABLES}.\text{var}) \\
rv & = \range(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
\maxv & = \maxval(\text{VARIABLES}.\text{var})
\end{align*} \]

Purpose
\( \text{VALUE} \) is the minimal surface of occurrences of the \textit{DECREASING\_SEQUENCE} pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value \(+\infty\).

An occurrence of the pattern \textit{DECREASING\_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression ‘\( \geq > (\geq | =)^* > | \)’.

Assume that the occurrence of the pattern \textit{DECREASING\_SEQUENCE} starts at position \( i \) and ends at position \( j \). The feature \text{SURF} computes the sum of the values from index \( i \) to index \( j + 1 \).

Example
\( \langle 6, 3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4 \rangle \)
Figure 4.860 provides an example where the \textit{MIN\_SURF\_DECREASING\_SEQUENCE} constraint holds.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure.png}
\caption{Illustrating the \textit{MIN\_SURF\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot}
\end{figure}

\begin{itemize}
\item \textbf{Typical} \hfill \texttt{|VARIABLES|} > 1
\item \texttt{\texttt{range(VARIABLES.var)}} > 1
\end{itemize}

\begin{itemize}
\item \textbf{Arg. properties} \hfill \texttt{Functional dependency: VALUE determined by VARIABLES.}
\end{itemize}
Automaton

Figure 4.861 depicts the automaton associated with the constraint \text{MIN\_SURF\_DECREASING\_SEQUENCE}.

\[
\begin{align*}
C \leftarrow & \text{default} \\
D \leftarrow & 0 \\
R \leftarrow & \text{default}
\end{align*}
\]

\[
\leq s
\]

\[
\begin{align*}
C \leftarrow & \text{default} \\
D \leftarrow & 0 \\
R \leftarrow & \min(R, C)
\end{align*}
\]

\[
\leq t \min(R, C)
\]

\[
\begin{align*}
C \leftarrow & \text{default} \\
D \leftarrow & 0 \\
R \leftarrow & \min(R, C)
\end{align*}
\]

\[
\begin{align*}
C \leftarrow & D + \text{VAR}_i + \text{VAR}_{i+1} \\
D \leftarrow & 0
\end{align*}
\]

\[
\geq t \min(R, C)
\]

\[
\begin{align*}
C \leftarrow & \text{default} \\
D \leftarrow & 0 \\
R \leftarrow & \min(R, C)
\end{align*}
\]

\[
\begin{align*}
C \leftarrow & D + \text{VAR}_i + \text{VAR}_{i+1} \\
D \leftarrow & 0
\end{align*}
\]

\[
\leq t \min(R, C)
\]

\[
\begin{align*}
C \leftarrow & \text{default} \\
D \leftarrow & 0 \\
R \leftarrow & \min(R, C)
\end{align*}
\]

\[
\begin{align*}
C \leftarrow & C + D + \text{VAR}_{i+1} \\
D \leftarrow & 0
\end{align*}
\]

\[
\leq t \min(R, C)
\]

\[
\begin{align*}
C \leftarrow & \text{default} \\
D \leftarrow & 0 \\
R \leftarrow & \min(R, C)
\end{align*}
\]

\[
\begin{align*}
C \leftarrow & D + \text{VAR}_i + \text{VAR}_{i+1} \\
D \leftarrow & 0
\end{align*}
\]

\[
\geq t \min(R, C)
\]

\[
\begin{align*}
C \leftarrow & \text{default} \\
D \leftarrow & 0 \\
R \leftarrow & \min(R, C)
\end{align*}
\]

\[
\begin{align*}
C \leftarrow & \text{default} \\
D \leftarrow & 0 \\
R \leftarrow & \min(R, C)
\end{align*}
\]

Figure 4.861: Automaton for the \text{MIN\_SURF\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \text{DECREASING\_SEQUENCE} pattern where \text{default} is \(\infty\); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

\[
\begin{array}{|c|c|}
\hline
s & t \\
\hline
s & \min(\bar{C}, \bar{C}) & \min(\bar{C}, \bar{C}) \\
\hline
\min(\bar{C}, \bar{C}) & \bar{C} + \bar{C} + \bar{D} + \bar{D} - \text{VAR}_{i+1} \\
\hline
\end{array}
\]

Table 4.171: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the \text{MIN\_SURF\_DECREASING\_SEQUENCE} constraint defined as the composition of the \text{DECREASING\_SEQUENCE} pattern, the feature \text{SURF}, and the aggregator \text{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_SURF_DECREASING_TERRACE

DESCRIPTION AUTOMATON

Origin
Based on the DECREASING_TERRACE pattern.

Constraint
MIN_SURF_DECREASING_TERRACE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var − dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 2 ⇒ VALUE = +∞
VALUE ≥ min(2 * (minv + 1), (sv − 2) * (minv + 1))
VALUE = +∞ ∨ VALUE ≤ max(2 * (maxv − 1), (sv − 2) * (maxv − 1))
among([n1, VARIABLES[sv − 1], (maxv − 1)])
VALUE < +∞ ⇒ n1 ≥ VALUE − max(0, (sv − 2) * (maxv − 2))
among([n2, VARIABLES[sv − 1], (minv + 1)])
VALUE < +∞ ⇒ n2 ≥ min(0, (sv − 2) * (minv + 2)) − VALUE
required(VARIABLES, var)

where
minv = minval(VARIABLES.var)
sv = |VARIABLES|
maxv = maxval(VARIABLES.var)
rv = range(VARIABLES.var)

VALUE is the minimal surface of occurrences of the DECREASING_TERRACE pattern in
the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE
takes the default value +∞.

Purpose
An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence
which matches the regular expression ‘> = t >’.
Assume that the occurrence of the pattern DECREASING_TERRACE starts at position i
and ends at position j. The feature SURF computes the sum of the values from index
i + 1 to index j.

Example
(4, 6, 4, 4, 5, 2, 2, 1, 3, 5, 4, 4, 3, 3, 3))

Figure 4.862 provides an example where the MIN_SURF_DECREASING_TERRACE
(4, 6, 4, 4, 5, 2, 2, 1, 3, 5, 4, 4, 3, 3, 3)) constraint holds.

Typical
|VARIABLES| > 3
range(VARIABLES.var) > 2
Figure 4.862: Illustrating the MIN_SURF_DECREASING_TERRACE constraint of the Example slot

Arg. properties  Functional dependency: VALUE determined by VARIABLES.
Figures 4.863 and 4.864 respectively depict the automaton associated with the constraint \textsc{MIN\_SURF\_DECREASING\_TERRACE} and its simplified form.

\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default} \\
\leq s &\leq \\
\leq &
\end{align*}

\begin{align*}
{D \leftarrow D + \text{VAR}_i} &
\end{align*}

\begin{align*}
> r &> \\
< &
\end{align*}

\begin{align*}
{D \leftarrow 0} \\
R &\leftarrow \min(R, D + \text{VAR}_i)
\end{align*}

\begin{align*}
{D \leftarrow 0}
\end{align*}

Figure 4.863: Automaton for the \textsc{MIN\_SURF\_DECREASING\_TERRACE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsc{DECREASING\_TERRACE} pattern where \texttt{default} is $+\infty$.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
\textbf{s} & \textbf{r} & \textbf{t} \\
\hline
$s$ & $\min(\vec{C}, \vec{C})$ & $\min(\vec{C}, \vec{C})$ \\
$r$ & $\min(\vec{C}, \vec{C})$ & $\overline{D} + \overline{D} + \text{VAR}_{i+1}$ \\
$t$ & $\min(\vec{C}, \vec{C})$ & $\overline{D} + \overline{D} + \text{VAR}_{i+1}$ \\
\hline
\end{tabular}
\caption{Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the \textsc{MIN\_SURF\_DECREASING\_TERRACE} constraint defined as the composition of the \textsc{DECREASING\_TERRACE} pattern, the feature \textsc{SURF}, and the aggregator $\min$; cells of the glue matrix are coloured with the colour of the constituent to which they are related.}
\end{table}
Figure 4.864: Simplified automaton for the MIN_SURF_DECREASING_TERRACE constraint obtained by applying decoration Table 3.26 to the seed transducer of the DECREASING_TERRACE pattern where \texttt{default} is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$+\infty$</td>
<td>$+\infty$</td>
<td>$+\infty$</td>
</tr>
<tr>
<td>$r$</td>
<td>$+\infty$</td>
<td>$+\infty$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>$t$</td>
<td>$+\infty$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

Table 4.173: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the simplified automaton of the MIN_SURF_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature SURF, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_SURF_DIP_ON_INCREASING_SEQUENCE

Origin
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

Constraint
MIN_SURF_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 5 ∨ rv ≤ 2 ⇒ VALUE = +∞
VALUE ≥ 3 * minv + 3x
VALUE = +∞ ∨ VALUE ≤ 3 * maxv − 3
among(n1, VARIABLES[3, sv − 1], (maxv − 2, maxv − 1, maxv))
VALUE < +∞ ⇒ n1 ≥ VALUE − 3 − max(maxv − 3, 0)
among(n2, VARIABLES[3, sv − 1], (minv, minv + 1, minv + 2))
VALUE < +∞ ⇒ n2 ≥ min(minv + 3, 0) − 3 − VALUE
required(VARIABLES, var)

where
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
VALUE is the minimal surface of occurrences of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression ‘<<><<’.

Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 2 to index j.

Example
(9, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))

Figure 4.865 provides an example where the MIN_SURF_DIP_ON_INCREASING_SEQUENCE (9, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4]) constraint holds.

Typical
|VARIABLES| > 5
range(VARIABLES.var) > 2
Figure 4.865: Illustrating the MIN_SURF_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot

Arg. properties: Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.866 and 4.867 respectively depict the automaton associated with the constraint MIN_SURF_DIP_ON_INCREASING_SEQUENCE and its simplified form.

Figure 4.866: Automaton for the MIN_SURF_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is $+\infty$.
Figure 4.867: Simplified automaton for the \texttt{MIN\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.26 to the seed transducer of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Based on the GORGE pattern.

**Constraint**

\[
\text{MIN\_SURF\_GORGE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- \(\text{VALUE} : \text{dvar}\)
- \(\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})\)

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{rv} & = 2 \Rightarrow \text{VALUE} \geq \text{minv} \\
\text{rv} & \geq 3 \Rightarrow \text{VALUE} \geq \min(\text{minv}^2, (\text{sv} - 2) \times (\text{minv} + 1) - 1) \\
\text{rv} & = 2 \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} - 1 \\
\text{rv} & \geq 3 \Rightarrow \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \max(\text{maxv} - 1, (\text{sv} - 2) \times (\text{maxv} - 1) - 1) \\
\text{among}(\text{n1}, \text{VARIABLES}[2, \text{sv} - 1], (\text{maxv} - 1)) & \\
\text{VALUE} & < +\infty \land (\text{rv} = 2 \lor \text{maxv} = 1) \Rightarrow \text{n1} \geq \text{VALUE} \geq \max(0, \text{maxv} - 2) \\
\text{VALUE} & < +\infty \land \text{rv} > 2 \land \text{maxv} > 1 \Rightarrow \text{n1} \geq \text{VALUE} - (\text{sv} - 2) \times (\text{maxv} - 2) \\
\text{among}(\text{n2}, \text{VARIABLES}[2, \text{sv} - 1], (\text{minv}, \text{minv} + 1)) & \\
\text{VALUE} & < +\infty \land (\text{rv} = 2 \lor \text{minv} = -1) \Rightarrow \text{n2} \geq \min(0, \text{minv} + 1) - \text{VALUE} \\
\text{VALUE} & < +\infty \land \text{rv} > 2 \land \text{minv} < -1 \Rightarrow \\
\text{n2} & \geq (\text{sv} - 2) \times (\text{minv} + 2) - 1 - \text{VALUE} \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘\((\text{vs} \mid > (\text{vs} \mid >) > (\text{ls} \mid < (\text{ls} \mid <)) <)\)’. Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(5, (1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7))\]

Figure 4.868 provides an example where the MIN\_SURF\_GORGE \((5, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7])\) constraint holds.
Figure 4.868: Illustrating the MIN_SURF_GORGE constraint of the Example slot

Typical

| VARIABLES | > 2
\[ \text{range}(\text{VARIABLES}.\var) > 1 \]

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.869 depicts the automaton associated with the constraint MIN_SURF_GORGE.

\[
\begin{align*}
\{ C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \text{default} \}
\rightarrow 
\leq s & \rightarrow \begin{cases} 
\leq \text{min}(R, C) \\
\}\end{cases} \\
\rightarrow > r & \rightarrow \begin{cases} 
C & \leftarrow D + \text{VAR}_i \\
D & \leftarrow 0 \}
\end{cases} \\
\end{align*}
\]

Figure 4.869: Automaton for the MIN_SURF_GORGE constraint obtained by applying decoration Table 3.37 to the seed transducer of the GORGE pattern where default is $+\infty$ (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$); $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.174: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the MIN_SURF_GORGE constraint defined as the composition of the GORGE pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MIN_SURF_INCREASING**

**Description**

- **Origin**: Based on the INCREASING pattern.
- **Constraint**: MIN_SURF_INCREASING(VALUE, VARIABLES)
- **Arguments**:
  - VALUE : dvar
  - VARIABLES : collection(var−dvar)
- **Restrictions**:
  - \(sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty\)
  - \(\text{VALUE} \geq 2 \times \text{minv} + 1\)
  - \(\text{VALUE} = +\infty \lor \text{VALUE} \leq 2 \times \text{maxv} - 1\)
  - \(\text{required}(\text{VARIABLES}.\text{var})\)
  
  where
  - \(\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})\)
  - \(\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})\)
  - \(sv = |\text{VARIABLES}|\)
  - \(rv = \text{range}(\text{VARIABLES}.\text{var})\)

**Purpose**

VALUE is the minimal surface of occurrences of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

**Example**

\((4, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\)

Figure 4.870 provides an example where the MIN_SURF_INCREASING \((4, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.870: Illustrating the MIN_SURF_INCREASING constraint of the Example slot
Figures 4.871 and 4.872 respectively depict the automaton associated with the constraint `MIN_SURF_INCREASING` and its simplified form.

Figure 4.871: Automaton for the `MIN_SURF_INCREASING` constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING pattern where `default` is $+\infty$.

**Automaton**

```plaintext
\begin{align*}
D &\leftarrow 0 \\
R &\leftarrow \min(R, D + \text{VAR}_i + \text{VAR}_{i+1}) \\
C &\leftarrow \text{default}
\end{align*}
```

Figure 4.872: Simplified automaton for the `MIN_SURF_INCREASING` constraint obtained by applying decoration Table 3.40 to the seed transducer of the INCREASING pattern where `default` is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

**Table 4.175:** Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the `MIN_SURF_INCREASING` constraint defined as the composition of the INCREASING pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.176: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the simplified automaton of the \texttt{MIN\_SURF\_INCREASING} constraint defined as the composition of the \texttt{INCREASING} pattern, the feature \texttt{SURF}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_SURF_INCREASING_SEQUENCE

### Origin
Based on the INCREASING_SEQUENCE pattern.

### Constraint
\[
\text{MIN\_SURF\_INCREASING\_SEQUENCE(\text{VALUE, VARIABLES})}
\]

### Arguments
- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection(var - dvar)} \)

### Restrictions
\[
\begin{align*}
\text{sv} \leq 1 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{rv} = 2 & \Rightarrow \text{VALUE} \geq 2 \times \text{minv} + 1 \\
\text{rv} \geq 3 & \Rightarrow \text{VALUE} \geq \min(2 \times \text{minv} + 1, \text{sv} \times (\text{minv} + 1)) \\
\text{rv} = 2 & \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq 2 \times \text{maxv} - 1 \\
\text{rv} \geq 3 & \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq \max(2 \times \text{maxv} - 1, \text{sv} \times (\text{maxv} - 1)) \\
\text{among}(\text{n1, VARIABLES}[1, \text{sv}], (\text{maxv} - 1, \text{maxv})) & \\
\text{VALUE} < +\infty & \land (\text{rv} = 2 \lor \text{maxv} = 1) \Rightarrow \\
\text{n1} \geq \text{VALUE} - \max(0, 2 \times \text{maxv} - 3) & \\
\text{VALUE} < +\infty & \land \text{rv} > 2 \land \text{maxv} > 1 \Rightarrow \text{n1} \geq \text{VALUE} - 1 - \text{sv} \times (\text{maxv} - 2) \\
\text{among}(\text{n2, VARIABLES}[1, \text{sv}], (\text{minv}, \text{minv} + 1)) & \\
\text{VALUE} < +\infty & \land (\text{rv} = 2 \lor \text{minv} = -1) \Rightarrow \\
\text{n2} \geq \min(0, 2 \times \text{minv} + 3) - \text{VALUE} & \\
\text{VALUE} < +\infty & \land \text{rv} > 2 \land \text{minv} < -1 \Rightarrow \text{n2} \geq \text{sv} \times (\text{minv} + 2) - 1 - \text{VALUE} \\
\text{required}(\text{VARIABLES, var}) & \\
\text{where} & \\
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) & \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var}) & \\
\text{sv} = |\text{VARIABLES}| & \\
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
\end{align*}
\]

### Purpose
\( \text{VALUE} \) is the minimal surface of occurrences of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \( \text{VALUE} \) takes the default value \( +\infty \).

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\( (< | =)^* < | < \)’. Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature \text{SURF} computes the sum of the values from index \( i \) to index \( j + 1 \).

### Example
\( (4, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)) \)
Figure 4.873 provides an example where the MIN_SURF_INCREASING_SEQUENCE (4, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

Figure 4.873: Illustrating the MIN_SURF_INCREASING_SEQUENCE constraint of the Example slot

Typical

| | VARIABLES | > 1 
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.874 depicts the automaton associated with the constraint `MIN_SURF_INCREASING_SEQUENCE`.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*}
\]

Figure 4.874: Automaton for the `MIN_SURF_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `INCREASING_SEQUENCE` pattern where `default` is \(+\infty\); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\min(\overline{C}, \overline{C}))</td>
<td>(\min(\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>t</td>
<td>(\min(\overline{C}, \overline{C}))</td>
<td>(\overline{C} + \overline{C} + \overline{D} + \overline{D} - \text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>

Table 4.177: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the `MIN_SURF_INCREASING_SEQUENCE` constraint defined as the composition of the `INCREASING_SEQUENCE` pattern, the `SURF` feature, and the aggregator \(\min\): cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin Based on the INCREASING_TERRACE pattern.

Constraint MIN_SURF_INCREASING_TERRACE(VALUE, VARIABLES)

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(} \text{var} - \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} \leq 3 & \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq \min(2 \ast (\text{minv} + 1), (\text{sv} - 2) \ast (\text{minv} + 1)) & \\
\text{VALUE} = +\infty & \lor \text{VALUE} \leq \max(2 \ast (\text{maxv} - 1), (\text{sv} - 2) \ast (\text{maxv} - 1)) \\
\text{among} & [n_1, \text{VARIABLES}[2, \text{sv} - 1], (\text{maxv} - 1)] \\
\text{VALUE} < +\infty & \Rightarrow n_1 \geq \text{VALUE} - \max(0, (\text{sv} - 2) \ast (\text{maxv} - 2)) \\
\text{among} & [n_2, \text{VARIABLES}[2, \text{sv} - 1], (\text{minv} + 1)] \\
\text{VALUE} < +\infty & \Rightarrow n_2 \geq \min(0, (\text{sv} - 2) \ast (\text{minv} + 2)) - \text{VALUE} \\text{required} & [\text{VARIABLES}, \text{var}] \\
\text{where} & \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var}) \\
\end{align*}
\]

\[
\begin{align*}
\text{VALUE} \geq 2 & = (12 - 2) \ast (-4 + 1) \\
\text{VALUE} \geq -30 & = (12 - 2) \ast (-4 + 1)
\end{align*}
\]

VALUE is the minimal surface of occurrences of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $+\infty$.

Purpose An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression '<=' $+$ '<'. Assume that the occurrence of the pattern INCREASING_TERRACE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

Example

\[
(9, (1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4))
\]

Figure 4.875 provides an example where the MIN_SURF_INCREASING_TERRACE (9, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4]) constraint holds.

Typical

\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]
Figure 4.875: Illustrating the MIN_SURF_INCREASING_TERRACE constraint of the Example slot

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figures 4.876 and 4.877 respectively depict the automaton associated with the constraint \texttt{MIN\_SURF\_INCREASING\_TERRACE} and its simplified form.

Figure 4.876: Automaton for the \texttt{MIN\_SURF\_INCREASING\_TERRACE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{INCREASING\_TERRACE} pattern where \texttt{default} is $+\infty$.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(C, C)$</td>
<td>$\min(C, C)$</td>
<td>$\min(C, C)$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(C, C)$</td>
<td>$\min(C, C)$</td>
<td>$D + D + \text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(C, C)$</td>
<td>$D + D + \text{VAR}_{i+1}$</td>
<td>$D + D + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

Table 4.178: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the \texttt{MIN\_SURF\_INCREASING\_TERRACE} constraint defined as the composition of the \texttt{INCREASING\_TERRACE} pattern, the feature \texttt{SURF}, and the aggregator $\min$; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.877: Simplified automaton for the MIN_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 3.26 to the seed transducer of the INCREASING_TERRACE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

Table 4.179: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the simplified automaton of the MIN_SURF_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin
Based on the INFLEXION pattern.

Constraint

\[
\text{MIN\_SURF\_INFLEXION}(\text{VALUE, VARIABLES})
\]

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(}\var - \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} \leq 2 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \min(\text{minv}, (\text{sv} - 2) \cdot \text{minv}) \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \max(\text{maxv}, (\text{sv} - 2) \cdot \text{maxv}) \\
\text{among}(\text{n1}, \text{VARIABLES}[2, \text{sv} - 1], \{\text{maxv}\}) \\
\text{VALUE} & < +\infty \Rightarrow \text{n1} \geq \text{VALUE} - \max(0, (\text{sv} - 2) \cdot (\text{maxv} - 1)) \\
\text{VALUE} & < +\infty \Rightarrow \text{n2} \geq \min(0, (\text{sv} - 2) \cdot (\text{minv} + 1)) - \text{VALUE} \\
\text{among}(\text{n2}, \text{VARIABLES}[2, \text{sv} - 1], \{\text{minv}\}) \\
\text{required}(\text{VARIABLES}, \var)
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} & = \minval(\text{VARIABLES}.\var) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{maxv} & = \maxval(\text{VARIABLES}.\var) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\var)
\end{align*}
\]

Purpose

VALUE is the minimal surface of occurrences of the INFLEXION pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression '\(< (< | =)^* | > (>| =)^* <\)'. Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature \text{SURF} computes the sum of the values from index \(i + 1\) to index \(j\).

Example

\[
(1, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))
\]

Figure 4.878 provides an example where the \text{MIN\_SURF\_INFLEXION} (1, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4]) constraint holds.

Typical

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\var) > 1
\]
Figure 4.878: Illustrating the MIN_SURF_INFLEXION constraint of the Example slot

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figures 4.879 and 4.880 respectively depict the automaton associated with the constraint MIN_SURF_INFLEXION and its simplified form.

Figure 4.879: Automaton for the MIN_SURF_INFLEXION constraint obtained by applying decoration Table 3.37 to the seed transducer of the INFLEXION pattern where default is $+\infty$ (transition $r \to t$ has the same registers updates as transition $t \to r$).

Figure 4.880: Simplified automaton for the MIN_SURF_INFLEXION constraint obtained by applying decoration Table 3.26 to the seed transducer of the INFLEXION pattern where default is $+\infty$ (transition $r \to t$ has the same registers updates as transition $t \to r$); $-R_i + R_{i-1} \geq 0$ is a linear invariant.
MIN_SURF_PEAK

DESCRIPTION

Origin

Based on the PEAK pattern.

Constraint

MIN_SURF_PEAK(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \]
\[ VALUE \geq \min(\text{minv} + 1, (sv - 2) \times (\text{minv} + 1)) \]
\[ VALUE = +\infty \lor VALUE \leq \max(\text{maxv}, (sv - 2) \times \text{maxv}) \]
\[ \text{among}(n1, \text{VARIABLES}[2, sv - 1], \text{maxv}) \]
\[ VALUE < +\infty \Rightarrow n1 \geq VALUE = \max(0, (sv - 2) \times (\text{maxv} - 1)) \]
\[ \text{among}(n2, \text{VARIABLES}[2, sv - 1], (\text{minv} + 1)) \]
\[ VALUE < +\infty \Rightarrow n2 \geq \min(0, (sv - 2) \times (\text{minv} + 2)) - VALUE \]

where

\[ \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \]
\[ \text{sv} = |\text{VARIABLES}| \]
\[ \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \]
\[ \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \]

VALUE is the minimal surface of occurrences of the PEAK pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

Purpose

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression ‘\(< (\sim | <)^*(\sim | >=)^* >\)’. Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example

\[(9, \langle 7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1 \rangle)\]

Figure 4.881 provides an example where the MIN_SURF_PEAK (9, \langle 7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1 \rangle) constraint holds.

Typical

\[|\text{VARIABLES}| > 2\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]
Figure 4.881: Illustrating the MIN_SURF_PEAK constraint of the Example slot

**Symmetry**  
Items of VARIABLES can be reversed.

**Arg. properties**  
Functional dependency: VALUE determined by VARIABLES.
Figure 4.882 depicts the automaton associated with the constraint MIN_SURF_PEAK.

Figure 4.882: Automaton for the MIN_SURF_PEAK constraint obtained by applying decoration Table 3.37 to the seed transducer of the PEAK pattern where default is $\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$r$</td>
<td>$t$</td>
</tr>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{VAR}_{i+1}$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

Table 4.180: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the MIN_SURF_PEAK constraint defined as the composition of the PEAK pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin Based on the PLAIN pattern.

Constraint

\[
\text{MIN}_\text{SURF}_\text{PLAIN}(\text{VALUE}, \text{VARIABLES})
\]

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \min(\text{minv}, (\text{sv} - 2) + \text{minv}) \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \max(\text{maxv} - 1, (\text{sv} - 2) \times (\text{maxv} - 1)) \\
\text{among}(\text{n1}, \text{VARIABLES}[2, \text{sv} - 1], (\text{maxv} - 1)) & \\
\text{VALUE} & < +\infty \Rightarrow \text{n1} \geq \text{VALUE} - \max(0, (\text{sv} - 2) \times (\text{maxv} - 2)) \\
\text{among}(\text{n2}, \text{VARIABLES}[2, \text{sv} - 1], (\text{minv})) & \\
\text{VALUE} & < +\infty \Rightarrow \text{n2} \geq \min(0, (\text{sv} - 2) \times (\text{minv} + 1)) - \text{VALUE} \\
\text{required}(\text{VARIABLES}.\text{var}) & \\
\text{where} & \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

VALUE is the minimal surface of occurrences of the PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

Purpose

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘\(\text{>} = \text{<}\)’. Assume that the occurrence of the pattern PLAIN starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example

\[
(4, (2, 3, 6, 5, 7, 6, 6, 4, 5, 4, 3, 3, 6, 6, 3))
\]

Figure 4.883 provides an example where the MIN_SURF_PLAIN \((4, [2, 3, 6, 5, 7, 6, 6, 4, 5, 4, 3, 3, 6, 6, 3])\) constraint holds.

Typical

\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]
Figure 4.883: Illustrating the MIN_SURFPLAIN constraint of the Example slot

Symmetry Items of VARIABLES can be reversed.

Arg. properties Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.884 and 4.885 respectively depict the automaton associated with the constraint MIN_SURF_PLAIN and its simplified form.

\[ s \]
\[ \begin{array}{c}
\{ C \leftarrow \text{default} \} \\
\{ D \leftarrow 0 \} \\
\{ R \leftarrow \text{default} \}
\end{array} \]

\[ r = t \]
\[ \min(R, C) \leq \]
\[ > \]
\[ = \]
\[ \{ D \leftarrow D + \text{VAR}_i \} < \]
\[ \{ D \leftarrow 0 \} \]
\[ \{ D \leftarrow D + \text{VAR}_i \} < \]
\[ \{ D \leftarrow 0 \} \]
\[ \{ D \leftarrow D + \text{VAR}_i \} < \]
\[ \{ D \leftarrow 0 \} \]

Figure 4.884: Automaton for the MIN_SURF_PLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLAIN pattern where \text{default} is \(+\infty\).

\[ s \]
\[ \begin{array}{c}
\{ C \leftarrow \text{default} \} \\
\{ D \leftarrow 0 \} \\
\{ R \leftarrow \text{default} \}
\end{array} \]

\[ r = t \]
\[ R \leq \]
\[ > \]
\[ = \]
\[ \{ D \leftarrow D + \text{VAR}_i \} < \]
\[ \{ D \leftarrow 0 \} \]
\[ \{ D \leftarrow D + \text{VAR}_i \} < \]
\[ \{ D \leftarrow 0 \} \]
\[ \{ D \leftarrow D + \text{VAR}_i \} < \]
\[ \{ D \leftarrow 0 \} \]

Figure 4.885: Simplified automaton for the MIN_SURF_PLAIN constraint obtained by applying decoration Table 3.26 to the seed transducer of the PLAIN pattern where \text{default} is \(+\infty\); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
### Table 4.181: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the \texttt{MIN\_SURF\_PLAIN} constraint defined as the composition of the \texttt{PLAIN} pattern, the feature \texttt{SURF}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\min(C, C))</td>
<td>(\min(C, C))</td>
<td>(\min(C, C))</td>
</tr>
<tr>
<td>(r)</td>
<td>(\min(C, C))</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1} )</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1} )</td>
</tr>
<tr>
<td>(t)</td>
<td>(\min(C, C))</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1} )</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>

### Table 4.182: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the simplified automaton of the \texttt{MIN\_SURF\_PLAIN} constraint defined as the composition of the \texttt{PLAIN} pattern, the feature \texttt{SURF}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(+\infty)</td>
<td>(+\infty)</td>
<td>(+\infty)</td>
</tr>
<tr>
<td>(r)</td>
<td>(+\infty)</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1} )</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1} )</td>
</tr>
<tr>
<td>(t)</td>
<td>(+\infty)</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1} )</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>
MIN_Surf_Plateau

Origin
Based on the PLATEAU pattern.

Constraint
MIN_Surf_Plateau(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ min(minv + 10, (sv − 2) * (minv + 1))
VALUE = +∞ ∀ VALUE ≤ max(maxv, (sv − 2) * maxv)
among([n1, VARIABLES[2, sv − 1], [maxv]])
VALUE < +∞ ⇒ n1 ≥ VALUE − max(0, (sv − 2) * (maxv − 1))
among([n2, VARIABLES[2, sv − 1], [minv + 1]])
VALUE < +∞ ⇒ n2 ≥ min(0, (sv − 2) * (minv + 2)) − VALUE
required(VARIABLES, var)

where
minv = minval(VARIABLES.var)
sv = |VARIABLES|
maxv = maxval(VARIABLES.var)
rv = range(VARIABLES.var)

VALUE is the minimal surface of occurrences of the PLATEAU pattern in the time-series
given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default
value +∞.

Purpose
An occurrence of the pattern PLATEAU is the maximal subsequence which matches the
regular expression '<=^>'.
Assume that the occurrence of the pattern PLATEAU starts at position i and ends at
position j. The feature SURF computes the sum of the values from index i + 1 to index
j.

Example
(3, (7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5))

Figure 4.886 provides an example where the MIN_Surf_Plateau
(3, [7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1
Figure 4.886: Illustrating the MIN_SURF_PLATEAU constraint of the Example slot

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.887 and 4.888 respectively depict the automaton associated with the constraint MIN_SURF_PLATEAU and its simplified form.

Figure 4.887: Automaton for the MIN_SURF_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLATEAU pattern where default is $+\infty$.

Figure 4.888: Simplified automaton for the MIN_SURF_PLATEAU constraint obtained by applying decoration Table 3.26 to the seed transducer of the PLATEAU pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.183: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the MIN_SURF_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\min(\overrightarrow{C}, \overleftarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overleftarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overleftarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\min(\overrightarrow{C}, \overleftarrow{C}))</td>
<td>(\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>t</td>
<td>(\min(\overrightarrow{C}, \overleftarrow{C}))</td>
<td>(\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>

Table 4.184: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the simplified automaton of the MIN_SURF_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(+\infty)</td>
<td>(+\infty)</td>
<td>(+\infty)</td>
</tr>
<tr>
<td>r</td>
<td>(+\infty)</td>
<td>(\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>t</td>
<td>(+\infty)</td>
<td>(\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{D} + \overleftarrow{D} + \text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>
MIN_SURF_PROPER.PLAIN

**Origin**
Based on the PROPER.PLAIN pattern.

**Constraint**
MIN_SURF_PROPER.PLAIN(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
\[ sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = +\infty \]
\[ VALUE \geq \min(2 * \text{minv}, (sv - 2) * \text{minv}) \]
\[ VALUE = +\infty \lor VALUE \leq \max(2 * (maxv - 1), (sv - 2) * (maxv - 1)) \]
\[ \text{among}(n1, \text{VARIABLES}[2, sv - 1], (maxv - 1)) \]
\[ VALUE < +\infty \Rightarrow n1 \geq \text{VALUE} - \max(0, (sv - 2) * (maxv - 2)) \]
\[ \text{among}(n2, \text{VARIABLES}[2, sv - 1], \text{minv}) \]
\[ VALUE < +\infty \Rightarrow n2 \geq \min(0, (sv - 2) * (minv + 1)) - VALUE \]
required(VARIABLES, var)

where
\[ \text{minv} = \text{minval}(\text{VARIABLES}.var) \]
\[ sv = |\text{VARIABLES}| \]
\[ \text{maxv} = \text{maxval}(\text{VARIABLES}.var) \]
\[ rv = \text{range}(\text{VARIABLES}.var) \]

VALUE is the minimal surface of occurrences of the PROPER.PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

**Purpose**
An occurrence of the pattern PROPER.PLAIN is the maximal subsequence which matches the regular expression ‘> = + <’.
Assume that the occurrence of the pattern PROPER.PLAIN starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+1\) to index \(j\).

**Example**
\((8, (2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5))\)

Figure 4.889 provides an example where the MIN_SURF_PROPER.PLAIN \((8, [2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5])\) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.var) > 1 \]
Symmetry: Items of VARIABLES can be reversed.

Arg. properties: Functional dependency: VALUE determined by VARIABLES.
Automaton  
Figures 4.890 and 4.891 respectively depict the automaton associated with the constraint MIN_SURF_PROPER.PLAIN and its simplified form.

Figure 4.890: Automaton for the MIN_SURF_PROPER.PLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER.PLAIN pattern where default is $+\infty$.

Figure 4.891: Simplified automaton for the MIN_SURF_PROPER.PLAIN constraint obtained by applying decoration Table 3.26 to the seed transducer of the PROPER.PLAIN pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.185: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the \texttt{MIN\_SURF\_PROPER\_PLAIN} constraint defined as the composition of the \texttt{PROPER\_PLAIN} pattern, the feature \texttt{SURF}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\min(C, C))</td>
<td>(\min(C, C))</td>
<td>(\min(C, C))</td>
</tr>
<tr>
<td>(r)</td>
<td>(\min(C, C))</td>
<td>(\min(C, C))</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>(t)</td>
<td>(\min(C, C))</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1})</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>

Table 4.186: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the simplified automaton of the \texttt{MIN\_SURF\_PROPER\_PLAIN} constraint defined as the composition of the \texttt{PROPER\_PLAIN} pattern, the feature \texttt{SURF}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(+\infty)</td>
<td>(+\infty)</td>
<td>(+\infty)</td>
</tr>
<tr>
<td>(r)</td>
<td>(+\infty)</td>
<td>(+\infty)</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>(t)</td>
<td>(+\infty)</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1})</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>
MIN_SURF_PROPER_PLATEAU

**Origin**
Based on the PROPER_PLATEAU pattern.

**Constraint**

\[
\text{MIN\_SURF\_PROPER\_PLATEAU}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(} \text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \min(2 \ast (\text{minv} + 1), (\text{sv} - 2) \ast (\text{minv} + 2)) \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \max(2 \ast \text{maxv}, (\text{sv} - 2) \ast \text{maxv}) \\
\text{VALUE} & < +\infty \Rightarrow \text{n1} \geq \text{VALUE} - \max(0, (\text{sv} - 2) \ast (\text{maxv} - 1)) \\
\text{VALUE} & < +\infty \Rightarrow \text{n2} \geq \min(0, (\text{sv} - 2) \ast (\text{minv} + 2)) - \text{VALUE} \\
\text{among}(\text{n1}, \text{VARIABLES}[2, \text{sv} - 1], \langle \text{maxv} \rangle) \\
\text{among}(\text{n2}, \text{VARIABLES}[2, \text{sv} - 1], \langle \text{minv} + 1 \rangle) \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

VALUE is the minimal surface of occurrences of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

**Purpose**

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression ‘< = + >’.
Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
(6, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))
\]

Figure 4.892 provides an example where the MIN_SURF_PROPER_PLATEAU \((6, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3])\) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]
Figure 4.892: Illustrating the \textsc{MIN\_SURF\_PROPER\_PLATEAU} constraint of the \texttt{Example} slot

\begin{tabular}{|c|c|c|c|}
\hline
\textbf{Symmetry} & Items of VARIABLES can be reversed. \\
\hline
\textbf{Arg. properties} & Functional dependency: VALUE determined by VARIABLES. \\
\hline
\end{tabular}
Automaton

Figures 4.893 and 4.894 respectively depict the automaton associated with the constraint MIN_SURF_PROPER_PLATEAU and its simplified form.

**Figure 4.893:** Automaton for the MIN_SURF_PROPER_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER_PLATEAU pattern where default is $+\infty$.

**Figure 4.894:** Simplified automaton for the MIN_SURF_PROPER_PLATEAU constraint obtained by applying decoration Table 3.26 to the seed transducer of the PROPER_PLATEAU pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.187: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the MIN_SURF_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$r$</td>
<td>$t$</td>
</tr>
<tr>
<td>$s$</td>
<td>$\min(\vec{C}, \vec{C})$</td>
<td>$\min(\vec{C}, \vec{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\vec{C}, \vec{C})$</td>
<td>$\min(\vec{C}, \vec{C})$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(\vec{C}, \vec{C})$</td>
<td>$\vec{D} + \vec{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

Table 4.188: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the simplified automaton of the MIN_SURF_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$r$</td>
<td>$t$</td>
</tr>
<tr>
<td>$s$</td>
<td>$+\infty$</td>
<td>$+\infty$</td>
</tr>
<tr>
<td>$r$</td>
<td>$+\infty$</td>
<td>$+\infty$</td>
</tr>
<tr>
<td>$t$</td>
<td>$+\infty$</td>
<td>$\vec{D} + \vec{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>
**MIN_SURF_STEADY**

**DESCRIPTION AUTOMATON**

**Origin**
Based on the STEADY pattern.

**Constraint**
MIN_SURF_STEADY(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
- $sv \leq 1 \Rightarrow VALUE = +\infty$
- $VALUE \geq 2 * \text{minv}$
- $VALUE = +\infty \lor VALUE \leq 2 * \text{maxv}$
  among([n1, VARIABLES[1, sv], (maxv)])
- $VALUE < +\infty \Rightarrow n1 \geq VALUE - 2 * (\text{maxv} - 1)$
  among([n2, VARIABLES[1, sv], (minv)])
- $VALUE < +\infty \Rightarrow n2 \geq 2 * (\text{minv} + 1) - VALUE$
  required(VARIABLES, var)

where
- $\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})$
- $\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})$
- $sv = |\text{VARIABLES}|$

VALUE is the minimal surface of occurrences of the STEADY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $+\infty$.

**Purpose**
An occurrence of the pattern STEADY is the subsequence which matches the regular expression '='. Assume that the occurrence of the pattern STEADY starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

**Example**
- $(2, (1, 1, 7, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6))$

Figure 4.895 provides an example where the MIN_SURF_STEADY (2, [1, 1, 7, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6]) constraint holds.

**Typical**
- $|\text{VARIABLES}| > 1$

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.895: Illustrating the MIN_SURF_STEADY constraint of the Example slot
Automaton

Figures 4.896 and 4.897 respectively depict the automaton associated with the constraint \textsc{MIN} \_ \textsc{SURF} \_ \textsc{STEADY} and its simplified form.

\[
\begin{align*}
\{ C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
\{ D &\leftarrow 0 \\
R &\leftarrow \min(R, D + \text{VAR}_i + \text{VAR}_{i+1}) \}
\end{align*}
\]

\[
\min(R, C)
\]

Figure 4.896: Automaton for the \textsc{MIN} \_ \textsc{SURF} \_ \textsc{STEADY} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsc{STEADY} pattern where \texttt{default} is $+\infty$.

\[
\begin{align*}
\{ R &\leftarrow \text{default} \}
\end{align*}
\]

\[
\{ R \leftarrow \min(R, \text{VAR}_i + \text{VAR}_{i+1}) \}
\]

\[
R
\]

Figure 4.897: Simplified automaton for the \textsc{MIN} \_ \textsc{SURF} \_ \textsc{STEADY} constraint obtained by applying decoration Table 3.40 to the seed transducer of the \textsc{STEADY} pattern where \texttt{default} is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

\[
\begin{array}{c}
s \quad \begin{array}{c}
\min(C, C) \\
\end{array}
\end{array}
\]

Table 4.189: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the \textsc{MIN} \_ \textsc{SURF} \_ \textsc{STEADY} constraint defined as the composition of the \textsc{STEADY} pattern, the feature \textsc{SURF}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.190: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the simplified automaton of the MIN_SURF_STEADY constraint defined as the composition of the STEADY pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin

Based on the STEADY_SEQUENCE pattern.

Constraint

MIN_SURF_STEADY_SEQUENCE(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var–dvar)

Restrictions

sv ≤ 1 ⇒ VALUE = +∞
rv = 1 ⇒ VALUE ≥ sv * minv
rv ≥ 2 ⇒ VALUE ≥ min(2 * minv, sv * minv)
rv = 1 ⇒ VALUE = +∞ ∨ VALUE ≤ sv * maxv
rv ≥ 2 ⇒ VALUE = +∞ ∨ VALUE ≤ max(2 * maxv, sv * maxv)
among(n1, VARIABLES[1, sv], (maxv))
VALUE < +∞ ⇒ n1 ≥ VALUE − max(0, sv * (maxv − 1))
among(n2, VARIABLES[2, sv], (minv))
VALUE < +∞ ⇒ n2 ≥ min(0, sv * (minv + 1)) − VALUE
required(VARIABLES, var)

where

minv = minval(VARIABLES.var)
rv = range(VARIABLES.var)
sv = |VARIABLES|
maxv = maxval(VARIABLES.var)

Purpose

VALUE is the minimal surface of occurrences of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression ‘=+’.

Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

Example

(2, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1))

Figure 4.898 provides an example where the MIN_SURF_STEADY_SEQUENCE (2, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1]) constraint holds.

Typical

|VARIABLES| > 1
Figure 4.898: Illustrating the MIN_SURF_STEADY_SEQUENCE constraint of the Example slot

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figures 4.899 and 4.900 respectively depict the automaton associated with the constraint MIN_SURF_STEADY_SEQUENCE and its simplified form.

Figure 4.899: Automaton for the MIN_SURF_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STEADY_SEQUENCE pattern where default is $+\infty$.

Figure 4.900: Simplified automaton for the MIN_SURF_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.25 to the seed transducer of the STEADY_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
### Table 4.191: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the MIN_SURF_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\overrightarrow{C} + \overleftarrow{C} + \overrightarrow{D} + \overleftarrow{D} - \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

### Table 4.192: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the simplified automaton of the MIN_SURF_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\overrightarrow{C} + \overleftarrow{C} - \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>
Based on the `Strictly_Decreasing_Sequence` pattern.

**Constraint**
```
MIN_SURF_STRICLY_DECREASING_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

**Restrictions**
- \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \)
- \( \minv < 0 \Rightarrow VALUE \geq \ell_1 + \minv + [\ell_1 (\ell_1 - 1)/2] \)
- \( \minv \geq 0 \Rightarrow VALUE \geq 2 + \minv + 1 \)
- \( \maxv > 0 \Rightarrow VALUE = +\infty \lor VALUE \leq \ell_2 + \maxv - [\ell_2 (\ell_2 - 1)/2] \)
- \( \maxv \leq 0 \Rightarrow VALUE = +\infty \lor VALUE \leq 2 + \maxv - 1 \)
- \( \text{among}(n1, VARIABLES[1, sv], (maxv - 1, maxv)) \)
- \( VALUE < +\infty \Rightarrow n1 \geq VALUE - \max(0, \min(sv, rv) \ast (maxv - 2)) - 1 \)
- \( \text{among}(n2, VARIABLES[1, sv], (minv, minv + 1)) \)
- \( VALUE < +\infty \Rightarrow n2 \geq \min(0, \min(sv, rv) \ast (minv + 2)) - 1 - VALUE \)

**Purpose**

An occurrence of the pattern `Strictly_Decreasing_Sequence` is the *maximal* subsequence which matches the regular expression `">+"'.

Assume that the occurrence of the pattern `Strictly_Decreasing_Sequence` starts at position \( i \) and ends at position \( j \). The feature `surf` computes the sum of the values from index \( i \) to index \( j + 1 \).

**Example**

\((7, 4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3)\)

Figure 4.901 provides an example where the `MIN_SURF_STRICLY_DECREASING_SEQUENCE` constraint holds.
Figure 4.901: Illustrating the MIN_SURF_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot

Typical

| VARIABLES | > 1
\[
\text{range(VARIABLES.var)} > 1
\]

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.902 and 4.903 respectively depict the automaton associated with the constraint MIN_SURF_STRICTLY_DECREASING_SEQUENCE and its simplified form.

Figure 4.902: Automaton for the MIN_SURF_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $+\infty$.

Figure 4.903: Simplified automaton for the MIN_SURF_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.25 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.193: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the MIN_SURF_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator \( \min \); cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \min(\overrightarrow{C}, \overleftarrow{C}) )</td>
<td>( \min(\overrightarrow{C}, \overleftarrow{C}) )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \min(\overrightarrow{C}, \overleftarrow{C}) )</td>
<td>( \overrightarrow{C} + \overleftarrow{C} + \overrightarrow{D} + \overleftarrow{D} - \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>

Table 4.194: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the MIN_SURF_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator \( \min \); cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \min(\overrightarrow{C}, \overleftarrow{C}) )</td>
<td>( \min(\overrightarrow{C}, \overleftarrow{C}) )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \min(\overrightarrow{C}, \overleftarrow{C}) )</td>
<td>( \overrightarrow{C} + \overleftarrow{C} - \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>
### Description

Based on the `STRICTLY_INCREASING_SEQUENCE` pattern.

#### Arguments

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var–dvar)`

#### Restrictions

\[
\begin{align*}
sv & \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
minv & < 0 \Rightarrow VALUE \geq \ell_1 \cdot \text{minv} + \lfloor \ell_1 \cdot (\ell_1 - 1)/2 \rfloor \\
minv & \geq 0 \Rightarrow VALUE \geq 2 \cdot \text{minv} + 1 \\
maxv & > 0 \Rightarrow VALUE = +\infty \lor VALUE \leq \ell_2 \cdot \text{maxv} - \lfloor \ell_2 \cdot (\ell_2 - 1)/2 \rfloor \\
maxv & \leq 0 \Rightarrow VALUE = +\infty \lor VALUE \leq 2 \cdot \text{maxv} - 1 \\
\text{among}(\text{n1, VARIABLES[1,sv]}, (\text{maxv} - 1, \text{maxv})) & \Rightarrow VALUE < +\infty \Rightarrow \text{n1} \geq \text{max}(0, \text{min}(sv, rv) \cdot (\text{maxv} - 2)) - 1 \\
\text{among}(\text{n2, VARIABLES[1,sv]}, (\text{minv}, \text{minv} + 1)) & \Rightarrow VALUE < +\infty \Rightarrow \text{n2} \geq \text{max}(0, \text{min}(sv, rv) \cdot (\text{minv} + 2)) - 1 - \text{VALUE} \\
\text{required(VARIABLES, var)} & \end{align*}
\]

where

- \( minv = \text{minval(VARIABLES.var)} \)
- \( rv = \text{range(VARIABLES.var)} \)
- \( sv = |\text{VARIABLES}| \)
- \( \ell_1 = \text{min}(\text{min}(sv, rv), |\text{minv}|) \)
- \( \ell_2 = \text{min}(\text{min}(sv, rv), |\text{maxv}|) \)
- \( \text{maxv} = \text{maxval(VARIABLES.var)} \)

**VALUE** is the minimal surface of occurrences of the `STRICTLY_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, **VALUE** takes the default value \(+\infty\).

An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the *maximal* subsequence which matches the regular expression ‘<’.

Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position \( i \) and ends at position \( j \). The feature `SURF` computes the sum of the values from index \( i \) to index \( j + 1 \).

### Example

\[(6, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))\]

Figure 4.904 provides an example where the `MIN_SURF_STRICTLY_INCREASING_SEQUENCE` constraint holds.
Figure 4.904: Illustrating the MIN_SURF_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot

Typical

| VARIABLES | > 1
range(VARIABLES.var) > 1

Arg. properties  

Functional dependency: VALUE determined by VARIABLES.
Figures 4.905 and 4.906 respectively depict the automaton associated with the constraint MIN_SURF STRICTLY INCREASING SEQUENCE and its simplified form.

Figure 4.905: Automaton for the MIN_SURF STRICTLY INCREASING SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICTLY INCREASING SEQUENCE pattern where default is $+\infty$.

Figure 4.906: Simplified automaton for the MIN_SURF STRICTLY INCREASING SEQUENCE constraint obtained by applying decoration Table 3.25 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.195: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the **MIN_SURF_STRICTLY_INCREASING_SEQUENCE** constraint defined as the composition of the **STRICTLY_INCREASING_SEQUENCE** pattern, the feature **SURF**, and the aggregator **min**; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>( \min(\overrightarrow{C}, \overleftarrow{C}) )</td>
<td>( \min(\overrightarrow{C}, \overleftarrow{C}) )</td>
</tr>
<tr>
<td>r</td>
<td>( \min(\overrightarrow{C}, \overleftarrow{C}) )</td>
<td>( \overrightarrow{C} + \overleftarrow{C} + \overrightarrow{D} + \overleftarrow{D} - \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>

Table 4.196: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the simplified automaton of the **MIN_SURF_STRICTLY_INCREASINGSEQUENCE** constraint defined as the composition of the **STRICTLY_INCREASINGSEQUENCE** pattern, the feature **SURF**, and the aggregator **min**; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>( \min(\overrightarrow{C}, \overleftarrow{C}) )</td>
<td>( \min(\overrightarrow{C}, \overleftarrow{C}) )</td>
</tr>
<tr>
<td>r</td>
<td>( \min(\overrightarrow{C}, \overleftarrow{C}) )</td>
<td>( \overrightarrow{C} + \overleftarrow{C} - \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>
Based on the SUMMIT pattern.

**Origin**  
Based on the SUMMIT pattern.

**Constraint**  
\( \text{MIN}_\text{SURF}_\text{SUMMIT}(\text{VALUE}, \text{VARIABLES}) \)

**Arguments**  
\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} : \text{dvar})
\end{align*}
\]

**Restrictions**  
\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{rv} & = 2 \Rightarrow \text{VALUE} \geq \text{minv} + 10 \\
\text{rv} & \geq 3 \Rightarrow \text{VALUE} \geq \min(\text{minv} + 10, (\text{sv} - 2) * (\text{minv} + 1) + 12) \\
\text{rv} & = 2 \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} \\
\text{rv} & \geq 3 \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq \max(\maxv, (\text{sv} - 2) * (\text{maxv} - 1) + 1) \\
\text{among}(n1, \text{VARIABLES}[2, \text{sv} - 1], (\text{maxv} - 1, \text{maxv})) & \\
\text{VALUE} & < +\infty \land (\text{rv} = 2 \lor \text{maxv} = 1) \Rightarrow n1 \geq \text{VALUE} - \max(0, \text{maxv} - 1) \\
\text{VALUE} & < +\infty \land \text{rv} > 2 \land \text{maxv} > 1 \Rightarrow \\
n1 & \geq \text{VALUE} - (\text{sv} - 2) * (\text{maxv} - 2) - 1 \\
\text{among}(n2, \text{VARIABLES}[2, \text{sv} - 1], (\text{minv} + 1)) & \\
\text{VALUE} & < +\infty \land (\text{rv} = 2 \lor \text{minv} = 1) \Rightarrow n2 \geq \min(0, \text{minv} + 2) - \text{VALUE} \\
\text{VALUE} & < +\infty \land \text{rv} > 2 \land \text{minv} < -1 \Rightarrow n2 \geq (\text{sv} - 2) * (\text{minv} + 2) - \text{VALUE} \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
\text{minv} & = \min(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \range(\text{VARIABLES}.\text{var}) \\
\text{sv} & = [\text{VARIABLES}] \\
\text{maxv} & = \max(\text{VARIABLES}.\text{var})
\end{align*}
\]

\( \text{VALUE} \) is the minimal surface of occurrences of the SUMMIT pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value \(+\infty\).

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression '| < (= | >)' \((= | >) \rangle \). Assume that the occurrence of the pattern SUMMIT starts at position \( i \) and ends at position \( j \). The feature \text{SURF} computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**  
\((3, (7, 1, 5, 4, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1))\)

Figure 4.907 provides an example where the \text{MIN}_\text{SURF}_\text{SUMMIT} \((3, [7, 1, 5, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1])\) constraint holds.
Figure 4.907: Illustrating the MIN_SURF_SUMMIT constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.908: Automaton for the MIN_SURF_SUMMIT constraint obtained by applying decoration Table 3.37 to the seed transducer of the SUMMIT pattern where default is $+\infty$ (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$); $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.197: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the \texttt{MIN\_SURF\_SUMMIT} constraint defined as the composition of the \texttt{SUMMIT} pattern, the feature \texttt{SURF}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\min(\vec{C}, \vec{C}))</td>
<td>(\min(\vec{C}, \vec{C}))</td>
<td>(\min(\vec{C}, \vec{C}))</td>
<td>(\min(\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\min(\vec{C}, \vec{C}))</td>
<td>(\vec{B} + \vec{D} + \text{VAR}_{i+1})</td>
<td>(\vec{C} + \vec{B} + \vec{D} + \text{VAR}_{i+1})</td>
<td>(\min(\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>t</td>
<td>(\min(\vec{C}, \vec{C}))</td>
<td>(\vec{C} + \vec{B} + \vec{D} + \text{VAR}_{i+1})</td>
<td>(\min(\vec{C}, \vec{C}))</td>
<td>(\vec{C} + \vec{B} + \vec{D} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>u</td>
<td>(\min(\vec{C}, \vec{C}))</td>
<td>(\min(\vec{C}, \vec{C}))</td>
<td>(\vec{C} + \vec{B} + \vec{D} + \text{VAR}_{i+1})</td>
<td>(\min(\vec{C}, \vec{C}))</td>
</tr>
</tbody>
</table>
Origin Based on the VALLEY pattern.

Constraint

\[ \text{MIN}_\text{SURF}_\text{VALLEY}(\text{VALUE}, \text{VARIABLES}) \]

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} \rightarrow \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} & \leq 2 \vee \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \min(\text{minv}, (\text{sv} - 2) + \text{minv}) \\
\text{VALUE} & \geq +\infty \Rightarrow \text{VALUE} \leq \max(0, (\text{sv} - 2) + (\text{maxv} - 2)) \\
\text{VALUE} & < +\infty \Rightarrow \text{VALUE} = \max(0, (\text{sv} - 2) + (\text{maxv} - 2)) \\
\text{required}(\text{VARIABLES}, \text{var}) & \geq \min(0, (\text{sv} - 2) + (\text{minv} + 1)) - \text{VALUE}
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

VALUE is the minimal surface of occurrences of the VALLEY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

Purpose

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \(> (\rightarrow) (< | =) \rightarrow <\).

Assume that the occurrence of the pattern VALLEY starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example

\[(7, 1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7)\]

Figure 4.909 provides an example where the MIN_SURF_VALLEY \((7, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7])\) constraint holds.

Typical

\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]
Figure 4.909: Illustrating the MIN_SURF_VALLEY constraint of the Example slot

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figure 4.910 depicts the automaton associated with the constraint MIN_SURF_VALLEY.

Figure 4.910: Automaton for the MIN_SURF_VALLEY constraint obtained by applying decoration Table 3.37 to the seed transducer of the VALLEY pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

Table 4.198: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the MIN_SURF_VALLEY constraint defined as the composition of the VALLEY pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin Based on the ZIGZAG pattern.

Constraint \( \text{MIN\_SURF\_ZIGZAG}(\text{VALUE}, \text{VARIABLES}) \)

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} &\leq 3 \lor \text{rv} \leq 1 \implies \text{VALUE} = +\infty \\
\text{VALUE} &\geq \min\left(2 \ast \text{minv} + 1, \left(\left\lfloor \text{sv} - 1 \right\rfloor / 2 \ast \text{minv} + \left\lfloor \text{sv} - 2 \right\rfloor / 2 * (\text{minv} + 1)\right)\right) \\
\lor \left(\text{VALUE} \leq \max\left(2 \ast \text{maxv} - 1, \left(\left\lfloor \text{sv} - 1 \right\rfloor / 2 \ast \text{maxv} + \left\lfloor \text{sv} - 2 \right\rfloor / 2 * (\text{maxv} - 1)\right)\right)\right) \\
\text{among}(\text{n1}, \text{VARIABLES}[2, \text{sv} - 1], \left(\text{maxv} - 1, \text{maxv}\right)) \\
\text{VALUE} &< +\infty \implies \text{n1} \geq \text{VALUE} - \left(\left\lfloor \text{sv} - 1 \right\rfloor / 2 \ast \max(0, (\text{sv} - 2) * (\text{maxv} - 2))\right) \\
\text{among}(\text{n2}, \text{VARIABLES}[2, \text{sv} - 1], \left(\text{minv}, \text{minv} + 1\right)) \\
\text{VALUE} &< +\infty \implies \text{n2} \geq \min(0, (\text{sv} - 2) * (\text{minv} + 2)) - \left\lfloor \text{sv} - 1 \right\rfloor / 2 - \text{VALUE} \\
\text{required}(\text{VARIABLES}.\text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

VALUE is the minimal surface of occurrences of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

Purpose An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \( '(<>)^* (< | <> ) | (<>)^* (> | <<)' \).
Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

Example \((5, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\)

Figure 4.911 provides an example where the \( \text{MIN\_SURF\_ZIGZAG} (5, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1]) \) constraint holds.
Figure 4.911: Illustrating the MIN_SURF_ZIGZAG constraint of the Example slot

Typical

\(|\text{VARIBALES}| > 3 \)
\(\text{range}(\text{VARIBALES.var}) > 1\)

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.912 and 4.913 respectively depict the automaton associated with the constraint MIN_SURF_ZIGZAG and its simplified form.
Figure 4.912: Automaton for the MIN_SURF_ZIGZAG constraint obtained by applying decoration Table 3.37 to the seed transducer of the ZIGZAG pattern where default is $+\infty$: (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=; (2)$ on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.
Figure 4.913: Simplified automaton for the MIN_SURF_ZIGZAG constraint obtained by applying decoration Table 3.24 to the seed transducer of the ZIGZAG pattern where default is $+\infty$; missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.199: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the MIN_SURF_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature SURF, and the aggregator min. Cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.200: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the simplified automaton of the MIN\_SURF\_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature SURF, and the aggregator \( \min \): cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
</tr>
<tr>
<td>a</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( C + D + VAR_{i+1} )</td>
<td>( \min(C, C) )</td>
<td>( C + D + VAR_{i+1} )</td>
<td>( \min(C, C) )</td>
</tr>
<tr>
<td>b</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( B + D + VAR_{i+1} )</td>
<td>( B + D + VAR_{i+1} )</td>
<td>( \min(C, C) )</td>
<td>( C + D + VAR_{i+1} )</td>
<td>( \min(C, C) )</td>
</tr>
<tr>
<td>c</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( C + B + D + VAR_{i+1} )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
</tr>
<tr>
<td>d</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( B + D + VAR_{i+1} )</td>
<td>( B + D + VAR_{i+1} )</td>
<td>( \min(C, C) )</td>
<td>( C + D + VAR_{i+1} )</td>
<td>( \min(C, C) )</td>
</tr>
<tr>
<td>e</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( C + B + D + VAR_{i+1} )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
</tr>
<tr>
<td>f</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( C + B + D + VAR_{i+1} )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
<td>( \min(C, C) )</td>
</tr>
</tbody>
</table>
### Origin
Based on the VALLEY pattern.

### Constraint
\[ \text{MIN}_\text{VALLEY} (\text{VARIABLES}, \text{FEATURES}, \text{DEFAULT}) \]

### Arguments
- **VARIABLES**: \( \text{collection}(\text{var} - \text{dvar}) \)
- **FEATURES**: \( \text{collection}(\text{var} - \text{dvar}) \)
- **DEFAULT**: \( \text{int} \)

### Restrictions
- \( \text{required}(\text{VARIABLES}, \text{var}) \)
- \( \text{required}(\text{FEATURES}, \text{var}) \)
- \( \text{sv} = |\text{FEATURES}| \)
- \( \text{sv} \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{FEATURES}.\text{var} = \text{DEFAULT} \)
- \( \text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \geq \text{minv} \)
- \( \text{FEATURES}.\text{var} = \text{DEFAULT} \lor \text{FEATURES}.\text{var} \leq \text{maxv} - 1 \)
- \( \text{DEFAULT} < \text{minv} \lor \text{DEFAULT} > \text{maxv} - 1 \)

where
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \)

Consider the time-series given by the VARIABLES collection. If position \( i \) does not correspond to the first position in VARIABLES where an occurrence of VALLEY is identified (even if this occurrence of pattern is not complete) then FEATURES\[i\] is the default value DEFAULT; otherwise FEATURES\[i\] gives the feature value of the corresponding occurrence of VALLEY.

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \( \geq (> (\geq |>) (\leq | =) ^* <) \).

Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

### Purpose
Consider the time-series given by the VARIABLES collection. If position \( i \) does not correspond to the first position in VARIABLES where an occurrence of VALLEY is identified (even if this occurrence of pattern is not complete) then FEATURES\[i\] is the default value DEFAULT; otherwise FEATURES\[i\] gives the feature value of the corresponding occurrence of VALLEY.

### Example
Figure 4.914 provides an example where the MIN_VALLEY constraint holds.

### Typical
- \( |\text{VARIABLES}| > 2 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

### Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.914: Illustrating the MIN_VALLEY constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
MIN_WIDTH_DECREASING_SEQUENCE

### Origin
Based on the **DECREASING_SEQUENCE** pattern.

### Constraint
```plaintext
MIN_WIDTH_DECREASING_SEQUENCE(VALUE, VARIABLES)
```

### Arguments
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

### Restrictions
- \(sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = sv + 1\)
- \(VALUE \geq 2\)
- \(rv = 2 \Rightarrow VALUE = sv + 1 \lor VALUE \leq 20\)
- \(rv \geq 3 \Rightarrow VALUE = sv + 1 \lor VALUE \leq sv\) is required(VARIABLES, var)

where
- \(sv = |VARIABLES|\)
- \(rv = \text{range}(VARIABLES, \text{var})\)

### Purpose
**VALUE** is the minimal width of occurrences of the **DECREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value \(|VARIABLES| + 1\).

An occurrence of the pattern **DECREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression ‘\(> (> | =) > | >\)’.

Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i + 2\).

### Example
\((2, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\)

Figure 4.915 provides an example where the **MIN_WIDTH_DECREASING_SEQUENCE** constraint holds.

### Typical
- \(|VARIABLES| > 1\)
- \(\text{range}(VARIABLES, \text{var}) > 1\)

### Symmetry
One and the same constant can be added to the \(\text{var}\) attribute of all items of **VARIABLES**.

### Arg. properties
**Functional dependency**: **VALUE** determined by **VARIABLES**.
Figure 4.915: Illustrating the MIN_WIDTH_DECREASING_SEQUENCE constraint of the Example slot
Figure 4.916 depicts the automaton associated with the constraint MIN_WIDTH_DECREASING_SEQUENCE.

\[
\begin{align*}
C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \text{default}
\end{align*}
\]

Figure 4.916: Automaton for the MIN_WIDTH_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING_SEQUENCE pattern where default is |VARIABLES| + 1; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

\[
\begin{array}{c|c|c}
 s & t & \\
\hline
 s & \min(\overline{C}, \overline{C}) & \min(\overline{C}, \overline{C}) \\
 t & \min(\overline{C}, \overline{C}) & \overline{C} + \overline{C} + \overline{D} + \overline{D} - M
\end{array}
\]

Table 4.201: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the MIN_WIDTH_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### MIN_WIDTH_DECREASING_TERRACE

**DESCRIPTION**

**Origin**
Based on the **DECREASING_TERRACE** pattern.

**Constraint**

\[
\text{MIN_WIDTH_DECREASING_TERRACE}(\text{VALUE, VARIABLES})
\]

**Arguments**

- VALUE : \text{dvar}
- VARIABLES : \text{collection}(\text{var-dvar})

**Restrictions**

- \(sv \leq 3 \lor rv \leq 2 \Rightarrow \text{VALUE} = sv + 1\)
- \(\text{VALUE} \geq 2\)
- \(\text{VALUE} = sv + 1 \lor \text{VALUE} \leq sv - 2\)
- \text{required}(\text{VARIABLES, var})

where

- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range}(\text{VARIABLES, var})\)

**Purpose**

VALUE is the minimal width of occurrences of the **DECREASING_TERRACE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(|\text{VARIABLES}| + 1\).

An occurrence of the pattern **DECREASING_TERRACE** is the maximal subsequence which matches the regular expression ‘\(\geq \rightarrow\)’.

Assume that the occurrence of the pattern **DECREASING_TERRACE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i\).

**Example**

\((2, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3))\)

Figure 4.917 provides an example where the **MIN_WIDTH_DECREASING_TERRACE** \((2, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3])\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 3\)
\(\text{range}(\text{VARIABLES, var}) > 2\)

**Symmetry**

One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.917: Illustrating the MIN_WIDTH_DECREASING_TERRACE constraint of the Example slot
Automaton

Figures 4.918 and 4.919 respectively depict the automaton associated with the constraint \textit{MIN_WIDTH_DECREASING_TERRACE} and its simplified form.

\[
\begin{align*}
C \leftarrow & \text{default} \\
D \leftarrow & 0 \\
R \leftarrow & \text{default}
\end{align*}
\]

\[
\begin{align*}
\leq s & & \leq \\
\{ D \leftarrow D + 1 & & > \\
\{ D \leftarrow D + 1 & & > \\
\leq s & & > \\
\{ D \leftarrow 0 & & > \\
\{ R \leftarrow \min(R, D + 1) & & > \\
\}\}
\end{align*}
\]

Figure 4.918: Automaton for the \textit{MIN_WIDTH_DECREASING_TERRACE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textit{DECREASING_TERRACE} pattern where \textit{default} is $|\text{VARIABLES}| + 1$.

<table>
<thead>
<tr>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>

Table 4.202: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the \textit{MIN_WIDTH_DECREASING_TERRACE} constraint defined as the composition of the \textit{DECREASING_TERRACE} pattern, the feature \textit{WIDTH}, and the aggregator $\min$; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.919: Simplified automaton for the \textsc{min\_width\_decreasing\_terrace} constraint obtained by applying decoration Table 3.26 to the seed transducer of the \textsc{decreasing\_terrace} pattern where \texttt{default} is $|\text{VARIABLES}| + 1$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

Table 4.203: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the simplified automaton of the \textsc{min\_width\_decreasing\_terrace} constraint defined as the composition of the \textsc{decreasing\_terrace} pattern, the feature \textsc{width}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_WIDTH_GORGE

**AGGREGATOR**  
**FEATURE**  
**PATTERN**  

**MIN_WIDTH_GORGE**

**Origin**  
Based on the GORGE pattern.

**Constraint**  
MIN_WIDTH_GORGE(VALUE, VARIABLES)

**Arguments**  
VALUE : dvar  
VARIABLES : collection(var–dvar)

**Restrictions**  
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = sv + 1  
VALUE ≥ 1  
rv = 2 ⇒ VALUE = sv + 1 ∨ VALUE ≤ 10  
rv ≥ 3 ⇒ VALUE = sv + 1 ∨ VALUE ≤ sv − 2  
required(VARIABLES, var)  
where  
sv = |VARIABLES|  
rv = range(VARIABLES.var)

**Purpose**  
An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘(>| (> (|= >) >)(< | < (= | <) <)’.

Assume that the occurrence of the pattern GORGE starts at position i and ends at position j. The feature WIDTH computes the value j − i.

**Example**  
(1, (1, 7, 3, 4, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7))

Figure 4.920 provides an example where the MIN_WIDTH_GORGE (1, [1, 7, 3, 4, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7]) constraint holds.

**Typical**  
|VARIABLES| > 2  
range(VARIABLES.var) > 1

**Symmetries**  
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**  
Functional dependency: VALUE determined by VARIABLES.
Figure 4.920: Illustrating the MIN_WIDTH_GORGE constraint of the Example slot
Automaton

Figure 4.921 depicts the automaton associated with the constraint MIN_WIDTH_GORGE.

\[
\begin{align*}
\{ C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default} \} \\
\leq \\
S \\
\{ D \leftarrow D + 1 \} \\
> \\
\min(R, C) \\
\{ C \leftarrow D + 1 \} \\
D \leftarrow 0 \\
\{ \leq \} \\
> \\
\{ \leq \} \\
> \\
\{ \} \\
\{ D \leftarrow D + 1 \} \\
< \\
\{ D \leftarrow D + 1 \} \\
\{ D \leftarrow D + 1 \} \\
\{ \} \\
\leq \\
\leq u \\
\{ D \leftarrow D + 1 \} \\
\{ D \leftarrow D + 1 \} \\
\{ D \leftarrow 0 \} \\
\{ \} \\
\leq t \\
\{ \} \\
\end{align*}
\]

Figure 4.921: Automaton for the MIN_WIDTH_GORGE constraint obtained by applying decoration Table 3.37 to the seed transducer of the GORGE pattern where default is |VARIABLES| + 1 (transition \( u \rightarrow r \) has the same register update as transition \( r \rightarrow u \)); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
Table 4.204: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the `MIN_WIDTH_GORGE` constraint defined as the composition of the `GORGE` pattern, the `feature WIDTH`, and the `aggregator min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MIN_WIDTH_INCREASING_SEQUENCE**

**Origin**
Based on the **INCREASING_SEQUENCE** pattern.

**Constraint**
\[\text{MIN_WIDTH_INCREASING_SEQUENCE}(\text{VALUE, VARIABLES})\]

**Arguments**
- **VALUE**: \( \text{dvar} \)
- **VARIABLES**: \( \text{collection}(\text{var} - \text{dvar}) \)

**Restrictions**
- \( sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = sv + 1 \)
- \( \text{VALUE} \geq 2 \)
- \( rv = 2 \Rightarrow \text{VALUE} = sv + 1 \lor \text{VALUE} \leq 2 \)
- \( rv \geq 3 \Rightarrow \text{VALUE} = sv + 1 \lor \text{VALUE} \leq sv + \quad \text{required}(\text{VARIABLES, var}) \)

where
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**
An occurrence of the pattern **INCREASING_SEQUENCE** is the maximal subsequence which matches the regular expression ‘\(< (< | =)^* < | <\)’. Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **WIDTH** computes the value \( j - i + 2 \).

**Example**
\((2, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\)

Figure 4.922 provides an example where the **MIN_WIDTH_INCREASING_SEQUENCE** (2, \([4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]\)) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**
One and the same constant can be added to the \( \text{var} \) attribute of all items of **VARIABLES**.

**Arg. properties**
Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.922: Illustrating the MIN_WIDTH_INCREASING_SEQUENCE constraint of the Example slot
Automaton Figure 4.923 depicts the automaton associated with the constraint MIN_WIDTH_INCREMENTING_SEQUENCE.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default} \\
\end{align*}
\]

≥

≥

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \min(R, C) \\
\end{align*}
\]

≥

<

\[
\begin{align*}
C &\leftarrow D + 2 \\
D &\leftarrow 0 \\
\end{align*}
\]

<

≤

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \min(R, C) \\
\end{align*}
\]

\[
\begin{align*}
D &\leftarrow D + 1 \\
\end{align*}
\]

<

≥

\[
\begin{align*}
C &\leftarrow C + D + 1 \\
D &\leftarrow 0 \\
\end{align*}
\]

Figure 4.923: Automaton for the MIN_WIDTH_INCREMENTING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_SEQUENCE pattern where default is |VARIABLES| + 1; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

\[
\begin{array}{|c|c|}
\hline
s & t \\
\hline
\min(C, C) & \min(C, C) \\
\min(C, C) & C + C + D + D - 1 \\
\hline
\end{array}
\]

Table 4.205: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the MIN_WIDTH_INCREMENTING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MIN_WIDTH_INCREASING_TERRACE**

**Origin**
Based on the **INCREASING_TERRACE** pattern.

**Constraint**
\[ \text{MIN_WIDTH_INCREASING_TERRACE}(\text{VALUE, VARIABLES}) \]

**Arguments**
- **VALUE** : dvar
- **VARIABLES** : collection(var−dvar)

**Restrictions**
- \[ sv \leq 3 \lor rv \leq 2 \Rightarrow \text{VALUE} = sv + 1 \]
- \[ \text{VALUE} \geq 2 \]
- \[ \text{VALUE} = sv + 1 \lor \text{VALUE} \leq sv - 2 \]
- \[ \text{required}(\text{VARIABLES}, \text{var}) \]
  
  where
  - \( sv = |\text{VARIABLES}| \)
  - \( rv = \text{range}(\text{VARIABLES.var}) \)

**Purpose**
VALUE is the minimal width of occurrences of the **INCREASING_TERRACE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, VALUE takes the default value \(|\text{VARIABLES}| + 1\).

An occurrence of the pattern **INCREASING_TERRACE** is the maximal subsequence which matches the regular expression \('< = + <'\).

Assume that the occurrence of the pattern **INCREASING_TERRACE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i\).

**Example**
\[(2, (1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4))\]

Figure 4.924 provides an example where the **MIN_WIDTH_INCREASING_TERRACE** \((2, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4])\) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES.var}) > 2\)

**Symmetry**
One and the same constant can be added to the var attribute of all items of **VARIABLES**.

**Arg. properties**
- Functional dependency: VALUE determined by **VARIABLES**.
Figure 4.924: Illustrating the MIN_WIDTH_INCREASING_TERRACE constraint of the Example slot
Automaton

Figures 4.925 and 4.926 respectively depict the automaton associated with the constraint MIN_WIDTH_INCREASING_TERRACE and its simplified form.

Figure 4.925: Automaton for the MIN_WIDTH_INCREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_TERRACE pattern where default is |VARIABLES| + 1

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>\min(\vec{C}, \vec{C})</td>
<td>\min(\vec{C}, \vec{C})</td>
<td>\min(\vec{C}, \vec{C})</td>
</tr>
<tr>
<td>r</td>
<td>\min(\vec{C}, \vec{C})</td>
<td>\min(\vec{C}, \vec{C})</td>
<td>\overrightarrow{D} + \overrightarrow{D} + 1 \text{ e}</td>
</tr>
<tr>
<td>t</td>
<td>\min(\vec{C}, \vec{C})</td>
<td>\overrightarrow{D} + \overrightarrow{D} + 1 \text{ e}</td>
<td>\overrightarrow{D} + \overrightarrow{D} + 1 \text{ e}</td>
</tr>
</tbody>
</table>

Table 4.206: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the MIN_WIDTH_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature WIDTH, and the aggregator min ; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_WIDTH_INCREASING_TERRACE

Figure 4.926: Simplified automaton for the MIN_WIDTH_INCREASING_TERRACE constraint obtained by applying decoration Table 3.26 to the seed transducer of the INCREASING_TERRACE pattern where default is $|\text{VARIABLES}| + 1$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

Table 4.207: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the simplified automaton of the MIN_WIDTH_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**Origin**
Based on the `INFLEXION` pattern.

**Constraint**
\[
\text{MIN\_WIDTH\_INFLEXION}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**
- `VALUE` : `dvar`
- `VARIABLES` : `collection(var-dvar)`

**Restrictions**
- \[sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = sv + 1\]
- \[\text{VALUE} \geq 1\]
- \[\text{VALUE} = sv + 1 \lor \text{VALUE} \leq sv - 2\]
- \[\text{required}(\text{VARIABLES}, \text{var})\]

where
- \[sv = |\text{VARIABLES}|\]
- \[rv = \text{range}(\text{VARIABLES}.\text{var})\]

**Purpose**
An occurrence of the pattern `INFLEXION` is the maximal subsequence which matches the regular expression `'< (< | =) > | > (>| =)* <'`. Assume that the occurrence of the pattern `INFLEXION` starts at position \(i\) and ends at position \(j\). The feature `WIDTH` computes the value \(j - i\).

**Example**
\[(1, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))\]

Figure 4.927 provides an example where the `MIN\_WIDTH\_INFLEXION` (1, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4]) constraint holds.

**Typical**
- `|\text{VARIABLES}| > 2`
- `\text{range}(\text{VARIABLES}.\text{var}) > 1`

**Symmetries**
- Items of `VARIABLES` can be reversed.
- One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**
- Functional dependency: `VALUE` determined by `VARIABLES`.
Figure 4.927: Illustrating the MIN_WIDTH_INFLEXION constraint of the Example slot
Automaton

Figures 4.928 and 4.929 respectively depict the automaton associated with the constraint MIN_WIDTH_INFLEXION and its simplified form.

**Figure 4.928:** Automaton for the MIN_WIDTH_INFLEXION constraint obtained by applying decoration Table 3.37 to the seed transducer of the INFLEXION pattern where default is |VARIABLES| + 1 (transition r → t has the same registers updates as transition t → r).

**Figure 4.929:** Simplified automaton for the MIN_WIDTH_INFLEXION constraint obtained by applying decoration Table 3.26 to the seed transducer of the INFLEXION pattern where default is |VARIABLES| + 1 (transition r → t has the same registers updates as transition t → r); −R_i + R_{i-1} ≥ 0 is a linear invariant.
**MIN_WIDTH_PEAK**

**DESCRIPTION**

Origin

Based on the PEAK pattern.

Constraint

MIN_WIDTH_PEAK(VALUE, VARIABLES)

Arguments

| VALUE  | : dvar |
| VARIABLES | : collection(var−dvar) |

Restrictions

\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \]
\[ VALUE \geq 1 \]
\[ VALUE = sv + 1 \lor VALUE \leq sv - 2 \]

required(VARIABLES, var)

where

\[ sv = |VARIABLES| \]
\[ rv = \text{range}(VARIABLES,\var) \]

**Purpose**

VALUE is the minimal width of occurrences of the PEAK pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(|VARIABLES| + 1\).

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression '<< | | > | = > | >'.

Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\((2, (7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1))\)

Figure 4.930 provides an example where the MIN_WIDTH_PEAK \((2, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1])\) constraint holds.

**Typical**

\[ |VARIABLES| > 2 \]
\[ \text{range}(VARIABLES,\var) > 1 \]

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.930: Illustrating the MIN_WIDTH_PEAK constraint of the Example slot
Automaton

Figure 4.931 depicts the automaton associated with the constraint MIN_WIDTH_PEAK.

\[
\begin{align*}
\{ C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default} \} & \quad \geq s \quad \geq \\
\min(R, C) & < \quad < \\
\{ D \leftarrow D + 1 \\
D \leftarrow 0 \} & \quad \geq t \\
\{ C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \min(R, C) \} & \quad \leq \quad \leq \\
\{ D \leftarrow D + 1 \} & \quad = \quad = \\
\end{align*}
\]

Figure 4.931: Automaton for the MIN_WIDTH_PEAK constraint obtained by applying decoration Table 3.37 to the seed transducer of the PEAK pattern where \text{default} is |VARIABLES| + 1; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>(r)</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
<tr>
<td>(t)</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
</tbody>
</table>

Table 4.208: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the MIN_WIDTH_PEAK constraint defined as the composition of the PEAK pattern, the feature WIDTH, and the aggregator \(\min\); cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_WIDTH_PEAK

1905
MIN_WIDTH_PLAIN

**DESCRIPTION**

**Origin**
Based on the PLAIN pattern.

**Constraint**
MIN_WIDTH_PLAIN(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \)
- \( VALUE \geq 1 \)
- \( VALUE = sv + 1 \lor VALUE \leq sv - 2 \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)

where
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**
An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘>∗<’. Assume that the occurrence of the pattern PLAIN starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**
\( (1, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3)) \)

Figure 4.932 provides an example where the MIN_WIDTH_PLAIN (1, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.932: Illustrating the MIN_WIDTHPLAIN constraint of the Example slot
Figures 4.933 and 4.934 respectively depict the automaton associated with the constraint MIN_WIDTHPLAIN and its simplified form.

Figure 4.933: Automaton for the MIN_WIDTHPLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLAIN pattern where default is |VARIABLES| + 1

Figure 4.934: Simplified automaton for the MIN_WIDTHPLAIN constraint obtained by applying decoration Table 3.26 to the seed transducer of the PLAIN pattern where default is |VARIABLES| + 1; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.209: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the MIN_WIDTH_PLAIN constraint defined as the composition of the PLAIN pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>

Table 4.210: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the simplified automaton of the MIN_WIDTH_PLAIN constraint defined as the composition of the PLAIN pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$</td>
<td>\text{VARIABLES}</td>
<td>+ 1$</td>
</tr>
<tr>
<td>$r$</td>
<td>$</td>
<td>\text{VARIABLES}</td>
<td>+ 1$</td>
</tr>
<tr>
<td>$t$</td>
<td>$</td>
<td>\text{VARIABLES}</td>
<td>+ 1$</td>
</tr>
</tbody>
</table>
Origin Based on the PLATEAU pattern.

Constraint

\[
\text{MIN\_WIDTH\_PLATEAU}(\text{VALUE}, \text{VARIABLES})
\]

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = sv + 1 \\
\text{VALUE} & \geq 1 \\
\text{VALUE} & = sv + 1 \lor \text{VALUE} \leq sv - 2 \text{ required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}, \text{var})
\end{align*}
\]

Purpose

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression ‘< = * >’. Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example

\[(3, (1, 3, 3, 5, 5, 5, 2, 4, 4, 4, 3, 3, 1, 5, 5))\]

Figure 4.935 provides an example where the MIN\_WIDTH\_PLATEAU (3, [1, 3, 3, 5, 5, 5, 2, 4, 4, 4, 3, 3, 1, 5, 5]) constraint holds.

Typical

\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES}, \text{var}) & > 1
\end{align*}
\]

Symmetries

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties

- Functional dependency: \(\text{VALUE}\) determined by \(\text{VARIABLES}\).
Figure 4.935: Illustrating the MIN_WIDTH_PLATEAU constraint of the Example slot
Automaton

Figures 4.936 and 4.937 respectively depict the automaton associated with the constraint MIN_WIDTH_PLATEAU and its simplified form.

Figure 4.936: Automaton for the MIN_WIDTH_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLATEAU pattern where default is $|\text{VARIABLES}| + 1$.

Figure 4.937: Simplified automaton for the MIN_WIDTH_PLATEAU constraint obtained by applying decoration Table 3.26 to the seed transducer of the PLATEAU pattern where default is $|\text{VARIABLES}| + 1$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.211: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the `MIN_WIDTH_PLATEAU` constraint defined as the composition of the `PLATEAU` pattern, the feature `WIDTH`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>

Table 4.212: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the simplified automaton of the `MIN_WIDTH_PLATEAU` constraint defined as the composition of the `PLATEAU` pattern, the feature `WIDTH`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$</td>
<td>\text{VARIABLES}</td>
<td>+ 1$</td>
</tr>
<tr>
<td>$r$</td>
<td>$</td>
<td>\text{VARIABLES}</td>
<td>+ 1$</td>
</tr>
<tr>
<td>$t$</td>
<td>$</td>
<td>\text{VARIABLES}</td>
<td>+ 1$</td>
</tr>
</tbody>
</table>
**Origin**
Based on the `PROPER_PLAIN` pattern.

**Constraint**
`MIN_WIDTH_PROPER_PLAIN(VALUE, VARIABLES)`

**Arguments**
- `VALUE : dvar`
- `VARIABLES : collection(var−dvar)`

**Restrictions**
\[
\text{sv} \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = \text{sv} + 1 \\
\text{VALUE} \geq 2 \\
\text{VALUE} = \text{sv} + 1 \lor \text{VALUE} \leq \text{sv} - 2 \Rightarrow \text{required}(\text{VARIABLES}, \text{var})
\]
where
\[
\text{sv} = |\text{VARIABLES}| \\
\text{rv} = \text{range}(\text{VARIABLES}, \text{var})
\]

**Purpose**
An occurrence of the pattern `PROPER_PLAIN` is the maximal subsequence which matches the regular expression `'>=+ '<`.
Assume that the occurrence of the pattern `PROPER_PLAIN` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

**Example**
\[
(2, (2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5))
\]

Figure 4.938 provides an example where the `MIN_WIDTH_PROPER_PLAIN(2, [2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5])` constraint holds.

**Typical**
- `|\text{VARIABLES}| > 3`
- `\text{range}(\text{VARIABLES}, \text{var}) > 1`

**Symmetries**
- Items of `VARIABLES` can be reversed.
- One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**
Functional dependency: `VALUE` determined by `VARIABLES`.
Figure 4.938: Illustrating the MIN_WIDTH_PROPER_PLAIN constraint of the Example slot
Figures 4.939 and 4.940 respectively depict the automaton associated with the constraint MIN_WIDTH_PROPER_PLAIN and its simplified form.

Figure 4.939: Automaton for the MIN_WIDTH_PROPER_PLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER_PLAIN pattern where default is |VARIABLES| + 1

Figure 4.940: Simplified automaton for the MIN_WIDTH_PROPER_PLAIN constraint obtained by applying decoration Table 3.26 to the seed transducer of the PROPER_PLAIN pattern where default is |VARIABLES| + 1; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
### Table 4.213: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the `MIN_WIDTH_PROPERPLAIN` constraint defined as the composition of the `PROPERPLAIN` pattern, the feature `WIDTH`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>

### Table 4.214: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the simplified automaton of the `MIN_WIDTH_PROPERPLAIN` constraint defined as the composition of the `PROPERPLAIN` pattern, the feature `WIDTH`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\text{VARIABLES} + 1$</td>
<td>$\text{VARIABLES} + 1$</td>
<td>$\text{VARIABLES} + 1$</td>
</tr>
<tr>
<td>r</td>
<td>$\text{VARIABLES} + 1$</td>
<td>$\text{VARIABLES} + 1$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>$\text{VARIABLES} + 1$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>
MIN_WIDTH_PROPER_PLATEAU

**Origin**
Based on the PROPER_PLATEAU pattern.

**Constraint**
MIN_WIDTH_PROPER_PLATEAU(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
\[sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = sv + 1\]
VALUE \geq 2
VALUE \geq sv + 1 \lor VALUE \leq sv - 2 \Rightarrow
\text{required}(VARIABLES, var)

where
sv = |VARIABLES|
rv = \text{range}(VARIABLES.var)

**Purpose**
An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression ‘< =+ >’.
Assume that the occurrence of the pattern PROPER_PLATEAU starts at position i and ends at position j. The feature WIDTH computes the value \(j - i\).

**Example**
\((2, \langle 7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3 \rangle)\)

Figure 4.941 provides an example where the MIN_WIDTH_PROPER_PLATEAU (2, \([7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3]\)) constraint holds.

**Typical**
|VARIABLES| > 3
range(VARIABLES.var) > 1

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.941: Illustrating the MIN_WIDTH_PROPER_PLATEAU constraint of the Example slot
Figures 4.942 and 4.943 respectively depict the automaton associated with the constraint MIN_WIDTH_PROPER_PLATEAU and its simplified form.

Figure 4.942: Automaton for the MIN_WIDTH_PROPER_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER_PLATEAU pattern where default is $|\text{VAR}A\text{IBL}E| + 1$

Figure 4.943: Simplified automaton for the MIN_WIDTH_PROPER_PLATEAU constraint obtained by applying decoration Table 3.26 to the seed transducer of the PROPER_PLATEAU pattern where default is $|\text{VAR}A\text{IBL}E| + 1$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.215: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the MIN_WIDTH_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

Table 4.216: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the simplified automaton of the MIN_WIDTH_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### Origin
Based on the `STEADY_SEQUENCE` pattern.

### Constraint
```
MIN_WIDTH_STEADY_SEQUENCE(VALUE, VARIABLES)
```

### Arguments
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

### Restrictions
- `sv ≤ 1 ⇒ VALUE = sv + 1`
- `rv = 1 ⇒ VALUE ≥ sv`
- `rv ≥ 2 ⇒ VALUE ≥ 2`
- `VALUE = sv + 1 ∨ VALUE ≤ sv
  required(VARIABLES,var)`
- where
  - `rv = range(VARIABLES.var)`
  - `sv = |VARIABLES|`

### Purpose
An occurrence of the pattern `STEADY_SEQUENCE` is the *maximal* subsequence which matches the regular expression `'[=+*]'`. Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

### Example
```
(2, ⟨3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1⟩)
```

Figure 4.944 provides an example where the `MIN_WIDTH_STEADY_SEQUENCE (2, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1])` constraint holds.

### Typical
```
|VARIABLES| > 1
```

### Symmetries
- Items of `VARIABLES` can be reversed.
- One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

### Arg. properties
- **Functional dependency**: `VALUE` determined by `VARIABLES`. 
Figure 4.944: Illustrating the MIN_WIDTH_STEADY_SEQUENCE constraint of the Example slot
Automaton

Figures 4.945 and 4.946 respectively depict the automaton associated with the constraint MIN_WIDTH_STEADY_SEQUENCE and its simplified form.

Figure 4.945: Automaton for the MIN_WIDTH_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STEADY_SEQUENCE pattern where default is |VARIABLES| + 1

Figure 4.946: Simplified automaton for the MIN_WIDTH_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.25 to the seed transducer of the STEADY_SEQUENCE pattern where default is |VARIABLES| + 1; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 4.217: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the MIN\_WIDTH\_STEADY\_SEQUENCE constraint defined as the composition of the STEADY\_SEQUENCE pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\min(\vec{C}, \vec{C}))</td>
<td>(\min(\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\min(\vec{C}, \vec{C}))</td>
<td>(\vec{C} + \vec{C} + \vec{D} + \vec{D} - 1)</td>
</tr>
</tbody>
</table>

Table 4.218: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the simplified automaton of the MIN\_WIDTH\_STEADY\_SEQUENCE constraint defined as the composition of the STEADY\_SEQUENCE pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\min(\vec{C}, \vec{C}))</td>
<td>(\min(\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\min(\vec{C}, \vec{C}))</td>
<td>(\vec{C} + \vec{C} - 1)</td>
</tr>
</tbody>
</table>
MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**

Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**

\[
\text{MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} &: \text{dvar} \\
\text{VARIABLES} &: \text{collection}(	ext{var} \rightarrow \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
sv &\leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = sv + 1 \\
\text{VALUE} &\geq 2 \\
\text{VALUE} &= sv + 1 \lor \text{VALUE} \leq \min(sv, rv) \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `>\+`.

Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `WIDTH` computes the value \(j - i + 2\).

**Example**

\[(2, 4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3)\]

Figure 4.947 provides an example where the `MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE` (2, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| &> 1 \\
\text{range}(\text{VARIABLES}.\text{var}) &> 1
\end{align*}
\]

**Symmetry**

One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`.  

Figure 4.947: Illustrating the MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.948 and 4.949 respectively depict the automaton associated with the constraint MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE and its simplified form.

Figure 4.948: Automaton for the MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is |VARIABLES| + 1

Figure 4.949: Simplified automaton for the MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.25 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is |VARIABLES| + 1; \(-R_t + R_{t-1} \geq 0\) is a linear invariant.
Table 4.219: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\min(\overrightarrow{C},\overleftarrow{C})$</td>
<td>$\min(\overrightarrow{C},\overleftarrow{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\min(\overrightarrow{C},\overleftarrow{C})$</td>
<td>$\overrightarrow{C} + \overleftarrow{C} + D + D - 1$</td>
</tr>
</tbody>
</table>

Table 4.220: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\min(\overrightarrow{C},\overleftarrow{C})$</td>
<td>$\min(\overrightarrow{C},\overleftarrow{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\min(\overrightarrow{C},\overleftarrow{C})$</td>
<td>$\overrightarrow{C} + \overleftarrow{C} - 1$</td>
</tr>
</tbody>
</table>
### MIN\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
</table>

#### Origin
Based on the `STRICTLY\_INCREASING\_SEQUENCE` pattern.

#### Constraint

\[
\text{MIN\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

#### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td><code>dvar</code></td>
</tr>
<tr>
<td>VARIABLES</td>
<td><code>collection(var\_\_dvar)</code></td>
</tr>
</tbody>
</table>

#### Restrictions

\[
\begin{align*}
\text{sv} \leq 1 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = \text{sv} + 1 \\
\text{VALUE} \geq 2 & \Rightarrow \text{VALUE} = \text{sv} + 1 \lor \text{VALUE} \leq \min(\text{sv}, \text{rv}) \\
\text{required}(\text{VARIABLES}, \text{var}) & \text{where} \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}, \text{var})
\end{align*}
\]

#### Purpose

VALUE is the minimal width of occurrences of the `STRICTLY\_INCREASING\_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $|\text{VARIABLES}| + 1$.

An occurrence of the pattern `STRICTLY\_INCREASING\_SEQUENCE` is the maximal sub-sequence which matches the regular expression '<\+

Assume that the occurrence of the pattern `STRICTLY\_INCREASING\_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `WIDTH` computes the value $j - i + 2$.

#### Example

\[
(2, 4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3)
\]

Figure 4.950 provides an example where the `MIN\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE` constraint holds.

#### Typical

$|\text{VARIABLES}| > 1$

$\text{range}(\text{VARIABLES}, \text{var}) > 1$

#### Symmetry
One and the same constant can be added to the `var` attribute of all items of VARIABLES.

#### Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.950: Illustrating the `MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint of the **Example** slot
Figures 4.951 and 4.952 respectively depict the automaton associated with the constraint MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE and its simplified form.

Figure 4.951: Automaton for the MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is |VARIABLES| + 1

Figure 4.952: Simplified automaton for the MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.25 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is |VARIABLES| + 1; $-R_t + R_{t-1} \geq 0$ is a linear invariant.
Table 4.221: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the \texttt{MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE} constraint defined as the composition of the \texttt{STRICTLY_INCREASING_SEQUENCE} pattern, the feature \texttt{WIDTH}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

Table 4.222: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the simplified automaton of the \texttt{MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE} constraint defined as the composition of the \texttt{STRICTLY_INCREASING_SEQUENCE} pattern, the feature \texttt{WIDTH}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**MIN_WIDTH_SUMMIT**

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the SUMMIT pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>$\text{MIN_WIDTH_SUMMIT}(\text{VALUE, VARIABLES})$</td>
</tr>
<tr>
<td><strong>Arguments</strong></td>
<td></td>
</tr>
</tbody>
</table>
VALUE : $dvar$  
VARIABLES : $\text{collection}(\text{var} - dvar)$ |
| **Restrictions** |  
$sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = sv + 1$
$\text{VALUE} \geq 1$
$rv = 2 \Rightarrow \text{VALUE} = sv + 1 \lor \text{VALUE} \leq 1$
$rv \geq 3 \Rightarrow \text{VALUE} = sv + 1 \lor \text{VALUE} \leq sv - 2$

required(VARIABLES, var)  
where
$sv = |\text{VARIABLES}|$
$rv = \text{range}(\text{VARIABLES}.\text{var})$ |
| **Purpose** | An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression ‘\((< | < | >= | < | < | < | > | > | > | > | > | >>)\)’.  
Assume that the occurrence of the pattern SUMMIT starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$. |
| **Example** | $(1, (7, 1, 5, 4, 3, 4, 6, 2, 3, 4, 2, 3, 1))$ |
| **Typical** | $|\text{VARIABLES}| > 2$
range(\text{VARIABLES}.\text{var}) > 1 |
| **Symmetries** |  
- Items of VARIABLES can be reversed.  
- One and the same constant can be added to the var attribute of all items of VARIABLES. |
| **Arg. properties** | Functional dependency: VALUE determined by VARIABLES. |
MIN_WIDTH_SUMMIT

Figure 4.953: Illustrating the MIN_WIDTH_SUMMIT constraint of the Example slot
Automaton

Figure 4.954 depicts the automaton associated with the constraint MIN_WIDTH_SUMMIT.

Figure 4.954: Automaton for the MIN_WIDTH_SUMMIT constraint obtained by applying decoration Table 3.37 to the seed transducer of the SUMMIT pattern where default is |VARIABLES| + 1 (transition u → r has the same register update as transition r → u); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
Table 4.223: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the MIN_WIDTH_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN WIDTH VALLEY

AGGREGATOR → FEATURE → PATTERN ↑

MIN WIDTH VALLEY

DESCRIPTION AUTOMATON

Origin
Based on the VALLEY pattern.

Constraint
MIN_WIDTH_VALLEY(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv \leq 2 \land rv \leq 1 \Rightarrow VALUE = sv + 1
VALUE \geq 1
VALUE = sv + 1 \lor VALUE \leq sv − 2 \land
required(VARIABLES, var)
where
sv = |VARIABLES|
rv = range(VARIABLES.var)

VALUE is the minimal width of occurrences of the VALLEY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value |VARIABLES| + 1.

Purpose
An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression ‘> (= | >) (< | =)* <’.
Assume that the occurrence of the pattern VALLEY starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j − i$.

Example
(2, (1,3,7,4,3,6,6,5,3,3,2,6,5,5,7))

Figure 4.955 provides an example where the MIN_WIDTH_VALLEY (2, [1,3,7,4,3,6,6,5,3,3,2,6,5,5,7]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1

Symmetries
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.955: Illustrating the MIN_WIDTH_VALLEY constraint of the Example slot
Automaton

Figure 4.956 depicts the automaton associated with the constraint MIN_WIDTH_VALLEY.

Figure 4.956: Automaton for the MIN_WIDTH_VALLEY constraint obtained by applying decoration Table 3.37 to the seed transducer of the VALLEY pattern where default is |VARIABLES| + 1; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$r$</td>
<td>$t$</td>
</tr>
<tr>
<td>$s$</td>
<td>min($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
<td>min($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
</tr>
<tr>
<td>$r$</td>
<td>min($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>$t$</td>
<td>min($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>

Table 4.224: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the MIN_WIDTH_VALLEY constraint defined as the composition of the VALLEY pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_WIDTH_VALLEY

1941
MIN_WIDTH_ZIGZAG

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ ▶ ▶ ▶</td>
<td>(&lt;&gt;)+(&lt;</td>
</tr>
</tbody>
</table>

Origin
Based on the ZIGZAG pattern.

Constraint
MIN_WIDTH_ZIGZAG(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = sv + 1
VALUE ≥ 2
VALUE = sv + 1 ∨ VALUE ≤ sv − 2

where
sv = |VARIABLES|
rv = range(VARIABLES.var)

VALUE is the minimal width of occurrences of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value |VARIABLES| + 1.

Purpose
An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘(<>)+(< | <>)| (>)+(| | <>)’.
Assume that the occurrence of the pattern ZIGZAG starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example
(2, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))

Figure 4.957 provides an example where the MIN_WIDTH_ZIGZAG (2, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1]) constraint holds.

Typical
| VARIABLES| > 3
| range(VARIABLES.var) | > 1

Symmetries
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.957: Illustrating the MIN_WIDTH_ZIGZAG constraint of the Example slot
Automaton

Figures 4.958 and 4.959 respectively depict the automaton associated with the constraint MIN_WIDTH_ZIGZAG and its simplified form.
Figure 4.958: Automaton for the MIN_WIDTH_ZIGZAG constraint obtained by applying decoration Table 3.37 to the seed transducer of the ZIGZAG pattern where default is |VARIABLES| + 1; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the register D is reset to its initial value; (3) on transitions from c, f to s the register R is updated wrt C and the register C is reset to its initial value.
Figure 4.959: Simplified automaton for the MIN_WIDTH_ZIGZAG constraint obtained by applying decoration Table 3.24 to the seed transducer of the ZIGZAG pattern where default is |VARIABLES| + 1; missing transitions from a, b, c, d, e, f to s are labelled by =: (2) on transitions from b, c, e, f to s the register D is reset to its initial value; (3) on transitions from c, f to s the register R is updated wrt C and the register C is reset to its initial value.; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
Table 4.225: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the MIN_WIDTH_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.26: Concrete glue matrix, derived from the parameterised glue matrix 3.22, for the simplified automaton of the MIN-\_WIDTH-\_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature Width, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
</tr>
<tr>
<td>a</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
</tr>
<tr>
<td>b</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>(\bar{B} + B + 1)</td>
<td>min((C, \bar{C}))</td>
<td>(\bar{B} + B + 1)</td>
<td>min((C, \bar{C}))</td>
</tr>
<tr>
<td>c</td>
<td>min((C, \bar{C}))</td>
<td>(\bar{C} + B + B + 1)</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>(\bar{C} + B + B + 1)</td>
<td>min((C, \bar{C}))</td>
</tr>
<tr>
<td>d</td>
<td>min((C, \bar{C}))</td>
<td>(\bar{B} + B + 1)</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>(\bar{B} + B + 1)</td>
<td>min((C, \bar{C}))</td>
</tr>
<tr>
<td>e</td>
<td>min((C, \bar{C}))</td>
<td>(\bar{C} + B + B + 1)</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>(\bar{C} + B + B + 1)</td>
<td>min((C, \bar{C}))</td>
</tr>
<tr>
<td>f</td>
<td>min((C, \bar{C}))</td>
<td>(\bar{C} + B + B + 1)</td>
<td>min((C, \bar{C}))</td>
<td>min((C, \bar{C}))</td>
<td>(\bar{C} + B + B + 1)</td>
<td>min((C, \bar{C}))</td>
</tr>
</tbody>
</table>
MIN_ZIGZAG

**Origin**  
Based on the ZIGZAG pattern.

**Constraint**  
MIN_ZIGZAG(VARIABLES, FEATURES, DEFAULT)

**Arguments**  
| VARIABLES | collection(var–dvar) |
| FEATURES  | collection(var–dvar) |
| DEFAULT   | int |

**Restrictions**  
- required(VARIABLES.var)  
- required(FEATURES.var)  
- sv = |FEATURES|  
- sv ≤ 3 ∨ rv ≤ 1 ⇒ FEATURES.var = DEFAULT  
- FEATURES.var = DEFAULT ∨ FEATURES.var ≥ minv  
- FEATURES.var = DEFAULT ∨ FEATURES.var ≤ maxv − 1  
- DEFAULT < minv ∨ DEFAULT > maxv − 1  

where  
- maxv = maxval(VARIABLES.var)  
- minv = minval(VARIABLES.var)  
- sv = |VARIABLES|  
- rv = range(VARIABLES.var)

**Purpose**  
Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of ZIGZAG is identified (even if this occurrence of pattern is not complete) then FEATURES[$i$] is the default value DEFAULT; otherwise FEATURES[$i$] gives the feature value of the corresponding occurrence of ZIGZAG.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘(<>)$^{*}$(< | <>)$|$ (>><)$|$ (> | >>)’.

Assume that the occurrence of the pattern ZIGZAG starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

**Example**  
Figure 4.960 provides an example where the MIN_ZIGZAG ([4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0], 0) constraint holds.

**Typical**  
- $|VARIABLES| > 3$  
- range(VARIABLES.var) > 1

**Arg. properties**  
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.960: Illustrating the MIN_ZIGZAG constraint of the Example slot.
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
NB_BUMP_ON_DECREASING_SEQUENCE

**Pattern**

**NB_BUMP_ON_DECREASING_SEQUENCE**

- **Description Automaton**

**Origin**
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**
NB_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
- \( sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = 0 \)
- \( VALUE \geq 0 \)
- \( VALUE \leq \max(0, \lfloor (sv - 3)/3 \rfloor) \)
- \( required(VARIABLES, var) \)

where
- \( sv = |VARIABLES| \)
- \( rv = \text{range}(VARIABLES, var) \)

**Purpose**
VALUE is the number of occurrences of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression ‘>>><’.

**Example**
(2, (7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3))

Figure 4.961 provides an example where the NB_BUMP_ON_DECREASING_SEQUENCE constraint holds.

**Typical**
\( |VARIABLES| > 5 \)
\( \text{range}(VARIABLES, var) > 2 \)

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.961: Illustrating the NB_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.962 and 4.963 respectively depict the automaton associated with the constraint `NB_BUMP_ON_DECREASING_SEQUENCE` and its simplified form.

Figure 4.962: Automaton for the `NB_BUMP_ON_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `BUMP_ON_DECREASING_SEQUENCE` pattern where `default` is 0.
Figure 4.963: Simplified automaton for the \texttt{NB\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-3} + 1 \geq 0 \) are linear invariants.
Specialisation

Figure 4.964: Automaton without registers for the NB_BUMP_ON_DECREASINGSEQUENCE_EQ_0 constraint; it describes all sequences containing no occurrence of the BUMP_ON_DECREASINGSEQUENCE pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the BUMP_ON_DECREASINGSEQUENCE pattern by removing the register $R$ and the found transition, i.e. the transition from state $v$ to state $t$ that increments $R$. 
Figure 4.965: (top) Automaton without registers for the `NB_BUMP_ON_DECREASING_SEQUENCE_EQ_UP` constraint; it describes all sequences containing the maximum number of occurrences of the `BUMP_ON_DECREASING_SEQUENCE` pattern on a sequence of `sv` variables, i.e. \( \max(0, \left\lfloor \frac{sv - 3}{3} \right\rfloor) \) of the `Restrictions` slot (see 4); transitions in blue correspond to a new occurrence of the `BUMP_ON_DECREASING_SEQUENCE` pattern, dashed transitions correspond to slack, and accepting states have a light-brown background; states `s`, `u`, `t'`, `e'`, `r''` are accepting when `sv mod 3 = 1`, states `r`, `v`, `s'`, `u'`, `t''`, `e''` are accepting when `sv mod 3 = 2`, states `t`, `r'`, `s''` are accepting when `sv mod 3 = 0`. (bottom) All corresponding solutions for `sv - 1 \in \{5, 6, 7\}`.
**Description**

Based on the **DECREASING** pattern.

**Constraint**

```
NB_DECREASING(VALUE, VARIABLES)
```

**Arguments**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0
\]
\[
VALUE \geq 0
\]
\[
VALUE \leq \max(0, (\ell - 1) * \lfloor sv/\ell \rfloor + \max(0, sv \mod \ell - 1))
\]
\[
\text{required}(\text{VARIABLES}.\text{var})
\]

**Purpose**

VALUE is the number of occurrences of the **DECREASING** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression ‘>’.

**Example**

```
(5, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))
```

Figure 4.966 provides an example where the **NB_DECREASING** (5, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]
\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**

One and the same constant can be added to the **var** attribute of all items of **VARIABLES**.

**Arg. properties**

Functional dependency: VALUE determined by **VARIABLES**.
Figure 4.966: Illustrating the NB_DECREASING constraint of the Example slot
Figures 4.967 and 4.968 respectively depict the automaton associated with the constraint \texttt{NB\textunderscore DECREASING} and its simplified form.

Figure 4.967: Automaton for the \texttt{NB\textunderscore DECREASING} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{DECREASING} pattern where \texttt{default} is 0.

Figure 4.968: Simplified automaton for the \texttt{NB\textunderscore DECREASING} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \texttt{DECREASING} pattern where \texttt{default} is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + 1 \geq 0$ are linear invariants.

Table 4.227: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the \texttt{NB\textunderscore DECREASING} constraint defined as the composition of the \texttt{DECREASING} pattern, the feature \texttt{ONE}, and the aggregator \texttt{sum}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.228: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the simplified automaton of the NB_DECREASING constraint defined as the composition of the DECREASING pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Specialisation

Figure 4.969: Automaton without registers for the $\text{NB}_{\text{DECREASING}}_\text{EQ}_0$ constraint; it describes all sequences containing no occurrence of the DECREASING pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the DECREASING pattern by removing the register $R$ and the found transition, i.e. the transition from state $s$ to state $s$ that increments $R$.

Figure 4.970: Automata without registers for the (A) $\text{NB}_{\text{DECREASING}}_\text{IS}_\text{EVEN}$ and the (B) $\text{NB}_{\text{DECREASING}}_\text{IS}_\text{ODD}$ constraints; they respectively achieve an even/odd number of occurrences of the DECREASING pattern on a sequence of $n$ variables; transitions in blue correspond to a new occurrence of the DECREASING pattern.
Origin: Based on the DECREASING_SEQUENCE pattern.

Constraint: \( \text{NB\_DECREASING\_SEQUENCE}(\text{VALUE, VARIABLES}) \)

Arguments:
- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection(\text{var}-\text{dvar})} \)

Restrictions:
- \( \text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \)
- \( \text{VALUE} \geq 0 \)
- \( \text{VALUE} \leq \lfloor \text{sv}/2 \rfloor \)
- \( \text{VALUE} \leq 6 \)

Purpose:
\( \text{VALUE} \) is the number of occurrences of the DECREASING_SEQUENCE pattern in the time-series given by the \( \text{VARIABLES} \) collection. If the pattern does not occur, \( \text{VALUE} \) takes the default value 0.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\( (> | =)^* > | > \)’.

Example: \((3, (3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 4, 6, 4, 4))\)

Figure 4.971 provides an example where the \( \text{NB\_DECREASING\_SEQUENCE} \) (3, \( [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 4, 6, 4, 4] \)) constraint holds.

Typical:
- \( |\text{VARIABLES}| > 1 \)
- \( \text{range(\text{VARIABLES.var})} > 1 \)

Symmetry: One and the same constant can be added to the \( \text{var} \) attribute of all items of \( \text{VARIABLES} \).

Arg. properties: Functional dependency: \( \text{VALUE} \) determined by \( \text{VARIABLES} \).
Figure 4.971: Illustrating the NB\_DECREASING\_SEQUENCE constraint of the Example slot
Figures 4.972 and 4.973 respectively depict the automaton associated with the constraint \texttt{NB\_DECREASING\_SEQUENCE} and its simplified form.

```plaintext
\begin{align*}
\{ C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \text{default} \}
\end{align*}
\begin{align*}
\leq s & \leq \\
\{ C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \text{default} \}
\end{align*}
\begin{align*}
\{ C & \leftarrow \max(D, 1) \\
D & \leftarrow 0 \}
\end{align*}
\begin{align*}
\leq t & \geq
\{ D & \leftarrow \max(D, 1) \}
\end{align*}
\begin{align*}
\{ C & \leftarrow \max(C, \max(D, 1)) \\
D & \leftarrow 0 \}
\end{align*}
\begin{align*}
\leq & \\
\{ D & \leftarrow \max(D, 1) \}
\end{align*}
\begin{align*}
\leq & \\
\{ R & \leftarrow \text{default} \}
\end{align*}
\begin{align*}
\leq & \\
\{ R & \leftarrow \text{default} \}
\end{align*}
\begin{align*}
\leq & \\
\{ R & \leftarrow \text{default} \}
\end{align*}
\begin{align*}
\leq & \\
\{ R & \leftarrow \text{default} \}
\end{align*}
\begin{align*}
\leq & \\
\{ R & \leftarrow \text{default} \}
\end{align*}
```

Figure 4.972: Automaton for the \texttt{NB\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{DECREASING\_SEQUENCE} pattern where \texttt{default} is 0.

Figure 4.973: Simplified automaton for the \texttt{NB\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \texttt{DECREASING\_SEQUENCE} pattern where \texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-2} + 1 \geq 0 \) are linear invariants.
Table 4.229: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the \textit{NB\_DECREASING\_SEQUENCE} constraint defined as the composition of the \textit{DECREASING\_SEQUENCE} pattern, the feature \textit{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{center}
\begin{tabular}{|c|c|}
\hline
$s$ & $t$ \\
\hline
$s$ & $\overline{C} + \overline{C}$ \\
$t$ & $\overline{C} + \overline{C}$ & 1 \\
\hline
\end{tabular}
\end{center}

Table 4.230: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the simplified automaton of the \textit{NB\_DECREASING\_SEQUENCE} constraint defined as the composition of the \textit{DECREASING\_SEQUENCE} pattern, the feature \textit{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{center}
\begin{tabular}{|c|c|}
\hline
$s$ & $t$ \\
\hline
$s$ & 0 & 0 \\
$t$ & 0 & $-1$ \\
\hline
\end{tabular}
\end{center}
Specialisation

Figure 4.974: Automaton without registers for the \texttt{NB\_DECREASING\_SEQUENCE\_EQ\_0} constraint; it describes all sequences containing no occurrence of the \texttt{DECREASING\_SEQUENCE} pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the \texttt{DECREASING\_SEQUENCE} pattern by removing the register $R$, the found transition from state $s$ to $t$ that increments $R$, and the state $t$ that becomes unreachable after removing transition $s \rightarrow t$.

Figure 4.975: (left) Automaton without registers for the \texttt{NB\_DECREASING\_SEQUENCE\_EQ\_UP} constraint; it describes all sequences containing the maximum number of occurrences of the \texttt{DECREASING\_SEQUENCE} pattern on a sequence of $\texttt{sv}$ variables, i.e. $\lfloor \frac{\text{sv}}{2} \rfloor$ of the \texttt{Restrictions} slot (see (i)); transitions in blue correspond to a new occurrence of the \texttt{DECREASING\_SEQUENCE} pattern, dashed transitions to slack, and accepting states have a light-brown background; state $t$ is accepting when $\texttt{sv} \mod 2 = 0$, while states $s$ and $t'$ are accepting when $\texttt{sv} \mod 2 = 1$. (right) All corresponding solutions for $\texttt{sv} - 1 \in \{5, 6\}$.
Figure 4.976: Automata without registers for the (A) \text{NB\_DECREASING\_SEQUENCE\_IS\_EVEN} and the (B) \text{NB\_DECREASING\_SEQUENCE\_IS\_ODD} constraints; they respectively achieve an even/odd number of occurrences of the \text{DECREASING\_SEQUENCE} pattern on a sequence of \( n \) variables; transitions in blue correspond to a new occurrence of the \text{DECREASING\_SEQUENCE} pattern.
DECREASING_TERRACE

DESCRIPTION

Origin
Based on the DECREASING_TERRACE pattern.

Constraint
NB_DECREASING_TERRACE(VALUE, VARIABLES)

Arguments
VALUE: dvar
VARIABLES: collection(var−dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 2 ⇒ VALUE = 0
VALUE ≥ 0
VALUE ≤ max(0, (ℓ − 2)/2 + |sv/ℓ| + max(0, |sv mod ℓ − 2|/2))
where
sv = |VARIABLES|
rv = range(VARIABLES.var)
ℓ = max(1, min(sv, 2 × rv − 2))

Purpose
VALUE is the number of occurrences of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.
An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression ‘> = +’.

Example
(2, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3))

Figure 4.977 provides an example where the NB_DECREASING_TERRACE (2, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3]) constraint holds.

Typical
|VARIABLES| > 3
range(VARIABLES.var) > 2

Symmetry
One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.977: Illustrating the \text{NB\_DECREASING\_TERRACE} constraint of the \text{Example} slot
**Automaton**

Figures 4.978 and 4.979 respectively depict the automaton associated with the constraint \( \text{NB\_DECREASING\_TERRACE} \) and its simplified form.

![Automaton diagram](image)

Figure 4.978: Automaton for the \( \text{NB\_DECREASING\_TERRACE} \) constraint obtained by applying decoration Table 3.37 to the seed transducer of the \( \text{DECREASING\_TERRACE} \) pattern where \( \text{default} \) is 0.

![Simplified automaton diagram](image)

Figure 4.979: Simplified automaton for the \( \text{NB\_DECREASING\_TERRACE} \) constraint obtained by applying decoration Table 3.39 to the seed transducer of the \( \text{DECREASING\_TERRACE} \) pattern where \( \text{default} \) is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-2} + 1 \geq 0 \) are linear invariants.
Table 4.231: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the **NB_DECREASING_TERRACE** constraint defined as the composition of the **DECREASING_TERRACE** pattern, the feature **ONE**, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
<td>1</td>
</tr>
<tr>
<td>t</td>
<td>(\overline{C} + \overline{C})</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4.232: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the simplified automaton of the **NB_DECREASING_TERRACE** constraint defined as the composition of the **DECREASING_TERRACE** pattern, the feature **ONE**, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>1</td>
<td><strong>C</strong></td>
</tr>
</tbody>
</table>
Specialisation

Figure 4.980: Automaton without registers for the NB_DECREASING_TERRACE_EQ_0 constraint; it describes all sequences containing no occurrence of the DECREASING_TERRACE pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the DECREASING_TERRACE pattern by removing the register $R$ and the found transition, i.e. the transition from state $t$ to state $r$ that increments $R$.

Figure 4.981: (left) Automaton without registers for the NB_DECREASING_TERRACE_EQ_UP constraint; it describes all sequences containing the maximum number of occurrences of the DECREASING_TERRACE pattern on a sequence of $sv$ variables, assuming no restriction on the domain of the variables, i.e. the simplified bound $\max(0, \lfloor \frac{sv - 2}{2} \rfloor)$ of the Restrictions slot (see (i)); transitions in blue correspond to a new occurrence of the DECREASING_TERRACE pattern, dashed transitions to slack, and accepting states have a light-brown background; states $r$ and $s'$ are accepting when $sv \mod 2 = 0$, states $s$, $t$, $r'$ and $s''$ are accepting when $sv \mod 2 = 1$. (right) All corresponding solutions for $sv - 1 \in \{5, 6\}$.
Figure 4.982: Automata without registers for the (A) `NB_DECREASING_TERRACE_IS_EVEN` and the (B) `NB_DECREASING_TERRACE_IS_ODD` constraints; they respectively achieve an even/odd number of occurrences of the `DECREASING_TERRACE` pattern on a sequence of `n` variables; transitions in blue correspond to a new occurrence of the `DECREASING_TERRACE` pattern.
**Origin**
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**
\[
\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**
- \text{VALUE} : dvar
- \text{VARIABLES} : collection(var−dvar)

**Restrictions**
- \(sv \leq 5 \lor rv \leq 2 \Rightarrow \text{VALUE} = 0\)
- \(\text{VALUE} \geq 0\)
- \(\text{VALUE} \leq \max(0, \lfloor (sv - 3)/3 \rfloor)\)
- \(\text{required}(\text{VARIABLES, var})\)

where
- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range}(\text{VARIABLES.var})\)

**Purpose**
VALUE is the number of occurrences of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression ‘<<><<<’.

**Example**
\[(2, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))\]

Figure 4.983 provides an example where the NB_DIP_ON_INCREASING_SEQUENCE (2, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4]) constraint holds.

**Typical**
\[|\text{VARIABLES}| > 5\]
\[\text{range}(\text{VARIABLES.var}) > 2\]

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.983: Illustrating the NB_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Figures 4.984 and 4.985 respectively depict the automaton associated with the constraint \texttt{NB\_DIP\_ON\_INCREASING\_SEQUENCE} and its simplified form.

Figure 4.984: Automaton for the \texttt{NB\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern where \texttt{default} is 0
Figure 4.985: Simplified automaton for the \texttt{NB\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern where \texttt{default} is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-3} + 1 \geq 0$ are linear invariants.
Specialisation

Figure 4.986: Automaton without registers for the `NB_DIP_ON_INCREASINGSEQUENCE` constraint; it describes all sequences containing no occurrence of the `DIP_ON_INCREASINGSEQUENCE` pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the `DIP_ON_INCREASINGSEQUENCE` pattern by removing the register $R$ and the `found` transition, i.e. the transition from state $v$ to state $t$ that increments $R$. 
Figure 4.987: (top) Automaton without registers for the NB_DIP_ON_INCREASING_SEQUENCE constraint; it describes all sequences containing the maximum number of occurrences of the DIP_ON_INCREASING_SEQUENCE pattern on a sequence of sv variables, i.e. max(0, ⌊sv−3/3⌋) of the Restrictions slot (see 4); transitions in blue correspond to a new occurrence of the DIP_ON_INCREASING_SEQUENCE pattern, dashed transitions correspond to slack, and accepting states have a light-brown background; states s, u, t', e', r'' are accepting when sv mod 3 = 1, states r, v, s', u', t'', e'' are accepting when sv mod 3 = 2, states t, r', s'' are accepting when sv mod 3 = 0. (bottom) All corresponding solutions for sv − 1 ∈ \{5, 6, 7\}. 
DESCRIPTION AUTOMATON

Origin
Based on the GORGE pattern.

Constraint
\( \text{NB\_GORGE(VALE, VARIABLES)} \)

Arguments
\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var\_dvar)}
\end{align*}
\]

Restrictions
\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \max(0, \lfloor (sv - 1)/2 \rfloor x) \\
\text{required(VARIABLES, var)} & \\
\text{where} & \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range(VARIABLES.var)}
\end{align*}
\]

Purpose
\( \text{VALUE} \) is the number of occurrences of the GORGE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \( \text{VALUE} \) takes the default value 0.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘\( (> | > (= | >)^* > | < | < (= | <)^* <) \)’.

Example
\((3, (1, 7, 3, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7))\)

Figure 4.988 provides an example where the \( \text{NB\_GORGE} \)
\((3, [1, 7, 3, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7])\) constraint holds.

Typical
\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range(VARIABLES.var)} & > 1
\end{align*}
\]

Symmetries
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: \( \text{VALUE} \) determined by VARIABLES.
Figure 4.988: Illustrating the NB_GORGE constraint of the Example slot
Automaton

Figures 4.989 and 4.990 respectively depict the automaton associated with the constraint NB_GORGE and its simplified form.

\[
\begin{align*}
{s} & \leq \{ C \leftarrow \text{default}, D \leftarrow 0, R \leftarrow \text{default} \} \\
{r} & > \{ D \leftarrow \max(D, 1) \} \\
{t} & \leq \{ C \leftarrow \max(D, 1), D \leftarrow 0 \} \\
{u} & = \{ D \leftarrow \max(D, 1) \}
\end{align*}
\]

- Figure 4.989: Automaton for the NB_GORGE constraint obtained by applying decoration Table 3.37 to the seed transducer of the GORGE pattern where default is 0 (transition \( u \rightarrow r \) has the same register update as transition \( r \rightarrow u \))
Figure 4.990: Simplified automaton for the NB_GORGE constraint obtained by applying decoration Table 3.39 to the seed transducer of the GORGE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-2} + 1 \geq 0 \) are linear invariants.

Table 4.233: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the NB_GORGE constraint defined as the composition of the GORGE pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.234: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the simplified automaton of the \textsc{nb\_gorge} constraint defined as the composition of the \textsc{gorge} pattern, the feature \textsc{one}, and the aggregator \textsc{sum}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Specialisation

Figure 4.991: Automaton without registers for the \texttt{NB\_GORGE\_EQ\_0} constraint; it describes all sequences containing no occurrence of the GORGE pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the GORGE pattern by removing the register $R$, the \texttt{found} transition from state $r$ to $t$ that increments $R$, and the state $t$ that becomes unreachable after removing transition $r \rightarrow t$.

Figure 4.992: (left) Automaton without registers for the \texttt{NB\_GORGE\_EQ\_UP} constraint; it describes all sequences containing the maximum number of occurrences of the GORGE pattern on a sequence of $sv$ variables, i.e. $\lfloor \frac{sv-1}{2} \rfloor$ of the \texttt{Restrictions} slot (see \ref{fig:restrictions}); transitions in blue correspond to a new occurrence of the GORGE pattern, dashed transitions to slack, and accepting states have a light-brown background; state $t$ is accepting when $sv \mod 2 = 1$, while states $r$ and $t'$ are accepting when $sv \mod 2 = 0$. (right) All corresponding solutions for $sv - 1 \in \{4, 5\}$.
Figure 4.993: Automata without registers for the (A) NB_GORGE_IS_EVEN and the (B) NB_GORGE_IS_ODD constraints; they respectively achieve an even/odd number of occurrences of the GORGE pattern on a sequence of $n$ variables; transitions in blue correspond to a new occurrence of the GORGE pattern.
<table>
<thead>
<tr>
<th><strong>Origin</strong></th>
<th>Based on the INCREASING pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Constraint</strong></td>
<td>( \text{NB}_\text{INCREASING}(\text{VALUE}, \text{VARIABLES}) )</td>
</tr>
<tr>
<td><strong>Arguments</strong></td>
<td>\text{VALUE} : dvar &lt;br&gt; \text{VARIABLES} : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restrictions</strong></td>
<td>( \text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 ) &lt;br&gt; ( \text{VALUE} \geq 0 ) &lt;br&gt; ( \text{VALUE} \leq \max(0, (\ell - 1) \times \lfloor \text{sv}/\ell \rfloor + \max(0, \text{sv} \mod \ell - 1)) ) &lt;br&gt; ( \text{required}(\text{VARIABLES}.\text{var}) ) &lt;br&gt; ( \text{where} ) &lt;br&gt; ( \text{sv} =</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>( \text{VALUE} ) is the number of occurrences of the INCREASING pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value 0. &lt;br&gt;An occurrence of the pattern INCREASING is the subsequence which matches the regular expression ‘&lt;’.</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>((5, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)))</td>
</tr>
<tr>
<td><strong>Typical</strong></td>
<td>(</td>
</tr>
<tr>
<td><strong>Symmetry</strong></td>
<td>One and the same constant can be added to the var attribute of all items of \text{VARIABLES}.</td>
</tr>
<tr>
<td><strong>Arg. properties</strong></td>
<td>Functional dependency: \text{VALUE} determined by \text{VARIABLES}.</td>
</tr>
</tbody>
</table>
Figure 4.994: Illustrating the NB_INCREASING constraint of the Example slot
Figures 4.995 and 4.996 respectively depict the automaton associated with the constraint `NB_INCREASING` and its simplified form.

Figure 4.995: Automaton for the `NB_INCREASING` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `INCREASING` pattern where `default` is 0

Figure 4.996: Simplified automaton for the `NB_INCREASING` constraint obtained by applying decoration Table 3.39 to the seed transducer of the `INCREASING` pattern where `default` is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + 1 \geq 0$ are linear invariants.

Table 4.235: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the `NB_INCREASING` constraint defined as the composition of the `INCREASING` pattern, the feature `ONE`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.236: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the simplified automaton of the \texttt{NB\_INCREASING} constraint defined as the composition of the \texttt{INCREASING} pattern, the feature \texttt{ONE}, and the aggregator \texttt{sum}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Specialisation

Figure 4.997: Automaton without registers for the \texttt{NB\_INCREASING\_EQ\_0} constraint; it describes all sequences containing no occurrence of the \texttt{INCREASING} pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the \texttt{INCREASING} pattern by removing the register $R$ and the \texttt{found} transition, i.e. the transition from state $s$ to state $s$ that increments $R$.

Figure 4.998: Automata without registers for the (A) \texttt{NB\_INCREASING\_ IS\_EVEN} and the (B) \texttt{NB\_INCREASING\_IS\_ODD} constraints; they respectively achieve an even/odd number of occurrences of the \texttt{INCREASING} pattern on a sequence of $n$ variables; transitions in blue correspond to a new occurrence of the \texttt{INCREASING} pattern.
Origin

Based on the INCREASING_SEQUENCE pattern.

Constraint

NB_INCREASING_SEQUENCE(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE \geq 0 \]
\[ VALUE \leq \lfloor sv/2 \rfloor \]
\[ \text{required}(VARIABLES, var) \]

where

\[ sv = |VARIABLES| \]
\[ rv = \text{range}(VARIABLES.var) \]

Purpose

VALUE is the number of occurrences of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘< (< | =)* < | <’.

Example

(3, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))

Figure 4.999 provides an example where the NB_INCREASING_SEQUENCE (3, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

Typical

|VARIABLES| > 1
range(VARIABLES.var) > 1

Symmetry

One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.999: Illustrating the NB_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.1000 and 4.1001 respectively depict the automaton associated with the constraint \texttt{NB\_INCREASING\_SEQUENCE} and its simplified form.

\begin{align*}
\{ \begin{array}{l}
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default}
\end{array} \right.
\end{align*}

Figure 4.1000: Automaton for the \texttt{NB\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern where \texttt{default} is 0

\begin{align*}
\{ R \leftarrow \text{default} \}
\end{align*}

Figure 4.1001: Simplified automaton for the \texttt{NB\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern where \texttt{default} is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-2} + 1 \geq 0$ are linear invariants.
Table 4.237: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the `NB_INCREASING_SEQUENCE` constraint defined as the composition of the `INCREASINGSEQUENCE` pattern, the feature `ONE`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
</tr>
<tr>
<td>(\overline{C} + \overline{C})</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4.238: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the simplified automaton of the `NB_INCREASING_SEQUENCE` constraint defined as the composition of the `INCREASINGSEQUENCE` pattern, the feature `ONE`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
</tr>
</tbody>
</table>
Specialisation

Figure 4.1002: Automaton without registers for the `NB_INCREASING_SEQUENCE_EQ_0` constraint; it describes all sequences containing no occurrence of the `INCREASING_SEQUENCE` pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the `INCREASING_SEQUENCE` pattern by removing the register $R$, the found transition from state $s$ to $t$ that increments $R$, and the state $t$ that becomes unreachable after removing transition $s \rightarrow t$.

Figure 4.1003: (left) Automaton without registers for the `NB_INCREASING_SEQUENCE_EQ_UP` constraint; it describes all sequences containing the maximum number of occurrences of the `INCREASING_SEQUENCE` pattern on a sequence of $sv$ variables, i.e. $\lfloor \frac{sv}{2} \rfloor$ of the `Restrictions` slot (see 1); transitions in blue correspond to a new occurrence of the `INCREASING_SEQUENCE` pattern, dashed transitions to slack, and accepting states have a light-brown background; state $t$ is accepting when $sv \mod 2 = 0$, while states $s$ and $t'$ are accepting when $sv \mod 2 = 1$. (right) All corresponding solutions for $sv - 1 \in \{5, 6\}$. 

\[ \geq s \rightarrow >, = \]
\[ > s \rightarrow >, = \]
Figure 4.1004: Automata without registers for the (A) `NB_INCREASING_SEQUENCE_IS_EVEN` and the (B) `NB_INCREASING_SEQUENCE_IS_ODD` constraints; they respectively achieve an even/odd number of occurrences of the `INCREASING_SEQUENCE` pattern on a sequence of $n$ variables; transitions in blue correspond to a new occurrence of the `INCREASING_SEQUENCE` pattern.
Origin: Based on the INCREASING_TERRACE pattern.

Constraint: \( \text{NB}_\text{INCREASING}_\text{TERRACE}(\text{VALUE}, \text{VARIABLES}) \)

Arguments:
- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection}(\text{var} \rightarrow \text{dvar}) \)

Restrictions:
- \( \text{sv} \leq 3 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = 0 \)
- \( \text{VALUE} \geq 0 \)
- \( \text{VALUE} \leq \max(0, |(\ell - 2)/2| + |\text{sv}/\ell| + \max(0, |(\text{sv} \mod \ell - 2)/2|))) \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)

where:
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \)
- \( \ell = \max(1, \min(\text{sv} \times 2 - \text{rv} - 2)) \)

Purpose: \( \text{VALUE} \) is the number of occurrences of the INCREASING_TERRACE pattern in the time-series given by the \( \text{VARIABLES} \) collection. If the pattern does not occur, \( \text{VALUE} \) takes the default value 0.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression ‘\(<=^+<\)’.

Example: \((2, (1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 4, 4))\)

Figure 4.1005 provides an example where the \( \text{NB}_\text{INCREASING}_\text{TERRACE} \) \((2, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 4, 4]) \) constraint holds.

Typical:
- \( |\text{VARIABLES}| > 3 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 2 \)

Symmetry: One and the same constant can be added to the \( \text{var} \) attribute of all items of \( \text{VARIABLES} \).

Arg. properties: Functional dependency: \( \text{VALUE} \) determined by \( \text{VARIABLES} \).
Figure 4.1005: Illustrating the `NB_INCREASING_TERRACE` constraint of the `Example` slot
Automaton

Figures 4.1006 and 4.1007 respectively depict the automaton associated with the constraint **NB_INCREASING_TERRACE** and its simplified form.

Figure 4.1006: Automaton for the **NB_INCREASING_TERRACE** constraint obtained by applying decoration Table 3.37 to the seed transducer of the **INCREASING_TERRACE** pattern where default is 0.

Figure 4.1007: Simplified automaton for the **NB_INCREASING_TERRACE** constraint obtained by applying decoration Table 3.39 to the seed transducer of the **INCREASING_TERRACE** pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-2} + 1 \geq 0\) are linear invariants.
### Table 4.239: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the `NB_INCREASING_TERRACE` constraint defined as the composition of the `INCREASING_TERRACE` pattern, the feature `ONE`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>1</td>
</tr>
<tr>
<td>t</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>1</td>
<td>(\overrightarrow{C})</td>
</tr>
</tbody>
</table>

### Table 4.240: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the simplified automaton of the `NB_INCREASING_TERRACE` constraint defined as the composition of the `INCREASING_TERRACE` pattern, the feature `ONE`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>1</td>
<td>(\overrightarrow{C})</td>
</tr>
</tbody>
</table>
Specialisation

Figure 4.1008: Automaton without registers for the NB_INCREASING_TERRACE_EQ_0 constraint; it describes all sequences containing no occurrence of the INCREASING_TERRACE pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the INCREASING_TERRACE pattern by removing the register $R$ and the found transition, i.e. the transition from state $t$ to state $r$ that increments $R$.

Figure 4.1009: (left) Automaton without registers for the NB_INCREASING_TERRACE_EQ_UP constraint; it describes all sequences containing the maximum number of occurrences of the INCREASING_TERRACE pattern on a sequence of $sv$ variables, assuming no restriction on the domain of the variables, i.e. the simplified bound $\max(0, \lfloor \frac{sv-2}{2} \rfloor)$ of the Restrictions slot (see 1); transitions in blue correspond to a new occurrence of the INCREASING_TERRACE pattern, dashed transitions to slack, and accepting states have a light-brown background; states $r$ and $s'$ are accepting when $sv$ mod 2 = 0, states $s$, $t$, $r'$ and $s''$ are accepting when $sv$ mod 2 = 1. (right) All corresponding solutions for $sv - 1 \in \{5, 6\}$.
Figure 4.1010: Automata without registers for the (A) NB_INCREASING_TERRACE_IS_EVEN and the (B) NB_INCREASING_TERRACE_IS_ODD constraints; they respectively achieve an even/odd number of occurrences of the INCREASING_TERRACE pattern on a sequence of $n$ variables; transitions in blue correspond to a new occurrence of the INCREASING_TERRACE pattern.
Origin: Based on the INFLEXION pattern.

Constraint: NB_INFLEXION(VALUE, VARIABLES)

Arguments:
- VALUE : dvar
- VARIABLES : collection(var−dvar)

Restrictions:
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE \geq 0 \)
- \( VALUE \leq \text{max}(0, sv - 2v) \)

required(VARIABLES, var)

where:
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

Purpose:
VALUE is the number of occurrences of the INFLEXION pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \( < (< | =) > | > (> | =) > = < \)∗.

Example:
\((8, 1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 4, 4)\)

Figure 4.1011 provides an example where the NB_INFLEXION \((8, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 4, 4])\) constraint holds.

Typical:
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

Symmetries:
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties:
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1011: Illustrating the NB_INFLEXION constraint of the Example slot
Automaton

Figures 4.1012 and 4.1013 respectively depict the automaton associated with the constraint NB_INFLEXION and its simplified form.

Figure 4.1012: Automaton for the NB_INFLEXION constraint obtained by applying decoration Table 3.37 to the seed transducer of the INFLEXION pattern where default is 0 (transition $r \rightarrow t$ has the same register updates as transition $t \rightarrow r$)

Figure 4.1013: Simplified automaton for the NB_INFLEXION constraint obtained by applying decoration Table 3.39 to the seed transducer of the INFLEXION pattern where default is 0 (transition $r \rightarrow t$ has the same register update as transition $t \rightarrow r$); $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + 1 \geq 0$ are linear invariants.
Specialisation

Figure 4.1014: Automaton without registers for the \texttt{NB\_INFLEXION\_EQ\_0} constraint; it describes all sequences containing no occurrence of the \texttt{INFLEXION} pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the \texttt{INFLEXION} pattern by removing the register \texttt{R} and the transitions labelled by \texttt{found}, i.e. transitions \( r \rightarrow t \) and \( t \rightarrow r \) that increment \texttt{R}.

Figure 4.1015: (left) Automaton without registers for the \texttt{NB\_INFLEXION\_EQ\_UP} constraint; it describes all sequences containing the maximum number of occurrences of the \texttt{INFLEXION} pattern on a sequence of \texttt{sv} variables, i.e. \( \max(0, \texttt{sv} - 2) \) of the \texttt{Restrictions} slot (see 3); transitions in blue correspond to a new occurrence of the \texttt{INFLEXION} pattern; the three accepting states have a light-brown background. (right) All corresponding solutions for \( \texttt{sv} - 1 \in \{1, 2, 3, 4\} \).
Figure 4.1016: Automata without registers for the (A) \texttt{NB\_INFLEXION\_IS\_EVEN} and the (B) \texttt{NB\_INFLEXION\_IS\_ODD} constraints; they respectively achieve an even/odd number of occurrences of the \texttt{INFLEXION} pattern on a sequence of \(n\) variables; transitions in blue correspond to a new occurrence of the \texttt{INFLEXION} pattern.
### Origin

Based on the `PEAK` pattern.

### Constraint

\[
\text{NB\_PEAK}(\text{VALUE}, \text{VARIABLES})
\]

### Arguments

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`

### Restrictions

\[
\begin{align*}
\text{sv} \leq 2 \vee \text{rv} \leq 1 \Rightarrow & \text{VALUE} = 0 \\
\text{VALUE} \geq 0 \\
\text{VALUE} \leq \max(0, \lfloor (\text{sv} - 1)/2 \rfloor) & \text{required}(\text{VARIABLES}, \text{var}) \\
\text{where} \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

### Purpose

`VALUE` is the number of occurrences of the `PEAK` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value 0. An occurrence of the pattern `PEAK` is the maximal subsequence which matches the regular expression `\'< (= | <)\%'`.

### Example

\[(3, \langle 7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1 \rangle)\]

Figure 4.1017 provides an example where the `NB\_PEAK` constraint holds.

### Typical

\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

### Symmetries

- Items of `VARIABLES` can be reversed.
- One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

### Arg. properties

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 4.1017: Illustrating the NB_Peak constraint of the Example slot
Automaton

Figures 4.1018 and 4.1019 respectively depict the automaton associated with the constraint NB_Peak and its simplified form.

Figure 4.1018: Automaton for the NB_Peak constraint obtained by applying decoration Table 3.37 to the seed transducer of the PEAK pattern where default is 0.

Figure 4.1019: Simplified automaton for the NB_Peak constraint obtained by applying decoration Table 3.39 to the seed transducer of the PEAK pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-2} + 1 \geq 0 \) are linear invariants.
Table 4.241: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the NB\_PEAK constraint defined as the composition of the PEAK pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>t</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
</tbody>
</table>

Table 4.242: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the simplified automaton of the NB\_PEAK constraint defined as the composition of the PEAK pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Specialisation

![Specialisation Diagram]

Figure 4.1020: Automaton without registers for the NB_PEAK_EQ_0 constraint; it describes all sequences containing no occurrence of the PEAK pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the PEAK pattern by removing the register $R$, the found transition from state $r$ to $t$ that increments $R$, and the state $t$ that becomes unreachable after removing transition $r \rightarrow t$.

![Specialisation Diagram]

Figure 4.1021: (left) Automaton without registers for the NB_PEAK_EQ_UP constraint; it describes all sequences containing the maximum number of occurrences of the PEAK pattern on a sequence of $sv$ variables, i.e. $\max(0, \lfloor \frac{sv-1}{2} \rceil)$ of the Restrictions slot (see 1); transitions in blue correspond to a new occurrence of the PEAK pattern, dashed transitions to slack, and accepting states have a light-brown background; state $t$ is accepting when $sv \mod 2 = 1$, while states $r$ and $t'$ are accepting when $sv \mod 2 = 0$. (right) All corresponding solutions for $sv - 1 \in \{4, 5\}$.

![Specialisation Diagram]

Figure 4.1022: Automata without registers for the (A) NB_PEAK_IS_EVEN and the (B) NB_PEAK_IS_ODD constraints; they respectively achieve an even/odd number of occurrences of the PEAK pattern on a sequence of $n$ variables; transitions in blue correspond to a new occurrence of the PEAK pattern.
AGGREGATOR

NB_PLAIN

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <code>PLAIN</code> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><code>NB_PLAIN(VALUE, VARIABLES)</code></td>
</tr>
</tbody>
</table>
| **Arguments** | `VALUE : dvar`
| | `VARIABLES : collection(var−dvar)` |
| **Restrictions** | `sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0`
| | `VALUE ≥ 0`
| | `VALUE ≤ max(0, [(sv − 1)/2]) ∧ required(VARIABLES, var)`
| | where
| | `sv = |VARIABLES|`
| | `rv = range(VARIABLES.var)` |
| **Purpose** | `VALUE` is the number of occurrences of the `PLAIN` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value 0. An occurrence of the pattern `PLAIN` is the maximal subsequence which matches the regular expression ‘>∗<’.
| **Example** | `(3, (2,3,6,5,7,6,6,4,5,5,4,3,3,6,6,3))` |
| **Typical** | `|VARIABLES| > 2`
| | `range(VARIABLES.var) > 1` |
| **Symmetries** | • Items of `VARIABLES` can be reversed.
| | • One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.
| **Arg. properties** | Functional dependency: `VALUE` determined by `VARIABLES`.

Figure 4.1023 provides an example where the `NB_PLAIN (3, [2,3,6,5,7,6,6,4,5,5,4,3,3,6,6,3])` constraint holds.
Figure 4.1023: Illustrating the NB.PLAIN constraint of the Example slot
Automaton

Figures 4.1024 and 4.1025 respectively depict the automaton associated with the constraint NBPLAIN and its simplified form.

Figure 4.1024: Automaton for the NBPLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLAIN pattern where default is 0.

Figure 4.1025: Simplified automaton for the NBPLAIN constraint obtained by applying decoration Table 3.39 to the seed transducer of the PLAIN pattern where default is 0; \( R - R_{i-1} \geq 0 \) and \( -R_i + R_{i-2} + 1 \geq 0 \) are linear invariants.
Table 4.243: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the NB_PLAIN constraint defined as the composition of the PLAIN pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>C + C</td>
<td>C + C</td>
<td>C + C</td>
</tr>
<tr>
<td>r</td>
<td>C + C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>t</td>
<td>C + C</td>
<td>C</td>
<td>C</td>
</tr>
</tbody>
</table>

Table 4.244: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the simplified automaton of the NB_PLAIN constraint defined as the composition of the PLAIN pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>C</td>
<td>C</td>
</tr>
</tbody>
</table>
Specialisation

Figure 4.1026: Automaton without registers for the NB_PLAIN_EQ_0 constraint; it describes all sequences containing no occurrence of the PLAIN pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the PLAIN pattern by removing the register $R$ and the transitions labelled by `found`, i.e. transitions $r \rightarrow s$ and $t \rightarrow s$ that increment $R$.

Figure 4.1027: (left) Automaton without registers for the NB_PLAIN_EQ_UP constraint; it describes all sequences containing the maximum number of occurrences of the PLAIN pattern on a sequence of $sv$ variables, i.e. $\max(0, \lfloor \frac{sv-1}{2} \rfloor)$ of the Restrictions slot (see 1); transitions in blue correspond to a new occurrence of the PLAIN pattern, dashed transitions to slack, and accepting states have a light-brown background; state $s$ is accepting when $sv \mod 2 = 1$, while states $r$ and $s'$ are accepting when $sv \mod 2 = 0$. (right) All corresponding solutions for $sv - 1 \in \{4, 5\}$.

Figure 4.1028: Automata without registers for the (A) NB\_PLAIN\_IS\_EVEN and the (B) NB\_PLAIN\_IS\_ODD constraints; they respectively achieve an even/odd number of occurrences of the PLAIN pattern on a sequence of $n$ variables; transitions in blue correspond to a new occurrence of the PLAIN pattern.
### Description

**Origin**
Based on the `PLATEAU` pattern.

**Constraint**

\[ \text{NB\_PLATEAU}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

- `VALUE`: `dvar`
- `VARIABLES`: `collection(var\_dvar)`

**Restrictions**

\[ sv \leq 2 \land rv \leq 1 \Rightarrow \text{VALUE} = 0 \]
\[ \text{VALUE} \geq 0 \]
\[ \text{VALUE} \leq \max(0, [(sv - 1)/2]) \]
\[ \text{required}(\text{VARIABLES}, \text{var}) \]

where

- `sv = |\text{VARIABLES}|`
- `rv = \text{range}(\text{VARIABLES}.\text{var})`

**Purpose**

`VALUE` is the number of occurrences of the `PLATEAU` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value 0.

An occurrence of the pattern `PLATEAU` is the maximal subsequence which matches the regular expression `'<='`.

**Example**

\{(3, (7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5))\}

Figure 4.1029 provides an example where the `NB\_PLATEAU` constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Symmetries**

- Items of `VARIABLES` can be reversed.
- One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 4.1029: Illustrating the NB_PLATEAU constraint of the Example slot
Automaton

Figures 4.1030 and 4.1031 respectively depict the automaton associated with the constraint \texttt{NB\_PLATEAU} and its simplified form.

\begin{align*}
\begin{cases}
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default}
\end{cases}
\quad \text{if } s \geq R + C \\
\begin{cases}
D \leftarrow 0 \\
R \leftarrow R + \max(D, 1)
\end{cases}
\quad \text{if } s < R + C
\end{align*}

Figure 4.1030: Automaton for the \texttt{NB\_PLATEAU} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{PLATEAU} pattern where \texttt{default} is 0.

\begin{align*}
\begin{cases}
R \leftarrow \text{default} \\
D \leftarrow \max(D, 1)
\end{cases}
\quad \text{if } R \geq D \\
\begin{cases}
D \leftarrow 0 \\
R \leftarrow R + \max(D, 1)
\end{cases}
\quad \text{if } s < D + C
\end{align*}

\begin{align*}
\begin{cases}
R \leftarrow \text{default} \\
D \leftarrow \max(D, 1)
\end{cases}
\quad \text{if } R \geq D \\
\begin{cases}
D \leftarrow 0 \\
R \leftarrow R + \max(D, 1)
\end{cases}
\quad \text{if } s < D + C
\end{align*}

Figure 4.1031: Simplified automaton for the \texttt{NB\_PLATEAU} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \texttt{PLATEAU} pattern where \texttt{default} is 0; \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-2} + 1 \geq 0\) are linear invariants.
Table 4.245: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the \textit{NB\_PLATEAU} constraint defined as the composition of the \texttt{PLATEAU} pattern, the feature \texttt{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>t</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4.246: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the simplified automaton of the \textit{NB\_PLATEAU} constraint defined as the composition of the \texttt{PLATEAU} pattern, the feature \texttt{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Specialisation

Figure 4.1032: Automaton without registers for the \texttt{NB\_PLATEAU\_EQ\_0} constraint; it describes all sequences containing no occurrence of the \texttt{PLATEAU} pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the \texttt{PLATEAU} pattern by removing the register \( R \) and the transitions labelled by \texttt{found}, i.e. transitions \( r \to s \) and \( t \to s \) that increment \( R \).

Figure 4.1033: (left) Automaton without registers for the \texttt{NB\_PLATEAU\_EQ\_UP} constraint; it describes all sequences containing the maximum number of occurrences of the \texttt{PLATEAU} pattern on a sequence of \( sv \) variables, i.e. \( \max(0, \left\lfloor \frac{sv-1}{2} \right\rfloor) \) of the \textbf{Restrictions} slot (see 1); transitions in blue correspond to a new occurrence of the \texttt{PLATEAU} pattern, dashed transitions to slack, and accepting states have a light-brown background; state \( s' \) is accepting when \( sv \mod 2 = 1 \), while states \( r \) and \( s' \) are accepting when \( sv \mod 2 = 0 \). (right) All corresponding solutions for \( sv - 1 \in \{4, 5\} \).
Figure 4.1034: Automata without registers for the (A) \text{NB\_PLATEAU\_IS\_EVEN} and the (B) \text{NB\_PLATEAU\_IS\_ODD} constraints; they respectively achieve an even/odd number of occurrences of the \text{PLATEAU} pattern on a sequence of \(n\) variables; transitions in blue correspond to a new occurrence of the \text{PLATEAU} pattern.
**NB_PROPER_PLAIN**

<table>
<thead>
<tr>
<th>AGGREGATOR</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ ▶ ▶ ▶ ▶</td>
<td></td>
</tr>
</tbody>
</table>

**DESCRIPTION**

**Origin**
Based on the `PROPER_PLAIN` pattern.

**Constraint**
```
NB_PROPER_PLAIN(VALUE, VARIABLES)
```

**Arguments**
- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`

**Restrictions**
```
sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE ≥ 0
VALUE ≤ max(0, |(sv − 1)/3|)⇒
required(VARIABLES, var)
```
where

- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**
VALUE is the number of occurrences of the `PROPER_PLAIN` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern `PROPER_PLAIN` is the maximal subsequence which matches the regular expression ‘>=<’.

**Example**
```
(3, (2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5))
```

Figure 4.1035 provides an example where the `NB_PROPER_PLAIN (3, [2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5])` constraint holds.

**Typical**
```
|VARIABLES| > 3
range(VARIABLES.var) > 1
```

**Symmetries**
- Items of `VARIABLES` can be reversed.
- One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**
**Functional dependency:** VALUE determined by `VARIABLES`.

---

[Diagram]

---

**Figure 4.1035**

---

**Diag.**

---
Figure 4.1035: Illustrating the NB_PROPER_PLAIN constraint of the Example slot
Automaton

Figures 4.1036 and 4.1037 respectively depict the automaton associated with the constraint \texttt{NB\_PROPER\_PLAIN} and its simplified form.

\begin{itemize}
  \item Figure 4.1036: Automaton for the \texttt{NB\_PROPER\_PLAIN} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{PROPER\_PLAIN} pattern where default is 0.
  \item Figure 4.1037: Simplified automaton for the \texttt{NB\_PROPER\_PLAIN} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \texttt{PROPER\_PLAIN} pattern where default is 0; \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-3} + 1 \geq 0\) are linear invariants.
\end{itemize}
### Table 4.247: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the NB_PROPER_PLAIN constraint defined as the composition of the PROPERPLAIN pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>1</td>
</tr>
<tr>
<td>t</td>
<td>1</td>
<td>1</td>
<td>C</td>
</tr>
</tbody>
</table>

### Table 4.248: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the simplified automaton of the NB_PROPER_PLAIN constraint defined as the composition of the PROPERPLAIN pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Figure 4.1038: Automaton without registers for the NB_PROPERPLAIN_EQ_0 constraint; it describes all sequences containing no occurrence of the PROPERPLAIN pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the PROPERPLAIN pattern by removing the register $R$ and the found transition, i.e. the transition from state $t$ to state $s$ that increments $R$.

Figure 4.1039: (left) Automaton without registers for the NB_PROPERPLAIN_EQ_UP constraint; it describes all sequences containing the maximum number of occurrences of the PROPERPLAIN pattern on a sequence of $sv$ variables, i.e. $\max(0, \lfloor \frac{sv}{3} - 1 \rfloor)$ of the Restrictions slot (see 1); transitions in blue correspond to a new occurrence of the PROPERPLAIN pattern, dashed transitions to slack, and accepting states have a light-brown background; state $s$ is accepting when $sv \mod 3 = 1$, states $r$ and $s'$ are accepting when $sv \mod 3 = 2$, states $t$, $r'$, $s''$ are accepting when $sv \mod 3 = 0$. (right) All corresponding solutions for $sv - 1 \in \{3, 4, 5\}$.
Figure 4.1040: Automata without registers for the (A) `NB_PROPER_PLAIN_IS_EVEN` and the (B) `NB_PROPER_PLAIN_IS_ODD` constraints; they respectively achieve an even/odd number of occurrences of the `PROPER_PLAIN` pattern on a sequence of \( n \) variables; transitions in blue correspond to a new occurrence of the `PROPER_PLAIN` pattern.
### Origin
Based on the `PROPER_PLATEAU` pattern.

### Constraint
```plaintext
NB_PROPER_PLATEAU(VALUE, VARIABLES)
```

### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

### Restrictions

- \( sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE \geq 0 \)
- \( VALUE \leq \max(0, [(sv - 1)/3]) \) if required(VARIABLES, var)

Where
- \( sv = |VARIABLES| \)
- \( rv = \text{range}(VARIABLES, \text{var}) \)

### Purpose
 valeur is the number of occurrences of the `PROPER_PLATEAU` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern `PROPER_PLATEAU` is the maximal subsequence which matches the regular expression ‘\(<=+>\)’.

### Example

```
(3, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))
```

Figure 4.1041 provides an example where the `NB_PROPER_PLATEAU (3, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3])` constraint holds.

### Typical

- \(|VARIABLES| > 3 \)
- \(\text{range}(VARIABLES, \text{var}) > 1 \)

### Symmetries

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

### Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.1041: Illustrating the `NB_PROPER_PLATEAU` constraint of the `Example` slot
Figures 4.1042 and 4.1043 respectively depict the automaton associated with the constraint NB_PROPER_PLATEAU and its simplified form.

Figure 4.1042: Automaton for the NB_PROPER_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER_PLATEAU pattern where \textit{default} is 0

Figure 4.1043: Simplified automaton for the NB_PROPER_PLATEAU constraint obtained by applying decoration Table 3.39 to the seed transducer of the PROPER_PLATEAU pattern where \textit{default} is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-3} + 1 \geq 0$ are linear invariants.
Table 4.249: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the \texttt{NB\_PROPER\_PLATEAU} constraint defined as the composition of the \texttt{PROPER\_PLATEAU} pattern, the feature \texttt{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
</tr>
<tr>
<td>(r)</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
<td>1</td>
</tr>
<tr>
<td>(t)</td>
<td>(\overline{C} + \overline{C})</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4.250: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the simplified automaton of the \texttt{NB\_PROPER\_PLATEAU} constraint defined as the composition of the \texttt{PROPER\_PLATEAU} pattern, the feature \texttt{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(r)</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>(t)</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Specialisation

Figure 4.1044: Automaton without registers for the NB_PROPER_PLATEAU_EQ_0 constraint; it describes all sequences containing no occurrence of the PROPER_PLATEAU pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the PROPER_PLATEAU pattern by removing the register $R$ and the found transition, i.e. the transition from state $t$ to state $s$ that increments $R$.

Figure 4.1045: (left) Automaton without registers for the NB_PROPER_PLATEAU_EQ_UP constraint; it describes all sequences containing the maximum number of occurrences of the PROPER_PLATEAU pattern on a sequence of $sv$ variables, i.e. $\max(0, \lfloor \frac{sv-1}{3} \rfloor)$ of the Restrictions slot (see ©); transitions in blue correspond to a new occurrence of the PROPER_PLATEAU pattern, dashed transitions to slack, and accepting states have a light-brown background; state $s$ is accepting when $sv \mod 3 = 1$, states $r$ and $s'$ are accepting when $sv \mod 3 = 2$, states $t$, $r'$, $s''$ are accepting when $sv \mod 3 = 0$. (right) All corresponding solutions for $sv - 1 \in \{3, 4, 5\}$.
Figure 4.1046: Automata without registers for the (A) NB_PROPER_PLATEAU_IS_EVEN and the (B) NB_PROPER_PLATEAU_IS_ODD constraints; they respectively achieve an even/odd number of occurrences of the PROPER_PLATEAU pattern on a sequence of \( n \) variables; transitions in blue correspond to a new occurrence of the PROPER_PLATEAU pattern.
Origin
Based on the **STEADY** pattern.

Constraint
**NB_STEADY** (VALUE, VARIABLES)

Arguments
- **VALUE** : dvar
- **VARIABLES** : collection(var − dvar)

Restrictions
- 
  \[ sv \leq 1 \Rightarrow VALUE = 0 \]
  \[ rv = 1 \Rightarrow VALUE \geq sv - 1 \]
  \[ rv \geq 2 \Rightarrow VALUE \geq 0 \]
  \[ VALUE \leq \max(0, sv - 1) \]

  \[ \text{required}(\text{VARIABLES}, \text{var}) \]
  \[ \text{where} \]
  \[ sv = |\text{VARIABLES}| \]
  \[ rv = \text{range}(\text{VARIABLES}, \text{var}) \]

Purpose
**VALUE** is the number of occurrences of the **STEADY** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value 0.

An occurrence of the pattern **STEADY** is the subsequence which matches the regular expression ‘=’.

Example
\[(7, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6))\]

Figure 4.1047 provides an example where the **NB_STEADY** (7, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6]) constraint holds.

Typical
\[ |\text{VARIABLES}| > 1 \]

Symmetries
- Items of **VARIABLES** can be reversed.
- One and the same constant can be added to the **var** attribute of all items of **VARIABLES**.

Arg. properties
- **Functional dependency**: **VALUE** determined by **VARIABLES**.
Figure 4.1047: Illustrating the NB STEADY constraint of the Example slot
Figures 4.1048 and 4.1049 respectively depict the automaton associated with the constraint \textsc{nb\_steady} and its simplified form.

\begin{align*}
\{ C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default} \}
\end{align*}

\begin{align*}
\{ D \leftarrow 0 \\
R \leftarrow R + \max(D, 1) \}
\end{align*}

Figure 4.1048: Automaton for the \textsc{nb\_steady} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsc{steady} pattern where \texttt{default} is 0

\begin{align*}
\{ R \leftarrow \text{default} \}
\end{align*}

\begin{align*}
\{ R \leftarrow R + 1 \}
\end{align*}

Figure 4.1049: Simplified automaton for the \textsc{nb\_steady} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \textsc{steady} pattern where \texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + 1 \geq 0 \) are linear invariants.

Table 4.251: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the \textsc{nb\_steady} constraint defined as the composition of the \textsc{steady} pattern, the feature \texttt{one}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.252: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the simplified automaton of the NB_STEADY constraint defined as the composition of the STEADY pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Specialisation

Figure 4.1050: Automaton without registers for the `NB_STEADY_EQ_0` constraint; it describes all sequences containing no occurrence of the `STEADY` pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the `STEADY` pattern by removing the register `R` and the `found` transition, i.e. the transition from state `s` to state `s` that increments `R`.

Figure 4.1051: Automata without registers for the (A) `NB_STEADY_IS_EVEN` and the (B) `NB_STEADY_IS_ODD` constraints; they respectively achieve an even/odd number of occurrences of the `STEADY` pattern on a sequence of `n` variables; transitions in blue correspond to a new occurrence of the `STEADY` pattern.
NB_STEADY_SEQUENCE

**Origin**
Based on the **STEADY_SEQUENCE** pattern.

**Constraint**

\[ \text{NB\_STEADY\_SEQUENCE}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var–dvar)

**Restrictions**

- \[ sv \leq 1 \Rightarrow \text{VALUE} = 0 \]
- \[ rv = 1 \land sv \geq 2 \Rightarrow \text{VALUE} \geq 1 \]
- \[ rv \geq 2 \Rightarrow \text{VALUE} \geq 0 \]
- \[ rv = 1 \land sv \geq 2 \Rightarrow \text{VALUE} \leq 1 \]
- \[ rv \geq 2 \land sv \geq 2 \Rightarrow \text{VALUE} \leq \lfloor sv/2 \rfloor \]

where

- \[ sv = |\text{VARIABLES}| \]
- \[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

**Purpose**
VALUE is the number of occurrences of the **STEADY_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern **STEADY_SEQUENCE** is the maximal subsequence which matches the regular expression ‘*=+’.

**Example**

\((5, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1))\)

Figure 4.1052 provides an example where the **NB\_STEADY\_SEQUENCE** constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]

**Symmetries**

- Items of **VARIABLES** can be reversed.
- One and the same constant can be added to the var attribute of all items of **VARIABLES**.

**Arg. properties**

Functional dependency: VALUE determined by **VARIABLES**.
Figure 4.1052: Illustrating the NB_STEADY_SEQUENCE constraint of the Example slot
Automaton

Figures 4.1053 and 4.1054 respectively depict the automaton associated with the constraint NB_STEADY_SEQUENCE and its simplified form.

Figure 4.1053: Automaton for the NB_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STEADY_SEQUENCE pattern where default is 0.

Figure 4.1054: Simplified automaton for the NB_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.39 to the seed transducer of the STEADY_SEQUENCE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-2} + 1 \geq 0 \) are linear invariants.
Table 4.253: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the NB_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overline{C} + \overline{C}$</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4.254: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the simplified automaton of the NB_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>


Specialisation

Figure 4.1055: Automaton without registers for the `NB_STEADY_SEQUENCE_EQ_0` constraint; it describes all sequences containing no occurrence of the `STEADY_SEQUENCE` pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the `STEADY_SEQUENCE` pattern by removing the register $R$, the found transition from state $s$ to $r$ that increments $R$, and the state $r$ that becomes unreachable after removing transition $s \rightarrow r$.

Figure 4.1056: (left) Automaton without registers for the `NB_STEADY_SEQUENCE_EQ_UP` constraint; it describes all sequences containing the maximum number of occurrences of the `STEADY_SEQUENCE` pattern on a sequence of $sv$ variables, i.e. $\left\lfloor \frac{sv}{2} \right\rfloor$ of the `Restrictions` slot (see 1); transitions in blue correspond to a new occurrence of the `STEADY_SEQUENCE` pattern, dashed transitions to slack, and accepting states have a light-brown background; state $r$ is accepting when $sv \mod 2 = 0$, while states $s$ and $r'$ are accepting when $sv \mod 2 = 1$. (right) All corresponding solutions for $sv - 1 \in \{3, 4\}$. 

---

Figure 4.1055:

Figure 4.1056:
Figure 4.1057: Automata without registers for the (A) `NB_STEADY_SEQUENCE_IS_EVEN` and the (B) `NB_STEADY_SEQUENCE_IS_ODD` constraints; they respectively achieve an even/odd number of occurrences of the `STEADY_SEQUENCE` pattern on a sequence of $n$ variables; transitions in blue correspond to a new occurrence of the `STEADY_SEQUENCE` pattern.
**Origin**
Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**
```
NB STRICTLY DECREASING_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var–dvar)

**Restrictions**
- \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE \geq 0 \)
- \( VALUE \leq \lfloor sv/2 \rfloor \)

**Purpose**
VALUE is the number of occurrences of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the maximal sub-sequence which matches the regular expression `'>'`.

**Example**
```
(3, (4, 4, 6, 4, 1, 3, 4, 6, 6, 5, 2, 2, 4, 3))
```

Figure 4.1058 provides an example where the `NB STRICTLY DECREASING_SEQUENCE` (3, [4, 4, 6, 4, 1, 3, 4, 6, 6, 5, 2, 2, 4, 3]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 1 \)
- \(\text{range} (\text{VARIABLES}.\text{var}) > 1 \)

**Symmetry**
One and the same constant can be added to the `var` attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1058: Illustrating the \texttt{NB\_STRICTLY\_DECREASING\_SEQUENCE} constraint of the \textbf{Example} slot
Figures 4.1059 and 4.1060 respectively depict the automaton associated with the constraint \texttt{NB\_STRICTLY\_DECREASING\_SEQUENCE} and its simplified form.

\begin{figure}
\centering
\includegraphics[width=0.5\textwidth]{automaton.png}
\caption{Automaton for the \texttt{NB\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table~3.37 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0.}
\end{figure}

Table 4.255: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the \texttt{NB\_STRICTLY\_DECREASING\_SEQUENCE} constraint defined as the composition of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern, the feature \texttt{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.1060: Simplified automaton for the `NB_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.39 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-2} + 1 \geq 0\) are linear invariants.

Table 4.256: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the `NB_STRICTLY_DECREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_DECREASING_SEQUENCE` pattern, the feature `ONE`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Specialisation

Figure 4.1061: Automaton without registers for the NB_STRICTLY_DECREASING_SEQUENCE_EQ_0 constraint; it describes all sequences containing no occurrence of the STRICTLY_DECREASING_SEQUENCE pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the STRICTLY_DECREASING_SEQUENCE pattern by removing the register $R$, the found transition from state $s$ to $r$ that increments $R$, and the state $r$ that becomes unreachable after removing transition $s \rightarrow r$.

Figure 4.1062: (left) Automaton without registers for the NB_STRICTLY_DECREASING_SEQUENCE_EQ_UP constraint; it describes all sequences containing the maximum number of occurrences of the STRICTLY_DECREASING_SEQUENCE pattern on a sequence of sv variables, i.e. $\left\lfloor \frac{sv}{2} \right\rfloor$ of the Restrictions slot (see 1); transitions in blue correspond to a new occurrence of the STRICTLY_DECREASING_SEQUENCE pattern, dashed transitions to slack, and accepting states have a light-brown background; state $r$ is accepting when $sv \mod 2 = 0$, while states $s$ and $r'$ are accepting when $sv \mod 2 = 1$. (right) All corresponding solutions for $sv - 1 \in \{3, 4\}$.
Figure 4.1063: Automata without registers for the (A) `NB_STRICTLY_DECREASING_SEQUENCE` and the (B) `NB_STRICTLY_DECREASING_SEQUENCE` constraints; they respectively achieve an even/odd number of occurrences of the `STRICTLY_DECREASING` pattern on a sequence of \( n \) variables; transitions in blue correspond to a new occurrence of the `STRICTLY_DECREASING` pattern.
NB STRICTLY_INCREASING_SEQUENCE

Description

Origin
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

Constraint
NB STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var-dvar)

Restrictions
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0
VALUE \geq 0
VALUE \leq \lfloor sv/2 \rfloor \Rightarrow \text{required}(VARIABLES, var)
where
sv = |VARIABLES|
rv = \text{range}(VARIABLES.var)

Purpose
VALUE is the number of occurrences of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.
An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression \'<\+'.

Example
(3, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))

Figure 4.1064 provides an example where the NB STRICTLY_INCREASING_SEQUENCE (3, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES.var) > 1

Symmetry
One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1064: Illustrating the `NB_STRICTLY_INCREASING_SEQUENCE` constraint of the `Example` slot
Automaton

Figures 4.1065 and 4.1066 respectively depict the automaton associated with the constraint `NB_STRICTLY_INCREASING_SEQUENCE` and its simplified form.

\[
\begin{align*}
\{ & \begin{cases}
C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \text{default}
\end{cases} \\
\{ & \begin{cases}
C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow R + C
\end{cases}
\end{align*}
\]

Figure 4.1065: Automaton for the `NB_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where `default` is 0.

\[
\begin{array}{cc}
s & r \\
\hline
s & \overrightarrow{C + \\ C} & \overrightarrow{C + \\ C} \\
r & \overrightarrow{C + \\ C} & 1
\end{array}
\]

Table 4.257: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the `NB_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the `feature ONE`, and the `aggregator sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.1066: Simplified automaton for the NB_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.39 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-2} + 1 \geq 0$ are linear invariants.

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
</tr>
</tbody>
</table>
Figure 4.1067: Automaton without registers for the \texttt{NB\_STRICTLY\_INCREASING\_SEQUENCE\_EQ\_0} constraint; it describes all sequences containing no occurrence of the \texttt{STRICTLY\_INCREASING\_SEQUENCE} pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the \texttt{STRICTLY\_INCREASING\_SEQUENCE} pattern by removing the register $R$, the found transition from state $s$ to $r$ that increments $R$, and the state $r$ that becomes unreachable after removing transition $s \rightarrow r$.

Figure 4.1068: (left) Automaton without registers for the \texttt{NB\_STRICTLY\_INCREASING\_SEQUENCE\_EQ\_UP} constraint; it describes all sequences containing the maximum number of occurrences of the \texttt{STRICTLY\_INCREASING\_SEQUENCE} pattern on a sequence of $sv$ variables, i.e. $\lfloor \frac{sv}{2} \rfloor$ of the \texttt{Restrictions} slot (see \textcircled{1}); transitions in blue correspond to a new occurrence of the \texttt{STRICTLY\_INCREASING\_SEQUENCE} pattern, dashed transitions to slack, and accepting states have a light-brown background; state $r$ is accepting when $sv \mod 2 = 0$, while states $s$ and $r'$ are accepting when $sv \mod 2 = 1$. (right) All corresponding solutions for $sv - 1 \in \{3, 4\}$.
Figure 4.1069: Automata without registers for the (A) \textit{NB\textunderscore STRICTLY\textunderscore INCREASING\textunderscore IS\textunderscore EVEN} and the (B) \textit{NB\textunderscore STRICTLY\textunderscore INCREASING\textunderscore IS\textunderscore ODD} constraints; they respectively achieve an even/odd number of occurrences of the \textit{STRUCTLY\textunderscore INCREASING} pattern on a sequence of \( n \) variables; transitions in blue correspond to a new occurrence of the \textit{STRUCTLY\textunderscore INCREASING} pattern.
**NB_SUMMIT**

**Description**

**Origin**
Based on the SUMMIT pattern.

**Constraint**

\[
\text{NB\_SUMMIT}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var\text{-dvar})}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \max(0, \lfloor (\text{sv} - 1)/2 \rfloor) \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**
VALUE is the number of occurrences of the SUMMIT pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression ‘\((< | < (= | <)^* < >(> | > (= | >)^* >)\)’.

**Example**

\((3, (7, 1, 5, 4, 3, 6, 6, 2, 3, 4, 2, 3, 1))\)

Figure 4.1070 provides an example where the \text{NB\_SUMMIT} \((3, [7, 1, 5, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1])\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 2\]

\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Symmetries**

- Items of VARIABLES can be **reversed**.
- One and the same constant can be **added** to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.1070: Illustrating the NB_SUMMIT constraint of the Example slot
Automaton

Figures 4.1071 and 4.1072 respectively depict the automaton associated with the constraint NB_SUMMIT and its simplified form.

Figure 4.1071: Automaton for the NB_SUMMIT constraint obtained by applying decoration Table 3.37 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
Figure 4.1072: Simplified automaton for the \texttt{NB\_SUMMIT} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \texttt{SUMMIT} pattern where \texttt{default} is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-2} + 1 \geq 0$ are linear invariants.

Table 4.259: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the \texttt{NB\_SUMMIT} constraint defined as the composition of the \texttt{SUMMIT} pattern, the feature \texttt{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.260: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the simplified automaton of the NB_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.1073: Automaton without registers for the NB_SUMMIT_EQ_0 constraint; it describes all sequences containing no occurrence of the SUMMIT pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the SUMMIT pattern by removing the register $R$, the found transition from state $r$ to $t$ that increments $R$, and the state $t$ that becomes unreachable after removing transition $r \rightarrow t$.

Figure 4.1074: (left) Automaton without registers for the NB_SUMMIT_EQ_UP constraint; it describes all sequences containing the maximum number of occurrences of the SUMMIT pattern on a sequence of $sv$ variables, i.e. $\lfloor \frac{sv-1}{2} \rfloor$ of the Restrictions slot (see (1)); transitions in blue correspond to a new occurrence of the SUMMIT pattern, dashed transitions to slack, and accepting states have a light-brown background; state $t'$ is accepting when $sv \mod 2 = 1$, while states $r$ and $t'$ are accepting when $sv \mod 2 = 0$. (right) All corresponding solutions for $sv - 1 \in \{4, 5\}$.
Figure 4.1075: Automata without registers for the (A) NB_SUMMIT_IS_EVEN and the (B) NB_SUMMIT_IS_ODD constraints; they respectively achieve an even/odd number of occurrences of the SUMMIT pattern on a sequence of $n$ variables; transitions in blue correspond to a new occurrence of the SUMMIT pattern.
NB
VALLEY

DESCRIPTION

AUTOMATON

Origin
Based on the VALLEY pattern.

Constraint
\( \text{NB}_\text{VALLEY}(\text{VALUE}, \text{VARIABLES}) \)

Arguments
\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

Restrictions
\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \max(0, \lfloor (\text{sv} - 1)/2 \rfloor) \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where
\[
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Purpose
\( \text{VALUE} \) is the number of occurrences of the VALLEY pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \( \text{VALUE} \) takes the default value 0.

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression ‘\( > (= | >)^* (< | =)^* < \)’.

Example
\((3, (1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7))\)

Figure 4.1076 provides an example where the \text{NB}_\text{VALLEY} (3, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7]) constraint holds.

Typical
\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

Symmetries
- Items of \text{VARIABLES} can be reversed.
- One and the same constant can be added to the \text{var} attribute of all items of \text{VARIABLES}.

Arg. properties
Functional dependency: \( \text{VALUE} \) determined by \text{VARIABLES}. 
Figure 4.1076: Illustrating the NB_VALLEY constraint of the Example slot
Automaton

Figures 4.1077 and 4.1078 respectively depict the automaton associated with the constraint \textsc{nB\_Valley} and its simplified form.

\begin{align*}
\text{Fig. 4.1077: Automaton for the \textsc{nB\_Valley} constraint obtained by applying decoration Table 3.37 to the seed transducer of the VALLEY pattern where default is 0; } \\
R_i - R_{i-1} \geq 0 \text{ and } -R_i + R_{i-2} + 1 \geq 0 \text{ are linear invariants.}
\end{align*}
Table 4.261: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the `NB_VALLEY` constraint defined as the composition of the `VALLEY` pattern, the feature `ONE`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(\overrightarrow{c} + \overrightarrow{c})</td>
<td>(\overrightarrow{c} + \overrightarrow{r})</td>
<td>(\overrightarrow{c} + \overrightarrow{c})</td>
</tr>
<tr>
<td>(s)</td>
<td>(\overrightarrow{c} + \overrightarrow{c})</td>
<td>1</td>
<td>(\overrightarrow{c})</td>
</tr>
<tr>
<td>(r)</td>
<td>(\overrightarrow{c} + \overrightarrow{c})</td>
<td>1</td>
<td>(\overrightarrow{r})</td>
</tr>
<tr>
<td>(t)</td>
<td>(\overrightarrow{c} + \overrightarrow{c})</td>
<td>(\overrightarrow{c})</td>
<td>(\overrightarrow{c} + \overrightarrow{c})</td>
</tr>
</tbody>
</table>

Table 4.262: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the simplified automaton of the `NB_VALLEY` constraint defined as the composition of the `VALLEY` pattern, the feature `ONE`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(s)</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(r)</td>
<td>(\overrightarrow{c})</td>
<td>0</td>
<td>(\overrightarrow{r})</td>
</tr>
<tr>
<td>(t)</td>
<td>0</td>
<td>(\overrightarrow{c})</td>
<td>0</td>
</tr>
</tbody>
</table>
Specialisation

Figure 4.1079: Automaton without registers for the NB\_VALLEY\_EQ\_0 constraint; it describes all sequences containing no occurrence of the VALLEY pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the VALLEY pattern by removing the register $R$, the found transition from state $r$ to $t$ that increments $R$, and the state $t$ that becomes unreachable after removing transition $r \to t$.

Figure 4.1080: (left) Automaton without registers for the NB\_VALLEY\_EQ\_UP constraint; it describes all sequences containing the maximum number of occurrences of the VALLEY pattern on a sequence of $sv$ variables, i.e. $\max(0, \lfloor \frac{sv-1}{2} \rfloor)$ of the Restrictions slot (see 0); transitions in blue correspond to a new occurrence of the VALLEY pattern, dashed transitions to slack, and accepting states have a light-brown background; state $t'$ is accepting when $sv \mod 2 = 1$, while states $r$ and $t'$ are accepting when $sv \mod 2 = 0$. (right) All corresponding solutions for $sv - 1 \in \{4, 5\}$. 
Figure 4.1081: Automata without registers for the (A) NB_VALLEY_IS_EVEN and the (B) NB_VALLEY_IS_ODD constraints; they respectively achieve an even/odd number of occurrences of the VALLEY pattern on a sequence of $n$ variables; transitions in blue correspond to a new occurrence of the VALLEY pattern.
NB_ZIGZAG

DESCRIPTION

Origin
Based on the ZIGZAG pattern.

Constraint

\[ \text{NB\_ZIGZAG}(\text{VALUE}, \text{VARIABLES}) \]

Arguments

\[
\begin{align*}
\text{VALUE} : & \quad \text{dvar} \\
\text{VARIABLES} : & \quad \text{collection(var-dvar)}
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} \leq 3 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{rv} = 2 & \Rightarrow \text{VALUE} \leq \lfloor \text{sv}/4 \rfloor \\
\text{rv} \geq 3 & \Rightarrow \text{VALUE} \leq \max(0, \lfloor (sv-1)/3 \rfloor) \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} = & \quad |\text{VARIABLES}| \\
\text{rv} = & \quad \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Purpose

\text{VALUE} is the number of occurrences of the ZIGZAG pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value 0.

An occurrence of the pattern ZIGZAG is the \textit{maximal} subsequence which matches the regular expression '\((<>)^+ (< | <>)) | (<>)^+ (> | <>)'.'

Example

\[(3, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\]

Figure 4.1082 provides an example where the \text{NB\_ZIGZAG} \((3, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1])\) constraint holds.

Typical

\[
\begin{align*}
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\end{align*}
\]

Symmetries

- Items of \text{VARIABLES} can be \textit{reversed}.
- One and the same constant can be \textit{added} to the \text{var} attribute of all items of \text{VARIABLES}.

Arg. properties

\textbf{Functional dependency:} \text{VALUE} determined by \text{VARIABLES}.
Figure 4.1082: Illustrating the NB_ZIGZAG constraint of the Example slot
Figures 4.1083 and 4.1084 respectively depict the automaton associated with the constraint \texttt{NB\_ZIGZAG} and its simplified form.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( a )</th>
<th>( b )</th>
<th>( c )</th>
<th>( d )</th>
<th>( e )</th>
<th>( f )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
</tr>
<tr>
<td>( a )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
</tr>
<tr>
<td>( b )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
</tr>
<tr>
<td>( c )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( 1 )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
</tr>
<tr>
<td>( d )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
</tr>
<tr>
<td>( e )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( 1 )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
</tr>
<tr>
<td>( f )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \overline{\mathcal{C}} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
<td>( \mathcal{C} + \mathcal{C} )</td>
</tr>
</tbody>
</table>

Table 4.263: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the \texttt{NB\_ZIGZAG} constraint defined as the composition of the \texttt{ZIGZAG} pattern, the feature \texttt{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( a )</th>
<th>( b )</th>
<th>( c )</th>
<th>( d )</th>
<th>( e )</th>
<th>( f )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( a )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>( \mathcal{C} )</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( b )</td>
<td>0</td>
<td>0</td>
<td>( \mathcal{C} )</td>
<td>0</td>
<td>1</td>
<td>( \mathcal{C} )</td>
<td>0</td>
</tr>
<tr>
<td>( c )</td>
<td>0</td>
<td>0</td>
<td>( \mathcal{C} )</td>
<td>0</td>
<td>( \mathcal{C} )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( d )</td>
<td>0</td>
<td>0</td>
<td>( \mathcal{C} )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( e )</td>
<td>0</td>
<td>( \mathcal{C} )</td>
<td>0</td>
<td>( \mathcal{C} )</td>
<td>0</td>
<td>( \mathcal{C} )</td>
<td>0</td>
</tr>
<tr>
<td>( f )</td>
<td>0</td>
<td>0</td>
<td>( \mathcal{C} )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>( \mathcal{C} )</td>
</tr>
</tbody>
</table>

Table 4.264: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the simplified automaton of the \texttt{NB\_ZIGZAG} constraint defined as the composition of the \texttt{ZIGZAG} pattern, the feature \texttt{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.1083: Automaton for the NB_ZIGZAG constraint obtained by applying decoration Table 3.37 to the seed transducer of the ZIGZAG pattern where default is 0; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the register $D$ is reset to its initial value; (3) on transitions from c, f to s the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.
Figure 4.1084: Simplified automaton for the \textsc{NB\_ZIGZAG} constraint obtained by applying decoration Table 3.39 to the seed transducer of the ZIGZAG pattern where \texttt{default} is 0; missing transitions from \texttt{a}, \texttt{b}, \texttt{c}, \texttt{d}, \texttt{e}, \texttt{f} to \texttt{s} are labelled by \(=\); \(R_i - R_{i-1} \geq 0\) and \(\neg R_i + R_{i-3} + 1 \geq 0\) are linear invariants.
Specialisation

Figure 4.1085: Automaton without registers for the NB\_ZIGZAG\_EQ\_0 constraint; it describes all sequences containing no occurrence of the ZIGZAG pattern on a sequence of variables; it is derived from the automaton that counts the number of occurrences of the ZIGZAG pattern by removing (1) the register $R$, (2) the transitions labelled by found, i.e. transitions $b \rightarrow c$ and $e \rightarrow f$ that increment $R$, and (3) the states $c$ and $f$ that become unreachable after removing transitions $b \rightarrow c$ and $e \rightarrow f$. 
Figure 4.1086: Automaton without registers for the \texttt{NB\_ZIGZAG\_EQ\_UP\_WHEN\_RANGE\_EQ\_2} constraint; it describes all sequences containing the maximum number of occurrences of the ZIGZAG pattern when the difference between the maximum and the minimum of the variables plus one of the sequence of $sv$ variables is equal to 2, i.e. $\left\lfloor \frac{sv}{4} \right\rfloor$ of the \textbf{Restrictions} slot (see 1); transitions in blue correspond to a new occurrence of the ZIGZAG pattern, dashed transitions to slack, and accepting states have a light-brown background; states $c$ and $f$ are accepting when $sv \mod 4 = 0$, states $s$, $r$, $t$, $c'$ and $f'$ are accepting when $sv \mod 4 = 1$, states $a$, $d$, $s'$, $r'$, $t'$, $c''$ and $f''$ are accepting when $sv \mod 4 = 2$, states $b$, $e$, $a'$, $d'$, $s''$, $r''$, $t''$, $c'''$ and $f'''$ are accepting when $sv \mod 4 = 3$. 
Figure 4.1087: (top) Automaton without registers for the NB_ZIGZAG_EQ_UP constraint; it describes all sequences containing the maximum number of occurrences of the ZIGZAG pattern when the difference between the maximum and the minimum of the variables plus one of the sequence of $sv$ variables is greater than or equal to 3, i.e. the upper bound $\max(0, \lfloor \frac{sv-1}{3} \rfloor)$ of the Restrictions slot (see \S); transitions in blue correspond to a new occurrence of the ZIGZAG pattern, dashed transitions to slack, and accepting states have a light-brown background; states $s$, $c$, and $f$ are accepting when $sv \mod 3 = 1$, states $s'$, $a$, $d$, $c'$ and $f'$ are accepting when $sv \mod 3 = 2$, states $s''$, $b$, $e$, $d'$, $d''$, $c''$ and $f''$ are accepting when $sv \mod 3 = 0$. (bottom) All corresponding solutions for $sv - 1 \in \{3, 4, 5\}$.
**Origin** Based on constraint MAX_HEIGHT_DECREASING_TERRACE.

**Constraint**

POS_MAX_HEIGHT_DECREASING_TERRACE(VALUE, VARIABLES, FOUND)

**Arguments**

- VALUE: dvar
- VARIABLES: collection(var−dvar)
- FOUND: collection(var−dvar)

**Restrictions**

\[
\text{sv} \leq 3 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = -\infty
\]

\[
\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1
\]

\[
\text{VALUE} \leq \text{maxv} - 1
\]

required(VARIABLES, var)

required(FOUND, var)

\[
\text{|VARIABLES|} = \text{|FOUND|}
\]

where

\[
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
\]

\[
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})
\]

\[
\text{sv} = \text{|VARIABLES|}
\]

\[
\text{rv} = \text{range}(\text{VARIABLES}.\text{var})
\]

The constraint MAX_HEIGHT_DECREASING_TERRACE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING_TERRACE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression ‘\(>\rightarrow\)’.

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

**Purpose**

**Example**

\[
(4, [6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
\]

Figure 4.1088 provides an example where the POS_MAX_HEIGHT_DECREASING_TERRACE constraint holds.
Figure 4.1088: Illustrating the POS_MAX_HEIGHT_DECREASING_TERRACE constraint of the Example slot

Typical

- |VARIABLES| > 3
- range(VARIABLES.var) > 2

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the POS_MAX_HEIGHT_DECREASING_TERRACE constraint but use the decoration table 3.35.
**POS_MAX_HEIGHT_INCREASING_TERRACE**

**Origin**
Based on constraint MAX_HEIGHT_INCREASING_TERRACE.

**Constraint**
POS_MAX_HEIGHT_INCREASING_TERRACE(VALUE, VARIABLES, FOUND)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>FOUND</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
sv \leq 3 & \lor rv \leq 2 \Rightarrow VALUE = -\infty \\
VALUE = -\infty & \lor VALUE \geq \minv + 1 \\
VALUE \leq \maxv - 1 & \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
|\text{VARIABLES}| = |\text{FOUND}| &
\end{align*}
\]

where

\[
\begin{align*}
\maxv = & \maxval(\text{VARIABLES}.\text{var}) \\
\minv = & \minval(\text{VARIABLES}.\text{var}) \\
sv = & |\text{VARIABLES}| \\
rv = & \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint MAX_HEIGHT_INCREASING_TERRACE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING_TERRACE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \( '<=+=<\)'.

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Purpose**

**Example**

\[
\begin{pmatrix}
5, \langle 1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4 \rangle, \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1089 provides an example where the POS_MAX_HEIGHT_INCREASING_TERRACE \((5, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])\) constraint holds.
Figure 4.1089: Illustrating the POS_MAX_HEIGHT_INCREASING_TERRACE constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 2 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MAX_HEIGHT_INCREASING_TERRACE` constraint but use the decoration table 3.35.
Origin: Based on constraint MAX_HEIGHTPLAIN.

Constraint: POS_MAX_HEIGHTPLAIN(VALUE, VARIABLES, FOUND)

Arguments:
- VALUE: dvar
- VARIABLES: collection(var−dvar)
- FOUND: collection(var−dvar)

Restrictions:
- $sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty$
- $VALUE = -\infty \lor VALUE \geq \minv$
- $VALUE \leq \maxv - 1$
- required(VARIABLES, var)
- required(FOUND, var)
- $|VARIABLES| = |FOUND|$

where
- $\maxv = \maxval(VARIABLES.var)$
- $\minv = \minval(VARIABLES.var)$
- $sv = |VARIABLES|$
- $rv = \text{range}(VARIABLES.var)$

Purpose:
The constraint MAX_HEIGHTPLAIN(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PLAIN for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘$>$’=’$<$’.

Assume that the occurrence of the pattern PLAIN starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i + 1$ to index $j$.

Example:

$\begin{pmatrix} 5, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3), \ldots \end{pmatrix}$,

$\begin{pmatrix} 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \end{pmatrix}$

Figure 4.1090 provides an example where the POS_MAX_HEIGHTPLAIN
($5, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$)

constraint holds.
Figure 4.1090: Illustrating the $\text{POS\_MAX\_HEIGHT\_PLAIN}$ constraint of the Example slot

Typical

$|\text{VARIABLES}| > 2$

$\text{range} (\text{VARIABLES}.\text{var}) > 1$

Arg. properties

- Functional dependency: VALUE determined by $\text{VARIABLES}$.
- Functional dependency: FOUND determined by $\text{VARIABLES}$. 
Automaton  Similar to the automaton of the MAX_HEIGHT_PLAIN constraint but use the decoration table 3.35.
Origin
Based on constraint \textit{MAX\_HEIGHT\_PLATEAU}.

Constraint

\texttt{POS\_MAX\_HEIGHT\_PLATEAU(VALUE, VARIABLES, FOUND)}

Arguments

\begin{itemize}
  \item \texttt{VALUE} : \textit{dvar}
  \item \texttt{VARIABLES} : \textit{collection(var–dvar)}
  \item \texttt{FOUND} : \textit{collection(var–dvar)}
\end{itemize}

Restrictions

\begin{itemize}
  \item $sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty$
  \item $\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1$
  \item $\text{VALUE} \leq \text{maxv}$
  \item \texttt{required(VARIABLES, var)}
  \item \texttt{required(FOUND, var)}
  \item $|\text{VARIABLES}| = |\text{FOUND}|$
\end{itemize}

where

\begin{itemize}
  \item $\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})$
  \item $\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})$
  \item $sv = |\text{VARIABLES}|$
  \item $rv = \text{range}(\text{VARIABLES}.\text{var})$
\end{itemize}

Purpose

The constraint \textit{MAX\_HEIGHT\_PLATEAU(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{PLATEAU} for which the feature value is \texttt{VALUE}. The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{PLATEAU} is the maximal subsequence which matches the regular expression \texttt{'<=' '>'}. Assume that the occurrence of the pattern \texttt{PLATEAU} starts at position $i$ and ends at position $j$. The feature \texttt{MIN}, called \texttt{HEIGHT} in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i+1$ to index $j$.

Example

\begin{itemize}
  \item $\left( 5, (7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5), (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0) \right)$
\end{itemize}

Figure \ref{fig:constraint-example} provides an example where the \texttt{POS\_MAX\_HEIGHT\_PLATEAU} constraint holds.
Figure 4.1091: Illustrating the `POS_MAX_HEIGHT_PLATEAU` constraint of the Example slot

**Typical**

\[ |\text{VARIABLES} \rangle > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
| Automaton | Similar to the automaton of the MAX_HEIGHT_PLATEAU constraint but use the decoration table 3.35. |
Origin Based on constraint \texttt{MAX\_HEIGHT\_PROPER\_PLAIN}.

**Constraint**

\texttt{POS\_MAX\_HEIGHT\_PROPER\_PLAIN(VALUE, VARIABLES, FOUND)}

**Arguments**

- \texttt{VALUE : dvar}
- \texttt{VARIABLES : collection(var\textendash dvar)}
- \texttt{FOUND : collection(var\textendash dvar)}

**Restrictions**

\begin{align*}
\text{sv} & \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq \text{minv} \\
\text{VALUE} & \leq \text{maxv} - 1 \\
\text{required}(\text{VARIABLES}, \text{var}) & \Rightarrow \text{required}(\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| & = |\text{FOUND}| \\
\text{where} & \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}

The constraint \texttt{MAX\_HEIGHT\_PROPER\_PLAIN(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{PROPER\_PLAIN} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{PROPER\_PLAIN} is the maximal subsequence which matches the regular expression '\texttt{> = + <}'.

Assume that the occurrence of the pattern \texttt{PROPER\_PLAIN} starts at position \(i\) and ends at position \(j\). The feature \texttt{MIN}, called \texttt{HEIGHT} in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\begin{equation*}
\begin{pmatrix}
5, \langle 2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5 \rangle, \\
\langle 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle
\end{pmatrix}
\end{equation*}

Figure 4.1092 provides an example where the \texttt{POS\_MAX\_HEIGHT\_PROPER\_PLAIN} constraint holds.
Figure 4.1092: Illustrating the POS.MAX_HEIGHT.PROPER.PLAIN constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_HEIGHT_PROPER_PLAIN constraint but use the decoration table 3.35.
**Description**

Based on constraint `MAX_HEIGHT_PROPER_PLATEAU`.

**Constraint**

`POS_MAX_HEIGHT_PROPER_PLATEAU(VALUE, VARIABLES, FOUND)`

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`
- **FOUND**: `collection(var−dvar)`

**Restrictions**

- `sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ minv + 1`
- `VALUE ≤ maxv`
- `required(VARIABLES, var)`
- `required(FOUND, var)`
- `|VARIABLES| = |FOUND|`

where

- `maxv = maxval(VARIABLES.var)`
- `minv = minval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `MAX_HEIGHT_PROPER_PLATEAU(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `PROPER_PLATEAU` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `PROPER_PLATEAU` is the **maximal** subsequence which matches the regular expression `'< =+ >'`.

Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position `i` and ends at position `j`. The feature `MIN`, called `HEIGHT` in the name of the constraint since all feature values are identical, computes the minimum of the values from index `i + 1` to index `j`.

**Example**

```
( 5, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])
```

Figure 4.1093 provides an example where the `POS_MAX_HEIGHT_PROPER_PLATEAU` constraint holds.
Figure 4.1093: Illustrating the POS_MAX_HEIGHT_PROPER_PLATEAU constraint of the Example slot

**Typical**

\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Similar to the automaton of the MAX_HEIGHT_PROPER_PLATEAU constraint but use the decoration table 3.35.
POS_MAX_HEIGHT_PROPER_PLATEAU

2115
POS_MAX_HEIGHT_STEADY

Origin
Based on constraint MAX_HEIGHT_STEADY.

Constraint
POS_MAX_HEIGHT_STEADY(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|

The constraint MAX_HEIGHT_STEADY(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STEADY for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern STEADY is the subsequence which matches the regular expression ‘=’.

Assume that the occurrence of the pattern STEADY starts at position i and ends at position j. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index i to index j + 1.

Example
([6, 1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])

Figure 4.1094 provides an example where the POS_MAX_HEIGHT_STEADY (6, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.

Typical
|VARIABLES| > 1
Figure 4.1094: Illustrating the POS_MAX_HEIGHT_STEADY constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton Similar to the automaton of the MAX_HEIGHT_STEADY constraint but use the decoration table 3.35.
Origin: Based on constraint \textit{MAX\_HEIGHT\_STEADY\_SEQUENCE}.

Constraint: $\text{POS\_MAX\_HEIGHT\_STEADY\_SEQUENCE}(\text{VALUE, VARIABLES, FOUND})$

Arguments:

\begin{itemize}
\item \textbf{VALUE}: $\text{dvar}$
\item \textbf{VARIABLES}: \text{collection}(\text{var} \text{--} \text{dvar})
\item \textbf{FOUND}: \text{collection}(\text{var} \text{--} \text{dvar})
\end{itemize}

Restrictions:

\begin{align*}
sv \leq 1 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty & \lor \text{VALUE} \geq \text{minv} \\
\text{VALUE} & \leq \text{maxv} \\
\text{required} & (\text{VARIABLES, var}) \\
\text{required} & (\text{FOUND, var}) \\
|\text{VARIABLES}| & = |\text{FOUND}|
\end{align*}

where:

\begin{align*}
\text{maxv} & = \text{maxval}(\text{VARIABLES, var}) \\
\text{minv} & = \text{minval}(\text{VARIABLES, var}) \\
sv & = |\text{VARIABLES}|
\end{align*}

The constraint $\text{MAX\_HEIGHT\_STEADY\_SEQUENCE}(\text{VALUE, VARIABLES})$ holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the \textit{found} letter in those occurrences of the pattern \textit{STEADY\_SEQUENCE} for which the feature value is \textit{VALUE}.

The position of the \textit{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \textit{STEADY\_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression \texttt{\textasciitilde=+\texttt{\textasciitilde}}.

Assume that the occurrence of the pattern \textit{STEADY\_SEQUENCE} starts at position $i$ and ends at position $j$. The feature \textit{MIN}, called \textit{HEIGHT} in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i$ to index $j + 1$.

Purpose:

Example:

\begin{equation}
\begin{pmatrix}
5, (3, 1, 1, 4, 5, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 0), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\end{equation}

Figure 4.1095 provides an example where the $\text{POS\_MAX\_HEIGHT\_STEADY\_SEQUENCE}$ constraint holds.
Figure 4.1095: Illustrating the **POS_MAX_HEIGHT_STEADY_SEQUENCE** constraint of the **Example** slot

**Typical**

|VARIABLES| > 1

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_HEIGHT_STEADY_SEQUENCE constraint but use the decoration table 3.35.
Based on constraint MAX_MAX_BUMP_ON_DECREASING_SEQUENCE.

Aggregator Feature Pattern
POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE

Description Automaton

Origin

Constraint

Arguments

Restrictions

The constraint MAX_MAX_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern BUMP_ON_DECREASING_SEQUENCE for which the feature value is VALUE.

Purpose

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression ‘>>>’.

Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i + 2$ to index $j$.

Example

\[
\begin{pmatrix}
6, 7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1096 provides an example where the POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE (6, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 4.1096: Illustrating the POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

**Typical**

|VARIABLES| > 5  
range(VARIABLES.var) > 2

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MAX_BUMP_ON_DECREASING_SEQUENCE constraint but use the decoration table 3.35.
Based on constraint MAX_MAX_DECREASING.

**Constraint**

\[ \text{POS\_MAX\_MAX\_DECREASING}(\text{VALUE}, \text{VARIABLES}, \text{FOUND}) \]

**Arguments**

- **VALUE**: \text{dvar}
- **VARIABLES**: \text{collection(var\_dvar)}
- **FOUND**: \text{collection(var\_dvar)}

**Restrictions**

\[ \begin{align*}
sv \leq 1 \lor rv \leq 1 &\Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1 &\Rightarrow \text{VALUE} \leq \text{maxv} \\
\text{required}(\text{VARIABLES}, \text{var}) &\Rightarrow \text{required}(\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| = |\text{FOUND}| &\Rightarrow \text{where} \\
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \\
sv = |\text{VARIABLES}| \\
rv = \text{range}(\text{VARIABLES}.\text{var})
\end{align*} \]

The constraint \text{MAX\_MAX\_DECREASING}(\text{VALUE}, \text{VARIABLES}) holds. In addition, \text{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{DECREASING} for which the feature \text{value} is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{DECREASING} is the subsequence which matches the regular expression ‘\text{>}''.

Assume that the occurrence of the pattern \text{DECREASING} starts at position \(i\) and ends at position \(j\). The feature \text{MAX} computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
\begin{pmatrix}
6, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4), \& \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1097 provides an example where the \text{POS\_MAX\_MAX\_DECREASING} \(6, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0])\) constraint holds.
Figure 4.1097: Illustrating the POS_MAX_MAX_DECREASING constraint of the Example slot

Typical

$|\text{VARIABLES}| > 1$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MAX_DECREASING constraint but use the decoration table 3.35.
POS_MAX_MAX_DECREASING

2131
**POS_MAX_MAX_DECREASING_SEQUENCE**

<table>
<thead>
<tr>
<th>▶ ▷ ◄ ◀</th>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
</table>

**Origin**
Based on constraint `MAX_MAX_DECREASING_SEQUENCE`.

**Constraint**
`POS_MAX_MAX_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)`

**Arguments**
- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`
- `FOUND`: `collection(var−dvar)`

**Restrictions**
\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
VALUE = -\infty \lor VALUE \geq \text{minv} + 1 \\
VALUE \leq \text{maxv} \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{required}(\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| = |\text{FOUND}|
\]

where
- `maxv = \maxval(\text{VARIABLES}.\text{var})`
- `minv = \minval(\text{VARIABLES}.\text{var})`
- `sv = |\text{VARIABLES}|`
- `rv = \text{range}(\text{VARIABLES}.\text{var})`

The constraint `MAX_MAX_DECREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `DECREASING_SEQUENCE` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `’>(>|=)^{*}>|>’`

Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MAX` computes the maximum of the values from index `i` to index `j + 1`.

**Purpose**

**Example**

\[
\left(6, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 1, 6, 4, 4], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0] \right)
\]

Figure 4.1098 provides an example where the `POS_MAX_MAX_DECREASING_SEQUENCE` constraint holds.
Figure 4.1098: Illustrating the **POS_MAX_MAX_DECREASING_SEQUENCE** constraint of the **Example** slot

**Typical**

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MAX_DECREASING_SEQUENCE constraint but use the decoration table 3.35.
POS_MAX_MAX_DECREASING_SEQUENCE

2135
POS_MAX_MAX_DIP_ON_INCREMENTING_SEQUENCE

Origin
Based on constraint MAX_MAX_DIP_ON_INCREMENTING_SEQUENCE.

Constraint
POS_MAX_MAX_DIP_ON_INCREMENTING_SEQUENCE(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 5 ∨ rv ≤ 2 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv + 2
VALUE ≤ maxv
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

The constraint MAX_MAX_DIP_ON_INCREMENTING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DIP_ON_INCREMENTING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DIP_ON_INCREMENTING_SEQUENCE is the subsequence which matches the regular expression ‘<<><<<’.

Assume that the occurrence of the pattern DIP_ON_INCREMENTING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 2 to index j.

Purpose

Example

Figure 4.1099 provides an example where the POS_MAX_MAX_DIP_ON_INCREMENTING_SEQUENCE (6, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.
POS_MAX_MAX_DIP_ON_INCREASING_SEQUENCE

Figure 4.1099: Illustrating the POS_MAX_MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot

**Typical**

| VARIABLES | > 5  
range(VARIABLES.var) | > 2

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the **MAX_MAX_DIP_ON_INCREASING_SEQUENCE** constraint but use the decoration table 3.35.
POS_MAX_MAX_DIP_ON_INCREASING_SEQUENCE
Origin | Based on constraint MAX_MAX_INCREASING.

Constraint | POS_MAX_MAX_INCREASING(VALUE, VARIABLES, FOUND)

Arguments | VALUE : dvar
| VARIABLES : collection(var-dvar)
| FOUND : collection(var-dvar)

Restrictions | $sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty$
| $VALUE = -\infty \lor VALUE \geq minv + 1$
| $VALUE \leq maxv$
| required(VARIABLES, var)
| required(FOUND, var)
| $|VARIABLES| = |FOUND|$

where
| maxv = maxval(VARIABLES.var)
| minv = minval(VARIABLES.var)
| sv = |VARIABLES|
| rv = range(VARIABLES.var)

Purpose | The constraint MAX_MAX_INCREASING(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression ‘<’.

Assume that the occurrence of the pattern INCREASING starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i$ to index $j+1$.

Example | $(6,\{4,3,5,5,2,1,1,3,3,4,6,6,3,1,3,3\},\{0,0,0,0,0,0,0,0,1,0,0,0,0,0\})$

Figure 4.1100 provides an example where the POS_MAX_MAX_INCREASING (6, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]) constraint holds.
POS_MAX_MAX_INCREASING

Figure 4.1100: Illustrating the POS_MAX_MAX_INCREASING constraint of the Example slot

Typical

$|\text{VARIABLES}| > 1$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
| Automaton | Similar to the automaton of the MAX_MAX_INCREASING constraint but use the decoration table 3.35. |
POS_MAX_MAX_INCREASING_SEQUENCE

DESCRIPTION

Origin
Based on constraint \textit{MAX\_MAX\_INCREASING\_SEQUENCE}.

Constraint
\texttt{POS\_MAX\_MAX\_INCREASING\_SEQUENCE\(\text{VALUE, VARIABLES, FOUND}\)}

Arguments
\begin{itemize}
  \item \texttt{VALUE} : \texttt{dvar}
  \item \texttt{VARIABLES} : \texttt{collection(var\_dvar)}
  \item \texttt{FOUND} : \texttt{collection(var\_dvar)}
\end{itemize}

Restrictions
\begin{itemize}
  \item \(sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty\)
  \item \(VALUE = -\infty \lor VALUE \geq \text{minv} + 1\)
  \item \(VALUE \leq \text{maxv}\)
  \item \texttt{required(VARIABLES, var)}
  \item \texttt{required(FOUND, var)}
  \item \(|\text{VARIABLES}| = |\text{FOUND}|\)
\end{itemize}

where
\begin{itemize}
  \item \texttt{maxv = maxval(VARIABLES.var)}
  \item \texttt{minv = minval(VARIABLES.var)}
  \item \texttt{sv = |VARIABLES|}
  \item \texttt{rv = range(VARIABLES.var)}
\end{itemize}

Purpose
The constraint \textit{MAX\_MAX\_INCREASING\_SEQUENCE\(\text{VALUE, VARIABLES}\)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{INCREASING\_SEQUENCE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of the pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{INCREASING\_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression ‘\(< (< | =) \ast < | \ast."

Assume that the occurrence of the pattern \texttt{INCREASING\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{MAX} computes the maximum of the values from index \(i\) to index \(j + 1\).

Example
\begin{itemize}
  \item \(\langle 6, 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3 \rangle, \langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle\)
\end{itemize}

Figure 4.1101 provides an example where the \texttt{POS\_MAX\_MAX\_INCREASING\_SEQUENCE(6, \langle 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3 \rangle, \langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle)} constraint holds.
POS_MAX_MAX_INCREASING_SEQUENCE

Figure 4.1101: Illustrating the POS_MAX_MAX_INCREASING_SEQUENCE constraint of the Example slot

Typical

| VARIABLES | > 1
| range(VARIABLES.var) | > 1 |

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_MAX\_INCREASING\_SEQUENCE} constraint but use the decoration table \ref{tab:3.35}. 
Origin

Based on constraint MAX_MAX_INFLEXION.

Constraint

POS_MAX_MAX_INFLEXION(VALUE, VARIABLES, FOUND)

Arguments

VALUE : dvar
VARIABLES : collection(var–dvar)
FOUND : collection(var–dvar)

Restrictions

sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE > minv
VALUE ≤ maxv
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose

The constraint MAX_MAX_INFLEXION(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INFLEXION for which the feature value is VALUE.

An occurrence of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

Assume that the occurrence of the pattern INFLEXION starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 1 to index j.

Example

\[
\begin{bmatrix}
6, (1, 2, 6, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4),
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1
\end{bmatrix}
\]

Figure 4.1102 provides an example where the POS_MAX_MAX_INFLEXION constraint holds.
Figure 4.1102: Illustrating the POS_MAX_MAX_INFLEXION constraint of the Example slot

Typical

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
|\text{VARIABLES}| > 2
\]
\[
\text{range(\text{VARIABLES}.var)} > 1
\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MAX_INFLEXION constraint but use the decoration table 3.35.
Origin

Based on constraint \texttt{MAX\_MAX\_PEAK}.

Constraint

\texttt{POS\_MAX\_MAX\_PEAK(VALUE, VARIABLES, FOUND)}

Arguments

\begin{itemize}
  \item \texttt{VALUE} : \texttt{dvar}
  \item \texttt{VARIABLES} : \texttt{collection(var-dvar)}
  \item \texttt{FOUND} : \texttt{collection(var-dvar)}
\end{itemize}

Restrictions

\begin{align*}
sv \leq 2 & \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
VALUE & = -\infty \lor VALUE \geq \text{minv} + 1 \\
VALUE & \leq \text{maxv}
\end{align*}

\begin{itemize}
  \item \texttt{required(VARIABLES, var)}
  \item \texttt{required(FOUND, var)}
  \item \texttt{|VARIABLES| = |FOUND|}
\end{itemize}

where

\begin{itemize}
  \item \texttt{maxv = maxval(VARIABLES.var)}
  \item \texttt{minv = minval(VARIABLES.var)}
  \item \texttt{sv = |VARIABLES|}
  \item \texttt{rv = range(VARIABLES.var)}
\end{itemize}

Purpose

The constraint \texttt{MAX\_MAX\_PEAK(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{PEAK} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{PEAK} is the \textit{maximal} subsequence which matches the regular expression ‘\texttt{\(=\ | <\)(\(>|\)=\)*}’.

Assume that the occurrence of the pattern \texttt{PEAK} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{MAX} computes the maximum of the values from index \texttt{i + 1} to index \texttt{j}.

Example

\begin{itemize}
  \item \texttt{(6,\{7,5,5,1,4,5,2,2,3,5,6,2,3,3,3,1\},\{0,0,0,0,0,0,0,0,0,1,0,0,0,0\})}
\end{itemize}

Figure 4.1103 provides an example where the \texttt{POS\_MAX\_MAX\_PEAK} constraint holds.
Figure 4.1103: Illustrating the POS_MAX_MAX_PEAK constraint of the Example slot

Typical

\[
\text{Variables} \geq 2 \\
\text{range}(\text{Variables}.\text{var}) > 1
\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
**Automaton**

Similar to the automaton of the MAX\_MAX\_PEAK constraint but use the decoration table 3.35.
POS_MAX_MAX_PEAK

2155
The constraint \( \text{POS\_MAX\_MAX\_Strictly\_Decreasing\_Sequence} \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \( \text{found} \) letter in those occurrences of the pattern \( \text{Strictly\_Decreasing\_Sequence} \) for which the feature value is \( \text{VALUE} \).

The position of the \( \text{found} \) letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern \( \text{Strictly\_Decreasing\_Sequence} \) is the maximal subsequence which matches the regular expression ‘\( >^+ \)’.

Assume that the occurrence of the pattern \( \text{Strictly\_Decreasing\_Sequence} \) starts at position \( i \) and ends at position \( j \). The feature \( \text{MAX} \) computes the maximum of the values from index \( i \) to index \( j + 1 \).

The \( \text{POS\_MAX\_MAX\_Strictly\_Decreasing\_Sequence} \) constraint holds.
Figure 4.1104: Illustrating the POS_MAX_MAX_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MAX.Strictly-Decreasing-Sequence constraint but use the decoration table 3.35.
POS_MAX_MAX_STRICTLY_DECREASING_SEQUENCE
Based on constraint \texttt{MAX\_MAX\_STRICTLY\_INCREASING\_SEQUENCE}.

The constraint \texttt{MAX\_MAX\_STRICTLY\_INCREASING\_SEQUENCE}(VALUE, VARIABLES, FOUND) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{STRICTLY\_INCREASING\_SEQUENCE} for which the feature value is VALUE.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

Assume that the occurrence of the pattern \texttt{STRICTLY\_INCREASING\_SEQUENCE} starts at position \texttt{i} and ends at position \texttt{j}. The feature MAX computes the maximum of the values from index \texttt{i} to index \texttt{j} + 1.

Figure 4.1105 provides an example where the \texttt{POS\_MAX\_MAX\_STRICTLY\_INCREASING\_SEQUENCE} (6, [4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 4.1105: Illustrating the POS_MAX_MAX STRICTLY_INCREASINGSEQUENCE constraint of the Example slot

Typical

\[ |\text{VARIABLES} | > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_MAX\_STRICTLY\_INCREASING\_SEQUENCE} constraint but use the decoration table \texttt{3.35}. 
POS_MAX_MAX_STRICTLY_INCREASING_SEQUENCE  2163
The constraint MAX_MAX_SUMMIT(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern SUMMIT for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression ‘(< | < (= | <)∗ <) (> | = >)∗ >)’.

Assume that the occurrence of the pattern SUMMIT starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 1 to index j.

Example

\[
(5, [7, 1, 5, 4, 3, 4, 6, 2, 3, 4, 2, 3, 1], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
\]

Figure 4.1106 provides an example where the POS_MAX_MAX_SUMMIT constraint holds.
Figure 4.1106: Illustrating the POS_MAX_MAX_SUMMIT constraint of the Example slot

Typical

|VARIABLES| > 2
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MAX_MAX_SUMMIT` constraint but use the decoration table `3.35`. 
2168

Origin
Based on constraint MAX_MAX_ZIGZAG.

Constraint
POS_MAX_MAX_ZIGZAG(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv + 1
VALUE ≤ maxv
required(VARIABLES,var)
required(FOUND,var)
|VARIABLES| = |FOUND|
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
The constraint MAX_MAX_ZIGZAG(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern ZIGZAG for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘(<>)+(< | <>)(<>)+(| ><>). Assume that the occurrence of the pattern ZIGZAG starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 1 to index j.

Example

\[
(7, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
\]

Figure 4.1107 provides an example where the POS_MAX_MAX_ZIGZAG (7, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 4.1107: Illustrating the POS_MAX_MAX_ZIGZAG constraint of the Example slot

**Typical**

$|\text{VARIABLES}| > 3$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \text{MAX\_MAX\_ZIGZAG} constraint but use the decoration table 3.35.
Based on constraint MAX_MIN_BUMP_ON_DECREASING_SEQUENCE, the constraint

\[ \text{POS\_MAX\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE}(\text{VALUE, VARIABLES, FOUND}) \]

holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the \textit{found} letter in those occurrences of the pattern BUMP_ON_DECREASING_SEQUENCE for which the feature value is \textit{VALUE}.

The position of the \textit{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression ‘\texttt{>>><>}'.

Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 2\) to index \(j\).

\[ \text{MIN} = \min_{i+2}^{j} \]

Figure 4.1108 provides an example where the POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE constraint holds.
Figure 4.1108: Illustrating the POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

Typical

| VARIABLES | > 5  
| range(VARIABLES.var) | > 2

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MIN_BUMP_ON_DECREASING_SEQUENCE constraint but use the decoration table 3.35.
Based on constraint \texttt{MAX\_MIN\_DECREASING}.

\textbf{Arguments} \begin{itemize} \item \texttt{VALUE} : \texttt{dvar} \item \texttt{VARIABLES} : \texttt{collection(var-dvar)} \item \texttt{FOUND} : \texttt{collection(var-dvar)} \end{itemize}

\textbf{Restrictions} \begin{align*} sv &\leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
VALUE &\leq -\infty \lor VALUE \geq \text{minv} \\
VALUE &\leq \text{maxv} - 1 \\
\text{required} &\left(\text{VARIABLES}, \text{var} \right) \\
\text{required} &\left(\text{FOUND}, \text{var} \right) \\
|\text{VARIABLES}| &\leq |\text{FOUND}| \\
\text{where} & \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}\text{.var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES}\text{.var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}\text{.var}) \end{align*}

The \texttt{constraint \texttt{MAX\_MIN\_DECREASING}(VALUE, VARIABLES, FOUND)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{DECREASING} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{DECREASING} is the subsequence which matches the regular expression \texttt{'>'}. Assume that the occurrence of the pattern \texttt{DECREASING} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{MIN} computes the minimum of the values from index \texttt{i} to index \texttt{j} + 1.

\textbf{Example} \begin{align*} \begin{pmatrix} 4, \langle 3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 4, 6, 4, 4 \rangle, \\
\langle 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0 \rangle \end{pmatrix} \end{align*}

Figure 4.1109 provides an example where the \texttt{POS\_MAX\_MIN\_DECREASING} \texttt{(4, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 4, 6, 4, 4], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0])} constraint holds.
Figure 4.1109: Illustrating the **POS_MAX_MIN_DECREASING** constraint of the **Example** slot

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range(}\text{VARIABLES.var}) > 1
\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MIN_DECREASING constraint but use the decoration table \textit{3.35}. 
Origin
Based on constraint \texttt{MAX_MIN_DECREASING_SEQUENCE}.

Constraint
\texttt{POS_MAX_MIN_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)}

Arguments
\begin{itemize}
\item \texttt{VALUE} : \texttt{dvar}
\item \texttt{VARIABLES} : \texttt{collection(var\texttt{--}dvar)}
\item \texttt{FOUND} : \texttt{collection(var\texttt{--}dvar)}
\end{itemize}

Restrictions
\begin{align*}
& sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
& VALUE = -\infty \lor VALUE \geq \text{minv} \\
& VALUE \leq \text{maxv} - 1 \\
& \text{required}(\text{VARIABLES}, \text{var}) \\
& \text{required}(\text{FOUND}, \text{var}) \\
& |\text{VARIABLES}| = |\text{FOUND}|
\end{align*}

where
\begin{align*}
& \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \\
& \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \\
& sv = |\text{VARIABLES}| \\
& rv = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}

The constraint \texttt{MAX_MIN_DECREASING_SEQUENCE(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{DECREASING_SEQUENCE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{DECREASING_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression \texttt{’> (>|=)** > | ’}.

Assume that the occurrence of the pattern \texttt{DECREASING_SEQUENCE} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{MIN} computes the minimum of the values from index \texttt{i} to index \texttt{j} + 1.

Example
\[
\left(\begin{array}{cccccccccc}
4, & 3, & 4, & 2, & 2, & 5, & 6, & 6, & 4, & 4 \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0
\end{array}\right)
\]

Figure 4.1110 provides an example where the \texttt{POS_MAX_MIN_DECREASING_SEQUENCE} constraint holds.
Figure 4.1110: Illustrating the POS_MAX_MIN_DECREASING_SEQUENCE constraint of the Example slot

Typical

$|\text{VARIABLES}| > 1$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MIN_DECREASING_SEQUENCE constraint but use the decoration table 3.35.
Origin

Based on constraint \texttt{MAX_MIN_DIP_ON_INCREASING_SEQUENCE}.

Constraint

\[
\text{POS\_MAX\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar}) \\
\text{FOUND} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} \leq 5 \land \text{rv} \leq 2 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty & \lor \text{VALUE} \geq \text{minv} \\
\text{VALUE} & \leq \text{maxv} - 2 \\
\text{required}(&\text{VARIABLES}, \text{var}) \\
\text{required}(&\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| & = |\text{FOUND}|
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint \texttt{MAX_MIN_DIP_ON_INCREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES}) holds. In addition, \text{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \texttt{DIP\_ON\_INCREASING\_SEQUENCE} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{DIP\_ON\_INCREASING\_SEQUENCE} is the subsequence which matches the regular expression ‘<<><<’.

Assume that the occurrence of the pattern \texttt{DIP\_ON\_INCREASING\_SEQUENCE} starts at position \text{i} and ends at position \text{j}. The feature \text{MIN} computes the minimum of the values from index \text{i} + 2 to index \text{j}.

Example

\[
\begin{pmatrix}
2, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4), \\
(0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 4.1111 provides an example where the \texttt{POS\_MAX\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE} (2, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Typical

$|\text{VARIABLES}| > 5$

$\text{range}(\text{VARIABLES.var}) > 2$

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE} constraint but use the decoration table 3.35.
2188

AGGREGATOR FEATURE PATTERN
POS_MAX_MIN_GORGE

DESCRIPTION AUTOMATON

Origin
Based on constraint MAX_MIN_GORGE.

Constraint

\[ \text{POS_MAX_MIN_GORGE}(\text{VALUE}, \text{VARIABLES}, \text{FOUND}) \]

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>FOUND</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

Restrictions

\[ sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \]
\[ \text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} \]
\[ \text{VALUE} \leq \text{maxv} - 1 \]
\[ \text{required} (\text{VARIABLES}.\text{var}) \]
\[ \text{required} (\text{FOUND}.\text{var}) \]
\[ |\text{VARIABLES}| = |\text{FOUND}| \]

where

\[ \text{maxv} = \text{maxval} (\text{VARIABLES}.\text{var}) \]
\[ \text{minv} = \text{minval} (\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range} (\text{VARIABLES}.\text{var}) \]

The constraint MAX_MIN_GORGE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern GORGE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \( (^{>}(=|^{>})^{*})(^{<}(=|^{<})^{*})^{<} \).

Assume that the occurrence of the pattern GORGE starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

Purpose

Example

\[ 5, (1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7), (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0) \]

Figure 4.1112 provides an example where the POS_MAX_MIN_GORGE (5, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.
Figure 4.1112: Illustrating the POS_MAX_MIN_GORGE constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 2 \]

\[ \text{range}(\text{VARIABLES}\.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MAX_MIN_GORGE` constraint but use the decoration table 3.35.
AGGREGATOR FEATURE PATTERN

### POS_MAX_MIN_INCREASING

#### DESCRIPTION

**Origin**
Based on constraint `MAX_MIN_INCREASING`.

**Constraint**

```
POS_MAX_MIN_INCREASING(VALUE, VARIABLES, FOUND)
```

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`
- **FOUND**: `collection(var−dvar)`

**Restrictions**

```
sv ≤ 1 ∧ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv − 1
required(VARIABLES, var)
required(FOUND, var)
\(|\text{VARIABLES}| = |\text{FOUND}|
```

where

- `maxv = maxval(VARIABLES.var)`
- `minv = minval(VARIABLES.var)`
- `sv = |\text{VARIABLES}|`
- `rv = range(VARIABLES.var)`

The constraint `MAX_MIN_INCREASING(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `FOUND` letter in those occurrences of the pattern `INCREASING` for which the feature value is `VALUE`.

The position of the `FOUND` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern `INCREASING` is the subsequence which matches the regular expression '<'. Assume that the occurrence of the pattern `INCREASING` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Purpose**

**Example**

```
(4, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0])
```

Figure 4.1113 provides an example where the `POS_MAX_MIN_INCREASING` constraint holds.
Figure 4.1113: Illustrating the POS_MAX_MIN_INCREASING constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \textit{MAX\_MIN\_INCREASING} constraint but use the decoration table \ref{table:decoration}.
The constraint \( \text{MAX\_MIN\_INCREASING\_SEQUENCE} \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{INCREASING\_SEQUENCE} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern \text{INCREASING\_SEQUENCE} is the maximal subsequence which matches the regular expression ‘\(< (< | =) * < | <'’. Assume that the occurrence of the pattern \text{INCREASING\_SEQUENCE} starts at position \( i \) and ends at position \( j \). The feature \text{MIN} computes the minimum of the values from index \( i \) to index \( j + 1 \).

Figure 4.1114 provides an example where the \( \text{POS\_MAX\_MIN\_INCREASING\_SEQUENCE} \) constraint holds.
Figure 4.114: Illustrating the POS_MAX_MIN_INCREMENTING_SEQUENCE constraint of the Example slot

Typical

\[|\text{VARIABLES}| > 1\]
\[\text{range} (\text{VARIABLES}.\text{var}) > 1\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton Similar to the automaton of the MAX_MIN_INCREASING_SEQUENCE constraint but use the decoration table 3.35.
Origin
Based on constraint \texttt{MAX\_MIN\_INFLEXION}.

Constraint
\texttt{POS\_MAX\_MIN\_INFLEXION(VALUE, VARIABLES, FOUND)}

Arguments
\begin{itemize}
  \item \texttt{VALUE} : \texttt{dvar}
  \item \texttt{VARIABLES} : \texttt{collection(var\rightarrow dvar)}
  \item \texttt{FOUND} : \texttt{collection(var\rightarrow dvar)}
\end{itemize}

Restrictions
\begin{align*}
  & sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
  & VALUE = -\infty \lor VALUE \geq \text{minv} \\
  & VALUE \leq \text{maxv} \\
  & \text{req}uired(\text{VARIABLES}, \text{var}) \\
  & \text{required}(\text{FOUND, var}) \\
  & |\text{VARIABLES}| = |\text{FOUND}| \\
\end{align*}

where
\begin{itemize}
  \item \texttt{maxv} = \text{maxval}(\text{VARIABLES, var})
  \item \texttt{minv} = \text{minval}(\text{VARIABLES, var})
  \item \texttt{sv} = |\text{VARIABLES}|
  \item \texttt{rv} = \text{range}(\text{VARIABLES, var})
\end{itemize}

The constraint \texttt{MAX\_MIN\_INFLEXION(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{INFLEXION} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{INFLEXION} is the \texttt{maximal} subsequence which matches the regular expression \texttt{"< (</|=)∗ > | > (||=)∗ <"}.

Assume that the occurrence of the pattern \texttt{INFLEXION} starts at position \( i \) and ends at position \( j \). The feature \texttt{MIN} computes the minimum of the values from index \( i + 1 \) to index \( j \).

Example
\begin{align*}
  & (5,\{1,2,6,4,3,5,2,5,1,5,3,3,4,4\},) \\
  & (0,0,0,0,0,0,0,1,0,1,0,0,0,0,0)
\end{align*}

Figure 4.1115 provides an example where the \texttt{POS\_MAX\_MIN\_INFLEXION} constraint holds.
**Figure 4.1115:** Illustrating the POS_MAX_MIN_INFLEXION constraint of the **Example** slot

**Typical**

\[ |\text{VARIABLES} | > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MIN_INFLEXION constraint but use the decoration table 3.35.
Based on constraint \texttt{MAX\_MIN\_STRICTLY\_DECREASING\_SEQUENCE},

\begin{verbatim}
VALUE : dvar
VARIABLES : collection(var\textarrow dvar)
FOUND : collection(var\textarrow dvar)
\end{verbatim}

\begin{verbatim}
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty
VALUE = -\infty \lor VALUE \geq minv
VALUE \leq maxv - 1
required(VARIABLES, var)
required(FOUND, var)
\end{verbatim}

where
\begin{verbatim}
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)
\end{verbatim}

The constraint \texttt{MAX\_MIN\_STRICTLY\_DECREASING\_SEQUENCE(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} is the maximal subsequence which matches the regular expression '$>' or '$+$'.

Assume that the occurrence of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} starts at position $i$ and ends at position $j$. The feature \texttt{MIN} computes the minimum of the values from index $i$ to index $j + 1$.

Figure 4.1116 provides an example where the \texttt{POS\_MAX\_MIN\_STRICTLY\_DECREASING\_SEQUENCE (3, 4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3), [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])} constraint holds.
Figure 4.1116: Illustrating the POS_MAX_MIN.Strictly-Decreasing-Sequence constraint of the Example slot

**Typical**

$|\text{VARIABLES}| > 1$

$\text{range(VARIABLES.var)} > 1$

**Arg. properties**

- **Functional dependency:** VALUE determined by VARIABLES.
- **Functional dependency:** FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \textsc{Max Min Strictly Decreasing Sequence} constraint but use the decoration table 3.35.
**Origin**

Based on constraint `MAX_MIN_STRICTLY_INCREASING_SEQUENCE`.

**Constraint**

`POS_MAX_MIN_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)`

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`
- **FOUND**: `collection(var−dvar)`

**Restrictions**

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \]
\[ VALUE = -\infty \lor VALUE \geq minv \]
\[ VALUE \leq maxv - 1 \]
\[ required(VARIABLES, var) \]
\[ required(FOUND, var) \]
\[ |VARIABLES| = |FOUND| \]

where

- `maxv = maxval(VARIABLES.var)`
- `minv = minval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `MAX_MIN_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `STRICTLY_INCREASING_SEQUENCE` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `'<+'`. Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Purpose**

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

**Example**

\[
(3, [4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
\]

Figure 4.1117 provides an example where the `POS_MAX_MIN_STRICTLY_INCREASING_SEQUENCE` constraint holds.
Figure 4.1117: Illustrating the `POS_MAX_MIN_STRICTLY_INCREASING_SEQUENCE` constraint of the `Example` slot

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`.
Automaton

Similar to the automaton of the \texttt{MAX\_MIN\_STRICTLY\_INCREASING\_SEQUENCE} constraint but use the decoration table 3.35.
The constraint `MAX_MIN_VALLEY(VALUE, VARIABLES)`) holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `VALLEY` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `VALLEY` is the maximal subsequence which matches the regular expression `>(= | >)∗(< | =)∗<`. Assume that the occurrence of the pattern `VALLEY` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i+1` to index `j`.

Figure 4.1118 provides an example where the `POS_MAX_MIN_VALLEY((5, [1, 3, 7, 4, 3, 6, 5, 3, 2, 6, 5, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]))` constraint holds.
Figure 4.1118: Illustrating the \texttt{POS}_{\texttt{MAX}}_{\texttt{MIN}}_{\texttt{VALLEY}} constraint of the \textbf{Example} slot

**Typical**

\[
|\texttt{VARIABLES}| > 2 \\
\text{range}(\texttt{VARIABLES}.\texttt{var}) > 1
\]

**Arg. properties**

- Functional dependency: VALUE determined by \texttt{VARIABLES}.
- Functional dependency: FOUND determined by \texttt{VARIABLES}.
Automaton

Similar to the automaton of the MAX_MIN_VALLEY constraint but use the decoration table 3.35.
Origin
Based on constraint MAX_MIN_ZIGZAG.

Constraint
POS_MAX_MIN_ZIGZAG(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv − 1
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
maxv = maxval(VARIABLES, var)
minv = minval(VARIABLES, var)
sv = |VARIABLES|
rv = range(VARIABLES, var)

The constraint MAX_MIN_ZIGZAG(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern ZIGZAG for which the feature value is VALUE.

Purpose
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘<>(>< | <<>) | (>>< | >>)’.
Assume that the occurrence of the pattern ZIGZAG starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

Example
\[
\begin{pmatrix}
1, 4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1119 provides an example where the POS_MAX_MIN_ZIGZAG (1, [1, 4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0]) constraint holds.
Figure 4.1119: Illustrating the `POS_MAX_MIN_ZIGZAG` constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range(\text{VARIABLES.var})} > 1 \]

**Arg. properties**

- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`.
Automaton

Similar to the automaton of the MAX\_MIN\_ZIGZAG constraint but use the decoration table 3.35.
The constraint \( \text{MAX\_SURF\_BUMP\_ON\_DECREASING\_SEQUENCE} \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \textbf{found} letter in those occurrences of the pattern \( \text{BUMP\_ON\_DECREASING\_SEQUENCE} \) for which the feature value is \( \text{VALUE} \). The position of the \textbf{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern \( \text{BUMP\_ON\_DECREASING\_SEQUENCE} \) is the subsequence which matches the regular expression ‘\( >>>\)’. Assume that the occurrence of the pattern \( \text{BUMP\_ON\_DECREASING\_SEQUENCE} \) starts at position \( i \) and ends at position \( j \). The feature \( \text{SURF} \) computes the sum of the values from index \( i + 2 \) to index \( j \).
Figure 4.1120 provides an example where the POS_MAX_SURF_BUMP_ON_DECREASING_SEQUENCE constraint holds.

![Diagram](image)

Figure 4.1120: Illustrating the POS_MAX_SURF_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

**Typical**

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 2</td>
</tr>
</tbody>
</table>

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the MAX_SURF_BUMP_ON_DECREASING_SEQUENCE constraint but use the decoration table 3.35.
POS_MAX_SURF_BUMP_ON_DECREASING_SEQUENCE
Based on constraint \textit{MAX\_SURF\_DECREASING}.

\textbf{Constraint} \\
\texttt{POS\_MAX\_SURF\_DECREASING(VALUE, VARIABLES, FOUND)}

\textbf{Arguments} \\
\begin{itemize}
  \item \texttt{VALUE} : dvar
  \item \texttt{VARIABLES} : collection(var\_–dvar)
  \item \texttt{FOUND} : collection(var\_–dvar)
\end{itemize}

\textbf{Restrictions} \\
\begin{align*}
  sv \leq 1 & \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
  VALUE = -\infty & \lor VALUE \geq 2 \ast \text{minv} + 1 \\
  VALUE \leq 2 \ast \text{maxv} - 1 & \\
  \text{required(VARIABLES, var)} & \\
  \text{required(FOUND, var)} & \\
  |\text{VARIABLES}| = |\text{FOUND}| & \\
\end{align*}

where \\
\begin{align*}
  \text{maxv} &= \text{maxval(VARIABLES.var)} \\
  \text{minv} &= \text{minval(VARIABLES.var)} \\
  sv &= |\text{VARIABLES}| \\
  rv &= \text{range(VARIABLES.var)}
\end{align*}

The constraint \texttt{MAX\_SURF\_DECREASING(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{DECREASING} for which the feature \texttt{SURF} computes the sum of the values from index \(i\) to index \(j+1\).

\textbf{Purpose} \\
The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern \texttt{DECREASING} is the subsequence which matches the regular expression ‘\texttt{>}\texttt{}’. Assume that the occurrence of the pattern \texttt{DECREASING} starts at position \(i\) and ends at position \(j\). The feature \texttt{SURF} computes the sum of the values from index \(i\) to index \(j+1\).

\textbf{Example} \\
The pattern \texttt{DECREASING} starts at position \(i\) and ends at position \(j\) with \texttt{SURF} value \(2\times2 - 1 = 3\). Figure 4.1121 provides an example where the \texttt{POS\_MAX\_SURF\_DECREASING(10, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0])} constraint holds.
Figure 4.1121: Illustrating the POS_MAX_SURF_DECREASING constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range} (\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_DECREASING constraint but use the decoration table 3.35.
The constraint \texttt{MAX\_SURF\_DECREASING\_SEQUENCE}(\texttt{VALUE}, \texttt{VARIABLES}) holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{DECREASING\_SEQUENCE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{DECREASING\_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression \texttt{’\(\geq (\mid =)^{*} \mid >\)’}.

Assume that the occurrence of the pattern \texttt{DECREASING\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{SURF} computes the sum of the values from index \(i\) to index \(j + 1\).
Example

Figure 4.1122 provides an example where the POS_MAX_SURF_DECREASING_SEQUENCE constraint holds.

\[
\begin{pmatrix}
18, 3, 4, 2, 5, 6, 4, 4, 3, 1, 4, 6, 4, 4 \\
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1122: Illustrating the POS_MAX_SURF_DECREASING_SEQUENCE constraint of the Example slot

Typical

- \(|\text{VARIABLES}| > 1\)
- \(\text{range(VARIABLES.var)} > 1\)

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_DECREASING_SEQUENCE constraint but use the decoration table 3.35.
POS_MAX_SURF_DECREASING_SEQUENCE

2231
**POS_MAX_SURF_DECREASING_TERRACE**

**Description**

**Constraint**

Based on constraint **MAX_SURF_DECREASING_TERRACE**.

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

\[
sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = -\infty \\
VALUE = -\infty \lor VALUE \geq \min(2 \ast (\text{minv} + 1), (sv - 2) \ast (\text{minv} + 1)) \\
VALUE \leq \max(2 \ast (\text{maxv} - 1), (sv - 2) \ast (\text{maxv} - 1)) \\
\text{among}(n1, \text{VARIABLES}[2, sv - 1], (\text{maxv} - 1)) \\
n1 \geq VALUE - \max(0, (sv - 2) \ast (\text{maxv} - 2)) \\
\text{among}(n2, \text{VARIABLES}[2, sv - 1], (\text{minv} + 1)) \\
n2 \geq \min(0, (sv - 2) \ast (\text{minv} + 2)) - VALUE \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{required}(\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| = |\text{FOUND}|
\]

where

- **maxv** =\text{maxval}(\text{VARIABLES}.\text{var})
- **sv** = |\text{VARIABLES}|
- **minv** =\text{minval}(\text{VARIABLES}.\text{var})
- **rv** =\text{range}(\text{VARIABLES}.\text{var})

The constraint **MAX_SURF_DECREASING_TERRACE(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **DECREASING_TERRACE** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **DECREASING_TERRACE** is the maximal subsequence which matches the regular expression ‘\( > = + > \)’.

Assume that the occurrence of the pattern **DECREASING_TERRACE** starts at position \( i \) and ends at position \( j \). The feature **SURF** computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**

\[
\begin{pmatrix}
8, (6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]
Figure 4.1123 provides an example where the \text{POS\_MAX\_SURF\_DECREASING\_TERRACE}\ (8, [6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.

Figure 4.1123: Illustrating the \text{POS\_MAX\_SURF\_DECREASING\_TERRACE} constraint of the Example slot

**Typical**

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(|\text{VARIABLES}\_\text{var}|) > 2\)

**Arg. properties**

- Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
- Functional dependency: \text{FOUND} determined by \text{VARIABLES}.
Automaton

Similar to the automaton of the \texttt{MAX\_SURF\_DECREASING\_TERRACE} constraint but use the decoration table \texttt{3.35}.
POS_MAX_SURF_DECREASING_TERRACE 2235
Origin
Based on constraint \texttt{MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE}.

Constraint
\texttt{POS\_MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE}(\texttt{VALUE}, \texttt{VARIABLES}, \texttt{FOUND})

Arguments
\begin{itemize}
  \item \texttt{VALUE} : dvar
  \item \texttt{VARIABLES} : collection(var\textendash dvar)
  \item \texttt{FOUND} : collection(var\textendash dvar)
\end{itemize}

Restrictions
\begin{align*}
  \text{sv} & \leq 5 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = -\infty \\
  \text{VALUE} & = -\infty \lor \text{VALUE} \geq 3 \star \text{minv} + 3 \\
  \text{VALUE} & \leq 3 \star \text{maxv} - 3 \\
  \text{among} & (\text{n1}, \text{VARIABLES}[3, \text{sv} - 1], (\text{maxv} - 2, \text{maxv} - 1, \text{maxv})) \\
  \text{n1} & \geq \text{VALUE} - 3 - \text{max}(\text{maxv} - 3, 0) \\
  \text{among} & (\text{n2}, \text{VARIABLES}[3, \text{sv} - 1], (\text{minv}, \text{minv} + 1, \text{minv} + 2)) \\
  \text{n2} & \geq \text{min}(\text{minv} + 3, 0) - 3 - \text{VALUE} \\
  \text{required} & (\text{VARIABLES}, \text{var}) \\
  \text{required} & (\text{FOUND}, \text{var}) \\
  |\text{VARIABLES}| & = |\text{FOUND}|
\end{align*}

where
\begin{itemize}
  \item \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
  \item \text{minv} = \text{minval}(\text{VARIABLES}.\text{var})
  \item \text{sv} = |\text{VARIABLES}|
  \item \text{rv} = \text{range}(\text{VARIABLES}.\text{var})
\end{itemize}

Purpose
The constraint \texttt{MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE}(\texttt{VALUE}, \texttt{VARIABLES}) holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{DIP\_ON\_INCREASING\_SEQUENCE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{DIP\_ON\_INCREASING\_SEQUENCE} is the subsequence which matches the regular expression ‘\texttt{<<><><}’.

Assume that the occurrence of the pattern \texttt{DIP\_ON\_INCREASING\_SEQUENCE} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{SURF} computes the sum of the values from index \texttt{i} + 2 to index \texttt{j}.

Example
\begin{itemize}
  \item \texttt{(10, \{1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4\}, \{0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\})}
\end{itemize}
Figure 4.1124 provides an example where the POS_MAX_SURF_DIP_ON_INCREASING_SEQUENCE constraint holds.

\[ \{10, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]\} \]

Figure 4.1124: Illustrating the POS_MAX_SURF_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot

**Typical**

\[ |VARIABLES| > 5 \]
\[ \text{range}(VARIABLES.\text{var}) > 2 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE} constraint but use the decoration table 3.35.
POS_MAX_SURF_DIP_ON_INCREASING_SEQUENCE
Origin
Based on constraint MAX_SURF_GORGE.

Constraint
POS_MAX_SURF_GORGE(VALUE, VARIABLES, FOUND)

Arguments
- VALUE: dvar
- VARIABLES: collection(var−dvar)
- FOUND: collection(var−dvar)

Restrictions
- sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞
- rv = 2 ⇒ VALUE = −∞ ∨ VALUE ≥ minv
- rv ≥ 3 ⇒ VALUE = −∞ ∨ VALUE ≥ min(minv(sv−2)∗(maxv−1)−1)
- rv = 2 ⇒ VALUE ≤ maxv−1
- rv ≥ 3 ⇒ VALUE ≤ max(maxv−1sv, (sv−2)∗(maxv−1)−1)

among(n1, VARIABLES[2, sv−1], [maxv−1])
- rv > 2 ∧ maxv > 1 ⇒ n1 ≥ VALUE − (sv−2)∗(maxv−2)
- rv = 2 ∨ maxv = 1 ⇒ n1 ≥ VALUE − max(0, maxv−2)

among(n2, VARIABLES[2, sv−1], [minv, minv+1])
- rv > 2 ∧ minv < −1 ⇒ n2 ≥ (sv−2)∗(minv+2)−1−VALUE
- rv = 2 ∨ minv = −1 ⇒ n2 ≥ min(0, minv+1)−VALUE

required(VARIABLES, var)
required(FOUND, var)
|VARIABLES|= |FOUND|

where
- maxv = maxval(VARIABLES.var)
- rv = range(VARIABLES.var)
- sv = |VARIABLES|
- minv = minval(VARIABLES.var)

Purpose
The constraint MAX_SURF_GORGE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern GORGE for which the feature value is VALUE.
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘(> | > (= | >) | >) (< | < (= | <) | <)’.
Assume that the occurrence of the pattern GORGE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.
Example

\[
(11, (1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7), (0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))
\]

Figure 4.1125 provides an example where the \texttt{POS\_MAX\_SURF\_GORGE} constraint holds.

![Diagram of feature values](image)

Figure 4.1125: Illustrating the \texttt{POS\_MAX\_SURF\_GORGE} constraint of the Example slot

Typical

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
**Automaton**

Similar to the automaton of the MAX_SURF_GORGE constraint but use the decoration table 3.35.
POS_MAX_SURF_GORGE

2243
### Description

**Origin**
Based on constraint `MAX_SURF_INCREASING`.

**Constraint**

```plaintext
POS_MAX_SURF_INCREASING(VALUE, VARIABLES, FOUND)
```

**Arguments**

- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`
- `FOUND`: `collection(var−dvar)`

**Restrictions**

- `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ 2 * minv + 1`
- `VALUE ≤ 2 * maxv − 1`
- `required(VARIABLES, var)`
- `required(FOUND, var)`
- `|VARIABLES| = |FOUND|`

where

- `maxv = maxval(VARIABLES.var)`
- `minv = minval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `MAX_SURF_INCREASING(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `INCREASING` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `INCREASING` is the subsequence which matches the regular expression ‘<’.

Assume that the occurrence of the pattern `INCREASING` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.

### Purpose

The constraint `MAX_SURF_INCREASING(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `INCREASING` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `INCREASING` is the subsequence which matches the regular expression ‘<’.

Assume that the occurrence of the pattern `INCREASING` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.

### Example

| 10, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 3, 3), |
| 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0 |

Figure 4.1126 provides an example where the `POS_MAX_SURF_INCREASING (10, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0])` constraint holds.
Figure 4.1126: Illustrating the POS_MAX_SURF_INCREASING constraint of the Example slot

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MAX_SURF_INCREASING` constraint but use the decoration table 3.35.
POS_MAX_SURF_INCREASING
**POS_MAX_SURF_INCREASING_SEQUENCE**

**AGGREGATOR**  
**FEATURE**  
**PATTERN**  

**DESCRIPTION AUTOMATON**

**Origin**  
Based on constraint MAX_SURF_INCREASING_SEQUENCE.

**Constraint**  
POS_MAX_SURF_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**  
VALUE : dvar  
VARIABLES : collection(var−dvar)  
FOUND : collection(var−dvar)

**Restrictions**  
\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \]
\[ rv = 2 \Rightarrow VALUE = -\infty \lor VALUE \geq 2 \ast \minv + 1 \]
\[ rv \geq 3 \Rightarrow VALUE = -\infty \lor VALUE \geq \min(2 \ast \minv + 1, sv \ast \minv + 1)) \]
\[ rv = 2 \Rightarrow VALUE \leq 2 \ast \maxv - 1 \]
\[ rv \geq 3 \Rightarrow VALUE \leq \max(2 \ast \maxv - 1, sv \ast \minv + 1) \]
\[ among(n1, VARIABLES[1,sv], (\maxv - 1, \maxv)) \]
\[ rv = 2 \lor \maxv = 1 \Rightarrow n1 \geq VALUE - \max(0, 2 \ast \maxv - 3) \]
\[ rv > 2 \land \maxv > 1 \Rightarrow n1 \geq VALUE - \sv \ast (\maxv - 2) - 1 \]
\[ among(n2, VARIABLES[1,sv], (\minv, \minv + 1)) \]
\[ rv = 2 \lor \minv = -1 \Rightarrow n2 \geq \min(0, 2 \ast \minv + 3) - VALUE \]
\[ rv > 2 \land \minv < -1 \Rightarrow n2 \geq \sv \ast (\minv + 2) - 1 - VALUE \]
\[ required(VARIABLES, var) \]
\[ required(FOUND, var) \]
\[ |VARIABLES| = |FOUND| \]

where  
\[ \maxv = \maxval(VARIABLES.var) \]
\[ rv = \range(VARIABLES.var) \]
\[ sv = |VARIABLES| \]
\[ \minv = \minval(VARIABLES.var) \]

The constraint MAX_SURF_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\(< \leq =\)* \(< \leq \)’. Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j + 1\).
Example

Figure 4.1127 provides an example where the POS_MAX_SURF_INCREASING_SEQUENCE constraint holds.

\[
\begin{align*}
(17, \langle 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3 \rangle), \\
\langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle
\end{align*}
\]

Figure 4.1127: Illustrating the POS_MAX_SURF_INCREASING_SEQUENCE constraint of the Example slot

Typical

\[|\text{VARIABLES}| > 1\]
\[\text{range(\text{VARIABLES}.var)} > 1\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_INCREASING_SEQUENCE constraint but use the decoration table 3.35.
Based on constraint \texttt{MAX\_SURF\_INCREASING\_TERRACE}.

**Constraint**

\texttt{POS\_MAX\_SURF\_INCREASING\_TERRACE(VALUE, VARIABLES, FOUND)}

**Arguments**

- \texttt{VALUE} : \texttt{dvar}
- \texttt{VARIABLES} : \texttt{collection(var\text{-}dvar)}
- \texttt{FOUND} : \texttt{collection(var\text{-}dvar)}

**Restrictions**

\begin{align*}
sv & \leq 3 \lor rv \leq 2 \Rightarrow VALUE = -\infty \\
VALUE & = -\infty \lor VALUE \geq \min(2 \ast (\text{minv} + 1), (sv - 2) \ast (\text{minv} + 1)) \\
VALUE & \leq \max(2 \ast (\text{maxv} - 1) \ast (sv - 2) \ast (\text{maxv} - 1)) \\
\text{among}(n1, \text{VARIABLES}[2, sv - 1], (sv - 2) \ast (\text{maxv} - 1)) \\
n1 & \geq VALUE - \max(0, (sv - 2) \ast (\text{maxv} - 2)) \\
\text{among}(n2, \text{VARIABLES}[2, sv - 1], (\text{minv} + 1)) \\
n2 & \geq \min(0, (sv - 2) \ast (\text{minv} + 2)) - VALUE \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{required}(\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| &= |\text{FOUND}|
\end{align*}

where

- \texttt{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
- \texttt{sv} = |\text{VARIABLES}|
- \texttt{minv} = \text{minval}(\text{VARIABLES}.\text{var})
- \texttt{rv} = \text{range}(\text{VARIABLES}.\text{var})

The constraint \texttt{MAX\_SURF\_INCREASING\_TERRACE(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{INCREASING\_TERRACE} for which the feature value is \texttt{VALUE}.

**Purpose**

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{INCREASING\_TERRACE} is the maximal subsequence which matches the regular expression ‘\texttt{< = + <}’.

Assume that the occurrence of the pattern \texttt{INCREASING\_TERRACE} starts at position $i$ and ends at position $j$. The feature \texttt{SURF} computes the sum of the values from index $i + 1$ to index $j$.

**Example**

\[
(10, \langle 1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4 \rangle, \langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle)
\]
Figure 4.1128 provides an example where the POS_MAX_SURF_INCREASING_TERRACE constraint holds. The example slot constraint is:

$[10, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 4, 4, 0, 0, 0, 0, 0, 0, 0] \leq [10, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$

Figure 4.1128: Illustrating the POS_MAX_SURF_INCREASING_TERRACE constraint of the Example slot

Typical

$|\text{VARIABLES}| > 3$

range(\text{VARIABLES}.\text{var}) > 2$

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_INCREASING_TERRACE constraint but use the decoration table 3.35.
POS_MAX_SURF_INCREASING_TERRACE
The constraint `MAX_SURF_INFLEXION(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern `INFLEXION` for which the feature value is `VALUE`. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern `INFLEXION` is the maximal subsequence which matches the regular expression `'< (< | =) > | ( > | =)>'`. Assume that the occurrence of the pattern `INFLEXION` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 1` to index `j`.

**Example**

\[
(14, \langle 1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4 \rangle, (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))
\]
Figure 4.1129 provides an example where the POS_MAX_SURF_INFLEXION constraint holds. The feature values for the example slot are as follows: [14, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]].

**Typical**

$|\text{VARIABLES}| > 2$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_INFLEXION constraint but use the decoration table 3.35.
POS_MAX_SURF_INFLEXION
Origin

Based on constraint \texttt{MAX\_SURF\_PEAK}.

Constraint

\[
\text{POS\_MAX\_SURF\_PEAK}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

Arguments

\[
\begin{align*}
\text{VALUE} : & \quad \text{dvar} \\
\text{VARIABLES} : & \quad \text{collection}(\text{var} - \text{dvar}) \\
\text{FOUND} : & \quad \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \min(\minv + 1, (\text{sv} - 2) \times (\minv + 1)) \\
\text{VALUE} & \leq \max(\maxv, (\text{sv} - 2) \times \maxv) \\
\text{among}(\text{n1}, \text{VARIABLES}[2, \text{sv} - 1], (\text{maxv})) & \\
\text{n1} & \geq \text{VALUE} - \max(0, (\text{sv} - 2) \times (\maxv - 1)) \\
\text{among}(\text{n2}, \text{VARIABLES}[2, \text{sv} - 1], (\minv + 1)) & \\
\text{n2} & \geq \min(0, (\text{sv} - 2) \times (\minv + 2)) - \text{VALUE} \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
|\text{VARIABLES}| = |\text{FOUND}| & \\
\text{where} & \\
\text{minv} & = \minval(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \maxval(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \range(\text{VARIABLES}.\text{var})
\end{align*}
\]

Purpose

The constraint \texttt{MAX\_SURF\_PEAK}(\text{VALUE}, \text{VARIABLES}) holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{PEAK} for which the feature value is \text{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{PEAK} is the \textit{maximal} subsequence which matches the regular expression ‘\texttt{<(=|<)(>|=)>}’.

Assume that the occurrence of the pattern \texttt{PEAK} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{SURF} computes the sum of the values from index \texttt{i} + 1 to index \texttt{j}.

Example

\[
\begin{pmatrix}
14, (7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0
\end{pmatrix}
\]
Figure 4.1130 provides an example where the POS_MAX_SURF_PEAK (14, [7, 5, 1, 4, 5, 2, 3, 5, 6, 2, 3, 3, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Figure 4.1130: Illustrating the POS_MAX_SURF_PEAK constraint of the Example slot

**Typical**

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
| Automaton | Similar to the automaton of the MAX_SURF_PEAK constraint but use the decoration table 3.35. |
POS_MAX_SURF_PEAK

2263
Origin
Based on constraint MAX_SURF_PLAIN.

Constraint
POS_MAX_SURF_PLAIN(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ min(minv,(sv − 2) * minv)
VALUE ≤ max(maxv − 10,(sv − 2) * (maxv − 1)*)
among(n1, VARIABLES[2, sv − 1, [maxv − 1]])
n1 ≥ VALUE − max(0,(sv − 2) * (maxv − 2))
among(n2, VARIABLES[2, sv − 1, [minv]])
n2 ≥ min(0,(sv − 2) * (minv + 1)) − VALUE
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
maxv = maxval(VARIABLES, var)
v = |VARIABLES|
minv = minval(VARIABLES, var)
rv = range(VARIABLES, var)

The constraint MAX_SURF_PLAIN(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PLAIN for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘>’=’<’.
Assume that the occurrence of the pattern PLAIN starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
(6, ⟨2, 3, 6, 5, 7, 6, 6, 4, 5, 4, 3, 3, 6, 6, 3⟩, ⟨0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0⟩)
Figure 4.1131 provides an example where the POS_MAX_SURF_PLAIN constraint holds.

Figure 4.1131: Illustrating the POS_MAX_SURF_PLAIN constraint of the Example slot

**Typical**

| VARIABLES | > 2  
| range(VARIABLES.var) | > 1 |

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_PLAIN constraint but use the decoration table 3.35.
The constraint $\text{POS\_MAX\_SURF\_PLATEAU(VALUE, VARIABLES, FOUND)}$ holds. In addition, $\text{FOUND}$ is a collection of 0/1 variables where the value 1 indicates the position of the $\text{found}$ letter in those occurrences of the pattern $\text{PLATEAU}$ for which the feature value is $\text{VALUE}$.

The position of the $\text{found}$ letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern $\text{PLATEAU}$ is the maximal subsequence which matches the regular expression ‘$<"=">$’.

Assume that the occurrence of the pattern $\text{PLATEAU}$ starts at position $i$ and ends at position $j$. The feature $\text{SURF}$ computes the sum of the values from index $i+1$ to index $j$.

Example

\[
\begin{pmatrix}
10, (7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5) , \\
\{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0\}
\end{pmatrix}
\]
Figure 4.1132 provides an example where the POS_MAX_SURF_PLATEAU constraint holds.

Figure 4.1132: Illustrating the POS_MAX_SURF_PLATEAU constraint of the Example slot

**Typical**

| VARIABLES | > 2  
| range(VARIABLES.var) | > 1  

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_PLATEAU constraint but use the decoration table 3.35.
POS_MAX_SURF_PLATEAU

2271
The constraint `MAX_SURF_PROPER_PLAIN(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `PROPER_PLAIN` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `PROPER_PLAIN` is the maximal subsequence which matches the regular expression `'>'=''<`.

Assume that the occurrence of the pattern `PROPER_PLAIN` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 1` to index `j`.

Example:

```
(10, (2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5),
(0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))
```
Figure 4.1133 provides an example where the `POS_MAX_SURF_PROPERPLAIN` (10, [2, 7, 5, 6, 3, 7, 4, 5, 6, 5, 3, 3, 3, 5], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Figure 4.1133: Illustrating the `POS_MAX_SURF_PROPERPLAIN` constraint of the `Example` slot

**Typical**

| VARIABLES | > 3  
| range(VARIABLES.var) | > 1 

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SRF_PROPER.PLAIN constraint but use the decoration table 3.35.
Origin
Based on constraint \texttt{MAX\_SURF\_PROPER\_PLATEAU}.

Constraint
\texttt{POS\_MAX\_SURF\_PROPER\_PLATEAU}(\texttt{VALUE}, \texttt{VARIABLES}, \texttt{FOUND})

Arguments
\begin{itemize}
  \item \texttt{VALUE} : \texttt{dvar}
  \item \texttt{VARIABLES} : \texttt{collection(var – dvar)}
  \item \texttt{FOUND} : \texttt{collection(var – dvar)}
\end{itemize}

Restrictions
\begin{align*}
  sv \leq 3 \lor rv \leq 1 &\Rightarrow VALUE = -\infty \\
  VALUE &\leq -\infty \lor VALUE \geq \min(2 \times (\text{minv} + 1), (sv - 2) \times (\text{minv} + 1)) \\
  VALUE &\leq \max(2 \times \text{maxv}, (sv - 2) \times \text{maxv}) \\
  \text{among}(n1, \text{VARIABLES}[2, sv - 1], (\text{maxv})) &\lor sv \leq \text{min}((0, (sv - 2) \times (\text{maxv} - 1)) \\
  \text{among}(n2, \text{VARIABLES}[2, sv - 1], (\text{minv} + 1)) &\lor n2 \geq \min(0, (sv - 2) \times (\text{minv} + 2)) - \text{VALUE} \\
  \text{required} &\left(\text{VARIABLES}, \text{var} \right) \\
  \text{required} &\left(\text{FOUND}, \text{var} \right) \\
  |\text{VARIABLES}| &\leq |\text{FOUND}|
\end{align*}

where
\begin{align*}
  \text{maxv} &\equiv \text{maxval} (\text{VARIABLES}.\text{var}) \\
  sv &\equiv |\text{VARIABLES}| \\
  \text{minv} &\equiv \text{minval} (\text{VARIABLES}.\text{var}) \\
  rv &\equiv \text{range} (\text{VARIABLES}.\text{var})
\end{align*}

The constraint \texttt{MAX\_SURF\_PROPER\_PLATEAU}(\texttt{VALUE}, \texttt{VARIABLES}) holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{PROPER\_PLATEAU} for which the feature \texttt{value} is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

Purpose
An occurrence of the pattern \texttt{PROPER\_PLATEAU} is the \textit{maximal} subsequence which matches the regular expression '\texttt{< =+ >}'.

Assume that the occurrence of the pattern \texttt{PROPER\_PLATEAU} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{SURF} computes the sum of the values from index \texttt{i} + 1 to index \texttt{j}.

Example
\begin{itemize}
  \item \texttt{(15, [7, 1, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3]),}
  \item \texttt{(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)}
\end{itemize}
Figure 4.1134 provides an example where the \texttt{POS\_MAX\_SURF\_PROPER\_PLATEAU} constraint holds.

![Diagram illustrating the POS\_MAX\_SURF\_PROPER\_PLATEAU constraint](image)

Figure 4.1134: Illustrating the \texttt{POS\_MAX\_SURF\_PROPER\_PLATEAU} constraint of the Example slot

\textbf{Typical}

\begin{align*}
|\text{VARIABLES}| & > 3 \\
\text{range}(\text{VARIABLES.var}) & > 1
\end{align*}

\textbf{Arg. properties}

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MAX_SURF_PROPER_PLATEAU` constraint but use the decoration table 3.35.
Based on constraint **MAX_SURF_STEADY**.

**Constraint**

\[ \text{POS_MAX_SURF_STEADY(VALUE, VARIABLES, FOUND)} \]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var=dvar)`
- **FOUND**: `collection(var=dvar)`

**Restrictions**

\[
\begin{align*}
sv \leq 1 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} \leq -\infty & \lor \text{VALUE} \geq 2 \times \text{minv} \\
\text{VALUE} \leq 2 \times \text{maxv} & \\
\text{among}(n1, \text{VARIABLES}[1, sv], (\text{maxv})) & \\
n1 \geq \text{VALUE} - 2 \times (\text{maxv} - 1) & \\
\text{among}(n2, \text{VARIABLES}[1, sv], (\text{minv})) & \\
n2 \geq 2 \times (\text{minv} + 1) - \text{VALUE} & \\
\text{required}(& \text{VARIABLES, var}) \\
\text{required}(& \text{FOUND, var}) \\
|\text{VARIABLES}| & = |\text{FOUND}|
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} & = \text{maxval}(& \text{VARIABLES}.\text{var}) \\
\text{minv} & = \text{minval}(& \text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}|
\end{align*}
\]

The constraint **MAX_SURF_STEADY(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **STEADY** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **STEADY** is the subsequence which matches the regular expression ‘\(=\)’.

Assume that the occurrence of the pattern **STEADY** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
\begin{pmatrix}
12, (1, 1, 7, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
\end{pmatrix}
\]

Figure 4.1135 provides an example where the **POS_MAX_SURF_STEADY** constraint holds.
Figure 4.1135: Illustrating the POS\_MAX\_SURF\_STEADY constraint of the Example slot

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>2</th>
<th>6</th>
<th>10</th>
<th>10</th>
<th>10</th>
<th>12 (MAX)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_1$</td>
<td>$V_2$</td>
<td>$V_3$</td>
<td>$V_4$</td>
<td>$V_5$</td>
<td>$V_6$</td>
<td>$V_7$</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>

Typical
$|\text{VARIABLES}| > 1$

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_STEADY constraint but use the decoration table 3.35.
Origin
Based on constraint MAX_SURF_STEADY_SEQUENCE.

Constraint
POS_MAX_SURF_STEADY_SEQUENCE(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 1 ⇒ VALUE = −∞
rv = 1 ⇒ VALUE = −∞ ∨ VALUE ≥ sv * minv
rv ≥ 2 ⇒ VALUE = −∞ ∨ VALUE ≥ min(2 * minv, sv * minv)
rv = 1 ⇒ VALUE ≤ sv * maxv
rv ≥ 2 ⇒ VALUE ≤ max(2 * maxv, sv * maxv)
among([n1, VARIABLES[1, sv], (maxv)])
n1 ≥ VALUE − max(0, sv * (maxv − 1))
among([n2, VARIABLES[2, sv], (minv)])
n2 ≥ min(0, sv * (minv + 1)) − VALUE
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|

where
maxv = maxval(VARIABLES.var)
rv = range(VARIABLES.var)
sv = |VARIABLES|
minv = minval(VARIABLES.var)

The constraint MAX_SURF_STEADY_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STEADY_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression ‘=+’.

Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.
Example

Figure 4.1136 provides an example where the POS_MAX_SURF_STEADY_SEQUENCE constraint holds.

$$15, (3, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1),\ (0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0)$$

Figure 4.1136: Illustrating the POS_MAX_SURF_STEADY_SEQUENCE constraint of the Example slot

Typical

$|\text{VARIABLES}| > 1$

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_STEADY_SEQUENCE constraint but use the decoration table 3.35.
POS_MAX_SURF_STEADY_SEQUENCE
DESCRIPTION AUTOMATON

POS_MAX_SURF_STRICTLY_DECREASING_SEQUENCE

ARGUMENTS

VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

RESTRICTIONS

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \]
\[ \minv < 0 \Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq \ell_1 * \minv + \lfloor \ell_1 * (\ell_1 - 1)/2 \rfloor \]
\[ \minv \geq 0 \Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq 2 * \minv + 1 \]
\[ \maxv > 0 \Rightarrow \text{VALUE} \leq \ell_2 * \maxv - \lfloor \ell_2 * (\ell_2 - 1)/2 \rfloor \]
\[ \maxv \leq 0 \Rightarrow \text{VALUE} \leq 2 * \maxv - 2 \]
\[ \text{among}(n_1, \text{VARIABLES}[1, sv], (\maxv - 1, \maxv)) \]
\[ n_1 \geq \text{VALUE} - \max(0, \min(sv, rv) * (\maxv - 2)) - 1 \]
\[ \text{among}(n_2, \text{VARIABLES}[1, sv], (\minv, \minv + 1)) \]
\[ n_2 \geq \min(0, \min(sv, rv) * (\minv + 2)) - 1 - \text{VALUE} \]
\[ \text{required}(\text{VARIABLES}, \text{var}) \]
\[ \text{required}(\text{FOUND}, \text{var}) \]
\[ |\text{VARIABLES}| = |\text{FOUND}| \]

where
\[ \maxv = \maxval(\text{VARIABLES}.\text{var}) \]
\[ rv = \range(\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ \ell_1 = \min(\min(sv, rv), |\text{minv}|) \]
\[ \ell_2 = \min(\min(sv, rv), |\maxv|) \]
\[ \minv = \minval(\text{VARIABLES}.\text{var}) \]

The constraint \textit{MAX_SURF_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)} holds. In addition, \textit{FOUND} is a collection of 0/1 variables where the value \textit{1} indicates the position of the \textit{found} letter in those occurrences of the pattern \textit{STRICTLY_DECREASING_SEQUENCE} for which the feature value is \textit{VALUE}.

The position of the \textit{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \textit{STRICTLY_DECREASING_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression ‘\textit{>}’.

Assume that the occurrence of the pattern \textit{STRICTLY_DECREASING_SEQUENCE} starts at position \textit{i} and ends at position \textit{j}. The feature \textit{SURF} computes the sum of the values from index \textit{i} to index \textit{j} + 1.
Example

\[
\begin{align*}
13, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3), \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0)
\end{align*}
\]

Figure 4.1137 provides an example where the \textit{POS\_MAX\_SURF\_STRICTLY\_DECREASING\_SEQUENCE} constraint holds.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{example_diagram}
\caption{Illustrating the \textit{POS\_MAX\_SURF\_STRICTLY\_DECREASING\_SEQUENCE} constraint of the \textbf{Example} slot}
\end{figure}

Typical

\[
|VARIABLES| > 1 \\
range(VARIABLES.var) > 1
\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_STRICTLY_DECREASING_SEQUENCE constraint but use the decoration table 3.35.
POS_MAX_SURF_STRICTLY_DECREASING_SEQUENCE
Based on constraint MAX_SURF_STRICTLY_INCREASING_SEQUENCE.

### Arguments

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var–dvar)</td>
</tr>
<tr>
<td>FOUND</td>
<td>collection(var–dvar)</td>
</tr>
</tbody>
</table>

### Restrictions

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty
\]

\[
\minv < 0 \Rightarrow VALUE = -\infty \lor VALUE \geq \ell_1 \ast \minv + \left\lfloor \ell_1 \ast (\ell_1 - 1)/2 \right\rfloor
\]

\[
\minv \geq 0 \Rightarrow VALUE = -\infty \lor VALUE \geq 2 \ast \minv + 1
\]

\[
\maxv > 0 \Rightarrow VALUE \leq \ell_2 \ast \maxv - \left\lfloor \ell_2 \ast (\ell_2 - 1)/2 \right\rfloor
\]

\[
\maxv \leq 0 \Rightarrow VALUE \leq 2 \ast \maxv - 1
\]

\[
\text{among}(n_1, \text{VARIABLES}[1, sv], (\maxv - 1, \maxv))
\]

\[
n_1 \geq VALUE - \max(0, \min(sv, rv) \ast (\maxv - 2)) - 1
\]

\[
\text{among}(n_2, \text{VARIABLES}[1, sv], (\minv, \minv + 1))
\]

\[
n_2 \geq \min(0, \min(sv, rv) \ast (\minv + 2)) - 1 - VALUE
\]

\[
\text{required}(\text{VARIABLES}, \text{var})
\]

\[
\text{required}(\text{FOUND}, \text{var})
\]

\[
|\text{VARIABLES}| = |\text{FOUND}|
\]

where

\[
\maxv = \text{maxval}(\text{VARIABLES}.\text{var})
\]

\[
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

\[
sv = |\text{VARIABLES}|
\]

\[
\ell_1 = \min(\min(sv, rv), |\minv|)
\]

\[
\ell_2 = \min(\min(sv, rv), |\maxv|)
\]

\[
\minv = \text{minval}(\text{VARIABLES}.\text{var})
\]

### Purpose

The constraint MAX_SURF_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STRICTLY_INCREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘<’.

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j + 1\).
Example

Figure 4.1138 provides an example where the \texttt{POS\_MAX\_SURF\_STRRICTLY\_INCREASING\_SEQUENCE} constraint holds.

Figure 4.1138: Illustrating the \texttt{POS\_MAX\_SURF\_STRRICTLY\_INCREASING\_SEQUENCE} constraint of the \texttt{Example} slot

Typical

| VARIABLES | > 1  \\
\texttt{range(VARIABLES.var)} | > 1  \\

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF.Strictly_Increasing_Sequence constraint but use the decoration table 3.35.
Origin
Based on constraint MAX_SURF_SUMMIT.

Constraint
POS_MAX_SURF_SUMMIT(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty \]
\[ rv = 2 \Rightarrow VALUE = -\infty \lor VALUE \geq minv + 1 \]
\[ rv \geq 3 \Rightarrow \]
\[ VALUE = -\infty \lor VALUE \geq \min(minv + 1, (sv - 2) \cdot (minv + 1) + 1) \]
\[ rv = 2 \Rightarrow VALUE \leq \maxv \]
\[ rv \geq 3 \Rightarrow VALUE \leq \max(\maxv, (sv - 2) \cdot (\maxv - 1) + 1) \]
\[ \text{among}(n1, VARIABLES[2, sv - 1], (\maxv - 1, \maxv)) \]
\[ rv = 2 \Rightarrow n1 \geq VALUE - \max(0, \maxv - 1) \]
\[ rv > 2 \land \maxv > 1 \Rightarrow n1 \geq VALUE - (sv - 2) \cdot (\maxv - 2) - 1 \]
\[ \text{among}(n2, VARIABLES[2, sv - 1], (\minv + 1)) \]
\[ rv = 2 \Rightarrow \minv = -1 \Rightarrow n2 \geq \min(0, \minv + 2) - VALUE \]
\[ rv > 2 \land \minv < -1 \Rightarrow n2 \geq (sv - 2) \cdot (\minv + 2) - VALUE \]
\[ \text{required}(VARIABLES, var) \]
\[ \text{required}(FOUND, var) \]
\[ |VARIABLES| = |FOUND| \]

where
\[ maxv = \maxval(VARIABLES.var) \]
\[ rv = \range(VARIABLES.var) \]
\[ sv = |VARIABLES| \]
\[ minv = \minval(VARIABLES.var) \]

Purpose
The constraint MAX_SURF_SUMMIT(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern SUMMIT for which the feature value is VALUE.
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression \( '(< | (= | <) <)(> | (> | =) >)' \).
Assume that the occurrence of the pattern SUMMIT starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).
Example

\[
\begin{pmatrix}
13, (7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1), \\
(0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 4.1139 provides an example where the \textit{POS\_MAX\_SURF\_SUMMIT} constraint holds.

Figure 4.1139: Illustrating the POS\_MAX\_SURF\_SUMMIT constraint of the Example slot

Typical

- \(|\text{VARIABLES}| > 2\)
- \(\text{range(}\text{VARIABLES.var} > 1\)

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_SUMMIT constraint but use the decoration table 3.35.
POS_MAX_SURF_SUMMIT

2299
Origin: Based on constraint \texttt{MAX\_SURF\_VALLEY}.

Constraint:

\texttt{POS\_MAX\_SURF\_VALLEY}(\texttt{VALUE, VARIABLES, FOUND})

Arguments:

- \texttt{VALUE} : \texttt{dvar}
- \texttt{VARIABLES} : \texttt{collection(var\textendash dvar)}
- \texttt{FOUND} : \texttt{collection(var\textendash dvar)}

Restrictions:

\begin{align*}
sv \leq 2 \lor rv \leq 1 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \min(\text{minv}, (sv - 2) \cdot \text{minv}) & \\
\text{VALUE} \leq \max(\text{maxv} - 1, (sv - 2) \cdot (\text{maxv} - 1)) & \\
\text{among}(n1, \text{VARIABLES}[2, sv - 1], (\text{maxv} - 1)) & \\
n1 \geq \text{VALUE} - \max(0, (sv - 2) \cdot (\text{maxv} - 2)) & \\
\text{among}(n2, \text{VARIABLES}[2, sv - 1], (\text{minv})) & \\
n2 \geq \min(0, (sv - 2) \cdot (\text{minv} + 1)) - \text{VALUE} & \\
\text{required}(\text{VARIABLES}.\text{var}) & \\
\text{required}(\text{FOUND}.\text{var}) & \\
|\text{VARIABLES}| = |\text{FOUND}| & \\
\text{where} & \\
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) & \\
sv = |\text{VARIABLES}| & \\
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) & \\
rv = \text{range}(\text{VARIABLES}.\text{var}) & \\
\end{align*}

Purpose:

The constraint \texttt{MAX\_SURF\_VALLEY}(\texttt{VALUE, VARIABLES}) holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{VALLEY} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{VALLEY} is the \textit{maximal} subsequence which matches the regular expression ‘\textgreater (= | \textgreater) (\textless | =)’ \textless.

Assume that the occurrence of the pattern \texttt{VALLEY} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{SURF} computes the sum of the values from index \texttt{i} + 1 to index \texttt{j}.

Example:

\begin{align*}
(15, \langle 1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7 \rangle, \\
\langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0 \rangle)
\end{align*}
Figure 4.1140 provides an example where the POS_MAX_SURF_VALLEY constraint holds.

![Diagram illustrating POS_MAX_SURF_VALLEY constraint]

**Typical**

\[|\text{VARIABLES}| > 2\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton: Similar to the automaton of the MAX_SURF_VALLEY constraint but use the decoration table 3.35.
POS_MAX_SURF_VALLEY

2303
Positional Maximum Surface Zigzag

**Origin**
Based on constraint MAX_SURF_ZIGZAG.

**Constraint**
POS_MAX_SURF_ZIGZAG(VALUE, VARIABLES, FOUND)

**Arguments**
- **VALUE** : dvar
- **VARIABLES** : collection(var−dvar)
- **FOUND** : collection(var−dvar)

**Restrictions**
\[
\begin{align*}
sv \leq 3 \vee rv \leq 1 & \Rightarrow VALUE = -\infty \\
\bigwedge \left( VALUE \geq \min \left( 2 * \minv + 1, \left( (sv - 1)/2 \right) * \minv + \left( (sv - 2)/2 \right) * (\minv + 1) \right) \right) \\
VALUE \leq \max \left( 2 * \maxv - 10, \left( (sv - 1)/2 \right) * \maxv + \left( (sv - 2)/2 \right) * (\maxv - 1) \right) \\
& \quad \text{among} (n1, VARIABLES[2, sv - 1], (maxv - 1, maxv)) \\
n1 \geq VALUE - \left( (sv - 1)/2 \right) - \max(0, (sv - 2) * (\maxv - 2)) \\
& \quad \text{among} (n2, VARIABLES[2, sv - 1], (minv, minv + 1)) \\
n2 \geq \min(0, (sv - 2) * (\minv + 2)) - \left( (sv - 1)/2 \right) - VALUE \\
& \quad \text{required}(VARIABLES, var) \\
& \quad \text{required}(FOUND, var) \\
|VARIABLES| = |FOUND| \\
\end{align*}
\]

where

- \( \minv = \minval(VARIABLES.var) \)
- \( \maxv = \maxval(VARIABLES.var) \)
- \( sv = |VARIABLES| \)
- \( rv = \range(VARIABLES.var) \)

The constraint MAX_SURF_ZIGZAG(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern ZIGZAG for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘\((<>^+< | <>)(<>^+>) (|><)\)’.

Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).
Example

\[(21, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1), (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))\]

Figure 4.1141 provides an example where the \texttt{POS\_MAX\_SURF\_ZIGZAG} constraint holds.

![Diagram illustrating the POS\_MAX\_SURF\_ZIGZAG constraint](image)

Figure 4.1141: Illustrating the \texttt{POS\_MAX\_SURF\_ZIGZAG} constraint of the \textbf{Example} slot

**Typical**

\[|\text{VARIABLES}| > 3\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Arg. properties**

- Functional dependency: \texttt{VALUE} determined by \texttt{VARIABLES}.
- Functional dependency: \texttt{FOUND} determined by \texttt{VARIABLES}.
Automaton

Similar to the automaton of the \texttt{MAX\_SURF\_ZIGZAG} constraint but use the decoration table \texttt{3.35}.
Origin
Based on constraint MAX_WIDTH_DECREASING_SEQUENCE.

Constraint
POS_MAX_WIDTH_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 2
rv = 2 ⇒ VALUE ≤ 2
rv ≥ 3 ⇒ VALUE ≤ sv
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
sv = |VARIABLES|
rv = range(VARIABLES, var)

Purpose
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘>(>|=)’.” >’.
Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position i and ends at position j. The feature WIDTH computes the value j − i + 2.

Example
Figure 4.1142 provides an example where the POS_MAX_WIDTH_DECREASING_SEQUENCE (5, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 4, 6, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 4.1142: Illustrating the POS\_MAX\_WIDTH\_DECREASING\_SEQUENCE constraint of the Example slot

Typical

| VARIABLES | > 1  
| range(VARIABLES.var) | > 1  

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MAX_WIDTH_DECREASING_SEQUENCE` constraint but use the decoration table 3.35.
### Description

**Origin**
Based on constraint \texttt{MAX\_WIDTH\_DECREASING\_TERRACE}.

**Constraint**
\texttt{POS\_MAX\_WIDTH\_DECREASING\_TERRACE}(VALUE, VARIABLES, FOUND)

**Arguments**
- \texttt{VALUE} : \texttt{dvar}
- \texttt{VARIABLES} : \texttt{collection(var\_dvar)}
- \texttt{FOUND} : \texttt{collection(var\_dvar)}

**Restrictions**
\[
\begin{align*}
sv \leq 3 & \lor rv \leq 2 \Rightarrow VALUE = 0 \\
VALUE = 0 & \lor VALUE \geq 2 \\
VALUE & \leq \max(0, sv - 2k) \\
\text{required}(VARIABLES, var) & \\
\text{required}(FOUND, var) & \\
|VARIABLES| & = |FOUND|
\end{align*}
\]
where
\[
sv = |VARIABLES| \\
rv = \text{range}(VARIABLES.var)
\]

### Automaton

The constraint \texttt{MAX\_WIDTH\_DECREASING\_TERRACE}(VALUE, VARIABLES) holds. In addition, \texttt{FOUND} is a collection of 0\slash 1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern \texttt{DECREASING\_TERRACE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{DECREASING\_TERRACE} is the \texttt{maximal} subsequence which matches the regular expression \texttt{\text{"\(>=>+\)\}}.

Assume that the occurrence of the pattern \texttt{DECREASING\_TERRACE} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{WIDTH} computes the value \texttt{j} \texttt{\(\text{-} \text{i}\)}.

### Example

\[
\begin{pmatrix}
2, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3), \\
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1143 provides an example where the \texttt{POS\_MAX\_WIDTH\_DECREASING\_TERRACE}(2, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]) constraint holds.

**Typical**
- \texttt{|VARIABLES| > 3}
- \texttt{\text{range}(VARIABLES.var) > 2}
POS_MAX_WIDTH_DECREASING_TERRACE

Figure 4.1143: Illustrating the POS_MAX_WIDTH_DECREASING_TERRACE constraint of the Example slot

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_WIDTH_DECREASING_TERRACE constraint but use the decoration table 3.35.
The constraint \texttt{MAX_WIDTH_GORGE}(\texttt{VALUE}, \texttt{VARIABLES}, \texttt{FOUND}) holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{GORGE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern \texttt{GORGE} is the maximal subsequence which matches the regular expression '

\[ (> | > (= | >)*) (> | < (= | <)*) <) \]

Assume that the occurrence of the pattern \texttt{GORGE} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{WIDTH} computes the value \texttt{j} − \texttt{i}.

Figure 4.1144 provides an example where the \texttt{POS_MAX_WIDTH_GORGE} constraint holds.

\section*{Example}

\begin{equation}
(3, \{1, 7, 3, 4, 4.5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7\}, \{0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\})
\end{equation}

\section*{Typical}

| \texttt{VARIABLES} > 2  \\
| \texttt{range}(\texttt{VARIABLES.var}) > 1 |
POS_MAX_WIDTH_GORGE

Figure 4.1144: Illustrating the POS_MAX_WIDTH_GORGE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the MAX_WIDTH_GORGE constraint but use the decoration table 3.35.
| POS_MAX_WIDTH_GORGE | 2319 |
The constraint \texttt{MAX_WIDTH_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{INCREASING_SEQUENCE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern \texttt{INCREASING_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression ‘\textasciitilde (< | =)* < |’.

Assume that the occurrence of the pattern \texttt{INCREASING_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{WIDTH} computes the value \(j - i + 2\).

Figure 4.1145 provides an example where the \texttt{POS_MAX_WIDTH_INCREASING_SEQUENCE(5, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0])} constraint holds.
POS_MAX_WIDTH_INCREASING_SEQUENCE

Figure 4.1145: Illustrating the POS_MAX_WIDTH_INCREASING_SEQUENCE constraint of the Example slot

Typical

| VARIABLES | > 1
| range(VARIABLES.var) | > 1

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
| Automaton | Similar to the automaton of the **MAX_WIDTH_INCREASING_SEQUENCE** constraint but use the decoration table 3.35. |
POS_MAX_WIDTH_INCREASING_TERRACE

**Origin**
Based on constraint \texttt{MAX_WIDTH_INCREASING_TERRACE}.

**Constraint**
\texttt{POS_MAX_WIDTH_INCREASING_TERRACE(VALUE, VARIABLES, FOUND)}

**Arguments**
- \texttt{VALUE} : dvar
- \texttt{VARIABLES} : collection(var \rightarrow dvar)
- \texttt{FOUND} : collection(var \rightarrow dvar)

**Restrictions**
\[
sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = 0
\]
\[
VALUE = 0 \lor VALUE \geq 2
\]
\[
VALUE \leq \max(0, sv - 2d)
\]
\[
\text{required}(VARIABLES, \text{var})
\]
\[
\text{required}(FOUND, \text{var})
\]
\[
|VARIABLES| = |FOUND|
\]
where
\[
sv = |VARIABLES|
\]
\[
rv = \text{range}(VARIABLES, \text{var})
\]

The constraint \texttt{MAX_WIDTH_INCREASING_TERRACE(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{INCREASING_TERRACE} for which the feature value is \texttt{VALUE}.

**Purpose**
The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
An occurrence of the pattern \texttt{INCREASING_TERRACE} is the \texttt{maximal} subsequence which matches the regular expression \texttt{'< = + <'}.
Assume that the occurrence of the pattern \texttt{INCREASING_TERRACE} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{WIDTH} computes the value \texttt{j} \texttt{−} \texttt{i}.

**Example**
\[
\begin{pmatrix}
3, (1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4), \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0)
\end{pmatrix}
\]

Figure 4.1146 provides an example where the \texttt{POS_MAX_WIDTH_INCREASING_TERRACE} constraint holds.

**Typical**
- \texttt{|VARIABLES|} > 3
- \texttt{range}(VARIABLES, \text{var}) > 2
Figure 4.1146: Illustrating the POS_MAX_WIDTH_INCREASING_TERRACE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_WIDTH\_INCREASING\_TERRACE} constraint but use the decoration table 3.35.
POS_MAX_WIDTH_INCREASING_TERRACE
Origin Based on constraint MAX_WIDTH_INFLEXION.

Constraint POS_MAX_WIDTH_INFLEXION(VALUE, VARIABLES, FOUND)

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar}) \\
\text{FOUND} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \\
VALUE & \geq 0 \\
VALUE & \leq \max(0, sv - 2b) \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
|\text{VARIABLES}| & = |\text{FOUND}|
\end{align*}
\]

where

\[
\begin{align*}
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint MAX_WIDTH_INFLEXION(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INFLEXION for which the feature value is VALUE.

Purpose

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (< | =) > | > (>) = \)* <'.

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example

\[
\begin{pmatrix}
3, (1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4) \\
0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1147 provides an example where the POS_MAX_WIDTH_INFLEXION \((3, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4], [0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])\) constraint holds.

Typical

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.1147: Illustrating the POS_MAX_WIDTH_INFLEXION constraint of the Example slot

Arg. properties

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
| Automaton | Similar to the automaton of the MAX_WIDTH_INFLEXION constraint but use the decoration table 3.35. |
Origin
Based on constraint MAX_WIDTH_PEAK.

Constraint
POS_MAX_WIDTH_PEAK(VALUE, VARIABLES, FOUND)

Arguments
| VALUE  | dvar |
| VARIABLES | collection(var−dvar) |
| FOUND | collection(var−dvar) |

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE ≥ 0
VALUE ≤ max(0, sv − 2v)
required(VARIABLES, var)
required(FOUND, var)

where
sv = |VARIABLES|
rv = range(VARIABLES.var)

The constraint MAX_WIDTH_PEAK(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PEAK for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression '< (= | <) (> | =)* >'.

Assume that the occurrence of the pattern PEAK starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example
\[
\begin{pmatrix}
3, 7, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1, \v
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0
\end{pmatrix}
\]

Figure 4.1148 provides an example where the POS_MAX_WIDTH_PEAK constraint holds.

Typical
| VARIABLES | > 2 |
| range(VARIABLES.var) | > 1 |
Figure 4.1148: Illustrating the POS_MAX_WIDTH_PEAK constraint of the Example slot.

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_WIDTH_PEAK constraint but use the decoration table 3.35.
POS_MAX_WIDTH_PEAK

2335
Origin
Based on constraint \( \text{MAX\_WIDTH\_PLAIN} \).

Constraint
\[
\text{POS\_MAX\_WIDTH\_PLAIN}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

Arguments
- VALUE : dvar
- VARIABLES : \(\text{collection}(\text{var} - \text{dvar})\)
- FOUND : \(\text{collection}(\text{var} - \text{dvar})\)

Restrictions
- \(\text{sv} \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0\)
- \(\text{VALUE} \geq 0\)
- \(\text{VALUE} \leq \max(0, \text{sv} - 2\text{dvar})\)
- \(\text{required}(\text{VARIABLES}, \text{var})\)
- \(\text{required}(\text{FOUND}, \text{var})\)
- \(|\text{VARIABLES}| = |\text{FOUND}|\)
where
- \(\text{sv} = |\text{VARIABLES}|\)
- \(\text{rv} = \text{range}(\text{VARIABLES} . \text{var})\)

Purpose
An occurrence of the pattern \text{PLAIN} is the \textit{maximal} subsequence which matches the regular expression '\textgreater^*\textless'.
Assume that the occurrence of the pattern \text{PLAIN} starts at position \(i\) and ends at position \(j\). The feature \text{WIDTH} computes the value \(j - i\).

Example
\[
(2, \{2, 3, 6, 5, 7, 6, 4, 5, 4, 3, 3, 6, 3\}, \{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0\})
\]

Figure 4.1149 provides an example where the \text{POS\_MAX\_WIDTH\_PLAIN} constraint holds.

Typical
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES} . \text{var}) > 1\)
Figure 4.1149: Illustrating the POS_MAX_WIDTH_PLAIN constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_WIDTH_PLAIN constraint but use the decoration table 3.35.
The constraint \( \text{MAX\_WIDTH\_PLATEAU}(\text{VALUE}, \text{VARIABLES}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PLATEAU for which the feature value is \( \text{VALUE} \).

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression '\(<\leq\ast\rangle'.

Assume that the occurrence of the pattern PLATEAU starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

Figure 4.1150 provides an example where the \( \text{POS\_MAX\_WIDTH\_PLATEAU}(4, [1, 3, 5, 5, 5, 2, 4, 4, 3, 3, 1, 5, 5], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]) \) constraint holds.

Typical
\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.1150: Illustrating the POS_MAX_WIDTH_PLATEAU constraint of the Example slot

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_WIDTH\_PLATEAU} constraint but use the decoration table \texttt{3.35}.
Origin

Based on constraint \texttt{MAX\_WIDTH\_PROPER\_PLAIN}.

Constraint

\texttt{POS\_MAX\_WIDTH\_PROPER\_PLAIN}(VALUE, VARIABLES, FOUND)

Arguments

\begin{itemize}
\item \texttt{VALUE} : dvar
\item \texttt{VARIABLES} : collection(var–dvar)
\item \texttt{FOUND} : collection(var–dvar)
\end{itemize}

Restrictions

\begin{align*}
sv \leq 3 \lor rv \leq 1 & \Rightarrow VALUE = 0 \\
VALUE = 0 \lor VALUE \geq 2 & \\
VALUE \leq \max \left(0, sv - 2d\right) & \\
\text{required}(\text{VARIABLES}, var) & \\
\text{required}(\text{FOUND}, var) & \\
|\text{VARIABLES}| = |\text{FOUND}| & \\
\text{where} & \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.var)
\end{align*}

The constraint \texttt{MAX\_WIDTH\_PROPER\_PLAIN(VALUE, VARIABLES, FOUND)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{PROPER\_PLAIN} for which the feature value is \texttt{VALUE}.

Purpose

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{PROPER\_PLAIN} is the maximal subsequence which matches the regular expression ‘\texttt{> = + <}’.

Assume that the occurrence of the pattern \texttt{PROPER\_PLAIN} starts at position $i$ and ends at position $j$. The feature \texttt{WIDTH} computes the value $j - i$.

Example

\begin{align*}
(3, & \langle 2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5 \rangle, \\
& \langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0 \rangle)
\end{align*}

Figure \ref{fig:example} provides an example where the \texttt{POS\_MAX\_WIDTH\_PROPER\_PLAIN} \((3, [2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])\) constraint holds.

Typical

\begin{align*}
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.var) > 1
\end{align*}
Figure 4.1151: Illustrating the POS_MAX_WIDTH_PROPERPLAIN constraint of the Example slot

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
| Automaton | Similar to the automaton of the `MAX_WIDTH_PROPER_PLAIN` constraint but use the decoration table 3.35. |
**Origin**
Based on constraint `MAX_WIDTH_PROPER_PLATEAU`.

**Constraint**
```plaintext
POS_MAX_WIDTH_PROPER_PLATEAU(VALUE, VARIABLES, FOUND)
```

**Arguments**
- `VALUE`: dvar
- `VARIABLES`: collection(var−dvar)
- `FOUND`: collection(var−dvar)

**Restrictions**
- `sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = 0`
- `VALUE = 0 ∨ VALUE ≥ 2`
- `VALUE ≤ max(0, sv − 2v)`
- `required(VARIABLES, var)`
- `required(FOUND, var)`
- `|VARIABLES| = |FOUND|`

where
- `sv = |VARIABLES|`
- `rv = range(VARIABLES var)`

The constraint `MAX_WIDTH_PROPER_PLATEAU(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `PROPER_PLATEAU` for which the feature value is `VALUE`.

**Purpose**
The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
An occurrence of the pattern `PROPER_PLATEAU` is the maximal subsequence which matches the regular expression `'< =+ >'`.
Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

**Example**
```
3, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3),
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)
```

Figure 4.1152 provides an example where the `POS_MAX_WIDTH_PROPER_PLATEAU (3, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])` constraint holds.

**Typical**
- `|VARIABLES| > 3`
- `range(VARIABLES var) > 1`
Figure 4.1152: Illustrating the POS_MAX_WIDTH_PROPER_PLATEAU constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_WIDTH_PROPER_PLATEAU constraint but use the decoration table 3.35.
Origin Based on constraint \texttt{MAX\_WIDTH\_STEADY\_SEQUENCE}.

Constraint \texttt{POS\_MAX\_WIDTH\_STEADY\_SEQUENCE(VALUE, VARIABLES, FOUND)}

Arguments

\begin{itemize}
  \item \texttt{VALUE} : \texttt{dvar}
  \item \texttt{VARIABLES} : \texttt{collection(var\,–\,dvar)}
  \item \texttt{FOUND} : \texttt{collection(var\,–\,dvar)}
\end{itemize}

Restrictions

\begin{itemize}
  \item $sv \leq 1 \Rightarrow VALUE = 0$
  \item \hspace{1em} $VALUE = 0 \lor VALUE \geq 2$
  \item \hspace{1em} $rv = 1 \land \text{av} \geq 2 \Rightarrow VALUE \geq \text{av}$
  \item \hspace{1em} $rv \geq 2 \land \text{av} \geq 2 \Rightarrow VALUE \geq 0$
  \item \hspace{1em} $VALUE \leq \text{sv}$
  \item \hspace{1em} \texttt{required}(\texttt{VARIABLES}, \texttt{var})
  \item \hspace{1em} \texttt{required}(\texttt{FOUND}, \texttt{var})
  \item \hspace{1em} $|\texttt{VARIABLES}| = |\texttt{FOUND}|$
  \item \hspace{1em} where
  \begin{itemize}
    \item \hspace{1em} $sv = |\texttt{VARIABLES}|$
    \item \hspace{1em} $rv = \text{range}(\texttt{VARIABLES}, \texttt{var})$
  \end{itemize}
\end{itemize}

Purpose

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{STEADY\_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression ’\texttt{=\_+}’.

Assume that the occurrence of the pattern \texttt{STEADY\_SEQUENCE} starts at position $i$ and ends at position $j$. The feature \texttt{WIDTH} computes the value $j - i + 2$.

Example

\begin{itemize}
  \item $3, \langle 3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1 \rangle, \langle 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle$
\end{itemize}

Figure 4.1153 provides an example where the \texttt{POS\_MAX\_WIDTH\_STEADY\_SEQUENCE} constraint holds.
Figure 4.1153: Illustrating the POS_MAX_WIDTH_STEADY_SEQUENCE constraint of the Example slot

Typical

| VARIABLES | > 1 |

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
| Automaton | Similar to the automaton of the `MAX_WIDTH_STEADY_SEQUENCE` constraint but use the decoration table 3.35. |
Origin: Based on constraint \( \text{MAX}_\text{WIDTH}_\text{STRICLY}_{\text{DECREASING}}\text{SEQUENCE} \).

Constraint:

\[
\text{POS}_\text{MAX}_\text{WIDTH}_\text{STRICLY}_{\text{DECREASING}}\text{SEQUENCE}(\text{VALUE, VARIABLES, FOUND})
\]

Arguments:

- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection}(\text{var} \rightarrow \text{dvar}) \)
- \( \text{FOUND} : \text{collection}(\text{var} \rightarrow \text{dvar}) \)

Restrictions:

\[
\begin{align*}
sv & \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 & \lor \text{VALUE} \geq 2 \\
\text{VALUE} & \leq \min(sv, rv) \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
|\text{VARIABLES}| & = |\text{FOUND}|
\end{align*}
\]

where:

\[
\begin{align*}
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Purpose:

The constraint \( \text{MAX}_\text{WIDTH}_\text{STRICLY}_{\text{DECREASING}}\text{SEQUENCE}(\text{VALUE, VARIABLES}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \( \text{STRICLY}_{\text{DECREASING}}\text{SEQUENCE} \) for which the feature value is \( \text{VALUE} \).

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \( \text{STRICLY}_{\text{DECREASING}}\text{SEQUENCE} \) is the maximal subsequence which matches the regular expression ‘\( >^{+} \)’.

Assume that the occurrence of the pattern \( \text{STRICLY}_{\text{DECREASING}}\text{SEQUENCE} \) starts at position \( i \) and ends at position \( j \). The feature \( \text{WIDTH} \) computes the value \( j - i + 2 \).

Example:

\[
\begin{pmatrix}
3, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3), \\
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1154 provides an example where the \( \text{POS}_\text{MAX}_\text{WIDTH}_\text{STRICLY}_{\text{DECREASING}}\text{SEQUENCE} (3, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) \) constraint holds.

Typical:

\( |\text{VARIABLES}| > 1 \)

\( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)
Figure 4.1154: Illustrating the POS_MAX_WIDTH STRICTLY_DECREASING_SEQUENCE constraint of the Example slot

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX_WIDTH\_STRICLY\_DECREASING\_SEQUENCE} constraint but use the decoration table 3.35.
POS_MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE

Origin
Based on constraint \texttt{MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE}.

Constraint
\[
\text{POS_MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE} (\text{VALUE, VARIABLES, FOUND})
\]

Arguments
\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection} (\text{var} \rightarrow \text{dvar}) \\
\text{FOUND} & : \text{collection} (\text{var} \rightarrow \text{dvar})
\end{align*}
\]

Restrictions
\[
\begin{align*}
\text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 \lor \text{VALUE} \geq 2 \\
\text{VALUE} \leq \min (\text{sv}, \text{rv}) \\
\text{required} (\text{VARIABLES}, \text{var}) \\
\text{required} (\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| = |\text{FOUND}|
\end{align*}
\]

where
\[
\begin{align*}
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range} (\text{VARIABLES}.\text{var})
\end{align*}
\]

Purpose
The constraint \texttt{MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{STRICTLY_INCREASING_SEQUENCE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{STRICTLY_INCREASING_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression '\texttt{<}+'.

Assume that the occurrence of the pattern \texttt{STRICTLY_INCREASING_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{WIDTH} computes the value \(j - i + 2\).

Example
Figure 4.1155 provides an example where the \texttt{POS_MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE} (5, [4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical
\[
|\text{VARIABLES}| > 1 \\
\text{range} (\text{VARIABLES}.\text{var}) > 1
\]
Figure 4.1155: Illustrating the POS_MAX_WIDTH STRICTLY_INCREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE} constraint but use the decoration table 3.35.
POS_MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE
Origin Based on constraint \textit{MAX}\_\textit{WIDTH}\_\textit{SUMMIT}.

Constraint \texttt{POS\_MAX\_WIDTH\_SUMMIT}(VALUE, VARIABLES, FOUND)

Arguments
\begin{itemize}
\item VALUE : dvar
\item VARIABLES : collection(var\_dvar)
\item FOUND : collection(var\_dvar)
\end{itemize}

Restrictions
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{rv} = 2 & \Rightarrow \text{VALUE} \leq 1 \\
\text{rv} & \geq 3 \Rightarrow \text{VALUE} \leq \max(0, \text{sv} - 2) \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
|\text{VARIABLES}| & = |\text{FOUND}|
\end{align*}

where
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES} . \text{var})
\end{align*}

The constraint \textit{MAX}\_\textit{WIDTH}\_\textit{SUMMIT}(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{SUMMIT} for which the feature value is \textit{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

Purpose
An occurrence of the pattern \texttt{SUMMIT} is the \textit{maximal} subsequence which matches the regular expression \texttt{'(< | < = | < > | > > = | > > >)'}. Assume that the occurrence of the pattern \texttt{SUMMIT} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{WIDTH} computes the value \texttt{j – i}.

Example
\begin{align*}
(3, 7, 1, 5, 4, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1) ,
(0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{align*}

Figure 4.1156 provides an example where the \texttt{POS\_MAX\_WIDTH\_SUMMIT}(3, [7, 1, 5, 4, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES} . \text{var}) & > 1
\end{align*}
Figure 4.1156: Illustrating the POS_MAX_WIDTH_SUMMIT constraint of the Example slot

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_WIDTH_SUMMIT constraint but use the decoration table 3.35.
POS_MAX_WIDTH_SUMMIT  2367
DESCRIPTION AUTOMATON

Origin
Based on constraint MAX_WIDTH_VALLEY.

Constraint
POS_MAX_WIDTH_VALLEY (VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 2 \lor rv ≤ 1 \Rightarrow VALUE = 0
VALUE ≥ 0
VALUE ≤ max(0, sv − 2v)
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
sv = |VARIABLES|
rv =range(VARIABLES.var)

The constraint MAX_WIDTH_VALLEY(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern VALLEY for which the feature value is VALUE.
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

Purpose
An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression ‘> (= | >)*(< | =)*<’.
Assume that the occurrence of the pattern VALLEY starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example
\[
\begin{pmatrix}
4, 1, 3, 7, 4, 3, 6, 5, 5, 5, 5, 7 \\
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1157 provides an example where the POS_MAX_WIDTH_VALLEY (4, [1, 3, 7, 4, 3, 6, 5, 5, 5, 5, 5, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1
Figure 4.1157: Illustrating the POS_MAX_WIDTH_VALLEY constraint of the Example slot

Arg. properties

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_WIDTH_VALLEY constraint but use the decoration table 3.35.
POS_MAX_WIDTH_VALLEY

2371
**Origin**  
Based on constraint `MAX_WIDTH_ZIGZAG`.

**Constraint**  
`POS_MAX_WIDTH_ZIGZAG(VALUE, VARIABLES, FOUND)`

**Arguments**  
- `VALUE` : `dvar`
- `VARIABLES` : `collection(var−dvar)`
- `FOUND` : `collection(var−dvar)`

**Restrictions**  
- `sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = 0`
- `VALUE = 0 ∨ VALUE ≥ 2`
- `VALUE ≤ max(0, sv − 2v)`
- `required(VARIABLES, var)`
- `required(FOUND, var)`
- `|VARIABLES| = |FOUND|`
  
  where  
  - `sv = |VARIABLES|`
  - `rv = range(VARIABLES.var)`

The constraint `MAX_WIDTH_ZIGZAG(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `ZIGZAG` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

**Purpose**  
An occurrence of the pattern `ZIGZAG` is the maximal subsequence which matches the regular expression `'<>'*(< | <>) | ('>'*(> | <>)`.

Assume that the occurrence of the pattern `ZIGZAG` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

**Example**  


Figure 4.1158 provides an example where the `POS_MAX_WIDTH_ZIGZAG` constraint holds.

**Typical**  
- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 1`
Figure 4.1158: Illustrating the POS_MAX_WIDTH_ZIGZAG constraint of the **Example** slot

**Arg. properties**
- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_WIDTH_ZIGZAG constraint but use the decoration table 3.35.
The constraint MIN_HEIGHT_DECREASING_TERRACE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING_TERRACE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression ‘\(>+=\)’.

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i+1\) to index \(j\).

Example

\[
\left( 2, \{0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \right)
\]

Figure 4.1159 provides an example where the POS_MIN_HEIGHT_DECREASING_TERRACE constraint holds.
Figure 4.1159: Illustrating the POS_MIN_HEIGHT_DECREASING_TERRACE constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 2 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN\_HEIGHT\_DECREASING\_TERRACE constraint but use the decoration table 3.35.
POS_MIN_HEIGHT_DECREASING_TERRACE
**POS_MIN_HEIGHT_INCREMENTING_TERRACE**

**DESCRIPTION**

**Origin**
Based on constraint **MIN_HEIGHT_INCREMENTING_TERRACE**.

**Constraint**

\[
\text{POS\_MIN\_HEIGHT\_INCREMENTING\_TERRACE}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

**Arguments**

- **VALUE**: \( \text{dvar} \)
- **VARIABLES**: \( \text{collection(\text{var}-\text{dvar})} \)
- **FOUND**: \( \text{collection(\text{var}-\text{dvar})} \)

**Restrictions**

- \( \text{sv} \leq 3 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = +\infty \)
- \( \text{VALUE} \geq \text{minv} + 1 \)
- \( \text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} - 1 \)
- \( \text{required(\text{VARIABLES, var})} \)
- \( \text{required(\text{FOUND, var})} \)

where

- \( \text{minv = minval(\text{VARIABLES, var})} \)
- \( \text{maxv = maxval(\text{VARIABLES, var})} \)
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{rv = range(\text{VARIABLES, var})} \)

The constraint **MIN_HEIGHT_INCREMENTING_TERRACE(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **INCREMENTING_TERRACE** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **INCREMENTING_TERRACE** is the maximal subsequence which matches the regular expression ‘\(<=+<\)’.

Assume that the occurrence of the pattern **INCREMENTING_TERRACE** starts at position \( i \) and ends at position \( j \). The feature **MIN**, called **HEIGHT** in the name of the constraint since all feature values are identical, computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**

\[
(3, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0])
\]

Figure 4.1160 provides an example where the **POS_MIN_HEIGHT_INCREMENTING_TERRACE** constraint holds.
Figure 4.1160: Illustrating the POS_MIN_HEIGHT_INCREASING_TERRACE constraint of the Example slot

Typical

| VARIABLES | > 3  
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES)</td>
<td>&gt; 2</td>
</tr>
</tbody>
</table>

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MIN_HEIGHT_INCREASING_TERRACE` constraint but use the decoration table 3.35.
Origin
Based on constraint MIN_HEIGHTPLAIN.

Constraint

\[
\text{POS_MIN_HEIGHTPLAIN}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar}) \\
\text{FOUND} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{minv} \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \text{maxv} - 1 \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{required}(\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| & = |\text{FOUND}|
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} & = \text{minval}(\text{VARIABLES}, \text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}, \text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}, \text{var})
\end{align*}
\]

Purpose

The constraint MIN_HEIGHTPLAIN(\text{VALUE}, \text{VARIABLES}) holds. In addition, \text{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{PLAIN} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{PLAIN} is the maximal subsequence which matches the regular expression ' > = ^ < .

Assume that the occurrence of the pattern \text{PLAIN} starts at position \text{i} and ends at position \text{j}. The feature \text{MIN}, called \text{HEIGHT} in the name of the constraint since all feature values are identical, computes the minimum of the values from index \text{i} + 1 to index \text{j}.

Example

\[
\begin{pmatrix}
3, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1161 provides an example where the \text{POS_MIN_HEIGHTPLAIN} \( (3, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]) \) constraint holds.
Figure 4.161: Illustrating the POS_MIN_HEIGHT_PLAIN constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 2 \]

\[ \text{range}(\text{VARIABLES.var}) > 1 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN\_HEIGHT\_PLAIN constraint but use the decoration table 3.35.
The constraint \texttt{MIN\_HEIGHT\_PLATEAU(VARIABLES, FOUND)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{PLATEAU} for which the feature value is \texttt{VALUE}. The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern \texttt{PLATEAU} is the \textit{maximal} subsequence which matches the regular expression \\
\texttt{<\=\*>}'. Assume that the occurrence of the pattern \texttt{PLATEAU} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{MIN}, called \texttt{HEIGHT} in the name of the constraint since all feature values are identical, computes the minimum of the values from index \texttt{i} + 1 to index \texttt{j}.
Figure 4.1162: Illustrating the POS_MIN_HEIGHT_PLATEAU constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_HEIGHT_PLATEAU constraint but use the decoration table 3.35.
POS_MIN_HEIGHT_PLATEAU  2391
2392

POS_MIN_HEIGHT_PROPER_PLAIN

description

AUTOMATON

Origin
Based on constraint MIN_HEIGHT_PROPER_PLAIN.

Constraint
POS_MIN_HEIGHT_PROPER_PLAIN(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var=dvar)
FOUND : collection(var=dvar)

Restrictions
\( sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = +\infty \)
VALUE \( \geq \) minv\$
VALUE = +\infty \lor VALUE \leq maxv - 1
required(VARIABLES, var)
required(FOUND, var)
\(|VARIABLES| = |FOUND|\)
where
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

The constraint MIN_HEIGHT_PROPER_PLAIN(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PROPER_PLAIN for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern PROPER_PLAIN is the maximal subsequence which matches the regular expression ‘\(> = + <\)’. Assume that the occurrence of the pattern PROPER_PLAIN starts at position \(i\) and ends at position \(j\). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\).

Example
\[
\left( 3, \{2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5\}, \{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0\}\right)
\]

Figure 4.1163 provides an example where the POS_MIN_HEIGHT_PROPER_PLAIN (3, [2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.
Figure 4.1163: Illustrating the POS_MIN_HEIGHT_PROPER_PLAIN constraint of the Example slot

Typical

\[ \text{\textbf{VARIABLES}} > 3 \]
\[ \text{range(\text{\textbf{VARIABLES}}.var)} > 1 \]

Arg. properties

- Functional dependency: VALUE determined by \text{\textbf{VARIABLES}}.
- Functional dependency: FOUND determined by \text{\textbf{VARIABLES}}.
Automaton

Similar to the automaton of the MIN_HEIGHT_PROPER_PLAIN constraint but use the decoration table 3.35.
Origin: Based on constraint MIN_HEIGHT_PROPER_PLATEAU.

Constraint: POS_MIN_HEIGHT_PROPER_PLATEAU(VALUE, VARIABLES, FOUND)

Arguments:
- VALUE: dvar
- VARIABLES: collection(var - dvar)
- FOUND: collection(var - dvar)

Restrictions:
- \( sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = +\infty \)
- \( VALUE \geq \minv + 1 \)
- \( VALUE = +\infty \lor VALUE \leq \maxv \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)
- \( \text{required}(\text{FOUND}, \text{var}) \)
- \(|\text{VARIABLES}| = |\text{FOUND}| \)

where:
- \( \minv = \minval(\text{VARIABLES}.\text{var}) \)
- \( \maxv = \maxval(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

Purpose:
The constraint MIN_HEIGHT_PROPER_PLATEAU(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PROPER_PLATEAU for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression '\('<=+>'\).

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \( i \) and ends at position \( j \). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \( i + 1 \) to index \( j \).

Example:

\[
\begin{pmatrix}
3, \{7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3\}, \\
\{0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\}
\end{pmatrix}
\]

Figure 4.1164 provides an example where the POS_MIN_HEIGHT_PROPER_PLATEAU (3, \{7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3\}; \{0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\}) constraint holds.
Figure 4.1164: Illustrating the POS_MIN_HEIGHT_PROPER_PLATEAU constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 3 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_HEIGHT_PROPER_PLATEAU constraint but use the decoration table 3.35.
POS_MIN_HEIGHT_STEADY

Origin
Based on constraint MIN_HEIGHT_STEADY.

Constraint
POS_MIN_HEIGHT_STEADY(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
\[ sv \leq 1 \Rightarrow VALUE = +\infty \]
\[ VALUE \geq \minv \]
\[ VALUE = +\infty \lor VALUE \leq \maxv \]
required(VARIABLES, var)
required(FOUND, var)
\[ |VARIABLES| = |FOUND| \]

where
\[ \minv = \minval(VARIABLES, var) \]
\[ \maxv = \maxval(VARIABLES, var) \]
\[ sv = |VARIABLES| \]

Purpose
The constraint MIN_HEIGHT_STEADY(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STEADY for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern STEADY is the subsequence which matches the regular expression ‘=’.

Assume that the occurrence of the pattern STEADY starts at position \( i \) and ends at position \( j \). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \( i \) to index \( j + 1 \).

Example

\[
\begin{pmatrix}
1, (1, 1, 7, 3), (3, 5, 5, 6, 5, 5, 5, 7, 2, 6, 6), \ldots \\
(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 4.1165 provides an example where the POS_MIN_HEIGHT_STEADY (1, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 5, 7, 2, 6, 6], [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical
\[ |VARIABLES| > 1 \]
Figure 4.1165: Illustrating the POS_MIN_HEIGHT_STEADY constraint of the **Example** slot

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
| **Automaton** | Similar to the automaton of the **MIN_HEIGHT_STEADY** constraint but use the decoration table 3.35. |
Origin Based on constraint \textit{MIN\_HEIGHT\_STEADY\_SEQUENCE}.

Constraint \textit{POS\_MIN\_HEIGHT\_STEADY\_SEQUENCE}(VALUE, VARIABLES, FOUND)

Arguments \begin{itemize}
  \item VALUE : dvar
  \item VARIABLES : collection(var\text{-}dvar)
  \item FOUND : collection(var\text{-}dvar)
\end{itemize}

Restrictions \begin{itemize}
  \item \(sv \leq 1 \Rightarrow VALUE = +\infty\)
  \item \(VALUE \geq minv\)
  \item \(VALUE = +\infty \lor VALUE \leq maxv\)
  \item \(\text{required}(\text{VARIABLES}, \text{var})\)
  \item \(\text{required}(\text{FOUND}, \text{var})\)
  \item \(|\text{VARIABLES}| = |\text{FOUND}|\)
\end{itemize}

where \begin{itemize}
  \item \(minv = \text{minval}(\text{VARIABLES}.\text{var})\)
  \item \(maxv = \text{maxval}(\text{VARIABLES}.\text{var})\)
  \item \(sv = |\text{VARIABLES}|\)
\end{itemize}

\begin{itemize}
  \item The constraint \textit{MIN\_HEIGHT\_STEADY\_SEQUENCE}(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the \textit{found} letter in those occurrences of the pattern \textit{STEADY\_SEQUENCE} for which the feature value is VALUE.
  \item The position of the \textit{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
  \item An occurrence of the pattern \textit{STEADY\_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression \texttt{\textquoteleft \textquoteleft =\textquoteleft \textquoteleft +\textquoteleft \textquoteleft}.
  \item Assume that the occurrence of the pattern \textit{STEADY\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \textit{MIN}, called \textit{HEIGHT} in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i\) to index \(j+1\).
\end{itemize}

Example \(\left(\begin{array}{cccccccccc}
1, 3, 1, 4, 5, 5, 6, 2, 2, 4, 3, 2, 1, 1 \\
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
\end{array}\right)\)

Figure 4.1166 provides an example where the \textit{POS\_MIN\_HEIGHT\_STEADY\_SEQUENCE} \(\langle 1, 3, 1, 4, 5, 5, 6, 2, 2, 4, 3, 2, 1, 1 \rangle \langle 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0 \rangle\) constraint holds.
Figure 4.1166: Illustrating the POS_MIN_HEIGHT_STEADY_SEQUENCE constraint of the Example slot

Typical

$|\text{VARIABLES}| > 1$

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_HEIGHT_STEADY_SEQUENCE constraint but use the decoration table 3.35.
The constraint \( \text{MIN\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE} \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{BUMP\_ON\_DECREASING\_SEQUENCE} for which the feature value is \( \text{VALUE} \).

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{BUMP\_ON\_DECREASING\_SEQUENCE} is the subsequence which matches the regular expression ‘\( >><>> \)’. Assume that the occurrence of the pattern \text{BUMP\_ON\_DECREASING\_SEQUENCE} starts at position \( i \) and ends at position \( j \). The feature \text{MAX} computes the maximum of the values from index \( i + 2 \) to index \( j \).

Example

\[
(5, \{7, 6, 5, 6, 6, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3\}, \{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0\})
\]

Figure 4.1167 provides an example where the \( \text{POS\_MIN\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE} \) constraint holds.
Figure 4.1167: Illustrating the POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

Typical

$|\text{VARIABLES}| > 5$

$\text{range}(\text{VARIABLES}\text{.var}) > 2$

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the MIN(MAX,BUMP,ON,DECREASING,SEQUENCE) constraint but use the decoration table 3.35.
The constraint $\text{MIN\_MAX\_DECREASING}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})$ holds. In addition, $\text{FOUND}$ is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern DECREASING is the subsequence which matches the regular expression ‘>’.

Assume that the occurrence of the pattern DECREASING starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i$ to index $j + 1$.

Figure 4.1168 provides an example where the $\text{POS\_MIN\_MAX\_DECREASING}$ constraint holds.
Figure 4.1168: Illustrating the POS_MIN_MAX_DECREASING constraint of the Example slot

Typical

\[ |\text{VARIABLES} | > 1 \]

\[ \text{range}(\text{VARIABLES.var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  
Similar to the automaton of the `MIN_MAX_DECREASING` constraint but use the decoration table 3.35.
Based on constraint $\text{MIN\_MAX\_DECREASING\_SEQUENCE}$.

**Constraint**

$$\text{POS\_MIN\_MAX\_DECREASING\_SEQUENCE}((\text{VALUE}, \text{VARIABLES}, \text{FOUND}))$$

**Arguments**

$$\begin{align*}
\text{VALUE} &: \text{dvar} \\
\text{VARIABLES} &: \text{collection} (\text{var} - \text{dvar}) \\
\text{FOUND} &: \text{collection} (\text{var} - \text{dvar})
\end{align*}$$

**Restrictions**

$$\begin{align*}
sv &\leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} &\geq \minv + 1 \\
\text{VALUE} = +\infty \lor \text{VALUE} \leq \maxv \\
\text{required}(\text{VARIABLES}, \text{var}) &\Rightarrow \text{required}(\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| &= |\text{FOUND}|
\end{align*}$$

where

$$\begin{align*}
\minv &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\maxv &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}$$

The constraint $\text{MIN\_MAX\_DECREASING\_SEQUENCE}((\text{VALUE}, \text{VARIABLES}))$ holds. In addition, $\text{FOUND}$ is a collection of 0/1 variables where the value 1 indicates the position of the $\text{FOUND}$ letter in those occurrences of the pattern $\text{DECREASING\_SEQUENCE}$ for which the feature value is $\text{VALUE}$.

The position of the $\text{FOUND}$ letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern $\text{DECREASING\_SEQUENCE}$ is the maximal subsequence which matches the regular expression '$>(>|=)*>\mid>'$.

Assume that the occurrence of the pattern $\text{DECREASING\_SEQUENCE}$ starts at position $i$ and ends at position $j$. The feature $\text{MAX}$ computes the maximum of the values from index $i$ to index $j + 1$.

**Example**

$$\begin{pmatrix}
4, \langle 3, 4, 2, 5, 6, 4, 4, 3, 1, 4, 6, 4, 4 \rangle \\
\langle 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle
\end{pmatrix}$$

Figure 4.1169 provides an example where the $\text{POS\_MIN\_MAX\_DECREASING\_SEQUENCE}$ constraint holds.
Figure 4.1169: Illustrating the POS_MIN_MAX_DECREASING_SEQUENCE constraint of the Example slot

Typical

| | VARIABLES| > 1

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the \texttt{MIN\_MAX\_DECREASING\_SEQUENCE} constraint but use the decoration table \texttt{3.35}.
Based on constraint \texttt{MIN\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE}.

\textbf{Arguments}

\begin{itemize}
  \item \texttt{VALUE} : \texttt{dvar}
  \item \texttt{VARIABLES} : \texttt{collection(var-dvar)}
  \item \texttt{FOUND} : \texttt{collection(var-dvar)}
\end{itemize}

\textbf{Restrictions}

\begin{align*}
  sv &\leq 5 \lor rv \leq 2 \Rightarrow VALUE = +\infty \\
  VALUE &\geq minv + 2 \\
  VALUE &\geq +\infty \lor VALUE \leq maxv \\
  required &\text{\texttt{VARIABLES}.var} \\
  required &\text{\texttt{FOUND}.var} \\
  |\texttt{VARIABLES}| &\text{\texttt{FOUND}}
\end{align*}

where

\begin{align*}
  minv &= \text{\texttt{minval}\texttt{VARIABLES}.var} \\
  maxv &= \text{\texttt{maxval}\texttt{VARIABLES}.var} \\
  sv &= |\texttt{VARIABLES}| \\
  rv &= \text{\texttt{range}\texttt{VARIABLES}.var}
\end{align*}

The constraint \texttt{MIN\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{DIP\_ON\_INCREASING\_SEQUENCE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{DIP\_ON\_INCREASING\_SEQUENCE} is the subsequence which matches the regular expression ‘<<><<<’.

Assume that the occurrence of the pattern \texttt{DIP\_ON\_INCREASING\_SEQUENCE} starts at position \textit{i} and ends at position \textit{j}. The feature \texttt{MAX} computes the maximum of the values from index \textit{i} + 2 to index \textit{j}.

\begin{itemize}
  \item \texttt{Example}:
\end{itemize}

\begin{itemize}
  \item \texttt{Example}:
\end{itemize}

\begin{itemize}
  \item \texttt{Example}:
\end{itemize}

Figure 4.1170 provides an example where the \texttt{POS\_MIN\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE(5, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])} constraint holds.
Figure 4.1170: Illustrating the POS_MIN_MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 5 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 2 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MAX_DIP_ON_INCREASING_SEQUENCE constraint but use the decoration table 3.35.
Origin
Based on constraint MIN_MAX_INCREASING.

Constraint
POS_MIN_MAX_INCREASING(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty
VALUE = \infty \lor VALUE \leq maxv
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|

where
minv = minval(VARIABLES, var)
maxv = maxval(VARIABLES, var)
sv = |VARIABLES|
rv = range(VARIABLES, var)

The constraint MIN_MAX_INCREASING(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression ‘<’.

Assume that the occurrence of the pattern INCREASING starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i to index j + 1.

Example

\[
(3, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0])
\]

Figure 4.1171 provides an example where the POS_MIN_MAX_INCREASING (3, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0]) constraint holds.
Figure 4.1171: Illustrating the POS_MIN_MAX_INCREASING constraint of the Example slot

Typical

\[|\text{VARIABLES}| > 1\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Similar to the automaton of the MIN_MAX_INCREASING constraint but use the decoration table 3.35.
**Origin**
Based on constraint `MIN_MAX_INCREASING_SEQUENCE`.

**Constraint**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>FOUND</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Arguments**

- **VALUE**: $\{dvar\}$
- **VARIABLES**: $\{\text{collection}(\text{var}−\text{dvar})\}$
- **FOUND**: $\{\text{collection}(\text{var}−\text{dvar})\}$

**Restrictions**

- $sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty$
- $VALUE \geq \text{minv} + 1$
- $VALUE = +\infty \lor VALUE \leq \text{maxv}$
- $\text{required}(\text{VARIABLES}, \text{var})$
- $\text{required}(\text{FOUND}, \text{var})$
- $|\text{VARIABLES}| = |\text{FOUND}|$

where

- $\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})$
- $\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})$
- $sv = |\text{VARIABLES}|$
- $rv = \text{range}(\text{VARIABLES}.\text{var})$

The constraint `MIN_MAX_INCREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `INCREASING_SEQUENCE` for which the feature value is `VALUE`.

**Purpose**

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `\(< (< | =)^* < | < \)\}'.

Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `MAX` computes the maximum of the values from index $i$ to index $j + 1$.

**Example**

```
(3, [4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0)
```

Figure 4.1172 provides an example where the `POS_MIN_MAX_INCREASING_SEQUENCE` constraint holds.
Figure 4.1172: Illustrating the POS_MIN_MAX_INCREASING_SEQUENCE constraint of the Example slot

Typical

\[|\text{VARIABLES}| > 1\]

\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MAX_INCREASING_SEQUENCE constraint but use the decoration table 3.35.
POS_MIN_MAX_INCREASING_SEQUENCE 2431
Origin
Based on constraint MIN_MAX_INFLEXION.

Constraint
POS_MIN_MAX_INFLEXION(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ minv
VALUE = +∞ ∨ VALUE ≤ maxv
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
The constraint MIN_MAX_INFLEXION(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INFLEXION for which the feature value is VALUE.
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression '< (< | =)* > | > (> | =)* >'.
Assume that the occurrence of the pattern INFLEXION starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 1 to index j.

Example

Figure 4.1173 provides an example where the POS_MIN_MAX_INFLEXION
(1,[1,2,6,4,4,3,5,2,5,1,5,3,3,4,4],0,0,0,0,0,0,0,0,1,0,0,0,0,0) constraint holds.
Figure 4.1173: Illustrating the POS_MIN_MAX_INFLEXION constraint of the Example slot

Typical
| VARIABLES | > 2
range(VARIABLES.var) | > 1

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MAX_INFLEXION constraint but use the decoration table 3.35.
POS_MIN_MAX_INFLEXION

2435
Based on constraint MIN_MAX_PEAK.

| Constraint | VALUE : dvar | VARIABLES : collection(var−dvar) | FOUND : collection(var−dvar) |

Restrictions

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE & \geq minv + 1 \\
VALUE & = +\infty \lor VALUE \leq maxv \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
|\text{VARIABLES}| & = |\text{FOUND}|
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint \text{MIN_MAX_PEAK}(\text{VALUE, VARIABLES}) holds. In addition, \text{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{PEAK} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{PEAK} is the \text{maximal} subsequence which matches the regular expression ‘\(< (= | <) (> | =) > > \)’.

Assume that the occurrence of the pattern \text{PEAK} starts at position \(i\) and ends at position \(j\). The feature \text{MAX} computes the maximum of the values from index \(i + 1\) to index \(j\).

Example

\[
\begin{pmatrix}
3, 7, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1 \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
\end{pmatrix}
\]

Figure 4.1174 provides an example where the \text{POS_MIN_MAX_PEAK} constraint holds.
Figure 4.1174: Illustrating the POS_MIN_MAX_PEAK constraint of the Example slot

Typical

| VARIABLES | > 2
range(VARIABLES.var) | > 1

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the MIN_MAX PEAK constraint but use the decoration table 3.35.
The constraint MIN_MAX_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STRICTLY_DECREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘>’. Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

Figure 4.1175 provides an example where the POS_MIN_MAX_STRICTLY_DECREASING_SEQUENCE constraint holds.
Figure 4.1175: Illustrating the POS_MIN_MAX.Strictly Decreasing.Sequence constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN\_MAX\_STRICTLY\_DECREASING\_SEQUENCE} constraint but use the decoration table \texttt{3.35}.
POS_MIN_MAX_STRICTLY_DECREASING_SEQUENCE 2443
Origin
Based on constraint \text{MIN\_MAX\_STRICTLY\_INCREASING\_SEQUENCE}.

Constraint
\text{POS\_MIN\_MAX\_STRICTLY\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES, FOUND})

Arguments
\begin{itemize}
\item \text{VALUE} \ : \ \text{dvar}
\item \text{VARIABLES} \ : \ \text{collection} (\text{var} \rightarrow \text{dvar})
\item \text{FOUND} \ : \ \text{collection} (\text{var} \rightarrow \text{dvar})
\end{itemize}

Restrictions
\begin{itemize}
\item \text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty
\item \text{VALUE} \geq \text{minv} + 1$
\item \text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv}$
\item \text{required} (\text{VARIABLES}.\text{var})
\item \text{required} (\text{FOUND}.\text{var})
\item \text{|VARIABLES|} = \text{|FOUND|}
\end{itemize}

where
\begin{itemize}
\item \text{minv} = \text{minval} (\text{VARIABLES}.\text{var})
\item \text{maxv} = \text{maxval} (\text{VARIABLES}.\text{var})
\item \text{sv} = \text{|VARIABLES|}
\item \text{rv} = \text{range} (\text{VARIABLES}.\text{var})
\end{itemize}

Purpose
The constraint \text{MIN\_MAX\_STRICTLY\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES}) holds. In addition, \text{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{STRICTLY\_INCREASING\_SEQUENCE} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{STRICTLY\_INCREASING\_SEQUENCE} is the maximal sub-sequence which matches the regular expression '\text{<+}.'

Assume that the occurrence of the pattern \text{STRICTLY\_INCREASING\_SEQUENCE} starts at position \text{i} and ends at position \text{j}. The feature \text{MAX} computes the maximum of the values from index \text{i} to index \text{j} + 1.

Example
\begin{itemize}
\item \text{VALUE} \geq 1
\end{itemize}

Figure 4.1176 provides an example where the \text{POS\_MIN\_MAX\_STRICTLY\_INCREASING\_SEQUENCE}(3, [4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3], |0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.
Figure 4.1176: Illustrating the \texttt{POS\_MIN\_MAX\_STRICTLY\_INCREASING\_SEQUENCE} constraint of the \textbf{Example} slot

**Typical**

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range} (\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \text{MIN\_MAX\_STRICTLY\_INCREASING\_SEQUENCE} constraint but use the decoration table 3.35.
POS_MIN_MAX STRICTLY_INCREASING_SEQUENCE
Origin: Based on constraint MIN_MIN_MAX_SUMMIT.

Constraint: POS_MIN_MAX_SUMMIT(VALUE, VARIABLES, FOUND)

Arguments:
- VALUE: dvar
- VARIABLES: collection(var - dvar)
- FOUND: collection(var - dvar)

Restrictions:
- $sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty$
- $VALUE \geq minv + 1$
- $VALUE = +\infty \lor VALUE \leq maxv$

The constraint MIN_MIN_MAX_SUMMIT(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern SUMMIT for which the feature value is VALUE.

Purpose:
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

Assume that the occurrence of the pattern SUMMIT starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i + 1$ to index $j$.

Example:

Figure 4.1177 provides an example where the POS_MIN_MAX_SUMMIT (3, [7, 1, 5, 4, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 4.1177: Illustrating the POS_MIN_MAX_SUMMIT constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES} . \text{var}) > 1 \]

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MAX_SUMMIT constraint but use the decoration table 3.35.
POS_MIN_MAX_SUMMIT 2451
The constraint $\text{MIN\_MAX\_ZIGZAG}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})$ holds. In addition, $\text{FOUND}$ is a collection of 0/1 variables where the value 1 indicates the position of the $\text{found}$ letter in those occurrences of the pattern $\text{ZIGZAG}$ for which the feature value is $\text{VALUE}$.

The position of the $\text{found}$ letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern $\text{ZIGZAG}$ is the maximal subsequence which matches the regular expression ‘$\langle\rangle^+\langle|\langle\rangle\rangle|\langle\rangle^+\langle|\rangle\rangle$’.

Assume that the occurrence of the pattern $\text{ZIGZAG}$ starts at position $i$ and ends at position $j$. The feature $\text{MAX}$ computes the maximum of the values from index $i + 1$ to index $j$.

Figure 4.1178 provides an example where the $\text{POS\_MIN\_MAX\_ZIGZAG}$ constraint holds.
Figure 4.1178: Illustrating the POS_MIN_MAX_ZIGZAG constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 3 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MAX_ZIGZAG constraint but use the decoration table 3.35.
The constraint \texttt{MIN\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} is the subsequence which matches the regular expression ‘\texttt{>><<>>}’.

Assume that the occurrence of the pattern \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{MIN} computes the minimum of the values from index \texttt{i + 2} to index \texttt{j}.
Figure 4.179: Illustrating the POS_MIN_MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 5 \]
\[ \text{range(Variable.var)} > 2 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the MIN_MIN_BUMP_ON_DECREASING_SEQUENCE constraint but use the decoration table 3.35.
Based on constraint MIN_MIN_DECREASING.

The constraint \(\text{MIN\_MIN\_DECREASING}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})\) holds. In addition, \(\text{FOUND}\) is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING for which the feature value is \(\text{VALUE}\).

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern DECREASING is the subsequence which matches the regular expression ‘>’.

Assume that the occurrence of the pattern DECREASING starts at position \(i\) and ends at position \(j\). The feature \(\text{MIN}\) computes the minimum of the values from index \(i\) to index \(j + 1\).

Figure 4.1180 provides an example where the \(\text{POS\_MIN\_MIN\_DECREASING}\) constraint holds.
Figure 4.1180: Illustrating the POS_MIN_MIN_DECREASING constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MIN_DECREASING constraint but use the decoration table 3.35.
Origin
Based on constraint MIN_MIN_DECREASING_SEQUENCE.

Constraint
POS_MIN_MIN_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ minv
VALUE = +∞ ∨ VALUE ≤ maxv − 1
required(VARIABLES.var)
required(FOUND.var)
|VARIABLES| = |FOUND|

where
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

The constraint MIN_MIN_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘> (> | =)∗ > | >’. Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

Example
\[
\begin{pmatrix}
1, 3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4)
\end{pmatrix}
\]

Figure 4.1181 provides an example where the constraint holds.
POS_MIN_MIN_DECREASING_SEQUENCE

Figure 4.1181: Illustrating the POS_MIN_MIN_DECREASING_SEQUENCE constraint of the Example slot

Typical

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Similar to the automaton of the \texttt{MIN\_MIN\_DECREASING\_SEQUENCE} constraint but use the decoration table \texttt{3.35}. 
Constraint: $\text{POS\_MIN\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES, FOUND})$

Arguments:
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

Restrictions:
- $\text{sv} \leq 5 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = +\infty$
- $\text{VALUE} \geq \text{minv}$
- $\text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} - 2$

where:
- $\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})$
- $\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})$
- $\text{sv} = |\text{VARIABLES}|$
- $\text{rv} = \text{range}(\text{VARIABLES}.\text{var})$

Purpose:
The constraint $\text{MIN\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES})$ holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern **DIP\_ON\_INCREASING\_SEQUENCE** for which the feature value is **VALUE**.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **DIP\_ON\_INCREASING\_SEQUENCE** is the subsequence which matches the regular expression ‘<<><<<’.

Assume that the occurrence of the pattern **DIP\_ON\_INCREASING\_SEQUENCE** starts at position $i$ and ends at position $j$. The feature **MIN** computes the minimum of the values from index $i + 2$ to index $j$.

Example:

$$\begin{pmatrix}
1, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4),
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0
\end{pmatrix}$$

Figure 4.1182 provides an example where the **POS\_MIN\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE** constraint holds.
Figure 4.1182: Illustrating the POS_MIN_MIN_DIP_ON_INCREASING_SEQUENCE constraint of the 
Example slot

Typical
\[
|\text{VARIABLES}| > 5, \\
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MIN_DIP_ON_INCREASINGSEQUENCE constraint but use the decoration table 3.35.
Origin

Based on constraint MIN_MIN_GORGE.

Constraint

POS_MIN_MIN_GORGE(VALUE, VARIABLES, FOUND)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions

sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ minv
VALUE = +∞ ∨ VALUE ≤ maxv − 1
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
minv = minval(VARIABLES, var)
maxv = maxval(VARIABLES, var)
sv = |VARIABLES|
rv = range(VARIABLES, var)

Purpose

The constraint MIN_MIN_GORGE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern GORGE for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘(>|>(|=|>)*)<(<|<(|=|<)*<)’.

Assume that the occurrence of the pattern GORGE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

Example

Figure 4.1183 provides an example where the POS_MIN_MIN_GORGE (3, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 4.1183: Illustrating the POS_MIN_MIN_GORGE constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range(}\text{VARIABLES.var}) > 1 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MIN_GORGE constraint but use the decoration table 3.35.
Origin: Based on constraint \texttt{MIN\_MIN\_INCREASING}.

Constraint: \texttt{POS\_MIN\_MIN\_INCREASING}(VALUE, VARIABLES, FOUND)

Arguments:
- \texttt{VALUE} : dvar
- \texttt{VARIABLES} : collection(var\texttt{--dvar})
- \texttt{FOUND} : collection(var\texttt{--dvar})

Restrictions:
- \(sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty\)
- \(VALUE \geq \text{minv}\)
- \(VALUE = +\infty \lor VALUE \leq \text{maxv} - 1\)
- \texttt{required}(\texttt{VARIABLES}.var)
- \texttt{required}(\texttt{FOUND}.var)
- \(|\texttt{VARIABLES}| = |\texttt{FOUND}|\)

The position of the \texttt{FOUND} letter in an occurrence of the pattern \texttt{INCREASING} for which the feature value is \texttt{VALUE}.

Purpose: The constraint \texttt{MIN\_MIN\_INCREASING}(VALUE, VARIABLES) holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{FOUND} letter in those occurrences of the pattern \texttt{INCREASING} for which the feature value is \texttt{VALUE}.

Assume that the occurrence of the pattern \texttt{INCREASING} starts at position \(i\) and ends at position \(j\). The feature \texttt{MIN} computes the minimum of the values from index \(i\) to index \(j + 1\).

Example:

\[
\begin{array}{c}
1. \{4,3,5,5,2,1,1,3,3,4,6,6,3,1,3,3\}, \\
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0
\end{array}
\]

Figure 4.1184 provides an example where the \texttt{POS\_MIN\_MIN\_INCREASING} (1. \{4,3,5,5,2,1,1,3,3,4,6,6,3,1,3,3\}; 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0) constraint holds.
Figure 4.1184: Illustrating the POS_MIN_MIN_INCREASING constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MIN_INCREASING constraint but use the decoration table 3.35.
POS_MIN_MIN_INCREASING

2479
Origin: Based on constraint MINMIN_INCREASING_SEQUENCE.

Constraint: POS_MIN_MIN_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

Arguments:
- VALUE: dvar
- VARIABLES: collection(var−dvar)
- FOUND: collection(var−dvar)

Restrictions:
- \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \)
- \( VALUE \geq \text{minv} \)
- \( VALUE = +\infty \lor VALUE \leq \text{maxv} - 1 \)
- \text{required}(\text{VARIABLES}, \text{var})
- \text{required}(\text{FOUND}, \text{var})
- \|\text{VARIABLES}\| = |\text{FOUND}|

where:
- \text{minv} = \text{minval}(\text{VARIABLES}.\text{var})
- \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
- \text{sv} = |\text{VARIABLES}|
- \text{rv} = \text{range}(\text{VARIABLES}.\text{var})

The constraint MIN_MIN_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, \text{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern INCREASING_SEQUENCE for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\(< (< | =)\)∗ < | ‘. Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

Purpose

Example

\[
\begin{pmatrix}
1, \{4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3\},
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0
\end{pmatrix}
\]

Figure 4.1185 provides an example where the POS_MIN_MIN_INCREASING_SEQUENCE (1, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.
Figure 4.1185: Illustrating the **POS\_MIN\_MIN\_INCREASING\_SEQUENCE** constraint of the Example slot

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MIN_INCREMENTING_SEQUENCE constraint but use the decoration table 3.35.
POS_MIN_MIN_INCREASING_SEQUENCE 2483
Origin

Based on constraint MIN_MIN_INFLEXION.

Constraint

POS_MIN_MIN_INFLEXION(VALUE, VARIABLES, FOUND)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions

\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \]
\[ VALUE \geq minv \leq +\infty \lor VALUE \leq maxv \]
required(VARIABLES, var)
required(FOUND, var)

where

\[ minv = \text{minval}(\text{VARIABLES}.\text{var}) \]
\[ maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

Purpose

The constraint MIN_MIN_INFLEXION(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INFLEXION for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \'< (< | =)* > | > (> | =)* <'. Assume that the occurrence of the pattern INFLEXION starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

Example

\[
\begin{pmatrix}
1, 1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4 \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1186 provides an example where the POS_MIN_MIN_INFLEXION (1, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 4.1186: Illustrating the POS_MIN_MIN_INFLEXION constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES.var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MIN_INFLEXION constraint but use the decoration table 3.35.
The constraint \texttt{MIN\_MIN\_STRICTLY\_DECREASING\_SEQUENCE}(VALUE, VARIABLES, FOUND) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} for which the feature value is VALUE. The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} is the maximal subsequence which matches the regular expression ‘>’. Assume that the occurrence of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} starts at position \textit{i} and ends at position \textit{j}. The feature \texttt{MIN} computes the minimum of the values from index \textit{i} to index \textit{j} + 1.

Figure 4.1187 provides an example where the \texttt{POS\_MIN\_MIN\_STRICTLY\_DECREASING\_SEQUENCE} (1, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 4.1187: Illustrating the \texttt{POS\_MIN\_MIN\_STRICTLY\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot

**Typical**

\[
|\texttt{VARIABLES}| > 1 \\
\text{range}(\texttt{VARIABLES}.\texttt{var}) > 1
\]

**Arg. properties**

- Functional dependency: VALUE determined by \texttt{VARIABLES}.
- Functional dependency: FOUND determined by \texttt{VARIABLES}.
Automaton

Similar to the automaton of the $\text{MIN_MIN_STRICTLY_DECREASING_SEQUENCE}$ constraint but use the decoration table 3.35.
POS_MIN_MIN_STRCTLY_DECREASING_SEQUENCE 2491
The constraint \( \text{POS}_\text{MIN}_\text{MIN}_\text{MIN}_\text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE}(\text{VALUE}, \text{VARIABLES}, \text{FOUND}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern \text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE} is the maximal subsequence which matches the regular expression ‘\(<+\)’.

Assume that the occurrence of the pattern \text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE} starts at position \( i \) and ends at position \( j \). The feature \( \text{MIN} \) computes the minimum of the values from index \( i \) to index \( j + 1 \).

Example

\[
\begin{pmatrix}
1, 4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3 \\
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0
\end{pmatrix}
\]

Example 4.1188 provides an example where the \( \text{POS}_\text{MIN}_\text{MIN}_\text{MIN}_\text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE} \) constraint holds.
Figure 4.1188: Illustrating the `POS_MIN_MIN_STRICTLY_INCREASING_SEQUENCE` constraint of the `Example` slot

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the _MIN_MIN_STRICTLY_INCREASING_SEQUENCE_ constraint but use the decoration table 3.35.
POS_MIN_MIN STRICTLY_INCREASING_SEQUENCE

2495
Origin
Based on constraint MIN_MIN_VALLEY.

Constraint
POS_MIN_MIN_VALLEY(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ minv
VALUE = +∞ ∨ VALUE ≤ maxv − 1
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
minv = minval(VARIABLES, var)
maxv = maxval(VARIABLES, var)
sv = |VARIABLES|
rv = range(VARIABLES, var)

The constraint MIN_MIN_VALLEY(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern VALLEY for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression ' > (= | >)* (< | =)* < '.

Assume that the occurrence of the pattern VALLEY starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

Example

Figure 4.1189 provides an example where the POS_MIN_MIN_VALLEY
(2, [1, 3, 7, 4, 3, 6, 6, 5, 3, 2, 6, 5, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0])
constraint holds.
Figure 4.1189: Illustrating the POS_MIN_MIN_VALLEY constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 2 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton Similar to the automaton of the MIN_MIN_VALLEY constraint but use the decoration table 3.35.
POS_MIN_MIN_VALLEY

2499
The constraint \( \text{MIN\_MIN\_ZIGZAG}(\text{VALUE}, \text{VARIABLES}, \text{FOUND}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{ZIGZAG} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{ZIGZAG} is the \text{maximal} subsequence which matches the regular expression \((<>)^+ (< | <> ) | (<>)^+ (> | >> )^+ )^+\).

Assume that the occurrence of the pattern \text{ZIGZAG} starts at position \( i \) and ends at position \( j \). The feature \text{MIN} computes the minimum of the values from index \( i + 1 \) to index \( j \).

The constraint \( \text{MIN\_MIN\_ZIGZAG}(\text{VALUE}, \text{VARIABLES}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{ZIGZAG} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{ZIGZAG} is the \text{maximal} subsequence which matches the regular expression \((<>)^+ (< | <> ) | (<>)^+ (> | >> )^+ )^+\).

Assume that the occurrence of the pattern \text{ZIGZAG} starts at position \( i \) and ends at position \( j \). The feature \text{MIN} computes the minimum of the values from index \( i + 1 \) to index \( j \).

The constraint \( \text{MIN\_MIN\_ZIGZAG}(\text{VALUE}, \text{VARIABLES}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{ZIGZAG} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{ZIGZAG} is the \text{maximal} subsequence which matches the regular expression \((<>)^+ (< | <> ) | (<>)^+ (> | >> )^+ )^+\).

Assume that the occurrence of the pattern \text{ZIGZAG} starts at position \( i \) and ends at position \( j \). The feature \text{MIN} computes the minimum of the values from index \( i + 1 \) to index \( j \).
Figure 4.1190: Illustrating the POS_MIN_MIN_ZIGZAG constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 3 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
| **Automaton** | Similar to the automaton of the MIN_MIN_ZIGZAG constraint but use the decoration table 3.35. |
Origin

Based on constraint MIN_SURF_BUMP_ON_DECREASING_SEQUENCE.

Constraint

POS_MIN_SURF_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

Arguments

VALUE : dvar
VARIABLES : collection(var-dvar)
FOUND : collection(var-dvar)

Restrictions

sv ≤ 5 ∨ rv ≤ 2 ⇒ VALUE = +∞
VALUE ≥ 3 * minv + 3 ⇒
VALUE = +∞ ∨ VALUE ≤ 3 * maxv − 3
among(n1, VARIABLES[3, sv − 1], (maxv − 2, maxv − 1, maxv))
VALUE < +∞ ⇒ n1 ≥ VALUE − 3 − max(maxv − 3, 0)
among(n2, VARIABLES[3, sv − 1], (minv, minv + 1, minv + 2))
VALUE < +∞ ⇒ n2 ≥ min(minv + 3, 0) − 3 − VALUE
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|

where

minv = minval(VARIABLES.var)
rv = range(VARIABLES.var)
sv = |VARIABLES|
maxv = maxval(VARIABLES.var)

The constraint MIN_SURF_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern BUMP_ON_DECREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression ’>>><>’.

Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 2 to index j.

Example

( 11, ⟨7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3⟩ , 
  ⟨0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0⟩ )
Figure 4.1191 provides an example where the POS_MIN_SURF_BUMP_ON_DECREASING_SEQUENCE constraint holds.

Figure 4.1191: Illustrating the POS_MIN_SURF_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

Typical

\[|\text{VARIABLES}| > 5\]

\[\text{range(VARIABLES.var)} > 2\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
**Automaton**  
Similar to the automaton of the MIN_SURF_BUMP_ON_DECREASING_SEQUENCE constraint but use the decoration table 3.35.
Origin: Based on constraint MIN_SURF_DECREASING.

Constraint: POSSMIN_SURF_DECREASING(VALUE, VARIABLES, FOUND)

Arguments:
- VALUE: dvar
- VARIABLES: collection(var−dvar)
- FOUND: collection(var−dvar)

Restrictions:
- \(sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty\)
- \(VALUE \geq 2 \ast minv + 1\)
- \(VALUE = +\infty \lor VALUE \leq 2 \ast maxv - 1\)
- required(VARIABLES, var)
- required(FOUND, var)
- \(|VARIABLES| = |FOUND|\)

where:
- \(minv = minval(VARIABLES.var)\)
- \(maxv = maxval(VARIABLES.var)\)
- \(sv = |VARIABLES|\)
- \(rv = range(VARIABLES.var)\)

Purpose: The constraint MIN_SURF_DECREASING(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression '<'.

Assume that the occurrence of the pattern DECREASING starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j+1\).

Example:

\[
\begin{pmatrix}
4, 3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4 \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1192 provides an example where the POSSMIN_SURF_DECREASING \((4, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])\) constraint holds.
Figure 4.1192: Illustrating the POS_MIN_SURF_DECREASING constraint of the Example slot

Typical

\[ |\text{VARIABLES} | > 1 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_DECREASING constraint but use the decoration table 3.35.
**DESCRIPTION**

Based on constraint `MIN_SURF_DECREASING_SEQUENCE`.

**AUTOMATON**

<table>
<thead>
<tr>
<th>AGGREGATOR</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><code>POS_MIN_SURF_DECREASING_SEQUENCE</code></td>
</tr>
</tbody>
</table>

**VARIABLES**

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var−dvar)`
- `FOUND` : `collection(var−dvar)`

**Restrictions**

```plaintext
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty
rv = 2 \Rightarrow VALUE = 2 \times \text{minv} + 3
rv > 2 \Rightarrow VALUE = \min(2 \times \text{minv} + 1, sv \times (\text{minv} + 1) <)
rv = 2 \Rightarrow VALUE = +\infty \lor VALUE \leq 2 \times \text{maxv} - 1
rv > 3 \Rightarrow VALUE = +\infty \lor VALUE \leq \max(2 \times \text{maxv} - 1, sv \times (\text{maxv} - 1))
\text{among}(n1, \text{VARIABLES}[1], sv, (\text{maxv} - 1, \text{maxv}))
\text{VALUE} < +\infty \land (rv = 2 \land \text{maxv} = 1) \Rightarrow
n1 \geq \text{VALUE} - \max(0, 2 \times \text{maxv} - 3)
\text{VALUE} < +\infty \land rv > 2 \land \text{maxv} > 1 \Rightarrow n1 \geq \text{VALUE} - 1 - sv \times (\text{maxv} - 2)
\text{among}(n2, \text{VARIABLES}[1], (\text{minv}, \text{minv} + 1))
\text{VALUE} < +\infty \land (rv = 2 \land \text{minv} = -1) \Rightarrow
n2 \geq \min(0, 2 \times \text{minv} + 3) - \text{VALUE}
\text{VALUE} < +\infty \land rv > 2 \land \text{minv} < -1 \Rightarrow n2 \geq sv \times (\text{minv} + 2) - 1 - \text{VALUE}
\text{required}(\text{VARIABLES}, var)
\text{required}(\text{FOUND}, var)
| \text{VARIABLES} | = | \text{FOUND} |
```

where

- `minv =\text{minval}(\text{VARIABLES}, var)`
- `rv =\text{range}(\text{VARIABLES}, var)`
- `sv = |\text{VARIABLES}|`
- `maxv =\text{maxval}(\text{VARIABLES}, var)`

The constraint `MIN_SURF_DECREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `DECREASING_SEQUENCE` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `DECREASING_SEQUENCE` is the **maximal** subsequence which matches the regular expression `"> (> | =)* > | >'`.

Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`. 

**Purpose**

An occurrence of the pattern `DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `"> (> | =)* > | >'`.
Figure 4.1193 provides an example where the POS_MIN_SURF_DECREASING_SEQUENCE (6, [3, 4, 2, 5, 6, 4, 4, 3, 1, 4, 6, 4, 4], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Figure 4.1193: Illustrating the POS_MIN_SURF_DECREASING_SEQUENCE constraint of the Example slot

Typical

| VARIABLES | > 1
\[range(\text{VARIABLES}.\text{var}) > 1\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_DECREASING_SEQUENCE constraint but use the decoration table 3.35.
Based on constraint \texttt{MIN\_SURF\_DECREASING\_TERRACE}.

**Arguments**

\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var}-\text{dvar}) \\
\text{FOUND} & : \text{collection}(\text{var}-\text{dvar})
\end{align*}

**Restrictions**

\begin{align*}
\text{sv} & \leq 3 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \min(2 * (\text{minv} + 1), (\text{sv} - 2) * (\text{minv} + 1)) \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \max(2 * (\text{maxv} - 1), (\text{sv} - 2) * (\text{maxv} - 1)) \\
\text{among}(\text{n1, VARIABLES}[2, \text{sv} - 1], (\text{maxv} - 1)) & \\
\text{VALUE} & < +\infty \Rightarrow \text{n1} \geq \text{VALUE} - \max(0, (\text{sv} - 2) * (\text{maxv} - 2)) \\
\text{among}(\text{n2, VARIABLES}[2, \text{sv} - 1], (\text{minv} + 1)) & \\
\text{VALUE} & < +\infty \Rightarrow \text{n2} \geq \min(0, (\text{sv} - 2) * (\text{minv} + 2)) - \text{VALUE} \\
\text{required}(\text{VARIABLES}\_\text{var}) & \\
\text{required}(\text{FOUND}\_\text{var}) & \\
|\text{VARIABLES}| & = |\text{FOUND}|
\end{align*}

where

\begin{align*}
\text{minv} & = \text{minval}(\text{VARIABLES}\_\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}\_\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}\_\text{var})
\end{align*}

The constraint \texttt{MIN\_SURF\_DECREASING\_TERRACE}(\text{VALUE}, \text{VARIABLES}) holds. In addition, \text{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \textit{found} letter in those occurrences of the pattern \texttt{DECREASING\_TERRACE} for which the feature value is \text{VALUE}.

The position of the \textit{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{DECREASING\_TERRACE} is the \textit{maximal} subsequence which matches the regular expression \texttt{“>+=”}.

Assume that the occurrence of the pattern \texttt{DECREASING\_TERRACE} starts at position \textit{i} and ends at position \textit{j}. The feature \texttt{SURF} computes the sum of the values from index \textit{i} + 1 to index \textit{j}.

**Example**

\[
\left(4, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3), (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)\right)
\]
Figure 4.1194 provides an example where the `POS_MIN_SURF_DECREASING_TERRACE` constraint holds.

![Graph showing feature values and variables]

Figure 4.1194: Illustrating the `POS_MIN_SURF_DECREASING_TERRACE` constraint of the Example slot

**Typical**

- $|\text{VARIABLES}| > 3$
- $\text{range}(\text{VARIABLES}.\text{var}) > 2$

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_DECREASING_TERRACE constraint but use the decoration table 3.35.
The constraint `MIN_SURF_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `DIP_ON_INCREASING_SEQUENCE` for which the feature value is `VALUE`. The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` is the subsequence which matches the regular expression `<<<`.

Assume that the occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j`.

Example:

```
( 9, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4),
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0) )
```
Figure 4.1195 provides an example where the POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE constraint holds.

Figure 4.1195: Illustrating the POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 5 \]
\[ \text{range(VARIABLES.var)} > 2 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURT_DIP_ON_INCREASING_SEQUENCE constraint but use the decoration table 3.35.
POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE
The constraint `MIN_SURF_GORGE(VALUE, VARIABLES, FOUND)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `GORGE` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `GORGE` is the maximal subsequence which matches the regular expression `\((> | > (= | >)) > (< | < (= | <)) \)`. Assume that the occurrence of the pattern `GORGE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 1` to index `j`. 

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `GORGE` is the maximal subsequence which matches the regular expression `\((> | > (= | >)) > (< | < (= | <)) \)`. Assume that the occurrence of the pattern `GORGE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 1` to index `j`. 

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
Example

\[5, (1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7), (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)\]

Figure 4.1196 provides an example where the POS_MIN_SURF_GORGE constraint holds.

![Diagram of feature values](image)

Figure 4.1196: Illustrating the POS_MIN_SURF_GORGE constraint of the Example slot

Typical

- \[|\text{VARIABLES}| > 2\]
- \[\text{range} (\text{VARIABLES}.\text{var}) > 1\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_GORGE constraint but use the decoration table 3.35.
POS_MIN_SURF_GORGE

2527
Origin
Based on constraint MIN_SURF_INCREASING.

Constraint
POS_MIN_SURF_INCREASING(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var—dvar)
FOUND : collection(var—dvar)

Restrictions
$sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty$
VALUE $\geq 2 \times \minv + 1$
VALUE $= +\infty \lor VALUE \leq 2 \times \maxv - 1$
required(VARIABLES, var)
required(FOUND, var)
$|VARIABLES| = |FOUND|

where
$\minv = \minval(VARIABLES.var)$
$\maxv = \maxval(VARIABLES.var)$
$sv = |VARIABLES|
$rv = \range(VARIABLES.var)$

Purpose
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
An occurrence of the pattern INCREASING is the subsequence which matches the regular expression ‘<’.
Assume that the occurrence of the pattern INCREASING starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

Example

Figure 4.1197 provides an example where the POS_MIN_SURF_INCREASING
$\langle 4, [4, 3, 5, 2, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0] \rangle$
constraint holds.
Figure 4.1197: Illustrating the POS_MIN_SURF_INCREASING constraint of the Example slot

**Typical**

\[
\text{|VARIABLES|} > 1 \\
\text{range(VARIABLES.var)} > 1
\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_INCREASING constraint but use the decoration table 3.35.
Based on constraint MIN_SURF_INCREASING_SEQUENCE.

**Arguments**

- VALUE : dvar
- VARIABLES : collection(var−dvar)
- FOUND : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv &\leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
rv &= 2 \Rightarrow \text{VALUE} \geq 2 \cdot \text{minv} + 1 \\
rv &\geq 3 \Rightarrow \text{VALUE} \geq \min(2 \cdot \text{minv} + 1, sv \cdot (\text{minv} + 1) \cdot \text{x}) \\
rv &= 2 \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq 2 \cdot \text{maxv} - 1 \\
rv &\geq 3 \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq \max(2 \cdot \text{maxv} - 1, sv \cdot (\text{maxv} - 1)) \\
\text{among}(n1, \text{VARIABLES}[1, sv], (\text{maxv} - 1, \text{maxv})) \\
\text{VALUE} &< +\infty \land (rv = 2 \lor \text{maxv} = 1) \Rightarrow \\
n1 &\geq \text{VALUE} \land \text{max}(0, 2 \cdot \text{maxv} - 3) \\
\text{VALUE} &< +\infty \land rv > 2 \land \text{maxv} > 1 \Rightarrow n1 \geq \text{VALUE} - 1 - sv \cdot (\text{maxv} - 2) \\
\text{among}(n2, \text{VARIABLES}[1, sv], (\text{minv}, \text{minv} + 1)) \\
\text{VALUE} &< +\infty \land (rv = 2 \lor \text{minv} = -1) \Rightarrow \\
n2 &\geq \min(0, 2 \cdot \text{minv} + 3) - \text{VALUE} \\
\text{VALUE} &< +\infty \land rv > 2 \land \text{minv} < -1 \Rightarrow n2 \geq sv \cdot (\text{minv} + 2) - 1 - \text{VALUE} \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{required}(\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| = |\text{FOUND}|
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint MIN_SURF_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘<( | | )∗ < | < ’.

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.
Figure 4.1198 provides an example where the POS_MIN_SURF_INCREASING_SEQUENCE constraint holds.

Figure 4.1198: Illustrating the POS_MIN_SURF_INCREASING_SEQUENCE constraint of the Example slot

Typical

$|\text{VARIABLES}| > 1$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MIN_SURF_INCREASING_SEQUENCE` constraint but use the decoration table 3.35.
POS_MIN_SURF_INCREASING_SEQUENCE
The constraint $\text{MIN\_SURF\_INCREASING\_TERRACE}(\text{VALUE}, \text{VARIABLES})$ holds. In addition, $\text{FOUND}$ is a collection of 0/1 variables where the value 1 indicates the position of the $\text{found}$ letter in those occurrences of the pattern $\text{INCREASING\_TERRACE}$ for which the feature value is $\text{VALUE}$.

The position of the $\text{found}$ letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern $\text{INCREASING\_TERRACE}$ is the maximal subsequence which matches the regular expression ‘$< = + <$’.

Assume that the occurrence of the pattern $\text{INCREASING\_TERRACE}$ starts at position $i$ and ends at position $j$. The feature $\text{SURF}$ computes the sum of the values from index $i+1$ to index $j$.

**Example**

\[
\left(\begin{array}{c}
9, (1, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0
\end{array}\right)
\]
Figure 4.1199 provides an example where the POS_MIN_SURF_INCREASING_TERRACE constraint holds.

![Diagram](image)

Figure 4.1199: Illustrating the POS_MIN_SURF_INCREASING_TERRACE constraint of the Example slot

**Typical**

- $|\text{VARIABLES}| > 3$
- $\text{range(}\text{VARIABLES.var}) > 2$

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN\_SURF\_INCREASING\_TERRACE} constraint but use the decoration table \ref{tab:3.35}. 

Based on constraint **MIN_SURF_INFLEXION**.

**Constraint**

\[ \text{POS\_MIN\_SURF\_INFLEXION} \left( \text{VALUE, VARIABLES, FOUND} \right) \]

**Arguments**

- **VALUE**: \( \text{dvar} \)
- **VARIABLES**: \( \text{collection(var-dvar)} \)
- **FOUND**: \( \text{collection(var-dvar)} \)

**Restrictions**

1. \( sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \)
2. \( \text{VALUE} \geq \min(\text{minv}, (sv - 2) \ast \text{minv}) \)
3. \( \text{VALUE} = +\infty \lor \text{VALUE} \leq \max(\text{maxv}, (sv - 2) \ast \text{maxv}) \)
4. \( \text{among}(n_1, \text{VARIABLES}[2, sv - 1], \text{maxv}) \)
5. \( \text{VALUE} < +\infty \Rightarrow n_1 \geq \text{VALUE} - \max(0, (sv - 2) \ast (\text{maxv} - 1)) \)
6. \( \text{VALUE} < +\infty \Rightarrow n_2 \geq \min(0, (sv - 2) \ast (\text{minv} + 1)) - \text{VALUE} \)
7. \( \text{among}(n_2, \text{VARIABLES}[2, sv - 1], \text{minv}) \)
8. \text{required}(\text{VARIABLES}, \text{var})
9. \text{required}(\text{FOUND}, \text{var})

**Purpose**

The constraint \( \text{MIN\_SURF\_INFLEXION} \left( \text{VALUE, VARIABLES} \right) \) holds. In addition, \text{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{INFLEXION} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{INFLEXION} is the maximal subsequence which matches the regular expression ‘\( < (< | =) > | > (| =) > < \)’.

Assume that the occurrence of the pattern \text{INFLEXION} starts at position \( i \) and ends at position \( j \). The feature \text{SURF} computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**

\[ \left( 1, \langle 1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4 \rangle, \langle 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0 \rangle \right) \]
Figure 4.1200 provides an example where the POS_MIN_SURF_INFLEXION constraint holds.

\[
\begin{array}{cccccccccccccc}
1 & 1 & 2 & 6 & 6 & 4 & 4 & 3 & 5 & 2 & 5 & 1 & 5 & 3 & 3 & 4 & 4 & 4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

Figure 4.1200: Illustrating the POS_MIN_SURF_INFLEXION constraint of the **Example** slot

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_INFLEXION constraint but use the decoration table 3.35.
POS_MIN_SURF_INFLEXION

2543
Origin
Based on constraint MIN_SURF_PEAK.

Constraint
\[
\text{POS_MIN_SURF_PEAK}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

Arguments
\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var−dvar)} \\
\text{FOUND} & : \text{collection(var−dvar)}
\end{align*}
\]

Restrictions
\[
\begin{align*}
\text{sv} \leq 2 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \min(\text{minv} + 1, (\text{sv} - 2) \ast (\text{minv} + 1)) \\
\text{VALUE} & \geq +\infty \lor \text{VALUE} \leq \max(\text{maxv}, (\text{sv} - 2) \ast \text{maxv}) \lor \\
\text{among} & ([n1, \text{VARIABLES}[2, \text{sv} - 1], \text{maxv}]) \\
\text{VALUE} & < +\infty \Rightarrow n1 \geq \text{VALUE} - \max(0, (\text{sv} - 2) \ast (\text{maxv} - 1)) \\
\text{among} & ([n2, \text{VARIABLES}[2, \text{sv} - 1], \text{minv} + 1]) \\
\text{VALUE} & < +\infty \Rightarrow n2 \geq \min(0, (\text{sv} - 2) \ast (\text{minv} + 2)) - \text{VALUE} \\
\text{required} & (\text{VARIABLES}, \text{var}) \\
\text{required} & (\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| & = |\text{FOUND}|
\end{align*}
\]

where
\[
\begin{align*}
\text{minv} & = \minval(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{maxv} & = \maxval(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Purpose
The constraint \(\text{MIN\_SURF\_PEAK}(\text{VALUE}, \text{VARIABLES})\) holds. In addition, \(\text{FOUND}\) is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern \(\text{PEAK}\) for which the feature value is \(\text{VALUE}\).

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \(\text{PEAK}\) is the maximal subsequence which matches the regular expression \(<(= | <) (>| =)) >\).

Assume that the occurrence of the pattern \(\text{PEAK}\) starts at position \(i\) and ends at position \(j\). The feature \text{SURF} computes the sum of the values from index \(i + 1\) to index \(j\).

Example
\[
\left( \begin{array}{c} 9, (7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1), \\
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0 \end{array} \right)
\]
Figure 4.1201 provides an example where the POS\_MIN\_SURF\_PEAK constraint holds.

Figure 4.1201: Illustrating the POS\_MIN\_SURF\_PEAK constraint of the Example slot

**Typical**

\[|\text{VARIABLES}| > 2\]
\[\text{range(\text{VARIABLES}.\text{var})} > 1\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_PEAK constraint but use the decoration table 3.35.
AGGREGATOR FEATURE PATTERN

POS_MIN_SURF/plain

▶ ▷ ◄ ◄ DESCRIPTION AUTOMATON

Origin Based on constraint MIN_SURF/plain.

Constraint

POS_MIN_SURF/plain(VALUE, VARIABLES, FOUND)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions

\( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \)
\( VALUE \geq \min(\minv, (sv - 2) + \minv) \)
\( VALUE = +\infty \lor VALUE \leq \max(maxv - 1, (sv - 2) + (maxv - 1)) \)
\( \text{among}(n1, \text{VARIABLES}[2, sv - 1], [maxv - 1]) \)
\( VALUE < +\infty \Rightarrow n1 \geq VALUE - \max(0, (sv - 2) + (maxv - 2)) \)
\( \text{among}(n2, \text{VARIABLES}[2, sv - 1], [\minv]) \)
\( VALUE < +\infty \Rightarrow n2 \geq \min(0, (sv - 2) + (\minv + 1)) - VALUE \setminus \text{required}(\text{VARIABLES}, \text{var}) \)
\( \text{required}(\text{FOUND}, \text{var}) \)
\( |\text{VARIABLES}| = |\text{FOUND}| \)

where

\( \minv = \text{minval}((\text{VARIABLES}).\text{var}) \)
\( sv = |\text{VARIABLES}| \)
\( maxv = \text{maxval}((\text{VARIABLES}).\text{var}) \)
\( rv = \text{range}((\text{VARIABLES}).\text{var}) \)

Purpose

The constraint MIN_SURF/plain(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern plain for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern plain is the maximal subsequence which matches the regular expression ‘\(\geq\)’.

Assume that the occurrence of the pattern plain starts at position \(i\) and ends at position \(j\). The feature surf computes the sum of the values from index \(i + 1\) to index \(j\).

Example

\( (4, (2, 3, 6, 5, 7, 6, 6, 4, 5, 4, 3, 3, 6, 6, 3), (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)) \)
Figure 4.1202 provides an example where the POS_MIN_SURFPLAIN (4, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3], [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Figure 4.1202: Illustrating the POS_MIN_SURFPLAIN constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range(}\text{VARIABLES.}\text{var}) > 1 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_PLAIN constraint but use the decoration table 3.35.
Origin
Based on constraint MIN_SURF_PLATEAU.

Constraint
POS_MIN_SURF_PLATEAU(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ min(minv + 10, (sv − 2) * (minv + 1))
VALUE = +∞ ∨ VALUE ≤ max(maxv, (sv − 2) * maxv)
among(n1, VARIABLES[2, sv − 1], {maxv})
VALUE < +∞ ⇒ n1 ≥ VALUE − max(0, (sv − 2) * (maxv − 1))
among(n2, VARIABLES[2, sv − 1], {minv + 1})
VALUE < +∞ ⇒ n2 ≥ min(0, (sv − 2) * (minv + 2)) − VALUE
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|

where
minv = minval(VARIABLES.var)
sv = |VARIABLES|
maxv = maxval(VARIABLES.var)
rv = range(VARIABLES.var)

The constraint MIN_SURF_PLATEAU(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PLATEAU for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression '<="'>.

Assume that the occurrence of the pattern PLATEAU starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
(3, (7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5),
 (0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))
Figure 4.1203 provides an example where the POS_MIN_SURF_PLATEAU constraint holds.

Figure 4.1203: Illustrating the POS_MIN_SURF_PLATEAU constraint of the **Example** slot

**Typical**
- |VARIABLES| > 2
- range(VARIABLES.var) > 1

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_PLATEAU constraint but use the decoration table 3.35.
Origin
Based on constraint $MIN\_SURF\_PROPER\_PLAIN$.

Constraint
$POS\_MIN\_SURF\_PROPER\_PLAIN(VALUE, VARIABLES, FOUND)$

Arguments
- $VALUE$ : $dvar$
- $VARIABLES$ : $collection(var-dvar)$
- $FOUND$ : $collection(var-dvar)$

Restrictions
- $sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = +\infty$
- $VALUE \geq \text{min}(2 \ast \text{minv}, (sv - 2) \ast \text{minv}^2)$
- $VALUE = +\infty \lor VALUE \leq \text{max}(2 \ast (\text{maxv} - 1), (sv - 2) \ast (\text{maxv} - 1))$
- $\text{among}(n_1, VARIABLES[2, sv - 1], (\text{maxv} - 1))$
- $VALUE < +\infty \Rightarrow n_1 \geq VALUE - \text{max}(0, (sv - 2) \ast (\text{maxv} - 2))$
- $\text{among}(n_2, VARIABLES[2, sv - 1], (\text{minv}))$
- $VALUE < +\infty \Rightarrow n_2 \geq \text{min}(0, (sv - 2) \ast (\text{minv} + 1)) - VALUE$
- $\text{required}(VARIABLES, var)$
- $\text{required}(FOUND, var)$
- $|\text{VARIABLES}| = |\text{FOUND}|$

where
- $\text{minv} = \text{minval}(\text{VARIABLES}.var)$
- $sv = |\text{VARIABLES}|$
- $\text{maxv} = \text{maxval}(\text{VARIABLES}.var)$
- $rv = \text{range}(\text{VARIABLES}.var)$

The constraint $MIN\_SURF\_PROPER\_PLAIN(VALUE, VARIABLES)$ holds. In addition, $FOUND$ is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern $PROPER\_PLAIN$ for which the feature value is $VALUE$.

Purpose
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern $PROPER\_PLAIN$ is the maximal subsequence which matches the regular expression ‘$\geq + \prec$’.

Assume that the occurrence of the pattern $PROPER\_PLAIN$ starts at position $i$ and ends at position $j$. The feature $SURF$ computes the sum of the values from index $i + 1$ to index $j$.

Example
$\left( 8, (2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5) , \right)$
$\left( 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \right)$
Figure 4.1204 provides an example where the POS_MIN_SURF_PROPER.PLAIN constraint holds.

Figure 4.1204: Illustrating the POS_MIN_SURF_PROPER.PLAIN constraint of the Example slot

**Typical**

\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_PROPER_PLAIN constraint but use the decoration table 3.35.
Origin
Based on constraint MIN_SURF_PROPER_PLATEAU.

Constraint
POS_MIN_SURF_PROPER_PLATEAU(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ min(2 + (minv + 1)2, (sv − 2) * (minv + 1)2)
VALUE = +∞ ∨ VALUE ≤ max(2 * maxv, (sv − 2) * maxv)
VALUE < +∞ ⇒ n1 ≥ VALUE = max(0, (sv − 2) * (maxv − 1))
VALUE < +∞ ⇒ n2 ≥ min(0, (sv − 2) * (minv + 2)) − VALUE
among(n1, VARIABLES[2, sv − 1], (maxv))
among(n2, VARIABLES[2, sv − 1], (minv + 1))
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|

where
minv = minval(VARIABLES.var)
sv = |VARIABLES|
maxv = maxval(VARIABLES.var)
rv = range(VARIABLES.var)

The constraint MIN_SURF_PROPER_PLATEAU(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PROPER_PLATEAU for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression ‘< =+ >’.

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
\[
(6, (7, 1, 3, 2, 5, 1, 4, 3, 2, 3, 5, 5, 3), (0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))
\]
Figure 4.1205 provides an example where the \texttt{POS\_MIN\_SURF\_PROPER\_PLATEAU}\,(6, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3, 1], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

![Diagram](image)

Figure 4.1205: Illustrating the \texttt{POS\_MIN\_SURF\_PROPER\_PLATEAU} constraint of the \texttt{Example} slot

**Typical**

\[
|\text{VARIABLES}| > 3 \\
\text{range(\text{VARIABLES}.\text{var})} > 1
\]

**Arg. properties**

- Functional dependency: \texttt{VALUE} determined by \texttt{VARIABLES}.
- Functional dependency: \texttt{FOUND} determined by \texttt{VARIABLES}.
Automaton

Similar to the automaton of the `MIN_SURF_PROPER_PLATEAU` constraint but use the decoration table \[3.35\].
Based on constraint MIN_SURF_STEADY.

### Constraint

\[ \text{POS\_MIN\_SURF\_STEADY} (\text{VALUE}, \text{VARIABLES}, \text{FOUND}) \]

### Arguments

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var\,−\,dvar)`
- **FOUND**: `collection(var\,−\,dvar)`

### Restrictions

\[
\begin{align*}
\text{sv} \leq 1 & \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq 2 \ast \text{minv} & \\
\text{VALUE} = +\infty \lor \text{VALUE} \leq 2 \ast \text{maxv} & \\
\text{among}([\text{n1}, \text{VARIABLES}[1, \text{sv}]], (\text{maxv})) & \\
\text{VALUE} < +\infty & \Rightarrow \text{n1} \geq \text{VALUE} - 2 \ast (\text{maxv} - 1) \\
\text{among}([\text{n2}, \text{VARIABLES}[1, \text{sv}]], (\text{minv})) & \\
\text{VALUE} < +\infty & \Rightarrow \text{n2} \geq 2 \ast (\text{minv} + 1) - \text{VALUE} \\
\text{required} & (\text{VARIABLES}, \text{var}) \\
\text{required} & (\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| = |\text{FOUND}| & \\
\text{where} & \\
\text{minv} = \text{minval} & (\text{VARIABLES}.\text{var}) \\
\text{maxv} = \text{maxval} & (\text{VARIABLES}.\text{var}) \\
\text{sv} = |\text{VARIABLES}| &
\end{align*}
\]

The constraint `MIN_SURF_STEADY(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `STEADY` for which the feature value is `VALUE`. The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. Assume that the occurrence of the pattern `STEADY` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.

### Example

\[
\begin{align*}
2, (1, 1, 7, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6) & \\
(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) &
\end{align*}
\]

Figure 4.1206 provides an example where the `POS_MIN_SURF_STEADY (2, [1, 1, 7, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6], [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])` constraint holds.
Figure 4.1206: Illustrating the POS\_MIN\_SURF\_STEADY constraint of the Example slot

**Typical**

\[|\text{VARIABLES}| > 1\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
| Automaton | Similar to the automaton of the MIN_SURF_STEADY constraint but use the decoration table 3.35. |
The constraint MIN_SURF_STEADY_SEQUENCE(VALUE, VARIABLES, FOUND) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STEADY_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression '+='. Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$. 

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression '+='. Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$. 

### Origin
Based on constraint MIN_SURF_STEADY_SEQUENCE.

### Constraint
$$\text{POS\_MIN\_SURF\_STEADY\_SEQUENCE(VALUE, VARIABLES, FOUND)}$$

### Arguments
- **VALUE**: $d\text{var}$
- **VARIABLES**: $\text{collection(var-dvar)}$
- **FOUND**: $\text{collection(var-dvar)}$

### Restrictions
- $sv \leq 1 \Rightarrow VALUE = +\infty$
- $rv = 1 \Rightarrow VALUE \geq sv \cdot \text{minv}$
- $rv \geq 2 \Rightarrow VALUE \geq \min(2 \cdot \text{minv}, sv \cdot
- $rv = 1 \Rightarrow VALUE = +\infty \lor VALUE \leq sv \cdot \text{maxv}$
- $rv \geq 2 \Rightarrow VALUE = +\infty \lor VALUE \leq \max(2 \cdot \text{maxv}, sv \cdot \text{maxv})$
- $\text{among}(n_1, \text{VARIABLES}[1, sv], (\text{maxv}))$
- $\text{VALUE} < +\infty \Rightarrow n_1 \geq \text{VALUE} - \max(0, sv \cdot (\text{maxv} - 1))$
- $\text{among}(n_2, \text{VARIABLES}[2, sv], (\text{minv}))$
- $\text{VALUE} < +\infty \Rightarrow n_2 \geq \min(0, sv \cdot (\text{minv} + 1)) - \text{VALUE}$
- $\text{required}($VARIABLES, var$)$
- $\text{required}(\text{FOUND}, \text{var})$
- $|\text{VARIABLES}| = |\text{FOUND}|$

where
- $\text{minv} = \text{minval}($VARIABLES, var$)$
- $\text{rv} = \text{range}($VARIABLES, var$)$
- $sv = |\text{VARIABLES}|$
- $\text{maxv} = \text{maxval}($VARIABLES, var$)$

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression '+='. Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$. 

### Purpose
**Example**

\[
\begin{pmatrix}
2, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1) \\
(0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)
\end{pmatrix}
\]

Figure 4.1207 provides an example where the \textit{POS\_\texttt{MIN\_SURF\_STEADY\_SEQUENCE}} constraint holds.

Figure 4.1207: Illustrating the \textit{POS\_\texttt{MIN\_SURF\_STEADY\_SEQUENCE}} constraint of the Example slot

**Typical**

\[|\text{VARIABLES}| > 1\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton Similar to the automaton of the \texttt{MIN\_SURF\_STEADY\_SEQUENCE} constraint but use the decoration table 3.35.
POS_MIN_SURF_STEADY_SEQUENCE
\textbf{POS\_MIN\_SURF\_STRICTLY\_DECREASING\_SEQUENCE}\\
\textbf{DESCRIPTION}\\
\textbf{AUTOMATON}\\
\begin{tabular}{|c|c|}
\hline
\hline
\textbf{Origin} & Based on constraint \textbf{MIN\_SURF\_STRICTLY\_DECREASING\_SEQUENCE}. \\
\hline
\textbf{Constraint} & \textbf{POS\_MIN\_SURF\_STRICTLY\_DECREASING\_SEQUENCE(VALUE, VARIABLES, FOUND)} \\
\hline
\textbf{Arguments} & \begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var}^{-}\text{dvar}) \\
\text{FOUND} & : \text{collection(var}^{-}\text{dvar}) \\
\end{align*} \\
\hline
\textbf{Restrictions} & \begin{align*}
\text{sv} \leq 1 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{minv} < 0 & \Rightarrow \text{VALUE} \geq \ell_1 \ast \text{minv} + \lfloor \ell_1 \ast (\ell_1 - 1)/2 \rfloor \\
\text{minv} \geq 0 & \Rightarrow \text{VALUE} \geq 2 \ast \text{minv} + 1 \\
\text{maxv} > 0 & \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq \ell_2 \ast \text{maxv} - \lfloor \ell_2 \ast (\ell_2 - 1)/2 \rfloor \\
\text{maxv} \leq 0 & \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq 2 \ast \text{maxv} - 1 \\
\text{among}(n1, \text{VARIABLES}[1, \text{sv}], (\text{maxv} - 1, \text{maxv})) & \Rightarrow \text{VALUE} < +\infty \Rightarrow n1 \geq \text{VALUE} - \max(0, \min(\text{sv}, \text{rv}) \ast (\text{maxv} - 2)) - 1 \\
\text{among}(n2, \text{VARIABLES}[1, \text{sv}], (\text{minv}, \text{minv} + 1)) & \Rightarrow \text{VALUE} < +\infty \Rightarrow n2 \geq \min(0, \min(\text{sv}, \text{rv}) \ast (\text{minv} + 2)) - 1 - \text{VALUE} \\
\text{required}(\text{VARIABLES}, \text{var}) & \Rightarrow \text{required}(\text{FOUND}, \text{var}) \\
|\text{VARIABLES}| & = |\text{FOUND}| \\
\text{where} & \begin{align*}
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\ell_1 & = \min(\min(\text{sv}, \text{rv}), |\text{minv}|) \\
\ell_2 & = \min(\min(\text{sv}, \text{rv}), |\text{maxv}|) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\end{align*} \\
\hline
\end{tabular}\\
\begin{tabular}{|c|c|}
\hline
\hline
\text{Purpose} & The constraint \textbf{MIN\_SURF\_STRICTLY\_DECREASING\_SEQUENCE(VALUE, VARIABLES)} holds. In addition, \text{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \textbf{STRICTLY\_DECREASING\_SEQUENCE} for which the feature value is \text{VALUE}. \\
\text{The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.} \\
\text{An occurrence of the pattern \textbf{STRICTLY\_DECREASING\_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression ‘>’}. \\
\text{Assume that the occurrence of the pattern \textbf{STRICTLY\_DECREASING\_SEQUENCE} starts at position \textit{i} and ends at position \textit{j}. The feature \text{SURF} computes the sum of the values from index \textit{i} to index \textit{j} + 1.} \\
\end{tabular}
Figure 4.1208 provides an example where the \( \text{POS\_MIN\_SURF\_STRICTLY\_DECREASING\_SEQUENCE} \) constraint holds.

\[
\begin{align*}
7, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3), \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)
\end{align*}
\]

Figure 4.1208: Illustrating the \( \text{POS\_MIN\_SURF\_STRICTLY\_DECREASING\_SEQUENCE} \) constraint of the \textbf{Example} slot

Typical

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES.var}) > 1
\]

Arg. properties

- Functional dependency: VALUE determined by \text{VARIABLES}.
- Functional dependency: FOUND determined by \text{VARIABLES}. 

Example
Automaton

Similar to the automaton of the MIN_SURF_STRICTLY_DECREASING_SEQUENCE constraint but use the decoration table 3.35.
**Description**

- **Origin**: Based on constraint `MIN_SURF_STRICTLY_INCREASING_SEQUENCE`.
- **Constraint**: `MIN_SURF_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)`
- **Arguments**:
  - `VALUE`: `dvar`
  - `VARIABLES`: `collection(var−dvar)`
  - `FOUND`: `collection(var−dvar)`

**Restrictions**

- `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = +∞`
- `minv < 0 ⇒ VALUE ≥ ℓ1 * minv + ⌊ℓ1 * (ℓ1 − 1)/2⌋ ≥−∞`
- `minv ≥ 0 ⇒ VALUE ≥ 2 * minv + 1 ≥−∞`
- `maxv > 0 ⇒ VALUE = +∞ ∨ VALUE ≤ ℓ2 * maxv − ⌊ℓ2 * (ℓ2 − 1)/2⌋ ≤−∞`
- `maxv ≤ 0 ⇒ VALUE = +∞ ∨ VALUE ≤ 2 * maxv − 1 ≤−∞`
- `among(n1, VARIABLES[1, sv], (maxv − 1, maxv))`
- `VALUE < +∞ ⇒ n1 ≥ VALUE − max(0, min(sv, rv) * (maxv − 2)) − 1 ≤−∞`
- `among(n2, VARIABLES[1, sv], (minv, minv + 1))`
- `VALUE < +∞ ⇒ n2 ≥ min(0, min(sv, rv) * (minv + 2) − 1 − VALUE required(VARIABLES, var) ≤−∞`
- `required(FOUND, var)`
- `|VARIABLES| = |FOUND|`

**Purpose**

- The constraint `MIN_SURF_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `STRICTLY_INCREASING_SEQUENCE` for which the feature value is `VALUE`. The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
- An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the `maximal` subsequence which matches the regular expression `'<>'`.
- Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.

<table>
<thead>
<tr>
<th>VALUE</th>
<th>minv</th>
<th>maxv</th>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 0</td>
<td>≤ 0</td>
<td>&gt; 0</td>
<td>minv, maxv</td>
</tr>
</tbody>
</table>

- The automaton shows the transitions for `VALUE` and `surff` values, indicating the occurrence of the pattern in different parts of the sequence.
Example (6, (4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3),
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0))

Figure 4.1209 provides an example where the POS_MIN_SURF_STRICTLY_INCREASING_SEQUENCE constraint holds.

Figure 4.1209: Illustrating the POS_MIN_SURF_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot

Typical

| VARIABLES | > 1 
| range(VARIABLES.var) | > 1 

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN\_SURF\_STRICTLY\_INCREASING\_SEQUENCE} constraint but use the decoration table \texttt{3.35}. 
POS_MIN_SURF STRICTLY_INCREASING_SEQUENCE
DESCRIPTION AUTOMATON

Origin
Based on constraint MIN_SURF_SUMMIT.

Constraint
POS_MIN_SURF_SUMMIT(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = +∞
rv = 2 ⇒ VALUE ≥ minv + 1
rv ≥ 3 ⇒ VALUE ≥ min(minv + 1, (sv − 2) ∗ (minv + 1) + 1)
rv = 2 ⇒ VALUE = +∞ ∨ VALUE ≤ maxv
rv ≥ 3 ⇒ VALUE = +∞ ∨ VALUE ≤ max(maxv, (sv − 2) ∗ (maxv − 1) + 1)
among(n1, VARIABLES[2, sv − 1], (maxv − 1, maxv))
VALUE < +∞ ∧ (rv = 2 ∧ maxv = 1) ⇒ n1 ≥ VALUE − max(0, maxv − 1)
VALUE < +∞ ∧ rv > 2 ∧ maxv > 1 ⇒
n1 ≥ VALUE − (sv − 2) ∗ (maxv − 2) − 1
among(n2, VARIABLES[2, sv − 1], (minv + 1))
VALUE < +∞ ∧ (rv = 2 ∧ minv = −1) ⇒ n2 ≥ max(0, minv + 2) − VALUE
VALUE < +∞ ∧ rv > 2 ∧ minv < −1 ⇒ n2 ≥ (sv − 2) ∗ (minv + 2) − VALUE
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|

where
minv = minval(VARIABLES.var)
rv = range(VARIABLES.var)
sv = |VARIABLES|
maxv = maxval(VARIABLES.var)

The constraint MIN_SURF_SUMMIT(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern SUMMIT for which the feature value is VALUE.
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

Purpose
An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression ‘(< | < (|= | <)* (< | > (>| =)* >)’.
Assume that the occurrence of the pattern SUMMIT starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.
Example

\[
\begin{pmatrix}
3, (7, 1, 5, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1),
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)
\end{pmatrix}
\]

Figure 4.1210 provides an example where the \texttt{POS_MIN_SURF_SUMMIT} constraint holds.

\[
\begin{array}{cccccccccccccccc}
\end{array}
\]

Figure 4.1210: Illustrating the \texttt{POS_MIN_SURF_SUMMIT} constraint of the Example slot

Typical
\[
|\texttt{VARIABLES}| > 2
\]
\[
\text{range(\texttt{VARIABLES}.\texttt{var})} > 1
\]

Arg. properties
- Functional dependency: \texttt{VALUE} determined by \texttt{VARIABLES}.
- Functional dependency: \texttt{FOUND} determined by \texttt{VARIABLES}. 
Automaton

Similar to the automaton of the MIN_SURF_SUMMIT constraint but use the decoration table 3.35.
Origin

Based on constraint \textit{MIN\_SURF\_VALLEY}.

Constraint

\texttt{POS\_MIN\_SURF\_VALLEY(VALUE, VARIABLES, FOUND)}

Arguments

\begin{itemize}
\item \texttt{VALUE} : \texttt{dvar}
\item \texttt{VARIABLES} : \texttt{collection(var–dvar)}
\item \texttt{FOUND} : \texttt{collection(var–dvar)}
\end{itemize}

Restrictions

\begin{itemize}
\item \texttt{sv} \leq 2 \lor \texttt{rv} \leq 1 \Rightarrow \texttt{VALUE} = +\infty
\item \texttt{VALUE} = +\infty \lor \texttt{VALUE} \leq \texttt{max}(\texttt{maxv} - 1, (\texttt{sv} - 2) * (\texttt{maxv} - 1))
\item \texttt{among}([\texttt{n1}, \texttt{VARIABLES}][2, \texttt{sv} - 1], (\texttt{maxv} - 1))
\item \texttt{VALUE} < +\infty \Rightarrow \texttt{n1} \geq \texttt{VALUE} - \texttt{max}(0, (\texttt{sv} - 2) * (\texttt{maxv} - 2))
\item \texttt{among}([\texttt{n2}, \texttt{VARIABLES}][2, \texttt{sv} - 1], \texttt{minv})
\item \texttt{VALUE} < +\infty \Rightarrow \texttt{n2} \geq \texttt{min}(0, (\texttt{sv} - 2) * (\texttt{minv} + 1)) - \texttt{VALUE}
\end{itemize}

\texttt{required(VARIABLES, var)}

\texttt{required(FOUND, var)}

\texttt{|VARIABLES| = |FOUND|}

where

\begin{itemize}
\item \texttt{minv} = \texttt{minval(VARIABLES.var)}
\item \texttt{sv} = \texttt{|VARIABLES|}
\item \texttt{maxv} = \texttt{maxval(VARIABLES.var)}
\item \texttt{rv} = \texttt{range(VARIABLES.var)}
\end{itemize}

The constraint \texttt{MIN\_SURF\_VALLEY(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{VALLEY} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{VALLEY} is the \textit{maximal} subsequence which matches the regular expression \textquote{\textgreater \textgreater ( = \textless =) \textless \textless}. Assume that the occurrence of the pattern \texttt{VALLEY} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{SURF} computes the sum of the values from index \texttt{i} + 1 to index \texttt{j}.

Example

\begin{itemize}
\item \texttt{(7, 1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)}
\end{itemize}
Figure 4.1211 provides an example where the POS_MIN_SURF_VALLEY constraint holds.

Figure 4.1211: Illustrating the POS_MIN_SURF_VALLEY constraint of the Example slot

**Typical**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Automaton

Similar to the automaton of the MIN_SURF_VALLEY constraint but use the decoration table 3.35.
The constraint MIN_SURF_ZIGZAG(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern ZIGZAG for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘(<>)+(< | <>)| (<>)+(> | <>).’

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+1\) to index \(j\).
Example

\[
\begin{pmatrix}
5, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1), \\
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1212 provides an example where the \textit{POS\_MIN\_SURF\_ZIGZAG} constraint holds.

Figure 4.1212: Illustrating the \textit{POS\_MIN\_SURF\_ZIGZAG} constraint of the \textbf{Example} slot

Typical

\[|\text{VARIABLES}| > 3\]

\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_ZIGZAG constraint but use the decoration table 3.35.
Origin
Based on constraint MIN_WIDTH_DECREASING_SEQUENCE.

Constraint
POS_MIN_WIDTH_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var − dvar)
FOUND : collection(var − dvar)

Restrictions
\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \]
\[ VALUE \geq 2 \]
\[ rv = 2 \Rightarrow VALUE = sv + 1 \lor VALUE \leq 2 \]
\[ rv \geq 3 \Rightarrow VALUE = sv + 1 \lor VALUE \leq sv + 2 \]
\[ \text{required}(VARIABLES, \text{var}) \]
\[ \text{required}(\text{FOUND}, \text{var}) \]
\[ |\text{VARIABLES}| = |\text{FOUND}| \]
where
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES, var}) \]

Purpose
The constraint \( \text{MIN_WIDTH_DECREASING_SEQUENCE}(\text{VALUE, VARIABLES}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{DECREASING_SEQUENCE} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern \text{DECREASING_SEQUENCE} is the maximal subsequence which matches the regular expression ‘\( (> | =)* > | > \)’.

Assume that the occurrence of the pattern \text{DECREASING_SEQUENCE} starts at position \( i \) and ends at position \( j \). The feature \text{WIDTH} computes the value \( j - i + 2 \).

Example
\[
\begin{pmatrix}
2, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4),
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0
\end{pmatrix}
\]

Figure 4.1213 provides an example where the \( \text{POS_MIN_WIDTH_DECREASING_SEQUENCE} \) constraint holds.
Figure 4.1213: Illustrating the POS_MIN_WIDTH_DECREASING_SEQUENCE constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(|\text{VARIABLES}\var|) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton: Similar to the automaton of the MIN_WIDTH_DECREASING_SEQUENCE constraint but use the decoration table 3.35.
Origin
Based on constraint \textit{MIN\_WIDTH\_DECREASING\_TERRACE}.

Constraint
\textit{POS\_MIN\_WIDTH\_DECREASING\_TERRACE}(\textit{VALUE}, \textit{VARIABLES}, \textit{FOUND})

Arguments
\begin{itemize}
  \item \textit{VALUE} : dvar
  \item \textit{VARIABLES} : \textit{collection}(var−dvar)
  \item \textit{FOUND} : \textit{collection}(var−dvar)
\end{itemize}

Restrictions
\begin{align*}
sv & \leq 3 \land rv \leq 2 \Rightarrow \text{VALUE} = sv + 1 \\
\text{VALUE} & \geq 2 \\
\text{VALUE} = sv + 1 \land \text{VALUE} \leq sv - 2 \Rightarrow \\
\text{required}(\text{VARIABLES}.\text{var}) & \\
\text{required}(\text{FOUND}.\text{var}) & \\
|\text{VARIABLES}| & = |\text{FOUND}|
\end{align*}

where
\begin{align*}
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}

The constraint \textit{MIN\_WIDTH\_DECREASING\_TERRACE} holds. In addition, \textit{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \textit{found} letter in those occurrences of the pattern \textit{DECREASING\_TERRACE} for which the feature value is \textit{VALUE}.

Purpose
The position of the \textit{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \textit{DECREASING\_TERRACE} is the maximal subsequence which matches the regular expression ‘bable’.

Assume that the occurrence of the pattern \textit{DECREASING\_TERRACE} starts at position \(i\) and ends at position \(j\). The feature \textit{WIDTH} computes the value \(j - i\).

Example
\[
\begin{pmatrix}
  2, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3) \\
  \langle 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0 \rangle
\end{pmatrix}
\]

Figure 4.1214 provides an example where the \textit{POS\_MIN\_WIDTH\_DECREASING\_TERRACE} \((2, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0])\) constraint holds.

Typical
\begin{itemize}
  \item \(|\text{VARIABLES}| > 3\)
  \item \textit{range}(\text{VARIABLES}.\text{var}) > 2
\end{itemize}
Figure 4.1214: Illustrating the POS\_MIN\_WIDTH\_DECREASING\_TERRACE constraint of the Example slot

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MIN_WIDTH_DECREASING_TERRACE` constraint but use the decoration table 3.35.
POS_MIN_WIDTH_DECREASING_TERRACE 2599
**POS_MIN_WIDTH_GORGE**

**Origin**
Based on constraint `MIN_WIDTH_GORGE`.

**Constraint**
`POS_MIN_WIDTH_GORGE(VALUE, VARIABLES, FOUND)`

**Arguments**
- `VALUE` : `dvar`
- `VARIABLES` : `collection(var-dvar)`
- `FOUND` : `collection(var-dvar)`

**Restrictions**
\[
sv \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = sv + 1
\]
\[
\text{VALUE} \geq 1
\]
\[
\text{rv} = 2 \Rightarrow \text{VALUE} = sv + 1 \lor \text{VALUE} \leq 1
\]
\[
\text{rv} \geq 3 \Rightarrow \text{VALUE} = sv + 1 \lor \text{VALUE} \leq sv - 2
\]

`required(VARIABLES.var)`
`required(FOUND.var)`

\[|\text{VARIABLES}| = |\text{FOUND}|\]

where
\[
sv = |\text{VARIABLES}|
\]
\[
\text{rv} = \text{range}(\text{VARIABLES}.var)
\]

The constraint `MIN_WIDTH_GORGE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `GORGE` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `GORGE` is the *maximal* subsequence which matches the regular expression `'(>|=|>)^*)(<|=<|<)^'`.

Assume that the occurrence of the pattern `GORGE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j - i`.

**Purpose**

**Example**

\[
(1, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
\]

Figure 4.1215 provides an example where the `POS_MIN_WIDTH_GORGE (1, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])` constraint holds.

**Typical**

| `|VARIABLES| > 2`

| `range(VARIABLES.var) > 1`
Figure 4.1215: Illustrating the POS_MIN_WIDTH_GORGE constraint of the Example slot

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_WIDTH_GORGE constraint but use the decoration table 3.35.
Origin

Based on constraint MIN_WIDTH_INCREASING_SEQUENCE.

Constraint

\( \text{POS}_\text{MIN}_\text{WIDTH}_\text{INCREASING}_\text{SEQUENCE}(\text{VALUE}, \text{VARIABLES}, \text{FOUND}) \)

Arguments

\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar}) \\
\text{FOUND} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}

Restrictions

\begin{align*}
\text{sv} \leq 1 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = \text{sv} + 1 \\
\text{VALUE} \geq 2 & \\
\text{rv} = 2 & \Rightarrow \text{VALUE} = \text{sv} + 1 \lor \text{VALUE} \leq 20 \\
\text{rv} \geq 3 & \Rightarrow \text{VALUE} = \text{sv} + 1 \lor \text{VALUE} \leq \text{sv} \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
|\text{VARIABLES}| = |\text{FOUND}| & \\
\text{where} & \\
\text{sv} = |\text{VARIABLES}| & \\
\text{rv} = \text{range}(\text{VARIABLES}, \text{var}) &
\end{align*}

The constraint MIN_WIDTH_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING_SEQUENCE for which the feature value is VALUE.

Purpose

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\( (< | =)* < | \)’.

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i + 2 \).

Example

\[ (2, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) \]

Figure 4.1216 provides an example where the POS_MIN_WIDTH_INCREASING_SEQUENCE constraint holds.
Figure 4.1216: Illustrating the POS_MIN_WIDTH_INCREASING_SEQUENCE constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MIN_WIDTH_INCREASING_SEQUENCE` constraint but use the decoration table 3.35.
Origin Based on constraint \texttt{MIN\_WIDTH\_INCREASING\_TERRACE}.

Constraint

\begin{verbatim}
POS_MIN_WIDTH_INCREASING_TERRACE(VALUE, VARIABLES, FOUND)
\end{verbatim}

Arguments

\begin{verbatim}
VALUE : dvar
VARIABLES : collection(var--dvar)
FOUND : collection(var--dvar)
\end{verbatim}

Restrictions

\begin{verbatim}
sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = sv + 1
VALUE \geq 2
VALUE = sv + 1 \lor VALUE \leq sv - 2
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
\end{verbatim}

where

\begin{verbatim}
sv = |VARIABLES|
rv = range(VARIABLES.var)
\end{verbatim}

Purpose

The constraint \texttt{MIN\_WIDTH\_INCREASING\_TERRACE(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{INCREASING\_TERRACE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{INCREASING\_TERRACE} is the \textit{maximal} subsequence which matches the regular expression '\texttt{< = + <}.'

Assume that the occurrence of the pattern \texttt{INCREASING\_TERRACE} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{WIDTH} computes the value \texttt{j - i}.

Example

\begin{verbatim}
( 2,(1,3,3,3,2,5,5,6,4,4,2,3,3,3,4,4),
  (0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0) )
\end{verbatim}

Figure 4.1217 provides an example where the \texttt{POS\_MIN\_WIDTH\_INCREASING\_TERRACE (2, [1,3,3,3,2,5,5,6,4,4,2,3,3,3,4,4], [0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0])} constraint holds.

Typical

\begin{verbatim}
|VARIABLES| > 3
range(VARIABLES.var) > 2
\end{verbatim}
Figure 4.1217: Illustrating the POS_MIN_WIDTH_INCREASING_TERRACE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the \texttt{MIN\_WIDTH\_INCREASING\_TERRACE} constraint but use the decoration table \ref{tab:3.35}.
Origin
Based on constraint MIN_WIDTH_INFLEXION.

Constraint
POS_MIN_WIDTH_INFLEXION(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \\
VALUE & \geq 1 \\
VALUE & = sv + 1 \lor VALUE \leq sv - 2
\end{align*}
\]

required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
sv = |VARIABLES|
rv = range(VARIABLES.var)

The constraint MIN_WIDTH_INFLEXION(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INFLEXION for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

Purpose
An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression ‘\(< (< | =) | > (> | =) > '<. Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example
\[
\left( 1, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4), \langle 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0 \rangle \right)
\]

Figure 4.1218 provides an example where the POS_MIN_WIDTH_INFLEXION (1,[1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4], [0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1
Figure 4.1218: Illustrating the POS_MIN_WIDTH_INFLEXION constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_WIDTH_INFLEXION constraint but use the decoration table 3.35.
Origin
Based on constraint MIN_WIDTH_PEAK.

Constraint
POS_MIN_WIDTH_PEAK(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \]
\[ VALUE \geq 1 \]
\[ VALUE = sv + 1 \lor VALUE \leq sv - 2 \]
required(VARIABLES, var)
required(FOUND, var)
where
\[ sv = |VARIABLES| \]
\[ rv = \text{range}(VARIABLES, var) \]

Purpose
An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression '\(< (= | <)\)\>(> | ==)\)\>'.
Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example
\[
\begin{pmatrix}
2, 7, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1219 provides an example where the POS_MIN_WIDTH_PEAK
\((2, 7, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)\)
constraint holds.

Typical
\[ |\text{VARIABLES}| \geq 2 \]
\[ \text{range}(\text{VARIABLES, var}) > 1 \]
Figure 4.1219: Illustrating the POS_MIN_WIDTH_PEAK constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the MIN_WIDTH_PEAK constraint but use the decoration table 3.35.
POS_MIN_WIDTH_PEAK

2619
Origin: Based on constraint MIN_WIDTHPLAIN.

Constraint: POS_MIN_WIDTHPLAIN(VALUE, VARIABLES, FOUND)

Arguments:
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions:
\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \]
\[ VALUE \geq 1 \]
\[ VALUE = sv + 1 \lor VALUE \leq sv - 2 \]
required(VARIABLES, var)
required(FOUND, var)
\[ |VARIABLES| = |FOUND| \]

where
\[ sv = |VARIABLES| \]
\[ rv = \text{range}(VARIABLES, \text{var}) \]

Purpose:
The constraint MIN_WIDTHPLAIN(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PLAIN for which the feature value is VALUE.
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘>∗<’. Assume that the occurrence of the pattern PLAIN starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

Example:
\[
\begin{pmatrix}
1, 2, 3, 6, 5, 7, 6, 6, 4, 5, 4, 3, 3, 6, 6, 3, \\
0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 4.1220 provides an example where the POS_MIN_WIDTHPLAIN constraint holds.

Typical:
\[ |VARIABLES| > 2 \]
\[ \text{range}(VARIABLES, \text{var}) > 1 \]
Figure 4.1220: Illustrating the POS_MIN_WIDTHPLAIN constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_WIDTH_PLAIN constraint but use the decoration table 3.35.
**Origin**
Based on constraint MIN_WIDTH_PLATEAU.

**Constraint**
POS_MIN_WIDTH_PLATEAU(VALUE, VARIABLES, FOUND)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)
- FOUND : collection(var−dvar)

**Restrictions**
- $sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = sv + 1$
- $VALUE \geq 1$
- $VALUE = sv + 1 \lor VALUE \leq sv - 2$
- required(VARIABLES, var)
- required(FOUND, var)
- $|VARIABLES| = |FOUND|$

where
- $sv = |VARIABLES|$
- $rv = range(VARIABLES.var)$

The constraint MIN_WIDTH_PLATEAU(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PLATEAU for which the feature value is VALUE.

**Purpose**
An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression '<=*>'.
Assume that the occurrence of the pattern PLATEAU starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$.

**Example**

$$\begin{pmatrix}
3, \{1, 3, 5, 5, 5, 2, 4, 4, 3, 3, 1, 5, 5\}, \\
\{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0\}
\end{pmatrix}$$

Figure 4.1221 provides an example where the POS_MIN_WIDTH_PLATEAU (3, [1, 3, 5, 5, 5, 2, 4, 4, 3, 3, 1, 5, 5], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]) constraint holds.

**Typical**
- $|VARIABLES| > 2$
- $range(VARIABLES.var) > 1$
Figure 4.1221: Illustrating the POS_MIN_WIDTH_PLATEAU constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_WIDTH_PLATEAU constraint but use the decoration table \ref{table:3.35}. 
POS_MIN_WIDTH_PLATEAU 2627
POS_MIN_WIDTH_PROPERPLAIN

Origin
Based on constraint MIN_WIDTH_PROPERPLAIN.

Constraint
POS_MIN_WIDTH_PROPERPLAIN(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
\[ sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \]
VALUE \geq 2
VALUE = sv + 1 \lor VALUE \leq sv - 2 \Rightarrow
\text{required}(\text{VARIABLES}.var)
\text{required}(\text{FOUND}.var)

where
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES}.var) \]

The constraint MIN_WIDTH_PROPERPLAIN(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PROPERPLAIN for which the feature value is VALUE.

Purpose
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression ‘\(> =+<\)’.
Assume that the occurrence of the pattern PROPERPLAIN starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example
\[
\begin{pmatrix}
2, (2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5), \\
0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 4.1222 provides an example where the POS_MIN_WIDTH_PROPERPLAIN \((2, [2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5], [0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0])\) constraint holds.

Typical
\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.var) > 1 \]
Figure 4.1222: Illustrating the POS_MIN_WIDTH_PROPER_PLAIN constraint of the Example slot

Arg. properties

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN\_WIDTH\_PROPER\_PLAIN} constraint but use the decoration table \texttt{3.35}.
Origin Based on constraint MIN_WIDTH_PROPER_PLATEAU.

Constraint POS_MIN_WIDTH_PROPER_PLATEAU(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = sv + 1
VALUE ≥ 2
VALUE = sv + 1 ∨ VALUE ≤ sv − 2
required(VARIABLES.var)
required(FOUND.var)
|VARIABLES| = |FOUND|

where
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression ‘< =+ >’.

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example
\[
\left( 2, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3), \langle 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0 \rangle \right)
\]

Figure 4.1223 provides an example where the POS_MIN_WIDTH_PROPER_PLATEAU (2, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3], [0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]) constraint holds.

Typical
|VARIABLES| > 3
range(VARIABLES.var) > 1
Figure 4.1223: Illustrating the POS_MIN_WIDTH_PROPER_PLATEAU constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_WIDTH_PROPER_PLATEAU constraint but use the decoration table 3.35.
Origin Based on constraint \texttt{MIN\_WIDTH\_STEADY\_SEQUENCE}.

Constraint \texttt{POS\_MIN\_WIDTH\_STEADY\_SEQUENCE(VALUE, VARIABLES, FOUND)}

Arguments \begin{itemize}
\item \texttt{VALUE} : dvar
\item \texttt{VARIABLES} : collection(var--dvar)
\item \texttt{FOUND} : collection(var--dvar)
\end{itemize}

Restrictions
\begin{align*}
sv &\leq 1 \Rightarrow \text{VALUE} = sv + 1 \\
rv &\leq 1 \Rightarrow \text{VALUE} \geq sv \\
rv &\geq 2 \Rightarrow \text{VALUE} \geq 2 \\
\text{VALUE} &\leq sv + 1 \lor \text{VALUE} \leq sv \\
\text{required(VARIABLES, var)} &\lor \text{required(FOUND, var)} \\
|\text{VARIABLES}| &\leq |\text{FOUND}|
\end{align*}

where
\begin{itemize}
\item \texttt{rv} = \text{range(VARIABLES, var)}
\item \texttt{sv} = |\text{VARIABLES}|
\end{itemize}

Purpose
The constraint \texttt{MIN\_WIDTH\_STEADY\_SEQUENCE(VALUE, VARIABLES)} holds. In addition, \texttt{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \texttt{found} letter in those occurrences of the pattern \texttt{STEADY\_SEQUENCE} for which the feature value is \texttt{VALUE}.

The position of the \texttt{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \texttt{STEADY\_SEQUENCE} is the maximal subsequence which matches the regular expression \texttt{\ldots+\ldots}.

Assume that the occurrence of the pattern \texttt{STEADY\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{WIDTH} computes the value \(j - i + 2\).

Example
\begin{itemize}
\item \texttt{(2, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 3, 2, 1, 1), (0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0))}
\end{itemize}

Figure 4.1224 provides an example where the \texttt{POS\_MIN\_WIDTH\_STEADY\_SEQUENCE} \((2, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 3, 2, 1, 1], [0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1])\) constraint holds.

Typical
\(|\text{VARIABLES}| > 1\)
Figure 4.1224: Illustrating the $\text{POS}_\text{MIN}_\text{WIDTH}_\text{STEADY}_\text{SEQUENCE}$ constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN\_WIDTH\_STEADY\_SEQUENCE} constraint but use the decoration table 3.35.
**Description**

Based on constraint `MIN_WIDTH STRICTLY DECREASING SEQUENCE`.

**Constraint**

`POS_MIN_WIDTH STRICTLY DECREASING SEQUENCE(VALUE, VARIABLES, FOUND)`

**Arguments**

- `VALUE`: `dvar`
- `VARIABLES`: `collection(var-dvar)`
- `FOUND`: `collection(var-dvar)`

**Restrictions**

- `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = sv + 1`
- `VALUE ≥ 2`
- `VALUE = sv + 1 ∨ VALUE ≤ min(sv, rv)`
  - `required(VARIABLES, var)`
  - `required(FOUND, var)`
- `|VARIABLES| = |FOUND|`

where

- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**

The constraint `MIN_WIDTH STRICTLY DECREASING SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `STRICTLY DECREASING SEQUENCE` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `STRICTLY DECREASING SEQUENCE` is the maximal subsequence which matches the regular expression `‘>’+‘.’`.

Assume that the occurrence of the pattern `STRICTLY DECREASING SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j - i + 2`.

**Example**

```
( 2, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])
```

Figure 4.1225 provides an example where the `POS_MIN_WIDTH STRICTLY DECREASING SEQUENCE (2, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 4.1225: Illustrating the \texttt{POS\_MIN\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot

\begin{itemize}
  \item Functional dependency: VALUE determined by VARIABLES.
  \item Functional dependency: FOUND determined by VARIABLES.
\end{itemize}
Automaton  Similar to the automaton of the \texttt{MIN\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE} constraint but use the decoration table 3.35.
POS_MIN_WIDTH_STRICTLY_INCREMENTAL_SEQUENCE

Origin
Based on constraint MIN_WIDTH_STRICTLY_INCREMENTAL_SEQUENCE.

Constraint
POS_MIN_WIDTH_STRICTLY_INCREMENTAL_SEQUENCE(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var–dvar)
FOUND : collection(var–dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = sv + 1
VALUE ≥ 2
VALUE = sv + 1 ∨ VALUE ≤ min(sv, rv)
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
The constraint MIN_WIDTH_STRICTLY_INCREMENTAL_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STRICTLY_INCREMENTAL_SEQUENCE for which the feature value is VALUE.

An occurrence of the pattern STRICTLY_INCREMENTAL_SEQUENCE is the maximal subsequence which matches the regular expression '<+'.

Assume that the occurrence of the pattern STRICTLY_INCREMENTAL_SEQUENCE starts at position i and ends at position j. The feature WIDTH computes the value j − i + 2.

Example

Figure 4.1226 provides an example where the POS_MIN_WIDTH_STRICTLY_INCREMENTAL_SEQUENCE constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES.var) > 1
Figure 4.1226: Illustrating the POS_MIN_WIDTH STRICTLY_INCREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint but use the decoration table 3.35.
POS_MIN_WIDTH_SUMMIT

**Origin**
Based on constraint MIN_WIDTH_SUMMIT.

**Constraint**
POS_MIN_WIDTH_SUMMIT(VALUE, VARIABLES, FOUND)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

**Restrictions**
sv ≤ 2 \lor rv ≤ 1 \Rightarrow \text{VALUE} = sv + 1
VALUE ≥ 1
rv = 2 \Rightarrow \text{VALUE} = sv + 1 \lor \text{VALUE} ≤ 1
rv ≥ 3 \Rightarrow \text{VALUE} = sv + 1 \lor \text{VALUE} ≤ sv - 2
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
sv = |VARIABLES|
rv = range(VARIABLES.var)

The constraint MIN_WIDTH_SUMMIT(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern SUMMIT for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression ‘(< | < (= | <)* <) (> | > (= | >)* >)’.
Assume that the occurrence of the pattern SUMMIT starts at position i and ends at position j. The feature WIDTH computes the value j − i.

**Purpose**

**Example**

\[
\begin{pmatrix}
1, (7, 1, 5, 4, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
\end{pmatrix}
\]

Figure 4.1227 provides an example where the POS_MIN_WIDTH_SUMMIT (1, [7, 1, 5, 4, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.

**Typical**

|VARIABLES| > 2
rangex(VARIABLES.var) > 1
Figure 4.1227: Illustrating the POS_MIN_WIDTH_SUMMIT constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_WIDTH_SUMMIT constraint but use the decoration table 3.35.
POS_MIN_WIDTH_SUMMIT  2651
Origin
Based on constraint MIN_WIDTH_VALLEY.

Constraint
POS_MIN_WIDTH_VALLEY(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = sv + 1
VALUE ≥ 1
VALUE = sv + 1 ∨ VALUE ≤ sv − 2
required(VARIABLES, var)
required(FOUND, var)
|VARIABLES| = |FOUND|
where
sv = |VARIABLES|
rv =range(VARIABLES.var)

The constraint MIN_WIDTH_VALLEY(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern VALLEY for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression ‘> (= | >)∗ (< | =)∗ <’. Assume that the occurrence of the pattern VALLEY starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j − i$.

Purpose

Example

Figure 4.1228 provides an example where the POS_MIN_WIDTH_VALLEY (2, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical

|VARIABLES| > 2
range(VARIABLES.var) > 1
Figure 4.1228: Illustrating the POS_MIN_WIDTH_VALLEY constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the MIN_WIDTH_VALLEY constraint but use the decoration table 3.35.
POS_MIN_WIDTH_VALLEY  2655
POS_MIN_WIDTH_ZIGZAG

**Description**

Based on constraint MIN_WIDTH_ZIGZAG.

**Constraint**

POS_MIN_WIDTH_ZIGZAG(VALUE, VARIABLES, FOUND)

**Arguments**

<table>
<thead>
<tr>
<th>VALUE</th>
<th>VARIABLES</th>
<th>FOUND</th>
</tr>
</thead>
<tbody>
<tr>
<td>dvar</td>
<td>collection(var−dvar)</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \\
VALUE \geq 2 \\
VALUE = sv + 1 \lor VALUE \leq sv - 2 \\
required(VARIABLES, var) \\
required(FOUND, var) \\
|VARIABLES| = |FOUND| \\
where \\
sv = |VARIABLES| \\
rv = range(VARIABLES.var)
\]

The constraint MIN_WIDTH_ZIGZAG(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern ZIGZAG for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘(<>)+(< | <>) | (<>)+(>| <>)’.

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\[
\begin{pmatrix}
2, (4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1), \\
(0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 4.129 provides an example where the POS_MIN_WIDTH_ZIGZAG (2, [4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Purpose**

**Typical**

| | VARIABLES | > 3 |
|-----------------|-----------------|
| range(VARIABLES.var) | > 1 |
Figure 4.1229: Illustrating the POS_MIN_WIDTH_ZIGZAG constraint of the Example slot

Arg. properties

- **Functional dependency:** VALUE determined by VARIABLES.
- **Functional dependency:** FOUND determined by VARIABLES.
Automaton: Similar to the automaton of the MIN_WIDTH_ZIGZAG constraint but use the decoration table 3.35.
SUM_HEIGHT_DECREASING_TERRACE

**Origin**  
Based on the DECREASING_TERRACE pattern.

**Constraint**  
SUM_HEIGHT_DECREASING_TERRACE(VALUE, VARIABLES)

**Arguments**  
VALUE : dvar  
VARIABLES : collection(var−dvar)
Restrictions

\[ sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = 0 \]

\[
\begin{align*}
\text{VALUE} \geq \min_{q \in [1b1,ub1]} \min &
\begin{cases}
0, \sum \left( \prod \left( \min \left( 1, \max \left( 0, \frac{rv - 2}{q} \right) \right) \right), \\
q, \sum \left( \min \left( np1 \ast \min v, \frac{[np1 + 1] \ast np1}{2} \right) \right), \\
\end{cases}
\end{align*}
\]

\[
\begin{align*}
\text{VALUE} \leq \max_{q \in [1b2,ub2]} \max &
\begin{cases}
0, \sum \left( \prod \left( \min \left( 1, \max \left( 0, \frac{rv - 2}{q} \right) \right) \right), \\
q, \sum \left( \min \left( np2 \ast \max v - \left( [np2 + 1] \ast np2 \right) \right) \right), \\
\end{cases}
\end{align*}
\]

where

\[ sv = |\text{VARIABLES}| \]

\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

\[ np1 = \max \left( 0, \sum \left( \min \left( 0, \frac{\min (rv, \min v + 1) - [\max (1, sv - sv \ mod 2) - 2 \ast q)/2 \ast q]}{2} \right) \right) \right) \]

\[ np2 = \max \left( 0, \sum \left( \min \left( 0, \frac{\min (rv, \max v + 1) - [\max (1, sv - sv \ mod 2) - 2 \ast q)/2 \ast q]}{2} \right) \right) \right) \]

\[ maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \]

\[ minv = \text{minval}(\text{VARIABLES}.\text{var}) \]

\[ 1b1 = \min \left( \sum \left( \min \left( \max \left( 1, sv - sv \ mod 2 \right) / \max (1, sv - sv \ mod 2), \ \min \left( \max (1, sv - sv \ mod 2), \ \min (\max v + 1, rv) - 2 \ast 2 + 2 \right) \right) \right) \right) \]

\[ ub1 = \left[ \frac{sv}{4} \right] + 1 \]

\[ 1b2 = \min \left( \sum \left( \min \left( \max (1, sv - sv \ mod 2), \ \min (1, \min (\max v + 1, rv) - 2 \ast 2 + 2) \right) \right) \right) \]

\[ ub2 = \left[ \frac{sv}{4} \right] + 1 \]
VALUE is the sum of all minimum values in each occurrence of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression ‘> = >’. Assume that the occurrence of the pattern DECREASING_TERRACE starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i + 1$ to index $j$.

Example

$$(6, (6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3))$$

Figure 4.1230 provides an example where the SUM_HEIGHT_DECREASING_TERRACE constraint holds.

Typical

$|\text{VARIABLES}| > 3$

$\text{range}(\text{VARIABLES}.\text{var}) > 2$

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1231 and 4.1232 respectively depict the automaton associated with the constraint \textsc{SUM}\_\textsc{HEIGHT}\_\textsc{DECREASING}\_\textsc{TERRACE} and its simplified form.

\begin{align*}
&\{ C \leftarrow \text{default} \} \\
&\{ D \leftarrow +\infty \} \\
&\{ R \leftarrow \text{default} \} \\
&\leq s \\
&\leq \\
&= \\
&\{ D \leftarrow \min(D, \text{VAR}_i) \} \\
&\leq s \\
&\geq \{ R \leftarrow R + \text{VAR}_i \} \\
&> r = t \\
&> \\
&= \\
&\{ D \leftarrow \min(D, \text{VAR}_i) \} \\
&< \\
&\geq \{ D \leftarrow +\infty \} \\
&< \\
&\leq s \\
&\leq \\
&= \\
&\{ R \leftarrow R + \text{VAR}_i \} \\
&> r = t \\
&> \\
&= \\
&\{ D \leftarrow \min(D, \text{VAR}_i) \} \\
&< \\
&\geq \{ D \leftarrow +\infty \} \\
&< \\
&\leq s
\end{align*}

Figure 4.1231: Automaton for the \textsc{SUM}\_\textsc{HEIGHT}\_\textsc{DECREASING}\_\textsc{TERRACE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsc{DECREASING}\_\textsc{TERRACE} pattern where \texttt{default} is 0

\begin{align*}
&\{ R \leftarrow \text{default} \} \\
&\leq s \\
&\leq \\
&= \\
&\{ R \leftarrow R + \text{VAR}_i \} \\
&> r = t \\
&> \\
&= \\
&\{ R \leftarrow \text{default} \} \\
&\leq s \\
&\leq \\
&= \\
&\{ R \leftarrow R + \text{VAR}_i \} \\
&> r = t \\
&> \\
&= \\
&\{ R \leftarrow \min(D, \text{VAR}_i) \} \\
&< \\
&\geq \{ D \leftarrow +\infty \} \\
&< \\
&\leq s
\end{align*}

Figure 4.1232: Simplified automaton for the \textsc{SUM}\_\textsc{HEIGHT}\_\textsc{DECREASING}\_\textsc{TERRACE} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \textsc{DECREASING}\_\textsc{TERRACE} pattern where \texttt{default} is 0
Table 4.265: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the `SUM_HEIGHT_DECREASING_TERRACE` constraint defined as the composition of the `DECREASING_TERRACE` pattern, the feature `MIN`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\vec{D} + \vec{C})</td>
<td>(\vec{D} + \vec{C})</td>
<td>(\vec{D} + \vec{C})</td>
</tr>
<tr>
<td>(r)</td>
<td>(\vec{D} + \vec{C})</td>
<td>(\vec{D} + \vec{C})</td>
<td>(\min(\vec{D}, \vec{D}, \text{VAR}_{i+1}))</td>
</tr>
<tr>
<td>(t)</td>
<td>(\vec{D} + \vec{C})</td>
<td>(\min(\vec{D}, \vec{D}, \text{VAR}_{i+1}))</td>
<td>(\min(\vec{D}, \vec{D}, \text{VAR}_{i+1}))</td>
</tr>
</tbody>
</table>

Table 4.266: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the simplified automaton of the `SUM_HEIGHT_DECREASING_TERRACE` constraint defined as the composition of the `DECREASING_TERRACE` pattern, the feature `MIN`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(0)</td>
<td>(0)</td>
<td>(0)</td>
</tr>
<tr>
<td>(r)</td>
<td>(0)</td>
<td>(0)</td>
<td>(\text{VAR}_{i+1})</td>
</tr>
<tr>
<td>(t)</td>
<td>(\text{VAR}_{i+1})</td>
<td>(\text{VAR}_{i+1})</td>
<td>(\text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>
**SUM_HEIGHT_INCREASING_TERRACE**

**DESCRIPTION AUTOMATON**

**Origin**
Based on the **INCREASING_TERRACE** pattern.

**Constraint**

```
SUM_HEIGHT_INCREASING_TERRACE(VALUE, VARIABLES)
```

**Arguments**

```
VALUE      : dvar
VARIABLES  : collection(var−dvar)
```
Restrictions

\[ sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = 0 \]

\[
VALUE \geq \min_{q \in [131,136]} \min \left( \Pi \sum_{0}^{\min(v, \lfloor (np1 + 1) \cdot np1/2 \rfloor)} \left( \min \left( 1, \max \left( 0, \frac{rv}{2} - 2 \right) \right), \right) \right),
\]

\[
VALUE \leq \max_{q \in [123,129]} \max \left( \Pi \sum_{0}^{\min(v, \lfloor (np2 + 1) \cdot np2/2 \rfloor)} \left( \min \left( 1, \max \left( 0, \frac{rv}{2} - 2 \right) \right), \right) \right),
\]

where

- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range(\text{VARIABLES.var})} \)
- \( np1 = \max \left( \Pi \sum_{0}^{\min(v, \lfloor (max(1, sv - sv \mod 2) - 2 \cdot q)/2 \cdot q \rfloor)} \left( \min \left( 0, \frac{min(rv, minv + 1) - \lfloor (max(1, sv - sv \mod 2) - 2 \cdot q)/2 \cdot q \rfloor}{2} \right), \right) \right), \)
- \( np2 = \max \left( \Pi \sum_{0}^{\min(v, \lfloor (max(1, sv - sv \mod 2) - 2 \cdot q)/2 \cdot q \rfloor)} \left( \min \left( 0, \frac{min(rv, maxv + 1) - \lfloor (max(1, sv - sv \mod 2) - 2 \cdot q)/2 \cdot q \rfloor}{2} \right), \right) \right), \)
- \( maxv = \text{maxval(\text{VARIABLES.var})} \)
- \( minv = \text{minval(\text{VARIABLES.var})} \)

\[ lb1 = \min \left( \sum_{\lfloor sv/4 \rfloor + 1}^{\lfloor sv/4 \rfloor + 1} \left( \min \left( 0, \frac{min(max(1, sv - sv \mod 2), max(1, sv - sv \mod 2))}{max(1, sv - sv \mod 2 \mod max(1, sv - sv \mod 2), max(2, (min(minv + 1, rv) - 2) + 2 + 2)) - 3} \right), \right) \right), \]

\[ ub1 = \lfloor sv/4 \rfloor + 1 \]

\[ lb2 = \min \left( \sum_{\lfloor sv/4 \rfloor + 1}^{\lfloor sv/4 \rfloor + 1} \left( \min \left( 0, \frac{min(max(1, sv - sv \mod 2), max(1, sv - sv \mod 2))}{max(1, sv - sv \mod 2 \mod max(1, sv - sv \mod 2), max(2, (min(minv + 1, rv) - 2) + 2 + 2)) - 3} \right), \right) \right), \]

\[ ub2 = \lfloor sv/4 \rfloor + 1 \]
Purpose

VALUE is the sum of all minimum values in each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression ‘< = < ‘.

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \( i + 1 \) to index \( j \).

Example

\((8, (1, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4))\)

Figure 4.1233 provides an example where the SUM_HEIGHT_INCREASING_TERRACE \((8, [1, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4])\) constraint holds.

Typical

\(|\text{VARIABLES}| > 3\)

\(\text{range}(\text{VARIABLES}.\text{var}) > 2\)

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1234 and 4.1235 respectively depict the automaton associated with the constraint SUM_HEIGHT_INCREASING_TERRACE and its simplified form.

Figure 4.1234: Automaton for the SUM_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_TERRACE pattern where default is 0

Figure 4.1235: Simplified automaton for the SUM_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 3.39 to the seed transducer of the INCREASING_TERRACE pattern where default is 0
Table 4.267: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the SUM_HEIGHT_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}))</td>
</tr>
<tr>
<td>t</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}))</td>
<td>(\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}))</td>
</tr>
</tbody>
</table>

Table 4.268: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the simplified automaton of the SUM_HEIGHT_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>0</td>
<td>(\text{VAR}_{i+1})</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>(\text{VAR}_{i+1})</td>
<td>(\text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>
SUM_HEIGHT_INCREASING_TERRACE 2671
SUM_HEIGHTPLAIN

DESCRIPTION

Origin
Based on the PLAIN pattern.

Constraint
SUM_HEIGHTPLAIN(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0
VALUE = 0 \lor VALUE \geq \min(minv, minv \cdot np \cdot 2)
VALUE = 0 \lor VALUE \leq \max(maxv − 1, (maxv − 1) \cdot np)
VALUE \geq \min(VARIABLES, var)
where
sv = |VARIABLES|
np = \max(0, \left\lfloor (sv − 1)/2 \right\rfloor)
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
rv = range(VARIABLES.var)

Purpose
VALUE is the sum of all minimum values in each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘> = ‘<’.

Assume that the occurrence of the pattern PLAIN starts at position i and ends at position j. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index i + 1 to index j.

Example
(12, ⟨2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 6, 6, 3⟩)

Figure 4.1236 provides an example where the SUM_HEIGHTPLAIN (12, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 6, 6, 3]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1

Symmetry
Items of VARIABLES can be reversed.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1236: Illustrating the SUM_HEIGHT_PLAIN constraint of the Example slot
Automaton

Figures 4.1237 and 4.1238 respectively depict the automaton associated with the constraint SUM_HEIGHT_PLAIN and its simplified form.

Figure 4.1237: Automaton for the SUM_HEIGHT_PLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLAIN pattern where default is 0.

Figure 4.1238: Simplified automaton for the SUM_HEIGHT_PLAIN constraint obtained by applying decoration Table 3.39 to the seed transducer of the PLAIN pattern where default is 0.
Table 4.269: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the `SUM_HEIGHT.PLAIN` constraint defined as the composition of the `PLAIN` pattern, the `MIN` feature, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overline{C} + \overline{C})</td>
<td>min((\overline{D}, \overline{D}, \text{VAR}_{i+1}))</td>
<td>min((\overline{D}, \overline{D}, \text{VAR}_{i+1}))</td>
</tr>
<tr>
<td>t</td>
<td>(\overline{C} + \overline{C})</td>
<td>min((\overline{D}, \overline{D}, \text{VAR}_{i+1}))</td>
<td>min((\overline{D}, \overline{D}, \text{VAR}_{i+1}))</td>
</tr>
</tbody>
</table>

Table 4.270: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the simplified automaton of the `SUM_HEIGHT.PLAIN` constraint defined as the composition of the `PLAIN` pattern, the `MIN` feature, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>(\text{VAR}_{i+1})</td>
<td>(\text{VAR}_{i+1})</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>(\text{VAR}_{i+1})</td>
<td>(\text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>
SUM\_HEIGHT\_PLATEAU

\textbf{DESCRIPTION}

Origin
Based on the PLATEAU pattern.

Constraint
\begin{align*}
\text{SUM\_HEIGHT\_PLATEAU}(\text{VALUE}, \text{VARIABLES})
\end{align*}

Arguments
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}

Restrictions
\begin{align*}
\text{sv} \leq 2 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 \lor \text{VALUE} \geq \min(\minv + 1\ast, (\minv + 1) \ast \text{np}) & \\
\text{VALUE} = 0 \lor \text{VALUE} \leq \max(\maxv \ast \text{np}) & \\
\text{required}(\text{VARIABLES}.\text{var}) & \\
\text{where} & \\
\text{sv} & = |\text{VARIABLES}| \\
\text{np} & = \max(0, \lfloor (\text{sv} - 1)/2 \rfloor) \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}

VALUE is the sum of all minimum values in each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose
An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression '\textless \ast \textgreater'.
Assume that the occurrence of the pattern PLATEAU starts at position $i$ and ends at position $j$. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index $i + 1$ to index $j$.

Example
\begin{align*}
(12, \langle 7, 5, 2, 3, 1, 2, 2, 4, 3, 4, 5, 5, 2, 2, 5 \rangle)
\end{align*}

Figure 4.1239 provides an example where the SUM\_HEIGHT\_PLATEAU constraint holds.

Typical
\begin{align*}
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\end{align*}

Symmetry
Items of VARIABLES can be reversed.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1239: Illustrating the SUM_HEIGHT_PLATEAU constraint of the Example slot
Automaton

Figures 4.1240 and 4.1241 respectively depict the automaton associated with the constraint `SUM_HEIGHT_PLATEAU` and its simplified form.

![Automaton Diagram](image)

Figure 4.1240: Automaton for the `SUM_HEIGHT_PLATEAU` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `PLATEAU` pattern where `default` is 0

![Simplified Automaton Diagram](image)

Figure 4.1241: Simplified automaton for the `SUM_HEIGHT_PLATEAU` constraint obtained by applying decoration Table 3.39 to the seed transducer of the `PLATEAU` pattern where `default` is 0
Table 4.271: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the `SUM_HEIGHT_PLATEAU` constraint defined as the composition of the `PLATEAU` pattern, the feature `MIN`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1})$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1})$</td>
</tr>
<tr>
<td>t</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1})$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1})$</td>
</tr>
</tbody>
</table>

Table 4.272: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the `SUM_HEIGHT_PLATEAU` constraint defined as the composition of the `PLATEAU` pattern, the feature `MIN`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>$\text{VAR}_{i+1}$</td>
<td>$\text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>$\text{VAR}_{i+1}$</td>
<td>$\text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>
SUM_HEIGHT_PROPER_PLAIN

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the PROPER_PLAIN pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>SUM_HEIGHT_PROPER_PLAIN(VALUE, VARIABLES)</td>
</tr>
<tr>
<td>Arguments</td>
<td>VALUE : dvar</td>
</tr>
<tr>
<td></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td>Restrictions</td>
<td>sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = 0</td>
</tr>
<tr>
<td></td>
<td>VALUE = 0 ∨ VALUE ≥ min(minv, minv ⋅ np)</td>
</tr>
<tr>
<td></td>
<td>VALUE = 0 ∨ VALUE ≤ max(maxv – 1, (maxv – 1) ⋅ np)</td>
</tr>
<tr>
<td></td>
<td>required(VARIABLES.var)</td>
</tr>
<tr>
<td>where</td>
<td>sv =</td>
</tr>
<tr>
<td></td>
<td>np = max(0, [(sv – 1)/3])</td>
</tr>
<tr>
<td></td>
<td>minv = minval(VARIABLES.var)</td>
</tr>
<tr>
<td></td>
<td>maxv = maxval(VARIABLES.var)</td>
</tr>
<tr>
<td></td>
<td>rv = range(VARIABLES.var)</td>
</tr>
<tr>
<td>Purpose</td>
<td>VALUE is the sum of all minimum values in each occurrence of the PROPER_PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.</td>
</tr>
<tr>
<td></td>
<td>An occurrence of the pattern PROPER_PLAIN is the maximal subsequence which matches the regular expression ‘&gt; =+ &lt;’.</td>
</tr>
<tr>
<td></td>
<td>Assume that the occurrence of the pattern PROPER_PLAIN starts at position i and ends at position j. The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index i + 1 to index j.</td>
</tr>
<tr>
<td>Example</td>
<td>(12, [2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5])</td>
</tr>
<tr>
<td>Typical</td>
<td></td>
</tr>
<tr>
<td></td>
<td>range(VARIABLES.var) &gt; 1</td>
</tr>
<tr>
<td>Symmetry</td>
<td>Items of VARIABLES can be reversed.</td>
</tr>
</tbody>
</table>
Figure 4.1242: Illustrating the SUM_HEIGHT_PROPERPLAIN constraint of the Example slot

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1243 and 4.1244 respectively depict the automaton associated with the constraint SUM_HEIGHT_PROPERPLAIN and its simplified form.

Figure 4.1243: Automaton for the SUM_HEIGHT_PROPERPLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPERPLAIN pattern where default is 0

Figure 4.1244: Simplified automaton for the SUM_HEIGHT_PROPERPLAIN constraint obtained by applying decoration Table 3.39 to the seed transducer of the PROPERPLAIN pattern where default is 0
Table 4.273: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the \texttt{SUM\_HEIGHT\_PROPER\_PLAIN} constraint defined as the composition of the \texttt{PROPER\_PLAIN} pattern, the feature \texttt{MIN}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\min(\overline{D}, \overline{D}, \text{VAR}_{i+1})$</td>
</tr>
</tbody>
</table>

Table 4.274: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the simplified automaton of the \texttt{SUM\_HEIGHT\_PROPER\_PLAIN} constraint defined as the composition of the \texttt{PROPER\_PLAIN} pattern, the feature \texttt{MIN}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$r$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t$</td>
<td>\text{VAR}_{i+1}</td>
<td>\text{VAR}_{i+1}</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td><strong>Automaton</strong></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <em>proper_plateload</em> pattern.</td>
<td></td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>[ \text{SUM_HEIGHT_PROPER_PLATEAU}(\text{VALUE, VARIABLES}) ]</td>
<td></td>
</tr>
</tbody>
</table>
| **Arguments** | \[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var-dvar)}
\end{align*}
\] |
| **Restrictions** | \[
\begin{align*}
\text{sv} \leq 3 & \vee \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 & \vee \text{VALUE} \geq \min(\text{minv} + 1\&, (\text{minv} + 1) \ast \text{np}) \\
\text{VALUE} = 0 & \vee \text{VALUE} \leq \max(\text{maxv, maxv} \ast \text{np}) \\
\text{required}(\text{VARIABLES, var}) & \\
\text{where} & \\
\text{sv} & = |\text{VARIABLES}| \\
\text{np} & = \max(0, \lfloor (\text{sv} - 1)/3 \rfloor) \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\] |
| **Purpose** | \text{VALUE} is the sum of all minimum values in each occurrence of the *proper_plateload* pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value 0. An occurrence of the pattern *proper_plateload* is the maximal subsequence which matches the regular expression ‘< = + >’. Assume that the occurrence of the pattern *proper_plateload* starts at position \(i\) and ends at position \(j\). The feature \(\text{min} \), called \text{HEIGHT} in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i + 1\) to index \(j\). |
| **Example** | \[(12, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))\] |
| **Typical** | \[
\begin{align*}
|\text{VARIABLES}| & > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\] |
| **Symmetry** | Items of \text{VARIABLES} can be reversed. |
SUM_HEIGHT_PROPER_PLATEAU

Figure 4.1245: Illustrating the SUM_HEIGHT_PROPER_PLATEAU constraint of the Example slot

Arg. properties  
Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1246 and 4.1247 respectively depict the automaton associated with the constraint SUM_HEIGHT_PROPER_PLATEAU and its simplified form.

Figure 4.1246: Automaton for the SUM_HEIGHT_PROPER_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER_PLATEAU pattern where default is 0

Figure 4.1247: Simplified automaton for the SUM_HEIGHT_PROPER_PLATEAU constraint obtained by applying decoration Table 3.39 to the seed transducer of the PROPER_PLATEAU pattern where default is 0
**Table 4.275**: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the **SUM** \_ **HEIGHT** \_ **PROPER** \_ **PLATEAU** constraint defined as the composition of the **PROPER** \_ **PLATEAU** pattern, the feature **MIN**, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}))</td>
</tr>
<tr>
<td>t</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}))</td>
<td>(\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1}))</td>
</tr>
</tbody>
</table>

**Table 4.276**: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the simplified automaton of the **SUM** \_ **HEIGHT** \_ **PROPER** \_ **PLATEAU** constraint defined as the composition of the **PROPER** \_ **PLATEAU** pattern, the feature **MIN**, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>VAR_{i+1}</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>0</td>
<td>\text{VAR}_{i+1}</td>
<td>\text{VAR}_{i+1}</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>\text{VAR}_{i+1}</td>
<td>\text{VAR}_{i+1}</td>
<td></td>
</tr>
</tbody>
</table>
Origin: Based on the STEADY pattern.

Constraint: \(\text{SUM\_HEIGHT\_STEADY}(\text{VALUE}, \text{VARIABLES})\)

Arguments:
- \(\text{VALUE} : \text{dvar}\)
- \(\text{VARIABLES} : \text{collection}(\text{var-dvar})\)

Restrictions:
- \(sv \leq 1 \Rightarrow \text{VALUE} = 0\)
- \(rv = 1 \Rightarrow \text{VALUE} \geq \text{minv} \times \text{np}\)
- \(rv \geq 2 \Rightarrow \text{VALUE} = 0 \vee \text{VALUE} \geq \min(\text{minv} \times \text{np}, \text{minv} \times \text{np})\)
- \(rv = 1 \Rightarrow \text{VALUE} \leq \text{maxv} \times \text{np}\)
- \(rv \geq 2 \Rightarrow \text{VALUE} = 0 \vee \text{VALUE} \leq \max(\text{maxv} \times \text{np}, \text{maxv} \times \text{np})\)

where:
- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range}(\text{VARIABLES}.\text{var})\)
- \(\text{np} = \max(0, sv - 1)\)
- \(\text{minv} = \minval(\text{VARIABLES}.\text{var})\)
- \(\text{maxv} = \maxval(\text{VARIABLES}.\text{var})\)

VALUE is the sum of all minimum values in each occurrence of the STEADY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose:
An occurrence of the pattern STEADY is the subsequence which matches the regular expression ‘=’. Assume that the occurrence of the pattern STEADY starts at position \(i\) and ends at position \(j\). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i\) to index \(j + 1\).

Example:
\((30, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6))\)

Figure 4.1248 provides an example where the \(\text{SUM\_HEIGHT\_STEADY}(30, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6])\) constraint holds.

Typical:
- \(|\text{VARIABLES}| > 1\)

Symmetry:
Items of VARIABLES can be reversed.
Figure 4.1248: Illustrating the SUM_HEIGHT_STEADY constraint of the Example slot

Arg. properties Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1249 and 4.1250 respectively depict the automaton associated with the constraint SUM_HEIGHT_STEADY and its simplified form.

Figure 4.1249: Automaton for the SUM_HEIGHT_STEADY constraint obtained by applying decoration Table 3.37 to the seed transducer of the STEADY pattern where default is 0

Figure 4.1250: Simplified automaton for the SUM_HEIGHT_STEADY constraint obtained by applying decoration Table 3.39 to the seed transducer of the STEADY pattern where default is 0

Table 4.277: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the SUM_HEIGHT_STEADY constraint defined as the composition of the STEADY pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.278: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the simplified automaton of the SUM_HEIGHT_STEADY constraint defined as the composition of the STEADY pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Origin Based on the STEADY_SEQUENCE pattern.

Constraint \[ \text{SUM_HEIGHT_STEADY_SEQUENCE}(\text{VALUE, VARIABLES}) \]

Arguments

- **VALUE**: dvar
- **VARIABLES**: collection(var–dvar)

Restrictions

\[
\begin{align*}
sv \leq 1 & \Rightarrow \text{VALUE} = 0 \\
rv = 1 & \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \geq \text{minv} \\
rv \geq 2 \land \text{minv} = -1 & \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \geq -1 * \lfloor (sv - 2) / 3 \rfloor - 1 \\
rv \geq 2 \land \text{minv} \neq -1 \land sv \leq 4 & \Rightarrow \\
\text{VALUE} = 0, & \lor \text{VALUE} \geq \min(\text{minv}^2, \text{minv} * \lfloor (np + 1) / 2 \rfloor + (\text{minv} + 1) * \lfloor np / 2 \rfloor) \\
rv \geq 2 \land \text{minv} \neq -1 \land sv \leq 5 & \Rightarrow \\
\text{VALUE} = 0, & \lor \text{VALUE} \geq \min(\text{minv}, \text{minv} * \lfloor (np + 1) / 2 \rfloor + (\text{minv} + 1) * \lfloor np / 2 \rfloor - 1) \\
rv = 1 & \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \leq \text{maxv} \\
rv \geq 2 \land \text{maxv} = 1 & \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \leq \lfloor (sv - 2) / 3 \rfloor + 1 \\
rv \geq 2 \land \text{maxv} \neq 1 \land sv \leq 4 & \Rightarrow \\
\text{VALUE} = 0, & \lor \text{VALUE} \leq \max(\text{maxv}, \text{maxv} * \lfloor (np + 1) / 2 \rfloor + (\text{maxv} - 1) * \lfloor np / 2 \rfloor) \\
rv \geq 2 \land \text{maxv} \neq 1 \land sv \geq 5 & \Rightarrow \\
\text{VALUE} = 0, & \lor \text{VALUE} \leq \max(\text{maxv}, \sum \lfloor \text{maxv} * (np + 1) / 2 \rfloor, (\text{maxv} - 1) * \lfloor np / 2 \rfloor, (np + 1) mod 2 * sv mod 2) \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}, \text{var}) \)
- \( np = |sv/2| \)
- \( \text{minv} = \text{minval}(\text{VARIABLES}, \text{var}) \)
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}, \text{var}) \)
VALUE is the sum of all minimum values in each occurrence of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression ‘\(=+\)’.

Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN, called HEIGHT in the name of the constraint since all feature values are identical, computes the minimum of the values from index \(i\) to index \(j+1\).

Example

\((13, \langle 3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1 \rangle)\)

Figure 4.1251 provides an example where the SUM_HEIGHT_STEADY_SEQUENCE \((13, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1])\) constraint holds.

Figure 4.1251: Illustrating the SUM_HEIGHT_STEADY_SEQUENCE constraint of the Example slot

Typical

| VARIABLES | > 1

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1252 and 4.1253 respectively depict the automaton associated with the constraint SUM_HEIGHT_STEADY_SEQUENCE and its simplified form.

Figure 4.1252: Automaton for the SUM_HEIGHT_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STEADY_SEQUENCE pattern where default is 0

Figure 4.1253: Simplified automaton for the SUM_HEIGHT_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.39 to the seed transducer of the STEADY_SEQUENCE pattern where default is 0
Table 4.279: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the SUM_HEIGHT_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>C + C</td>
<td>C + C</td>
</tr>
<tr>
<td>r</td>
<td>C + C</td>
<td>(\text{min}(C, \bar{C}, \bar{D}, \bar{D}))</td>
</tr>
</tbody>
</table>

Table 4.280: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the simplified automaton of the SUM_HEIGHT_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>-\text{VAR}_{i+1}</td>
</tr>
</tbody>
</table>
SUM_MAX_BUMP_ON_DECREASING_SEQUENCE

**DESCRIPTION**

Origin

Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

Constraint

SUM_MAX_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

\[ sv \leq 5 \vee rv \leq 2 \Rightarrow VALUE = 0 \]
\[ VALUE = 0 \vee VALUE \geq \min \left( \minv + 2, \ (\minv + 2) \ast np \right) \]
\[ VALUE = 0 \vee VALUE \leq \max \left( \maxv, \maxv \ast np \right) \]

where

\[ sv = |VARIABLES| \]
\[ np = \max \left( 0, \left( \frac{sv - 3}{3} \right) \right) \]
\[ \minv = \minval (VARIABLES.var) \]
\[ \maxv = \maxval (VARIABLES.var) \]
\[ rv = \text{range} (VARIABLES.var) \]

VALUE is the sum of all maximum values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression ‘>>>’.

Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i + 2 \) to index \( j \).

Example

\((11, [7, 6, 5, 6, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3])\)

Figure 4.1254 provides an example where the SUM_MAX_BUMP_ON_DECREASING_SEQUENCE constraint holds.

Typical

\(|VARIABLES| > 5\]
\[ \text{range} (VARIABLES.var) > 2 \]

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.1254: Illustrating the \texttt{SUM\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Automaton

Figures 4.1255 and 4.1256 respectively depict the automaton associated with the constraint SUM_MAX_BUMP_ON_DECREASING_SEQUENCE and its simplified form.

Figure 4.1255: Automaton for the SUM_MAX_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is 0.
Figure 4.1256: Simplified automaton for the SUM_MAX_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.28 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is 0
SUM_MAX_DECREASING

### DESCRIPTION

Based on the **DECREASING** pattern.

### AUTOMATON

### Constraint

\[ \text{SUM\_MAX\_DECREASING(\text{VALUE, VARIABLES})} \]

### Arguments

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var\_dvar)`
Restrictions

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]

\[ VALUE \geq \min_{q \in [lb1, ub1]} \min \left( q \sum \left( \prod \left( \frac{(np - 1) \cdot \maxv - \minv + 1 + np}{np - 2}, \right), \right), \left( \prod \left( \frac{sv}{np - 1}, \minv + 1, \right), \right) \right) \]

\[ VALUE \leq \max_{q \in [lb2, ub2]} \max \left( q \sum \left( \prod \left( \frac{(np - 1) \cdot \maxv - \minv + 1 + np}{np - 2}, \right), \left( \prod \left( \frac{sv}{np - 1}, \minv + 1, \right) \right) \right) \]

required(VARIABLES, var)

where

\[ sv = |VARIABLES| \]
\[ np = \lfloor sv/q \rfloor \]
\[ maxv = \maxval(VARIABLES, var) \]
\[ minv = \minval(VARIABLES, var) \]
\[ rv = \text{range}(VARIABLES, var) \]
\[ lb1 = \min \left( \sum \left( \frac{sv}{min(v sv, minv + 1)}, \right), \left( \min(1, sv \mod (sv, minv + 1)) \right) \right) \]
\[ ub1 = \lfloor sv/2 \rfloor + 1 \]
\[ lb2 = \min \left( \sum \left( \frac{sv}{min(v sv, maxv + 1)}, \right), \left( \min(1, sv \mod (sv, maxv + 1)) \right) \right) \]
\[ ub2 = \lfloor sv/2 \rfloor + 1 \]

VALUE is the sum of all maximum values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression ">".

Assume that the occurrence of the pattern DECREASING starts at position \( i \) and ends at position \( j \). The feature \( \text{MAX} \) computes the maximum of the values from index \( i \) to index \( j + 1 \).

Example

\((23, (3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\)
Figure 4.1257 provides an example where the SUM_MAX_DECREASING constraint holds.

\[ (23, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 4, 6, 4, 4]) \]

Figure 4.1257: Illustrating the SUM_MAX_DECREASING constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range} (\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
**SUM_MAX_DECREASING**

Figures 4.1258 and 4.1259 respectively depict the automaton associated with the constraint SUM_MAX_DECREASING and its simplified form.

![Automaton Diagram](image1)

**Figure 4.1258:** Automaton for the SUM_MAX_DECREASING constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING pattern where \( \text{default} \) is 0

![Simplified Automaton Diagram](image2)

**Figure 4.1259:** Simplified automaton for the SUM_MAX_DECREASING constraint obtained by applying decoration Table 3.39 to the seed transducer of the DECREASING pattern where \( \text{default} \) is 0

Table 4.281: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the SUM_MAX_DECREASING constraint defined as the composition of the DECREASING pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.282: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the simplified automaton of the `SUM_MAX_DECREASING` constraint defined as the composition of the `DECREASING` pattern, the `MAX` feature, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
</tr>
</tbody>
</table>
SUM_MAX_DECREASING

2705
SUM_MAX_DECREASING_SEQUENCE

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**
SUM_MAX_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE = 0 \lor VALUE \geq \min(minv + 1, (minv + 1) \cdot np) \]
\[ VALUE = 0 \lor VALUE \leq \max(maxv, maxv \cdot np) \]

required(VARIABLES, var)

where
\( sv = |VARIABLES| \)
\( np = \lfloor sv/2 \rfloor \)
\( minv = \text{minval}(VARIABLES.var) \)
\( maxv = \text{maxval}(VARIABLES.var) \)
\( rv = \text{range}(VARIABLES.var) \)

**Purpose**
VALUE is the sum of all maximum values in each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\( > (> | =)^n > | > \)’.

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i \) to index \( j + 1 \).

**Example**
\((16, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\)

Figure 4.1260 provides an example where the SUM_MAX_DECREASING_SEQUENCE (16, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

**Typical**
\(|VARIABLES| > 1 \)
\( \text{range}(VARIABLES.var) > 1 \)

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1260: Illustrating the SUM_MAX_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.1261 and 4.1262 respectively depict the automaton associated with the constraint \texttt{SUM\_MAX\_DECREASING\_SEQUENCE} and its simplified form.

\begin{align*}
\{ C &\leftarrow \text{default} \\
D &\leftarrow -\infty \\
R &\leftarrow \text{default} \}

\begin{tikzpicture}
  \node (A) [state] {$\leq s$};
  \node (B) [state] {$\geq t$};
  \node (C) [state] {$R \leftarrow \text{default}$};
  \node (D) [state] {$\leq$};
  \node (E) [state] {$\geq$};
  \node (F) [state] {$>\}$};
  \node (G) [state] {$=\}$};
  \node (H) [state] {$<\} \{D \leftarrow \text{max}(D, \text{VAR}_{i+1})\}$};
  \node (I) [state] {$\leq\}$};
  \node (J) [state] {$\geq\}$};

  \path[->, >=latex, thick]
    (A) edge node {$\leq$} (D)
    (D) edge node {$\leq$} (A)
    (B) edge node {$\geq$} (E)
    (E) edge node {$\geq$} (B)
    (C) edge node {$\leq\}$} (I)
    (I) edge node {$\leq\}$} (C)
    (D) edge node {$\geq\}$} (J)
    (J) edge node {$\geq\}$} (D)
    (H) edge node {$=\}$} (G)
    (G) edge node {$=\}$} (H)
    (F) edge node {$<\} \{D \leftarrow \text{max}(D, \text{VAR}_{i+1})\}$} (H)
    (H) edge node {$<\} \{D \leftarrow \text{max}(D, \text{VAR}_{i+1})\}$} (F)

\end{tikzpicture}
\end{align*}

Figure 4.1261: Automaton for the \texttt{SUM\_MAX\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{DECREASING\_SEQUENCE} pattern where \texttt{default} is 0

\begin{align*}
\{ R &\leftarrow \text{default} \}

\begin{tikzpicture}
  \node (A) [state] {$\leq s$};
  \node (B) [state] {$\geq t$};
  \node (C) [state] {$R \leftarrow \text{default}$};
  \node (D) [state] {$\leq$};
  \node (E) [state] {$\geq$};
  \node (F) [state] {$>\}$};
  \node (G) [state] {$<\}$};

  \path[->, >=latex, thick]
    (A) edge node {$\leq$} (D)
    (D) edge node {$\leq$} (A)
    (B) edge node {$\geq$} (E)
    (E) edge node {$\geq$} (B)
    (C) edge node {$\leq\}$} (G)
    (G) edge node {$\leq\}$} (C)
    (D) edge node {$\geq\}$} (F)
    (F) edge node {$\geq\}$} (D)
    (A) edge node {$>\}$} (F)
    (F) edge node {$>\}$} (A)

\end{tikzpicture}
\end{align*}

Figure 4.1262: Simplified automaton for the \texttt{SUM\_MAX\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \texttt{DECREASING\_SEQUENCE} pattern where \texttt{default} is 0
Table 4.283: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the SUM_MAX_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

Table 4.284: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the simplified automaton of the SUM_MAX_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_MAX_DIP_ON_INCREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**

\[
\text{SUM_MAX_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES)}
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 5 \lor rv \leq 2 \Rightarrow VALUE = 0 \\
\text{VALUE} &= 0 \lor \text{VALUE} \geq \min(minv + 2, (minv + 2) \times np) \\
\text{VALUE} &= 0 \lor \text{VALUE} \leq \max(maxv, maxv \times np) \\
\end{align*}
\]

where

- **sv** = |VARIABLES|
- **np** = max(0, |(sv − 3)/3|)
- **minv** = minval(VARIABLES.var)
- **maxv** = maxval(VARIABLES.var)
- **rv** = range(VARIABLES.var)

**Purpose**

VALUE is the sum of all maximum values in each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression ‘<<><<’. Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i+2\) to index \(j\).

**Example**

\((11, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))\)

Figure 4.1263 provides an example where the SUM_MAX_DIP_ON_INCREASING_SEQUENCE constraint holds.

**Typical**

| VARIABLES | > 5
---|---
range(VARIABLES.var) | > 2

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.1263: Illustrating the SUM_MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot.
Automaton

Figures 4.1264 and 4.1265 respectively depict the automaton associated with the constraint SUM_MAX_DIP_ON_INCREASING_SEQUENCE and its simplified form.

Figure 4.1264: Automaton for the SUM_MAX_DIP_ON_INCREASINGSEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is 0.
Figure 4.1265: Simplified automaton for the SUM_MAX_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.26 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is 0.
SUM_MAX_INCREASING

DESCRIPTION AUTOMATON

Origin
Based on the INCREASING pattern.

Constraint
SUM_MAX_INCREASING(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var--dvar)
Restrictions

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]

\[ VALUE \geq \min_{q \in \{lb1, ub1\}} \min \]

\[ VALUE \leq \max_{q \in \{lb2, ub2\}} \max \]

required(VARIABLES, var)

where

\[ sv = |VARIABLES| \]

\[ np = |sv/q| \]

\[ maxv = \text{maxval}(\text{VARIABLES}, \text{var}) \]

\[ minv = \text{minval}(\text{VARIABLES}, \text{var}) \]

\[ rv = |\text{range}(\text{VARIABLES}, \text{var})| \]

\[ lb1 = \min \left( \frac{|sv|}{2} + 1, \sum \left( \frac{|sv/\text{min}(sv, rv), |minv| + 1]}{\text{min}(1, sv \mod \text{min}(sv, rv), |minv| + 1)} \right) \right) \]

\[ ub1 = \frac{|sv|}{2} + 1 \]

\[ lb2 = \min \left( \frac{|sv|}{2} + 1, \sum \left( \frac{|sv/\text{min}(sv, rv), |maxv| + 1]}{\text{min}(1, sv \mod \text{min}(sv, rv), |maxv| + 1)} \right) \right) \]

\[ ub2 = \frac{|sv|}{2} + 1 \]

VALUE is the sum of all maximum values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression ‘<’. Assume that the occurrence of the pattern INCREASING starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i \) to index \( j + 1 \).

Example

\[ (21, \{4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3\}) \]
Figure 4.1266 provides an example where the `SUM_MAX_INCREASING` constraint holds.

```
 requisite [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3] ]
```

Figure 4.1266: Illustrating the `SUM_MAX_INCREASING` constraint of the **Example** slot

**Typical**

| \(|VARIABLES| > 1  
| range(VARIABLES.var) > 1 |

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figures 4.1267 and 4.1268 respectively depict the automaton associated with the constraint SUM\_MAX\_INCREASING and its simplified form.

Figure 4.1267: Automaton for the SUM\_MAX\_INCREASING constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING pattern where default is 0.

Figure 4.1268: Simplified automaton for the SUM\_MAX\_INCREASING constraint obtained by applying decoration Table 3.40 to the seed transducer of the INCREASING pattern where default is 0.

Table 4.285: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the SUM\_MAX\_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.286: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the simplified automaton of the SUM_MAX_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**SUM_MAX_INCREASING_SEQUENCE**

**DESCRIPTION**

Origin: Based on the INCREASING_SEQUENCE pattern.

Constraint: SUM_MAX_INCREASING_SEQUENCE(VALUE, VARIABLES)

Arguments:
- VALUE: dvar
- VARIABLES: collection(var−dvar)

Restrictions:
- sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0
- VALUE = 0 ∨ VALUE ≥ min(minv + 1, (minv + 1) * np)
- VALUE = 0 ∨ VALUE ≤ max(maxv, maxv * np)

where
- sv = |VARIABLES|
- np = |sv/2|
- minv = minval(VARIABLES.var)
- maxv = maxval(VARIABLES.var)
- rv = range(VARIABLES.var)

VALUE is the sum of all maximum values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose:
An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘< (< | =)∗ < | <’.
Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i to index j + 1.

Example:
(14, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))

Figure 4.1269 provides an example where the SUM_MAX_INCREASING_SEQUENCE constraint holds.

Typical:
- |VARIABLES| > 1
- range(VARIABLES.var) > 1

Arg. properties:
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1269: Illustrating the **SUM_MAX_INCREASING_SEQUENCE** constraint of the **Example** slot
Automaton

Figures 4.1270 and 4.1271 respectively depict the automaton associated with the constraint `SUM_MAX_INCREASING_SEQUENCE` and its simplified form.

Figure 4.1270: Automaton for the `SUM_MAX_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `INCREASING_SEQUENCE` pattern where `default` is 0.

Figure 4.1271: Simplified automaton for the `SUM_MAX_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.25 to the seed transducer of the `INCREASING_SEQUENCE` pattern where `default` is 0.
Table 4.287: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the SUM_MAX_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

Table 4.288: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the simplified automaton of the SUM_MAX_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_MAX_INFLEXION

DESCRIPTION AUTOMATON

Origin
Based on the INFLEXION pattern.

Constraint
SUM_MAX_INFLEXION(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var–dvar)

Restrictions
\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE \geq \min(minv, minv \cdot \frac{(np + 1)}{2} + (minv + 1) \cdot \lfloor np/2 \rfloor) \]
\[ VALUE = 0 \]
\[ VALUE \leq \max(maxv, maxv \cdot \frac{(np + 1)}{2} + (maxv - 1) \cdot \lfloor np/2 \rfloor) \]

Example
\((31, (1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))\)

Purpose
VALUE is the sum of all maximum values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression ‘\(< (< | =) > | > (> | =) > <\)’.

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

Figure 4.1272 provides an example where the SUM_MAX_INFLEXION \((31, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])\) constraint holds.

Typical
\(|VARIABLES| > 2 \]
\(\text{range}(VARIABLES.var) > 1\)
Figure 4.1272: Illustrating the SUM_MAX_INFLEXION constraint of the Example slot

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figures 4.1273 and 4.1274 respectively depict the automaton associated with the constraint SUM_MAX_INFLEXION and its simplified form.

**Automaton**

Figures 4.1273 and 4.1274 respectively depict the automaton associated with the constraint SUM_MAX_INFLEXION and its simplified form.

**Figure 4.1273:** Automaton for the SUM_MAX_INFLEXION constraint obtained by applying decoration Table 3.37 to the seed transducer of the INFLEXION pattern where default is 0 (transition r → t has the same registers updates as transition t → r)

**Figure 4.1274:** Simplified automaton for the SUM_MAX_INFLEXION constraint obtained by applying decoration Table 3.26 to the seed transducer of the INFLEXION pattern where default is 0 (transition r → t has the same registers updates as transition t → r)
SUM MAX PEAK

**Origin**
Based on the PEAK pattern.

**Constraint**
SUM_MAX_PEAK(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ min(minv + 1, (minv + 1) * np)
VALUE = 0 ∨ VALUE ≤ max(maxv, maxv * np)
required(VARIABLES, var)
where
sv = |VARIABLES|
np = max(0, [(sv − 1)/2])
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
rv = range(VARIABLES.var)

VALUE is the sum of all maximum values in each occurrence of the PEAK pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

**Purpose**
An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression '< (|= <)' (>|=)' >'.
Assume that the occurrence of the pattern PEAK starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 1 to index j.

**Example**
(14, (7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1))

Figure 4.1275 provides an example where the SUM_MAX_PEAK (14, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1]) constraint holds.

**Typical**
|VARIABLES| > 2
range(VARIABLES.var) > 1

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1275: Illustrating the SUM_MAX_PEAK constraint of the Example slot
Figures 4.1276 and 4.1277 respectively depict the automaton associated with the constraint \textsc{sum\_max\_peak} and its simplified form.

Figure 4.1276: Automaton for the \textsc{sum\_max\_peak} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsc{peak} pattern where \texttt{default} is 0

Figure 4.1277: Simplified automaton for the \textsc{sum\_max\_peak} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \textsc{peak} pattern where \texttt{default} is 0
Table 4.289: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the SUM\_MAX\_PEAK constraint defined as the composition of the PEAK pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C}$</td>
<td>$\overrightarrow{C}$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\overrightarrow{C}$</td>
<td>$\max(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1})$</td>
<td>$\overrightarrow{C}$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\overrightarrow{D}$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i+1})$</td>
<td>$\overrightarrow{C}$</td>
</tr>
</tbody>
</table>

Table 4.290: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the simplified automaton of the SUM\_MAX\_PEAK constraint defined as the composition of the PEAK pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$r$</td>
<td>0</td>
<td>\text{VAR}_{i+1}</td>
<td>0</td>
</tr>
<tr>
<td>$t$</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
**SUM_MAX_STRICTLY_DECREASING_SEQUENCE**

**Description**

- **Origin**: Based on the `SUM_MAX_STRICTLY_DECREASING_SEQUENCE` pattern.
- **Constraint**: 
  
  \[
  \text{SUM\_MAX\_STRICTLY\_DECREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
  \]

- **Arguments**
  
  \[
  \text{VALUE} : \text{dvar} \\
  \text{VARIABLES} : \text{collection(\text{var}--\text{dvar})}
  \]

- **Restrictions**
  
  \[
  \begin{align*}
  &sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \\
  &\text{VALUE} = 0 \lor \text{VALUE} \geq \min(\minv + 1, (\minv + 1) \ast np) \\
  &\text{VALUE} = 0 \lor \text{VALUE} \leq \max(\maxv, \maxv \ast np) \\
  
  &\text{required}(\text{VARIABLES}, \text{var})
  \end{align*}
  \]

  where
  
  \[
  \begin{align*}
  &sv = |\text{VARIABLES}| \\
  &np = \lfloor sv/2 \rfloor \\
  &\minv = \text{minval}(\text{VARIABLES}.\text{var}) \\
  &\maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \\
  &rv = \text{range}(\text{VARIABLES}.\text{var})
  \end{align*}
  \]

**Purpose**

- **VALUE** is the sum of all maximum values in each occurrence of the `SUM_MAX_STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, **VALUE** takes the default value 0.

- **An occurrence of the pattern `SUM_MAX_STRICTLY_DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression ‘>’**.

**Purpose**

- Assume that the occurrence of the pattern `SUM_MAX_STRICTLY_DECREASING_SEQUENCE` starts at position \( i \) and ends at position \( j \). The feature **MAX** computes the maximum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
(16,\{4,4,6,4,1,1,3,4,4,6,6,5,2,2,4,3\})
\]

Figure 4.1278 provides an example where the `SUM_MAX_STRICTLY_DECREASING_SEQUENCE` (16,\{4,4,6,4,1,1,3,4,4,6,6,5,2,2,4,3\}) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

- **Functional dependency**: **VALUE** determined by **VARIABLES**.
Figure 4.1278: Illustrating the SUM_MAX_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton: Figures 4.1279 and 4.1280 respectively depict the automaton associated with the constraint SUM_MAX_STRICTLY_DECREASING_SEQUENCE and its simplified form.

Figure 4.1279: Automaton for the SUM_MAX_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is 0

Figure 4.1280: Simplified automaton for the SUM_MAX_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.39 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is 0
Table 4.291: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the SUM_MAX_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>C_+</td>
<td>C_+</td>
</tr>
<tr>
<td>r</td>
<td>C_+</td>
<td>max(C, C, C, D, D)</td>
</tr>
</tbody>
</table>

Table 4.292: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the SUM_MAX_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>C</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
SUM_MAX.Strictly.Increasing.Sequence

**Description**

Based on the `Strictly.Increasing.Sequence` pattern.

**Constraint**

\[ \text{SUM\_MAX\_STRICTLY\_INCREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var-dvar)`

**Restrictions**

\[
\begin{align*}
sv \leq 1 & \lor rv \leq 1 \Rightarrow VALUE = 0 \\
VALUE = 0 & \lor VALUE \geq \min(minv + 1, (minv + 1) \times np) \\
VALUE = 0 & \lor VALUE \leq \max(maxv, maxv \times np) \\
\end{align*}
\]

where

- `sv = |VARIABLES|`
- `np = \lfloor sv/2 \rfloor`
- `minv = \text{minval}(\text{VARIABLES}\text{.var})`
- `maxv = \text{maxval}(\text{VARIABLES}\text{.var})`
- `rv = \text{range}(\text{VARIABLES}\text{.var})`

**Purpose**

`VALUE` is the sum of all maximum values in each occurrence of the `Strictly.Increasing.Sequence` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value 0.

An occurrence of the pattern `Strictly.Increasing.Sequence` is the maximal sub-sequence which matches the regular expression `'<+'`. Assume that the occurrence of the pattern `Strictly.Increasing.Sequence` starts at position `i` and ends at position `j`. The feature `MAX` computes the maximum of the values from index `i` to index `j + 1`.

**Example**

\[(14, \langle 4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3 \rangle)\]

Figure 4.1281 provides an example where the `SUM\_MAX\_STRICTLY\_INCREASING\_SEQUENCE` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `\text{range}(\text{VARIABLES}\text{.var}) > 1`

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 4.1281: Illustrating the \texttt{SUM\_MAX\_STRICKLY\_INCREASING\_SEQUENCE} constraint of the \textbf{Example} slot
Automaton

Figures 4.1282 and 4.1283 respectively depict the automaton associated with the constraint \texttt{SUM\_MAX\_STRICTLY\_INCREASING\_SEQUENCE} and its simplified form.

Figure 4.1282: Automaton for the \texttt{SUM\_MAX\_STRICTLY\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{STRICTLY\_INCREASING\_SEQUENCE} pattern where \texttt{default} is 0.

Figure 4.1283: Simplified automaton for the \texttt{SUM\_MAX\_STRICTLY\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.25 to the seed transducer of the \texttt{STRICTLY\_INCREASING\_SEQUENCE} pattern where \texttt{default} is 0.
Table 4.293: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the `SUM_MAX_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the feature `MAX`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}))</td>
</tr>
</tbody>
</table>

Table 4.294: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the simplified automaton of the `SUM_MAX_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the feature `MAX`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>(\overrightarrow{C})</td>
<td>0</td>
</tr>
</tbody>
</table>
SUM_MAX_SUMMIT

### Description

**Origin**
Based on the SUMMIT pattern.

**Constraint**

\[
\text{SUM\_MAX\_SUMMIT}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var\text{-}dvar)`

**Restrictions**

- \[sv \leq 2 \vee rv \leq 1 \Rightarrow \text{VALUE} = 0\]
- \[\text{VALUE} = 0 \vee \text{VALUE} \geq \min(\text{minv} + 1, (\text{minv} + 1) \cdot \text{np})\]
- \[\text{VALUE} = 0 \vee \text{VALUE} \leq \max(\text{maxv}, \text{maxv} \cdot \text{np})\]

**Purpose**

\[
\text{VALUE} \leq \left\lceil \frac{sv - 1}{2} \right\rceil
\]

VALUE is the sum of all maximum values in each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression ‘\((< | < (= | <)* <) (> | > (= | >)*>)\)’.

Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(12, (7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1))\]

Figure 4.1284 provides an example where the SUM_MAX_SUMMIT (12, [7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1]) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES.var}) > 1\)

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
Figure 4.1284: Illustrating the SUM_MAX_SUMMIT constraint of the Example slot
Figures 4.1285 and 4.1286 respectively depict the automaton associated with the constraint SUM_MAX_SUMMIT and its simplified form.

Figure 4.1285: Automaton for the SUM_MAX_SUMMIT constraint obtained by applying decoration Table 3.37 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
Figure 4.1286: Simplified automaton for the SUM_MAX_SUMMIT constraint obtained by applying decoration Table 3.39 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
<th>$u$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\max(\overrightarrow{d}, \overrightarrow{d}, \overrightarrow{VAR}_{i+1})$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\max(\overrightarrow{c}, \overrightarrow{d}, \overrightarrow{d}, \overrightarrow{VAR}_{i+1})$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\max(\overrightarrow{c}, \overrightarrow{d}, \overrightarrow{d}, \overrightarrow{VAR}_{i+1})$</td>
</tr>
<tr>
<td>$u$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
</tr>
</tbody>
</table>

Table 4.295: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the SUM_MAX_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.296: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the simplified automaton of the **SUM_MAX_SUMMIT** constraint defined as the composition of the **SUMMIT** pattern, the feature **MAX**, and the aggregator **sum**; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>VAR_{i+1}</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>0</td>
<td>L</td>
<td>U</td>
</tr>
<tr>
<td>u</td>
<td>0</td>
<td>0</td>
<td>R</td>
<td>0</td>
</tr>
</tbody>
</table>
SUM_MAX_ZIGZAG

Origin
Based on the ZIGZAG pattern.

Constraint
\( \text{SUM_MAX_ZIGZAG}(\text{VALUE, VARIABLES}) \)

Arguments
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} \rightarrow \text{dvar})
\end{align*}

Restrictions
\begin{align*}
\text{sv} & \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq \min_{q \in [0, \text{qub}]} \min \\
\sum_{0} & \left( \prod_{\minv} \left( \left[ \frac{(\text{sv} - 3 \times q)/4}{\minv + 1} \right], \frac{q \times (\minv + 2)}{\minv + 1} \right), \min \left( \text{maxv} - \text{minv} - 1, 0 \right), \max \left( -1, \min \left( 0, q \times (\minv + 2) \right) \right), \min \left( (\text{sv} - \left[ (\text{sv} - 3 \times q)/4 \right]) \times 4^{\text{mod} 3} \right) \right) \right) \\
\text{rv} & = 2 \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \leq \max(\text{maxv} \times \text{np1}) \\
\text{rv} & \geq 3 \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \leq \max(\text{maxv} \times \text{maxv} \times \text{np1}, \text{maxv} \times \text{np2})
\end{align*}

required(\text{VARIABLES} \rightarrow \text{var})

\text{where}
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES} \rightarrow \text{var}) \\
\text{np1} & = \lfloor \text{sv}/4 \rfloor \\
\text{np2} & = \max(0, \lfloor (\text{sv} - 1)/3 \rfloor) \\
\minv & = \text{minval}(\text{VARIABLES} \rightarrow \text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES} \rightarrow \text{var}) \\
\text{qub} & = \min(1, \max(\text{maxv} - \text{minv} - 1, 0)) \times \text{np2}
\end{align*}

Purpose
\begin{align*}
\text{VALUE} & \text{ is the sum of all maximum values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.} \\
\text{An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression } & '(<>+|<><|><>^+|>)' \\
\text{Assume that the occurrence of the pattern ZIGZAG starts at position } i & \text{ and ends at position } j. \text{ The feature MAX computes the maximum of the values from index } i + 1 \text{ to index } j.
\end{align*}

Example
\begin{align*}
(16, \langle 4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1 \rangle)
\end{align*}
SUM_MAX_ZIGZAG

Figure 4.1287 provides an example where the SUM_MAX_ZIGZAG (16, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1]) constraint holds.

Figure 4.1287: Illustrating the SUM_MAX_ZIGZAG constraint of the Example slot

Typical

| VARIABLES | > 3
range(VARIABLES.var) | > 1

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1288 and 4.1289 respectively depict the automaton associated with the constraint SUM_MAX_ZIGZAG and its simplified form.
Figure 4.1288: Automaton for the SUM_MAX_ZIGZAG constraint obtained by applying
decoration Table 3.37 to the seed transducer of the ZIGZAG pattern where default is 0;
(1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions
from b, c, e, f to s the register D is reset to its initial value; (3) on transitions from c, f to s
the register R is updated wrt C and the register C is reset to its initial value.
Figure 4.1289: Simplified automaton for the SUM_MAX_ZIGZAG constraint obtained by applying decoration Table 3.24 to the seed transducer of the ZIGZAG pattern where 

default is 0; missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the register D is reset to its initial value; (3) on transitions from c, f to s the register R is updated wrt C and the register C is reset to its initial value.
Table 4.297: Concrete glue matrix derived from the parametrised glue matrix 3.22, for the 
\text{SUM}_{MAX}\_ZIGZAG constraint defined as the composition of the \text{ZIGZAG} pattern, 
the \text{MAX} feature and the aggregator \text{sum}; cells of the glue matrix are coloured with 
the \text{MAX} and the aggregator \text{sum} to indicate the constituent to which they are related.

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{s}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
</tr>
<tr>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
</tr>
<tr>
<td>\text{max}(\text{c}, \text{c}, \text{VAR}_{i+1})</td>
<td>\text{max}(\text{c}, \text{c}, \text{VAR}_{i+1})</td>
<td>\text{max}(\text{c}, \text{c}, \text{VAR}_{i+1})</td>
<td>\text{max}(\text{c}, \text{c}, \text{VAR}_{i+1})</td>
<td>\text{max}(\text{c}, \text{c}, \text{VAR}_{i+1})</td>
<td></td>
</tr>
<tr>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
</tr>
<tr>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
<td>\text{s} + \text{c}</td>
</tr>
<tr>
<td>\text{max}(\text{c}, \text{c}, \text{VAR}_{i+1})</td>
<td>\text{max}(\text{c}, \text{c}, \text{VAR}_{i+1})</td>
<td>\text{max}(\text{c}, \text{c}, \text{VAR}_{i+1})</td>
<td>\text{max}(\text{c}, \text{c}, \text{VAR}_{i+1})</td>
<td>\text{max}(\text{c}, \text{c}, \text{VAR}_{i+1})</td>
<td>\text{max}(\text{c}, \text{c}, \text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>
Table 4.298: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the simplified automaton of the SUM_MAX_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_MAX_ZIGZAG

2753
SUM_MIN_BUMP_ON_DECREASING_SEQUENCE

**SUM_MIN_BUMP_ON_DECREASING_SEQUENCE**

**DESCRIPTION**

**Automaton**

**Origin**
Based on the **BUMP_ON_DECREASING_SEQUENCE** pattern.

**Constraint**

\[ \text{SUM_MIN_BUMP_ON_DECREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} \rightarrow \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 5 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & = 0 \lor \text{VALUE} \geq \min(\text{minv}, \text{minv} \times \text{np}) \\
\text{VALUE} & = 0 \lor \text{VALUE} \leq \max(\text{maxv} - 2, (\text{maxv} - 2) \times \text{np}) \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{np} & = \max(0, (|\text{sv} - 3|)/3) \\
\text{minv} & = \text{minval}(\text{VARIABLES}\text{.var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}\text{.var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}\text{.var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern **BUMP_ON_DECREASING_SEQUENCE** is the subsequence which matches the regular expression ‘\(>><>>\)’. Assume that the occurrence of the pattern **BUMP_ON_DECREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i + 2\) to index \(j\).

**Example**

\[(7, (7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3))\]

Figure 4.1290 provides an example where the **SUM_MIN_BUMP_ON_DECREASING_SEQUENCE** \((7, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3])\) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 5 \\
\text{range}(\text{VARIABLES}\text{.var}) & > 2
\end{align*}
\]

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.1290: Illustrating the \texttt{SUM\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint of the Example slot
Automaton

Figures 4.1291 and 4.1292 respectively depict the automaton associated with the constraint summary \textsf{SUM\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE} and its simplified form.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{figure}
\caption{Automaton for the \textsf{SUM\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsf{BUMP\_ON\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0}
\end{figure}
Figure 4.1292: Simplified automaton for the \textsc{sum\_min\_bump\_on\_decreasing\_sequence} constraint obtained by applying decoration Table 3.26 to the seed transducer of the \textsc{bump\_on\_decreasing\_sequence} pattern where default is 0.
SUM_MIN_DECREASING

DESCRIPTION

Constraint

<table>
<thead>
<tr>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUM_MIN_DECREASING(VALUE, VARIABLES)</td>
</tr>
</tbody>
</table>

Arguments

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Value</th>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
<td></td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
<td></td>
</tr>
</tbody>
</table>
Restrictions

\( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \)

\[
VALUE \geq \min_{q \in [lb1, ub1]} \min \left( 0, \sum \left( \prod_{j=1}^{q} \max \left( 0, \min \left( \frac{\left( np - 1 \right) \cdot minv}{np - 1, np - 2} \right), \right) \right) \right)
\]

\[
VALUE \leq \max_{q \in [lb2, ub2]} \max \left( 0, \sum \left( \prod_{j=1}^{q} \max \left( 0, \min \left( 1, 2 - np \right) \right) \right) \right)
\]

\( \text{required(VARIABLES, var)} \)

where

\( sv = |\text{VARIABLES}| \)
\( np = \lfloor sv/q \rfloor \)
\( \text{maxv} = \text{maxval}(\text{VARIABLES, var}) \)
\( \text{minv} = \text{minval}(\text{VARIABLES, var}) \)
\( rv = \text{range}(\text{VARIABLES, var}) \)
\( lb1 = \min \left( \lfloor sv/2 \rfloor + 1, \sum \left( \lfloor sv/\min(\min(sv, rv), |\text{minv}| + 1) \rfloor \right) \right) \)
\( ub1 = \lfloor sv/2 \rfloor + 1 \)
\( lb2 = \min \left( \lfloor sv/2 \rfloor + 1, \sum \left( \lfloor sv/\min(\min(sv, rv), |\text{maxv}| + 1) \rfloor \right) \right) \)
\( ub2 = \lfloor sv/2 \rfloor + 1 \)

VALUE is the sum of all minimum values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression ‘\>|’. Assume that the occurrence of the pattern DECREASING starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i \) to index \( j + 1 \).

Example

\((14, \langle 3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4 \rangle)\)
Figure 4.1293 provides an example where the SUM_MIN_DECREASING constraint holds.

| Typical                     | \(|\text{VARIABLES}\) > 1  |
|-----------------------------|---------------------------|
|                             | range(\text{VARIABLES.var}) > 1 |

**Arg. properties**  
Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1294 and 4.1295 respectively depict the automaton associated with the constraint SUM_MIN_DECREASING and its simplified form.

Figure 4.1294: Automaton for the SUM_MIN_DECREASING constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING pattern where default is 0

Figure 4.1295: Simplified automaton for the SUM_MIN_DECREASING constraint obtained by applying decoration Table 3.40 to the seed transducer of the DECREASING pattern where default is 0

Table 4.299: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the SUM_MIN_DECREASING constraint defined as the composition of the DECREASING pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.300: Concrete glue matrix, derived from the parametrised glue matrix $3.4$, for the simplified automaton of the SUM_MIN_DECREASING constraint defined as the composition of the DECREASING pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_MIN_DECREASING

2763
Origin | Based on the DECREASING_SEQUENCE pattern.
---|---
Constraint | SUM_MIN_DECREASING_SEQUENCE(VALUE, VARIABLES)
Arguments | VALUE: dvar
| VARIABLES: collection(var−dvar)
Restrictions | \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
| \( VALUE = 0 \lor VALUE \geq \min(minv, minv \ast np) \)
| \( VALUE = 0 \lor VALUE \leq \max(maxv − 1\ast, (maxv − 1) \ast np) \)
\[\text{required}(\text{VARIABLES}, \text{var})\]
where
| \( sv = |\text{VARIABLES}| \)
| \( np = \lfloor sv/2 \rfloor \)
| \( minv = \text{minval}(\text{VARIABLES}, \text{var}) \)
| \( maxv = \text{maxval}(\text{VARIABLES}, \text{var}) \)
| \( rv = \text{range}(\text{VARIABLES}, \text{var}) \)

VALUE is the sum of all minimum values in each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose | An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\(> (> | =) \ast > | >\)’. Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

Example | \((7, 3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4)\)

Figure 4.1296 provides an example where the SUM_MIN_DECREASING_SEQUENCE \((7, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\) constraint holds.

Typical | \(|\text{VARIABLES}| > 1 \)
| range(\text{VARIABLES}, \text{var}) > 1 \)

Arg. properties | Functional dependency: VALUE determined by VARIABLES.
Figure 4.1296: Illustrating the SUM_MIN_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.1297 and 4.1298 respectively depict the automaton associated with the constraint \textsc{sum\_min\_decreasing\_sequence} and its simplified form.

Figure 4.1297: Automaton for the \textsc{sum\_min\_decreasing\_sequence} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsc{decreasing\_sequence} pattern where \texttt{default} is 0

Figure 4.1298: Simplified automaton for the \textsc{sum\_min\_decreasing\_sequence} constraint obtained by applying decoration Table 3.25 to the seed transducer of the \textsc{decreasing\_sequence} pattern where \texttt{default} is 0
Table 4.301: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the SUM_MIN_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>t</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\min(\overline{C}, \overline{C}, \overline{D}, \overline{D})$</td>
</tr>
</tbody>
</table>

Table 4.302: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the simplified automaton of the SUM_MIN_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>t</td>
<td>$\overline{C}$</td>
<td>0</td>
</tr>
</tbody>
</table>
SUM_MIN_DIP_ON_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
</table>

**Origin**
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**
SUM_MIN_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
- VALUE: dvar
- VARIABLES: collection(var−dvar)

**Restrictions**
- \( sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = 0 \)
- \( VALUE = 0 \lor VALUE \geq \min(minv \cdot \text{np}) \)
- \( VALUE = 0 \lor VALUE \leq \max(maxv - 2, (maxv - 2) \cdot \text{np}) \)

**Purpose**
VALUE is the sum of all minimum values in each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression ‘<<><<’.
Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 2 \) to index \( j \).

**Example**
\( (3, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4)) \)

Figure 4.1299 provides an example where the SUM_MIN_DIP_ON_INCREASING_SEQUENCE (3, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 5 \)
- \( \text{range} \text{(VARIABLES.var)} > 2 \)

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1299: Illustrating the SUM_MIN_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Figures 4.1300 and 4.1301 respectively depict the automaton associated with the constraint SUM_MIN_DIP_ON_INCREASING_SEQUENCE and its simplified form.

Figure 4.1300: Automaton for the SUM_MIN_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is 0.
Figure 4.1301: Simplified automaton for the \texttt{SUM\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.28 to the seed transducer of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern where default is 0.
Origin
Based on the GORGE pattern.

Constraint
\[ \text{SUM\_MIN\_GORGE}(\text{VALUE, VARIABLES}) \]

Arguments
\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(\text{var}–\text{dvar})}
\end{align*}
\]

Restrictions
\[
\begin{align*}
\text{sv} \leq 2 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 & \lor \text{VALUE} \geq \min(\text{minv}\times, \text{minv} \times \text{np}\times) \\
\text{VALUE} = 0 & \lor \text{VALUE} \leq \max(\text{maxv} - 1\times, (\text{maxv} - 1) \times \text{np}\times) \\
\text{required}(\text{VARIABLES, var}) & \\
\text{where} & \\
\text{sv} & = |\text{VARIABLES}| \\
\text{np} & = \max(0, \lfloor (\text{sv} - 1)/2 \rfloor) \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Purpose
An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression '\( (> | > (= | >)^* ) < | < (= | <)^* < \)'.
Assume that the occurrence of the pattern GORGE starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

Example
\( (12, (1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7)) \)

Figure 4.1302 provides an example where the SUM\_MIN\_GORGE (12, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7]) constraint holds.

Typical
\[
\begin{align*}
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\end{align*}
\]

Symmetry
Items of VARIABLES can be reversed.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1302: Illustrating the SUM_MIN_GORGE constraint of the Example slot
Automaton

Figures 4.1303 and 4.1304 respectively depict the automaton associated with the constraint SUM_MIN_GORGE and its simplified form.

Figure 4.1303: Automaton for the SUM_MIN_GORGE constraint obtained by applying decoration Table 3.37 to the seed transducer of the GORGE pattern where default is 0 (transition u \to r has the same register update as transition r \to u)
Figure 4.1304: Simplified automaton for the SUM_MIN_GORGE constraint obtained by applying decoration Table 3.39 to the seed transducer of the GORGE pattern where default is 0

Table 4.303: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the SUM_MIN_GORGE constraint defined as the composition of the GORGE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.304: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the simplified automaton of the \textit{SUM\_MIN\_GORGE} constraint defined as the composition of the \textit{GORGE} pattern, the feature \textit{MIN}, and the aggregator \textit{sum}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_MIN_INCREASING

DESCRIPTION

Origin
Based on the INCREASING pattern.

Constraint
SUM_MIN_INCREASING(VALUE, VARIABLES)

Arguments
- VALUE : dvar
- VARIABLES : collection(var–dvar)
Restrictions
\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[
VALUE \geq \min_{q \in [lb1, ub1]} \min \left\{ \begin{array}{l}
0, \\
\sum q \cdot \left( \prod \left( \left[ \frac{(np - 1) \cdot \minv, np - 1, np - 2}{2} \right] \right) \right), \\
\max \left( 0, \min \left( 1, np - 1 \right) \right), \\
\minv + np - 1 \\
\max \left( 0, \min \left( 1, 2 - np \right) \right), \\
\minv. \\
q - 1
\end{array} \right. \\
\right\}
\]
\[
VALUE \leq \max_{q \in [lb2, ub2]} \max \left\{ \begin{array}{l}
0, \\
\sum q \cdot \left( \prod \left( \left[ \frac{(np - 1) \cdot \maxv - 1, np - 1, np - 2}{2} \right] \right) \right), \\
\max \left( 0, \min \left( 1, np - 1 \right) \right), \\
\maxv - np + 1 \\
\max \left( 0, \min \left( 1, 2 - np \right) \right), \\
\maxv - 1, \\
q - 1
\end{array} \right. \\
\right\}
\]
\[
\text{required(VARIABLES, var)}
\]
where
\[ sv = |VARIABLES| \]
\[ np = \left\lfloor \frac{sv}{q} \right\rfloor \]
\[ maxv = \maxval(VARIABLES, var) \]
\[ minv = \minval(VARIABLES, var) \]
\[ rv = \text{range}(VARIABLES, var) \]
\[ lb1 = \min \left( \left\lfloor \frac{sv}{2} \right\rfloor + 1, \left\lfloor \frac{sv}{\min(\min(sv, \minv) + 1)} \right\rfloor \right) \]
\[ ub1 = \left\lceil \frac{sv}{2} \right\rceil + 1 \]
\[ lb2 = \min \left( \left\lfloor \frac{sv}{2} \right\rfloor + 1, \left\lfloor \frac{sv}{\min(\min(sv, \maxv) + 1)} \right\rfloor \right) \]
\[ ub2 = \left\lceil \frac{sv}{2} \right\rceil + 1 \]

VALUE is the sum of all minimum values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression ‘<’.

Assume that the occurrence of the pattern INCREASING starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i \) to index \( j + 1 \).

Example
\[ (12, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) \]
Figure 4.1305 provides an example where the SUM_MIN_INCREASING constraint holds.

![Diagram showing feature values and increasing order for variables V1 to V16]

Figure 4.1305: Illustrating the SUM_MIN_INCREASING constraint of the Example slot

**Typical**

| | VARIABLES | > 1
-- | -- | --

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
Figures 4.1306 and 4.1307 respectively depict the automaton associated with the constraint \( \text{SUM}\_\text{MIN}\_\text{INCREASING} \) and its simplified form.

\[
\begin{align*}
&\{ C \leftarrow \text{default} \} \\
&\{ D \leftarrow +\infty \} \\
&\{ R \leftarrow \text{default} \} \\
\end{align*}
\]

\[
R + C \geq < \{ D \leftarrow +\infty \} \\
\begin{array}{c}
R \leftarrow R + \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
\end{array}
\]

Figure 4.1306: Automaton for the \( \text{SUM}\_\text{MIN}\_\text{INCREASING} \) constraint obtained by applying decoration Table 3.37 to the seed transducer of the \( \text{INCREASING} \) pattern where \( \text{default} \) is 0

\[
\begin{align*}
&\{ R \leftarrow \text{default} \} \\
\end{align*}
\]

\[
\leq \{ R \leftarrow R + \text{VAR}_i \} \\
\geq
\]

Figure 4.1307: Simplified automaton for the \( \text{SUM}\_\text{MIN}\_\text{INCREASING} \) constraint obtained by applying decoration Table 3.39 to the seed transducer of the \( \text{INCREASING} \) pattern where \( \text{default} \) is 0

Table 4.305: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the \( \text{SUM}\_\text{MIN}\_\text{INCREASING} \) constraint defined as the composition of the \( \text{INCREASING} \) pattern, the feature \( \text{MIN} \), and the aggregator \( \text{sum} \); cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.306: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the simplified automaton of the SUM_MIN_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_MIN_INCREASING  2783
**SUM_MIN_INCREASING_SEQUENCE**

**Description**
- **Origin**: Based on the **INCREASING_SEQUENCE** pattern.
- **Constraint**: `SUM_MIN_INCREASING_SEQUENCE(VALUE, VARIABLES)`
- **Arguments**:
  - `VALUE`: `dvar`
  - `VARIABLES`: `collection(var--dvar)`

**Restrictions**
- `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0`
- `VALUE = 0 ∨ VALUE ≥ min(minv, minv * np)`
- `VALUE = 0 ∨ VALUE ≤ max(maxv − 1, (maxv − 1) * np)`
- `required(VARIABLES, var)`

**Purpose**
- **VALUE** is the sum of all minimum values in each occurrence of the **INCREASING_SEQUENCE** pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, **VALUE** takes the default value 0.  
- An occurrence of the pattern **INCREASING_SEQUENCE** is the maximal subsequence which matches the regular expression `'(< | =)* < |'`.  
- Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position `i` and ends at position `j`. The feature **MIN** computes the minimum of the values from index `i` to index `j + 1`.

**Example**
- `(5, 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)`

**Typical**
- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Arg. properties**
- Functional dependency: **VALUE** determined by `VARIABLES`.

Figure 4.1308 provides an example where the `SUM_MIN_INCREASING_SEQUENCE (5, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])` constraint holds.
Figure 4.1308: Illustrating the SUM_MIN_INCREASING_SEQUENCE constraint of the Example slot
Figures 4.1309 and 4.1310 respectively depict the automaton associated with the constraint SUM_MIN_INCREASING_SEQUENCE and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow +\infty \\
& R \leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
\geq s & > \\
\{ & C \leftarrow \text{default} \\
& D \leftarrow +\infty \\
& R \leftarrow R + C \}
\end{align*}
\]

\[
\begin{align*}
& \{ D \leftarrow \min(D, \text{VAR}_i+1) \}
\end{align*}
\]

\[
\begin{align*}
\leq t & < \\
& \{ & C \leftarrow \min(C, \min(D, \text{VAR}_i+1)) \\
& D \leftarrow +\infty \}
\end{align*}
\]

Figure 4.1309: Automaton for the SUM_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0

\[
\begin{align*}
\{ & R \leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
\geq s & > \\
\{ & R \leftarrow R + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
& \leq t < \\
& =
\end{align*}
\]

Figure 4.1310: Simplified automaton for the SUM_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.39 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0
Table 4.307: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the SUM\_MIN\_INCREASING\_SEQUENCE constraint defined as the composition of the INCREASING\_SEQUENCE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>min($\overline{C}, \overline{C}, \overline{D}, \overline{D}$)</td>
</tr>
</tbody>
</table>

Table 4.308: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the simplified automaton of the SUM\_MIN\_INCREASING\_SEQUENCE constraint defined as the composition of the INCREASING\_SEQUENCE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>0</td>
<td>$\overline{C}$</td>
</tr>
<tr>
<td>$t$</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Origin

Based on the INFLEXION pattern.

Constraint

\[ \text{SUM\_MIN\_INFLEXION}(\text{VALUE, VARIABLES}) \]

Arguments

\[
\text{VALUE} : \ dvar \\
\text{VARIABLES} : \ \text{collection}(\text{var} - \text{dvar})
\]

Restrictions

\[
\begin{align*}
sv \leq 2 \lor rv \leq 1 &\implies \text{VALUE} = 0 \\
\lor & \quad (\text{VALUE} \geq \min(\text{minv}, \text{minv} \ast \lceil (\text{np} + 1)/2 \rceil + (\text{minv} + 1) \ast \lfloor \text{np}/2 \rfloor)) \\
\lor & \quad (\text{VALUE} \leq \max(\text{maxv}, \text{maxv} \ast \lceil (\text{np} + 1)/2 \rceil + (\text{maxv} - 1) \ast \lfloor \text{np}/2 \rfloor))
\end{align*}
\]

\[
\text{required}(\text{VARIABLES, var})
\]

where

\[
\begin{align*}
sv &= |\text{VARIABLES}| \\
np &= \max(0, sv - 2) \\
\text{minv} &= \text{minval}(\text{VARIABLES, var}) \\
\text{maxv} &= \text{maxval}(\text{VARIABLES, var}) \\
rv &= \text{range}(\text{VARIABLES, var})
\end{align*}
\]

Purpose

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression ‘\(< (\mid =)\ast > | > (\mid =)\ast <\)’. Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

Example

\((26, \{1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4\})\)

Figure 4.1311 provides an example where the \text{SUM\_MIN\_INFLEXION} \((26, \{1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4\})\) constraint holds.

Typical

\[
\begin{align*}
|\text{VARIABLES}| &> 2 \\
\text{range}(\text{VARIABLES, var}) &> 1
\end{align*}
\]
Figure 4.1311: Illustrating the SUM_MIN_INFLEXION constraint of the Example slot

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figures 4.1312 and 4.1313 respectively depict the automaton associated with the constraint 
SUM_MIN_INFLEXION and its simplified form.

Figure 4.1312: Automaton for the SUM_MIN_INFLEXION constraint obtained by applying decoration Table 3.37 to the seed transducer of the INFLEXION pattern where default is 0 (transition \( r \to t \) has the same registers updates as transition \( t \to r \)).

Figure 4.1313: Simplified automaton for the SUM_MIN_INFLEXION constraint obtained by applying decoration Table 3.26 to the seed transducer of the INFLEXION pattern where default is 0 (transition \( r \to t \) has the same registers updates as transition \( t \to r \)).
SUM_MIN_INFLEXION
SUM_MIN_STRICTLY_DECREASING_SEQUENCE

### Origin
Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

### Constraint
`SUM_MIN_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)`

### Arguments
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

### Restrictions
- `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0`
- `VALUE = 0 ∨ VALUE ≥ min(minv, minv * np)`
- `VALUE = 0 ∨ VALUE ≤ max(maxv − 1, (maxv − 1) * np)`

#### required(VARIABLES, var)

- `sv = |VARIABLES|`
- `np = [sv/2]`
- `minv = minval(VARIABLES.var)`
- `maxv = maxval(VARIABLES.var)`
- `rv = range(VARIABLES.var)`

**VALUE** is the sum of all minimum values in each occurrence of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, **VALUE** takes the default value 0.

### Purpose
An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `^>`.

Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

### Example

```
(6, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))
```

Figure 4.1314 provides an example where the `SUM_MIN_STRICTLY_DECREASING_SEQUENCE (6, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3])` constraint holds.

### Typical
- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

### Arg. properties
- Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.1314: Illustrating the `SUM_MIN_STRICTLY_DECREASING_SEQUENCE` constraint of the Example slot
Automaton

Figures 4.1315 and 4.1316 respectively depict the automaton associated with the constraint \( \text{SUM\_MIN\_STRICTLY\_DECREASING\_SEQUENCE} \) and its simplified form.

Figure 4.1315: Automaton for the \( \text{SUM\_MIN\_STRICTLY\_DECREASING\_SEQUENCE} \) constraint obtained by applying decoration Table 3.37 to the seed transducer of the \( \text{STRICTLY\_DECREASING\_SEQUENCE} \) pattern where \text{default} is 0.

Figure 4.1316: Simplified automaton for the \( \text{SUM\_MIN\_STRICTLY\_DECREASING\_SEQUENCE} \) constraint obtained by applying decoration Table 3.25 to the seed transducer of the \( \text{STRICTLY\_DECREASING\_SEQUENCE} \) pattern where \text{default} is 0.
Table 4.309: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the SUM_MIN_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}))</td>
</tr>
</tbody>
</table>

Table 4.310: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the SUM_MIN_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>(\overrightarrow{C})</td>
<td>0</td>
</tr>
</tbody>
</table>
**Origin** Based on the `SUM_MIN_STRICTLY_INCREASING_SEQUENCE` pattern.

**Constraint**

```
SUM_MIN_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**

- `VALUE`: `dvar`
- `VARIABLES`: `collection(var - dvar)`

**Restrictions**

- `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0`
- `VALUE = 0 ∨ VALUE ≥ min(minv, minv * np)`
- `VALUE = 0 ∨ VALUE ≤ max(maxv - 1, (maxv - 1) * np)`

where:

- `sv = |VARIABLES|`
- `np = |sv/2|`
- `minv = minval(VARIABLES.var)`
- `maxv = maxval(VARIABLES.var)`
- `rv = range(VARIABLES.var)`

**Purpose**

VALUE is the sum of all minimum values in each occurrence of the `SUM_MIN_STRICTLY_INCREASING_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern `SUM_MIN_STRICTLY_INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `<+`. Assume that the occurrence of the pattern `SUM_MIN_STRICTLY_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

```
(5, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))
```

Figure 4.1317 provides an example where the `SUM_MIN_STRICTLY_INCREASING_SEQUENCE (5, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3])` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.1317: Illustrating the **SUM_MIN_STRICTLY_INCREASING_SEQUENCE** constraint of the **Example** slot
Automaton

Figures 4.1318 and 4.1319 respectively depict the automaton associated with the constraint \( \text{SUM\_MIN\_STRICTLY\_INCREASING\_SEQUENCE} \) and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow \text{default}
\end{align*}
\]

Figure 4.1318: Automaton for the \( \text{SUM\_MIN\_STRICTLY\_INCREASING\_SEQUENCE} \) constraint obtained by applying decoration Table 3.37 to the seed transducer of the \( \text{STRICTLY\_INCREASING\_SEQUENCE} \) pattern where \( \text{default} \) is 0.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow R + C
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow \min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D &\leftarrow +\infty
\end{align*}
\]

Figure 4.1319: Simplified automaton for the \( \text{SUM\_MIN\_STRICTLY\_INCREASING\_SEQUENCE} \) constraint obtained by applying decoration Table 3.39 to the seed transducer of the \( \text{STRICTLY\_INCREASING\_SEQUENCE} \) pattern where \( \text{default} \) is 0.
Table 4.311: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the SUM_MIN_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\bar{C} + \bar{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\bar{C} + \bar{C}$</td>
</tr>
</tbody>
</table>

Table 4.312: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the simplified automaton of the SUM_MIN_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
</tr>
</tbody>
</table>
SUM_MIN_VALLEY

Description

Origin

Based on the VALLEY pattern.

Constraint

SUM_MIN_VALLEY(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ min(minvV, minv∗np)*
VALUE = 0 ∨ VALUE ≤ max(maxv−1, (maxv−1)∗np)*
required(VARIABLES, var)

where

sv = |VARIABLES|
np = max(0, (sv−1)/2)
minv =minval(VARIABLES.var)
maxv =maxval(VARIABLES.var)
rv =range(VARIABLES.var)

Purpose

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression ’>(= | >)*(< | =)*’.<
Assume that the occurrence of the pattern VALLEY starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

Example

(10, ⟨1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7⟩)

Figure 4.1320 provides an example where the SUM_MIN_VALLEY (10, [1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7]) constraint holds.

Typical

|VARIABLES| > 2
range(VARIABLES.var) > 1

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.1320: Illustrating the SUM\_MIN\_VALLEY constraint of the Example slot
Figures 4.1321 and 4.1322 respectively depict the automaton associated with the constraint \textsc{SUM\_MIN\_VALLEY} and its simplified form.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{figure4_1321.png}
\caption{Automaton for the \textsc{SUM\_MIN\_VALLEY} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsc{VALLEY} pattern where \texttt{default} is 0.}
\end{figure}

\begin{figure}
\centering
\includegraphics[width=\textwidth]{figure4_1322.png}
\caption{Simplified automaton for the \textsc{SUM\_MIN\_VALLEY} constraint obtained by applying decoration Table 3.39 to the seed transducer of the \textsc{VALLEY} pattern where \texttt{default} is 0.}
\end{figure}
Table 4.313: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the SUM_MIN_VALLEY constraint defined as the composition of the VALLEY pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

Table 4.314: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the simplified automaton of the SUM_MIN_VALLEY constraint defined as the composition of the VALLEY pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_MIN_ZIGZAG

DESCRIPTION

Origin

Based on the ZIGZAG pattern.

Constraint

\[ \text{SUM_MIN_ZIGZAG}(\text{VALUE}, \text{VARIABLES}) \]

Arguments

\[ \begin{align*}
\text{VALUE} &: \text{ dvar} \\
\text{VARIABLES} &: \text{ collection(var\text{-}dvar)} 
\end{align*} \]

Restrictions

\[ \begin{align*}
\text{sv} &\leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{rv} &\leq 2 \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \geq \min(\text{minv},\text{minv} \ast \text{np1}) \\
\text{rv} &\geq 3 \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \geq \min(\min(\text{minv},\text{minv} \ast \text{np1},\text{minv} \ast \text{np2})) \\
\end{align*} \]

\[ \text{VALUE} \leq \max_{q \in [0, \text{qub}]} \max \left( 0, \sum_{q} \prod_{0}^{\left\lceil \frac{\text{sv} - 3 \ast q}{4} \right\rceil} \left( \prod_{0}^{\min(\maxv - \text{minv} - 1, 0)} \right), \min(1, \max(0, \text{qub} - 2)), \min(1, \max(0, \text{qub} - 2)), \text{minv} \right) \right) \]

required(\text{VARIABLES}.\text{var})

where

\[ \begin{align*}
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var}) \\
\text{np1} &= \left\lceil \frac{\text{sv}}{4} \right\rceil \\
\text{np2} &= \max(0, |\text{sv} - 1|/3) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{qub} &= \min(1, \max(\maxv - \text{minv} - 1, 0)) \ast \text{np2} 
\end{align*} \]

VALUE is the sum of all minimum values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \( ^{\ast}(<>)^{\ast}(<>)(<>)(<>)(<>)^{\ast} \).

Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature \text{MIN} computes the minimum of the values from index \( i + 1 \) to index \( j \).

Example

\[ (3, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1)) \]
Figure 4.1323 provides an example where the \texttt{SUM\_MIN\_ZIGZAG}\((3, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1])\) constraint holds.

![Graph illustrating the \texttt{SUM\_MIN\_ZIGZAG} constraint](image)

Figure 4.1323: Illustrating the \texttt{SUM\_MIN\_ZIGZAG} constraint of the \textbf{Example} slot

**Typical**

\[
|VARIABLES| > 3 \\
\text{range}(VARIABLES.var) > 1
\]

**Symmetry**

Items of \texttt{VARIABLES} can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by \texttt{VARIABLES}.
Automaton

Figures 4.1324 and 4.1325 respectively depict the automaton associated with the constraint `SUM_MIN_ZIGZAG` and its simplified form.
Figure 4.1324: Automaton for the SUM_MIN_ZIGZAG constraint obtained by applying decoration Table 3.37 to the seed transducer of the ZIGZAG pattern where default is 0; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the register D is reset to its initial value; (3) on transitions from c, f to s the register R is updated wrt C and the register C is reset to its initial value.
Figure 4.1325: Simplified automaton for the SUM_MIN_ZIGZAG constraint obtained by applying decoration Table 3.24 to the seed transducer of the ZIGZAG pattern where default is 0; missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the register D is reset to its initial value; (3) on transitions from c, f to s the register R is updated wrt C and the register C is reset to its initial value.
Table 4.315: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the SUM-MIN-ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.316: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the simplified automaton of the SUM_MIN_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**SUM_RANGE_DECREASING**

**DESCRIPTION**

**Origin**
Based on the **DECREASING** pattern.

**Constraint**

\[
\text{SUM\_RANGE\_DECREASING}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE** : `dvar`
- **VARIABLES** : `collection(var−dvar)`

**Restrictions**

\[
\begin{align*}
sv \leq 1 \vee rv \leq 1 & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \left\lfloor \frac{sv}{2} \right\rfloor \times (rv - 1)_0 \\
\end{align*}
\]

where

- \(rv = \text{range}(\text{VARIABLES}.\text{var})\)
- \(sv = |\text{VARIABLES}|\)

**Purpose**

\(\text{VALUE}\) is the sum of the differences between the largest and smallest value in each occurrence of the **DECREASING** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, \(\text{VALUE}\) takes the default value 0.

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression `'>'.

Assume that the occurrence of the pattern **DECREASING** starts at position \(i\) and ends at position \(j\). The feature **RANGE** computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\[(9, (3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 4.1326 provides an example where the **SUM\_RANGE\_DECREASING** \((9, [3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**

One and the same constant can be added to the `var` attribute of all items of **VARIABLES**.

**Arg. properties**

Functional dependency: \(\text{VALUE}\) determined by **VARIABLES**.
Figure 4.1326: Illustrating the `SUM_RANGE_DECREASING` constraint of the Example slot
Automaton

Figures 4.1327 and 4.1328 respectively depict the automaton associated with the constraint `SUM_RANGE_DECREASING` and its simplified form.

**Figure 4.1327:** Automaton for the `SUM_RANGE_DECREASING` constraint obtained by applying decoration Table 3.48 to the seed transducer of the `DECREASING` pattern where `default` is 0

**Figure 4.1328:** Simplified automaton for the `SUM_RANGE_DECREASING` constraint obtained by applying decoration Table 3.46 to the seed transducer of the `DECREASING` pattern where `default` is 0
SUM_RANGE_DECREASING

2815
**SUM_RANGE_DECREASING SEQUENCE**

- **Origin**: Based on the **DECREASING SEQUENCE** pattern.

- **Constraint**: 
  \[ \text{SUM\_RANGE\_DECREASING\_SEQUENCE} (\text{VALUE}, \text{VARIABLES}) \]

- **Arguments**:
  - **VALUE**: `dvar`
  - **VARIABLES**: `collection(var\_dvar)`

- **Restrictions**:
  - \( sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \)
  - \( \text{VALUE} \geq 0 \)
  - \( \text{VALUE} \leq \lfloor sv/2 \rfloor \times (rv - 1) \)
  - \( \text{required(VARIABLES, var)} \)
  - \( rv = \text{range(VARIABLES, var)} \)
  - \( sv = |\text{VARIABLES}| \)

- **Purpose**: 
  - **VALUE** is the sum of the differences between the largest and smallest value in each occurrence of the **DECREASING SEQUENCE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value 0.
  - An occurrence of the pattern **DECREASING SEQUENCE** is the maximal subsequence which matches the regular expression `\( > (>|=|^)* > | > \)`.
  - Assume that the occurrence of the pattern **DECREASING SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **RANGE** computes the range of the values from index \( i \) to index \( j + 1 \).

- **Example**: 
  \( (9, (3, 4, 2, 5, 6, 4, 4, 3, 1, 4, 6, 4, 4)) \)

- **Typical**:
  - \(|\text{VARIABLES}| > 1 \)
  - \( \text{range(VARIABLES, var)} > 1 \)

- **Symmetry**: One and the same constant can be added to the **var** attribute of all items of **VARIABLES**.

- **Arg. properties**: Functional dependency: **VALUE** determined by **VARIABLES**.

---

**Figure 4.1329** provides an example where the **SUM\_RANGE\_DECREASING\_SEQUENCE** \((9, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\) constraint holds.
Figure 4.1329: Illustrating the SUM_RANGE_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.1330 and 4.1331 respectively depict the automaton associated with the constraint SUM_RANGE_DECREASING_SEQUENCE and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
H &\leftarrow \text{VAR}_1 \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
\leq s &\quad \leftarrow \quad \frac{H}{\leq \text{VAR}_{i+1}} \\
\{ C &\leftarrow \text{default} \\
\{ H &\leftarrow \text{VAR}_{i+1} \\
\{ R &\leftarrow R + C
\end{align*}
\]

\[
\begin{align*}
\leq &\quad \frac{1}{\leq} \\
\{ C &\leftarrow \text{default} \\
\{ H &\leftarrow \text{VAR}_{i+1} \\
\{ R &\leftarrow R + C
\end{align*}
\]

Figure 4.1330: Automaton for the SUM_RANGE_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.48 to the seed transducer of the DECREASING_SEQUENCE pattern where default is 0

\[
\begin{align*}
\{ R &\leftarrow \text{default} \\
\leq &\quad \frac{1}{\leq} \\
\{ R &\leftarrow R + \max(0, \text{VAR}_i - \text{VAR}_{i+1})
\end{align*}
\]

\[
\begin{align*}
\leq &\quad \frac{1}{\leq} \\
\{ R &\leftarrow R + \max(0, \text{VAR}_i - \text{VAR}_{i+1})
\end{align*}
\]

Figure 4.1331: Simplified automaton for the SUM_RANGE_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.46 to the seed transducer of the DECREASING_SEQUENCE pattern where default is 0
SUM_RANGE_DECREASING_SEQUENCE

2819
**SUM_RANGE_INCREMENTING**

**DESCRIPTION**

**Origin**
Based on the INCREASING pattern.

**Constraint**
SUM_RANGE_INCREMENTING(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var–dvar)

**Restrictions**
- $sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0$
- $VALUE \geq 0$
- $VALUE \leq \lfloor sv/2 \rfloor \times (rv - 1)$
- required(VARIABLES, var)

where
- $rv = \text{range}(VARIABLES.var)$
- $sv = |\text{VARIABLES}|$

**Purpose**
VALUE is the sum of the differences between the largest and smallest value in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression '\<'. Assume that the occurrence of the pattern INCREASING starts at position $i$ and ends at position $j$. The feature RANGE computes the range of the values from index $i$ to index $j + 1$.

**Example**
(9, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))

Figure 4.1332 provides an example where the SUM_RANGE_INCREMENTING (9, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

**Typical**
- $|\text{VARIABLES}| > 1$
- $\text{range}(VARIABLES.var) > 1$

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1332: Illustrating the SUM\_RANGE\_INCREASING constraint of the Example slot
Automaton

Figures 4.1333 and 4.1334 respectively depict the automaton associated with the constraint SUM\_RANGE\_INCREASING and its simplified form.

\[
\begin{cases}
C \leftarrow \text{default} \\
H \leftarrow \text{VAR}_1 \\
R \leftarrow \text{default}
\end{cases}
\]

\[
\begin{cases}
H \leftarrow \text{VAR}_{i+1} \\
R \leftarrow R + |H - \text{VAR}_{i+1}|
\end{cases}
\]

Figure 4.1333: Automaton for the SUM\_RANGE\_INCREASING constraint obtained by applying decoration Table 3.48 to the seed transducer of the INCREASING pattern where default is 0.

\[
\begin{cases}
R \leftarrow \text{default}
\end{cases}
\]

\[
\begin{cases}
R \leftarrow R + \max(0, \text{VAR}_{i+1} - \text{VAR}_i)
\end{cases}
\]

Figure 4.1334: Simplified automaton for the SUM\_RANGE\_INCREASING constraint obtained by applying decoration Table 3.47 to the seed transducer of the INCREASING pattern where default is 0; \( R_i - R_{i-2} - X_i + X_{i-2} \geq 0 \) and \( R_i - R_{i-1} - X_i + X_{i-1} \geq 0 \) are linear invariants.
### SUM RANGE INCREASING SEQUENCE

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Feature</th>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUM RANGE INCREASING SEQUENCE</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the INCREASING SEQUENCE pattern.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Constraint</th>
<th>SUM RANGE INCREASING SEQUENCE(VALUE, VARIABLES)</th>
</tr>
</thead>
</table>

| Arguments  | VALUE : dvar  
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES : collection(var−dvar)</td>
<td></td>
</tr>
</tbody>
</table>

| Restrictions | sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0  
|--------------|-----------------------------|
| VALUE ≥ 0  
| VALUE ≤ [sv/2] * (rv − 1) ≤ required(VARIABLES, var)  
| where  
| rv = range(VARIABLES.var)  
| sv = |VARIABLES| |

**VALUE** is the sum of the differences between the largest and smallest value in each occurrence of the INCREASING SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, **VALUE** takes the default value 0.

An occurrence of the pattern **INCREASING SEQUENCE** is the maximal subsequence which matches the regular expression ‘< (< | =)* < | ’. Assume that the occurrence of the pattern **INCREASING SEQUENCE** starts at position i and ends at position j. The feature **RANGE** computes the range of the values from index i to index j + 1.

<table>
<thead>
<tr>
<th>Purpose</th>
<th></th>
</tr>
</thead>
</table>

**Example**

(9, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))

Figure 4.1335 provides an example where the SUM RANGE INCREASING SEQUENCE (9, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

| Typical | | |
|---------|-----------------|
| | | |

<table>
<thead>
<tr>
<th>Symmetry</th>
<th>One and the same constant can be added to the var attribute of all items of VARIABLES.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Arg. properties</th>
<th>Functional dependency: VALUE determined by VARIABLES.</th>
</tr>
</thead>
</table>

---

**Figure 4.1335** provides an example where the SUM RANGE INCREASING SEQUENCE (9, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.
Figure 4.1335: Illustrating the SUM_RANGE_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.1336 and 4.1337 respectively depict the automaton associated with the constraint SUM_RANGE_INCREASING_SEQUENCE and its simplified form.

Figure 4.1336: Automaton for the SUM_RANGE_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.48 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0

Figure 4.1337: Simplified automaton for the SUM_RANGE_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.47 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0
### Constraint

$\text{SUM\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})$

### Arguments

- **VALUE**: $\text{dvar}$
- **VARIABLES**: collection($\text{var}\ldots\text{dvar}$)

### Restrictions

- $sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0$
- $\text{VALUE} \geq 0$
- $\text{VALUE} \leq \lfloor sv/2 \rfloor \ast (rv - 1) + \text{required}(\text{VARIABLES}, \text{var})$

where

- $rv = \text{range}(\text{VARIABLES}.\text{var})$
- $sv = |\text{VARIABLES}|$

### Purpose

A occurrence of the pattern $\text{STRICTLY\_DECREASING\_SEQUENCE}$ is the maximal subsequence which matches the regular expression ‘$>+$’.

Assume that the occurrence of the pattern $\text{STRICTLY\_DECREASING\_SEQUENCE}$ starts at position $i$ and ends at position $j$. The feature $\text{RANGE}$ computes the range of the values from index $i$ to index $j + 1$.

### Example

$(10, \langle 4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3 \rangle)$

Figure 4.138 provides an example where the $\text{SUM\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE}$ constraint holds.

### Typical

- $|\text{VARIABLES}| > 1$
- $\text{range}(\text{VARIABLES}.\text{var}) > 1$

### Symmetry

One and the same constant can be added to the $\text{var}$ attribute of all items of $\text{VARIABLES}$.

### Arg. properties

- **Functional dependency**: $\text{VALUE}$ determined by $\text{VARIABLES}$.
Figure 4.138: Illustrating the SUM_RANGE_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.1339 and 4.1340 respectively depict the automaton associated with the constraint SUM_RANGE STRICTLY DECREASING SEQUENCE and its simplified form.

Figure 4.1339: Automaton for the SUM_RANGE STRICTLY DECREASING SEQUENCE constraint obtained by applying decoration Table 3.48 to the seed transducer of the STRICTLY DECREASING SEQUENCE pattern where default is 0

Figure 4.1340: Simplified automaton for the SUM_RANGE STRICTLY DECREASING SEQUENCE constraint obtained by applying decoration Table 3.46 to the seed transducer of the STRICTLY DECREASING SEQUENCE pattern where default is 0
SUM_RANGE_STRICTLY_DECREASING_SEQUENCE
SUM\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE

**Origin**
Based on the **STRICTLY\_INCREASING\_SEQUENCE** pattern.

**Constraint**

SUM\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE(VALUE, VARIABLES)

**Arguments**

VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE \geq 0 \]
\[ VALUE \leq \lfloor sv/2 \rfloor \times (rv − 1)\]
\[ required(VARIABLES, var) \]

where

\[ rv = range(VARIABLES, var) \]
\[ sv = |VARIABLES| \]

**Purpose**

An occurrence of the pattern **STRICTLY\_INCREASING\_SEQUENCE** is the maximal subsequence which matches the regular expression ‘\(<+\)’. Assume that the occurrence of the pattern **STRICTLY\_INCREASING\_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **RANGE** computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\[(9, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))\]

Figure 4.1341 provides an example where the SUM\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE (9, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) constraint holds.

**Typical**

\[ |VARIABLES| > 1 \]
\[ range(VARIABLES, var) > 1 \]

**Symmetry**

One and the same constant can be added to the \(var\) attribute of all items of **VARIABLES**.

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.1341: Illustrating the SUM_RANGE.Strictly.Increasing.Sequence constraint of the Example slot
Automaton

Figures 4.1342 and 4.1343 respectively depict the automaton associated with the constraint 
SUM_RANGE_STRICTLY_INCREASING_SEQUENCE and its simplified form.

Figure 4.1342: Automaton for the SUM_RANGE_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.48 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0

Figure 4.1343: Simplified automaton for the SUM_RANGE_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.47 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0
SUM_RANGE STRICTLY_INCREASING_SEQUENCE

2835
Origin
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

Constraint
SUM_SURF_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
\( sv \leq 5 \land rv \leq 2 \Rightarrow VALUE = 0 \)
VALUE = 0 \lor VALUE \geq \min(3 * \text{minv} + 3, (3 * \text{minv} + 3) * np) \)
VALUE = 0 \lor VALUE \leq \max(3 * \text{maxv} - 3, (3 * \text{maxv} - 3) * np) \)
\( n1 \geq VALUE - 3 + np - \max((sv - 3) * (maxv - 3), 0) \)
\( n2 \geq \min((sv - 3) * (minv + 1), minv, minv + 1, minv + 2) \)
\( n1 \geq \min((sv - 3) * (minv + 3), 0) - 3 * np - VALUE \)
\( n2 \geq \min((sv - 3) * (minv + 3), 0) - 3 * np - VALUE \)
\( \text{VALUE} \) is the sum of the surface of occurrences of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \( \text{VALUE} \) takes the default value 0.

Purpose
An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression ‘>>>’.
Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 2 \) to index \( j \).

Example
\( (27, \langle 7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3 \rangle) \)

Figure 4.1344 provides an example where the SUM_SURF_BUMP_ON_DECREASING_SEQUENCE constraint holds.
Figure 4.1344: Illustrating the SUM_SURF_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

**Typical**

\[
|\text{VARIABLES}| > 5 \\
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1345 and 4.1346 respectively depict the automaton associated with the constraint SUM_SURF_BUMP_ON_DECREASING_SEQUENCE and its simplified form.

Figure 4.1345: Automaton for the SUM_SURF_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is 0.
Figure 4.1346: Simplified automaton for the `SUM_SURF_BUMP_ON_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.26 to the seed transducer of the `BUMP_ON_DECREASING_SEQUENCE` pattern where `default` is 0.
SUM_SURF_DECREASING

DESCRIPTION AUTOMATON

Origin
Based on the DECREASING pattern.

Constraint
SUM_SURF_DECREASING(VALUE, VARIABLES)

Arguments

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var–dvar)</td>
</tr>
</tbody>
</table>
Restrictions

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]

\[
\begin{align*}
\text{VALUE} \geq & \min_{q \in [lb1, ub1]} \min \left\{ \begin{array}{l}
q \left( \sum \left( 2 \times (np - 1) \times \minv, \right. \\
\left. np \times np \right) \right) - 1, \\
2 \times np \\
\max (0, \min (1, np - 1)), \\
sv \mod np \times q.
\end{array} \right\} \\
& \sum \Pi (q - 1)
\end{align*}
\]

\[
\begin{align*}
\text{VALUE} \leq & \max_{q \in [lb2, ub2]} \max \left\{ \begin{array}{l}
q \left( \sum \left( 2 \times (np - 1) \times \maxv, \right. \\
\left. np \times np \right) \right) - 1, \\
2 \times np \\
\max (0, \min (1, np - 1)), \\
sv \mod np \times q.
\end{array} \right\} \\
& \sum \Pi (q - 1)
\end{align*}
\]

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

where

\[ sv = |\text{VARIABLES}| \]
\[ np = \lfloor sv/q \rfloor \]
\[ \maxv = \maxval(\text{VARIABLES}.\text{var}) \]
\[ \minv = \minval(\text{VARIABLES}.\text{var}) \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]
\[ lb1 = \min \left( \lfloor sv/2 \rfloor + 1, \right. \\
\left. \sum \left( \lfloor sv/\min(\min(sv, rv), |\minv| + 1) \rfloor, \right. \right. \\
\left. \left. \min(1, sv \mod \min(\min(sv, rv), |\minv| + 1)) \right) \right) \]
\[ ub1 = \lfloor sv/2 \rfloor + 1 \]
\[ lb2 = \min \left( \lfloor sv/2 \rfloor + 1, \right. \\
\left. \sum \left( \lfloor sv/\min(\min(sv, rv), |\maxv| + 1) \rfloor, \right. \right. \\
\left. \left. \min(1, sv \mod \min(\min(sv, rv), |\maxv| + 1)) \right) \right) \]
\[ ub2 = \lfloor sv/2 \rfloor + 1 \]

VALUE is the sum of the surface of occurrences of the DECREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression ‘>’.

Assume that the occurrence of the pattern DECREASING starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i \) to index \( j + 1 \).

Example

\[ \{37, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 4, 6, 4, 4)\} \]
Figure 4.1347 provides an example where the \textsc{SUM\_SURF\_DECREASING} \((37, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 4, 6, 4, 4])\) constraint holds.

![Diagram showing decreasing sequences and feature values](image)

Figure 4.1347: Illustrating the \textsc{SUM\_SURF\_DECREASING} constraint of the \textbf{Example} slot

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1348 and 4.1349 respectively depict the automaton associated with the constraint \texttt{SUM\_SURF\_DECREASING} and its simplified form.

![Automaton Diagram](image)

Figure 4.1348: Automaton for the \texttt{SUM\_SURF\_DECREASING} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{DECREASING} pattern where \texttt{default} is 0

![Simplified Automaton Diagram](image)

Figure 4.1349: Simplified automaton for the \texttt{SUM\_SURF\_DECREASING} constraint obtained by applying decoration Table 3.40 to the seed transducer of the \texttt{DECREASING} pattern where \texttt{default} is 0

Table 4.317: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the \texttt{SUM\_SURF\_DECREASING} constraint defined as the composition of the \texttt{DECREASING} pattern, the feature \texttt{SURF}, and the aggregator \texttt{sum}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.318: Concrete glue matrix, derived from the parametrised glue matrix 3.4, for the simplified automaton of the SUM_SURF_DECREASING constraint defined as the composition of the DECREASING pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_SURF_DECREASING

2845
**Origin**
Based on the **DECREASING_SEQUENCE** pattern.

**Constraint**
SUM_SURF_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var–dvar)

**Restrictions**
\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \implies VALUE = 0 \\
rv = 2 & \implies VALUE = 0 \lor VALUE \geq \min(2*\text{minv} + 1, (2*\text{minv} + 1) + np) \\
rv \geq 3 & \implies VALUE = 0, \\
V & \lor VALUE \geq \min\left(2*\text{minv} + 1, (2*\text{minv} + 1) + n, \text{minv}, (2*\text{minv} + 1) + \text{np} \right) \\
rv = 2 & \implies VALUE = 0 \lor VALUE \leq \max(2*\text{maxv} - 1, (2*\text{maxv} - 1) + np) \\
rv \geq 3 & \implies VALUE = 0, \\
V & \lor VALUE \leq \max\left(2*\text{maxv} - 1, (2*\text{maxv} - 1) + \text{np} \right) \\
\text{among} & (n_1, \text{VARIABLES}[1, sv], (\text{maxv} - 1, \text{maxv})) \\
rv = 2 & \vee \text{maxv} = 1 \implies n_1 \geq VALUE - \max(0, \text{np} * (2*\text{maxv} - 3)) \\
rv \geq 2 & \wedge \text{maxv} > 1 \implies n_1 \geq VALUE - \text{np} - sv * (\text{maxv} - 2) \\
\text{among} & (n_2, \text{VARIABLES}[1, sv], (\text{minv}, \text{minv} + 1)) \\
rv = 2 & \wedge \text{minv} = -1 \implies n_2 \geq \min(0, \text{np} * (2*\text{minv} + 3)) - VALUE \\
rv \geq 2 & \wedge \text{minv} < -1 \implies n_2 \geq sv * (\text{minv} + 2) - \text{np} - VALUE \\
\text{required} & (\text{VARIABLES}, var)
\end{align*}
\]
where
- \(sv = |\text{VARIABLES}|\)
- \(np = |sv/2|\)
- \(\text{minval} = \text{minval}(\text{VARIABLES}, var)\)
- \(\text{maxval} = \text{maxval}(\text{VARIABLES}, var)\)
- \(rv = \text{range}(\text{VARIABLES}, var)\)

**Purpose**
VALUE is the sum of the surface of occurrences of the **DECREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern **DECREASING_SEQUENCE** is the maximal subsequence which matches the regular expression ‘\(> (\geq | =) \lor >\)’.

Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i\) to index \(j+1\).
**Example**

\[(34, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 4.1350 provides an example where the `SUM_SURF_DECREASING_SEQUENCE` constraint holds.

```
< > = < < > = > > = < < > =
```

Figure 4.1350: Illustrating the `SUM_SURF_DECREASING_SEQUENCE` constraint of the Example slot

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1351 and 4.1352 respectively depict the automaton associated with the constraint SUM_SURF_DECREASING_SEQUENCE and its simplified form.

Figure 4.1351: Automaton for the SUM_SURF_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING_SEQUENCE pattern where default is 0

Figure 4.1352: Simplified automaton for the SUM_SURF_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.30 to the seed transducer of the DECREASING_SEQUENCE pattern where default is 0
Table 4.319: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the SUM_SURF_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>t</td>
</tr>
<tr>
<td>s</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>t</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
</tbody>
</table>

Table 4.320: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the simplified automaton of the SUM_SURF_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>t</td>
</tr>
<tr>
<td>s</td>
<td>0</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
</tr>
</tbody>
</table>
### Description

**Origin**
Based on the `DECREASING_TERRACE` pattern.

**Constraint**

```
SUM_SURF_DECREASING_TERRACE(VALUE, VARIABLES)
```

**Arguments**

- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`

**Restrictions**

\[
\begin{align*}
sv \leq 3 & \lor \ rv \leq 2 \Rightarrow VALUE = 0 \\
VALUE = 0 & \lor VALUE \geq \min(2 \times (\minv + 1) \times (sv - 2) \times (\minv + 1) x) \\
VALUE = 0 & \lor VALUE \leq \max(2 \times (\maxv - 1) \times (sv - 2) \times (\maxv - 1) x) \\
\text{among} \{n1, VARIABLES[2, sv - 1], (\maxv - 1)\} & \geq VALUE \\
\text{among} \{n2, VARIABLES[2, sv - 1], (\minv + 1)\} & \geq \min(0, (sv - 2) \times (\maxv - 2)) \\
\text{required}(VARIABLES, var) & = VALUE \\
\end{align*}
\]

where

- `minv = minval(VARIABLES.var)`
- `maxv = maxval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**

`VALUE` is the sum of the surface of occurrences of the `DECREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value 0.

An occurrence of the pattern `DECREASING_TERRACE` is the maximal subsequence which matches the regular expression `’>+=’`. Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position \(i\) and ends at position \(j\). The feature `SURF` computes the sum of the values from index \(i+1\) to index \(j\).

**Example**

\((12, (6, 4, 4, 5, 2, 1, 3, 3, 5, 4, 4, 3, 3))\)

Figure 4.1353 provides an example where the `SUM_SURF_DECREASING_TERRACE (12, [6, 4, 4, 5, 2, 1, 3, 3, 5, 4, 4, 3, 3])` constraint holds.

**Typical**

- `|VARIABLES| \geq 3`
- `range(VARIABLES.var) \geq 2`
Figure 4.1353: Illustrating the SUM_SURF_DECREASING_TERRACE constraint of the Example slot

Arg. properties  Functional dependency: VALUE determined by VARIABLES.
Figures 4.1354 and 4.1355 respectively depict the automaton associated with the constraint SUM_SURF_DECURING_TERRACE and its simplified form.

Figure 4.1354: Automaton for the SUM_SURF_DECURING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING_TERRACE pattern where default is 0

Figure 4.1355: Simplified automaton for the SUM_SURF_DECURING_TERRACE constraint obtained by applying decoration Table 3.29 to the seed transducer of the DECENDING_TERRACE pattern where default is 0
Table 4.321: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the SUM_SURF_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{D} + \overline{D} + \text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>t</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{D} + \overline{D} + \text{VAR}_{i+1}$</td>
<td>$\overline{D} + \overline{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

Table 4.322: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the simplified automaton of the SUM_SURF_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>0</td>
<td>$\overline{D} + \overline{D} + \text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>$\overline{D} + \overline{D} + \text{VAR}_{i+1}$</td>
<td>$\overline{D} + \overline{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>
SUM_SURF_DIP_ON_INCREASING_SEQUENCE

**Origin**
Based on the **DIP_ON_INCREASING_SEQUENCE** pattern.

**Constraint**
SUM_SURF_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**

<table>
<thead>
<tr>
<th>VALUE</th>
<th>:</th>
<th>dvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>:</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
sv & \leq 5 \lor rv \leq 2 \Rightarrow VALUE = 0 \\
VALUE & = 0 \lor VALUE \geq \min(3 \ast \minv + 3, (3 \ast \minv + 3) \ast np) \\
VALUE & = 0 \lor VALUE \leq \max(3 \ast \maxv - 3, (3 \ast \maxv - 3) \ast np) \\
n_1 & \geq VALUE - 3 \ast np - \max((sv - 3) \ast (maxv - 3), 0) \\
n_2 & \geq \min((sv - 3) \ast (minv + 3), 0) - 3 \ast np - VALUE \\
\text{required}(VARIABLES, var) & \text{ where} \\
sv & = |\text{VARIABLES}| \\
np & = \max(0, [(sv - 3)/3]) \\
\minv & = \minval(\text{VARIABLES}.\text{var}) \\
\maxv & = \maxval(\text{VARIABLES}.\text{var}) \\
rv & = \text{range}(\text{VARIABLES}.\text{var}) \\
\text{VALUE} & \leq 18 = (3 \ast 4 - 3) + 2
\end{align*}
\]

**Purpose**

VALUE is the sum of the surface of occurrences of the **DIP_ON_INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** is the subsequence which matches the regular expression ‘<<><<’.

Assume that the occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i + 2\) to index \(j\).

**Example**

\[(19, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))\]

Figure 4.1356 provides an example where the SUM_SURF_DIP_ON_INCREASING_SEQUENCE (19,[1,2,3,2,5,6,7,4,1,3,4,6,1,2,4,4]) constraint holds.

**Typical**

| VARIABLES | > | 5 |
| range(VARIABLES.var) | > | 2 |
Figure 4.1356: Illustrating the SUM_SURF_DIP_ON_INCREASINGSEQUENCE constraint of the Example slot

**Arg. properties**

**Functional dependency:** VALUE determined by VARIABLES.
Automaton

Figures 4.1357 and 4.1358 respectively depict the automaton associated with the constraint SUM_SURF_DIP_ON_INCREASING_SEQUENCE and its simplified form.

Figure 4.1357: Automaton for the SUM_SURF_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is 0
Figure 4.1358: Simplified automaton for the SUM_SURF_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.26 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is 0
Purpose

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \( ((>| (> (= | >)*) | (< (= | <)*) <) | <) \). Assume that the occurrence of the pattern GORGE starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i+1 \) to index \( j \).
Example

$$(25, \langle 1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7 \rangle)$$

Figure 4.1359 provides an example where the SUM_SURF_GORGE constraint holds.

Figure 4.1359: Illustrating the SUM_SURF_GORGE constraint of the Example slot

Typical

- $|\text{VARIABLES}| > 2$
- $\text{range}(\text{VARIABLES}.\text{var}) > 1$

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1360 and 4.1361 respectively depict the automaton associated with the constraint SUM_SURF_GORGE and its simplified form.

Figure 4.1360: Automaton for the SUM_SURF_GORGE constraint obtained by applying decoration Table 3.37 to the seed transducer of the GORGE pattern where default is 0 (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
Figure 4.1361: Simplified automaton for the SUM_SURF_GORGE constraint obtained by applying decoration Table 3.26 to the seed transducer of the GORGE pattern where default is 0 (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
Table 4.323: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the SUM_SURF_GORGE constraint defined as the composition of the GORGE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td>t</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>u</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
</tbody>
</table>

Table 4.324: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the simplified automaton of the SUM_SURF_GORGE constraint defined as the composition of the GORGE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>u</td>
<td>0</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
<td>0</td>
</tr>
</tbody>
</table>
SUM_SURF_GORGE 2863
### Description

**Origin**
Based on the **INCREASING** pattern.

**Constraint**

```
SUM_SURF_INCREASING(VALUE, VARIABLES)
```

**Arguments**

- **VALUE** : dvar
- **VARIABLES** : `collection(var–dvar)`
Restrictions

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]

\[
\begin{align*}
\text{VALUE} & \geq \min_{q \in [lb1, ub1]} \min \\
\text{VALUE} & \leq \max_{q \in [lb2, ub2]} \max \\
\end{align*}
\]

\[
0, \sum \left( q \ast \left( \begin{array}{l}
\sum_{2 \ast np}^{\text{max} \left( 0, \min \left( 1, np - 1 \right) \right)} \\
\sum_{2 \ast np}^{\max \left( 0, \min \left( 1, np - 1 \right) \right)} \\
\sum_{2 \ast np}^{\max \left( 0, \min \left( 1, np - 1 \right) \right)} \\
\sum_{2 \ast np}^{\minv, np} \\
\end{array} \right) \right) \\
\end{align*}
\]

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

where

\[
\begin{align*}
sv &= |\text{VARIABLES}| \\
np &= |sv| \div q \\
maxv &= \maxval(\text{VARIABLES}, \text{var}) \\
minv &= \minval(\text{VARIABLES}, \text{var}) \\
rv &= \text{range}(\text{VARIABLES}, \text{var}) \\
lb1 &= \min \left( \left[ \frac{|sv|}{2} + 1, \right. \right. \\
& \left. \left. \sum_{\min(1, \text{sv} \mod \min(\text{sv}, \text{rv}), |\text{minv}| + 1)} \right) \right) \\
ub1 &= |sv| \div 2 + 1 \\
lb2 &= \min \left( \left[ \frac{|sv|}{2} + 1, \right. \right. \\
& \left. \left. \sum_{\min(1, \text{sv} \mod \min(\text{sv}, \text{rv}), |\text{maxv}| + 1)} \right) \right) \\
ub2 &= |sv| \div 2 + 1 \\
\end{align*}
\]

VALUE is the sum of the surface of occurrences of the increasing pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose

An occurrence of the pattern increasing is the subsequence which matches the regular expression ‘<’. Assume that the occurrence of the pattern increasing starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

Example

\[(33, \langle 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3 \rangle)\]
Figure 4.1362 provides an example where the \texttt{SUM\_SURF\_INCREASING}
\((33, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

![Diagram illustrating the \texttt{SUM\_SURF\_INCREASING} constraint](image)

\textbf{Typical}

- \(|\text{VARIABLES}| > 1\)
- \(\text{range(VARIABLES.var)} > 1\)

\textbf{Arg. properties}

- Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1363 and 4.1364 respectively depict the automaton associated with the constraint SUM_SURF_INCREASING and its simplified form.

\[
\begin{align*}
  \{ & C \leftarrow \text{default} \\
  & D \leftarrow 0 \\
  & R \leftarrow \text{default} \} \\
  \{ & D \leftarrow 0 \\
  & R \leftarrow R + D + \text{VAR}_i + \text{VAR}_{i+1} \}
\end{align*}
\]

Figure 4.1363: Automaton for the SUM_SURF_INCREASING constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING pattern where default is 0

\[
\begin{align*}
  \{ & R \leftarrow \text{default} \\
  \{ & R \leftarrow R + \text{VAR}_i + \text{VAR}_{i+1} \}
\end{align*}
\]

Figure 4.1364: Simplified automaton for the SUM_SURF_INCREASING constraint obtained by applying decoration Table 3.40 to the seed transducer of the INCREASING pattern where default is 0

Table 4.325: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the SUM_SURF_INCREASING constraint defined as the composition of the INCREASING pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.326: Concrete glue matrix, derived from the parametrised glue matrix 3.8, for the simplified automaton of the SUM_SURF_INCREASING constraint defined as the composition of the INCREASING pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Based on the **INCREASING_SEQUENCE** pattern.

**Arguments**
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var–dvar)`

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow VALUE = 0 \\
rv = 2 & \Rightarrow VALUE = 0 \lor VALUE \geq \min(2 \ast \text{minv} + 1, (2 \ast \text{minv} + 1) \ast np) \\
rv \geq 3 & \Rightarrow \\
\lor \\
\text{VALUE} = 0, \\
\text{VALUE} \geq \min \left( \begin{array}{l}
2 \ast \text{minv} + 13, \\
(2 \ast \text{minv} + 1) \ast np + \min(0, \text{sv mod } 2 \ast (\text{minv} + 2))
\end{array} \right)
\end{align*}
\]

**Purpose**

VALUE is the sum of the surface of occurrences of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘< (< | =)’ < |’. Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i \) to index \( j + 1 \).
Example

\[(29, \langle 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3 \rangle)\]

Figure 4.1365 provides an example where the \text{SUM}_{\text{SURF}}_{\text{INCREASING}}_{\text{SEQUENCE}} \ (29, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

Figure 4.1365: Illustrating the \text{SUM}_{\text{SURF}}_{\text{INCREASING}}_{\text{SEQUENCE}} constraint of the \textbf{Example} slot

**Typical**

\[|\text{VARIABLES}| > 1\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1366 and 4.1367 respectively depict the automaton associated with the constraint \texttt{SUM\_SURF\_INCREASING\_SEQUENCE} and its simplified form.

\begin{align*}
\{ C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default} \}
\end{align*}

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure4.1366}
\caption{Automaton for the \texttt{SUM\_SURF\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern where default is 0}
\end{figure}

\begin{align*}
\{ D \leftarrow 0 \\
R \leftarrow \text{default} \}
\end{align*}

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure4.1367}
\caption{Simplified automaton for the \texttt{SUM\_SURF\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.30 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern where default is 0}
\end{figure}
Table 4.327: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the SUM_SURF_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>C^| + C</td>
</tr>
<tr>
<td>t</td>
<td>C + C</td>
</tr>
</tbody>
</table>

Table 4.328: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the simplified automaton of the SUM_SURF_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
</tr>
</tbody>
</table>
Based on the INCREASING_TERRACE pattern.

**Constraint**

SUM_SURF_INCREASING_TERRACE(VALUE, VARIABLES)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var − dvar)

**Restrictions**

\[
\begin{align*}
sv &\leq 3 \lor rv \leq 2 \Rightarrow VALUE = 0 \\
VALUE &< 0 \lor VALUE \leq \min(2 + (\text{minv} + 1)x, (sv - 2) + (\text{minv} + 1)y) \\
VALUE &< 0 \lor VALUE \leq \max(2 + (\text{maxv} - 1)x, (sv - 2) + (\text{maxv} - 1)y) \\
n1 &\geq VALUE - \max(0, (sv - 2) + (\text{maxv} - 2)) \\
n2 &\geq \min(0, (sv - 2) + (\text{minv} + 2)) - VALUE
\end{align*}
\]

where

- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

VALUE is the sum of the surface of occurrences of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression ‘< = + <’.

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**

\[
(19, (1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4))
\]

Figure 4.1368 provides an example where the SUM_SURF_INCREASING_TERRACE (19, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]
Figure 4.1368: Illustrating the SUM_SURF_INCREASING_TERRACE constraint of the Example slot

Arg. properties Functional dependency: VALUE determined by VARIABLES.
Figures 4.1369 and 4.1370 respectively depict the automaton associated with the constraint SUM_SURF_INCREASING_TERRACE and its simplified form.

Figure 4.1369: Automaton for the SUM_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_TERRACE pattern where default is 0.

Figure 4.1370: Simplified automaton for the SUM_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 3.29 to the seed transducer of the INCREASING_TERRACE pattern where default is 0.
Table 4.329: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the **SUM_SURF_INCREASING_TERRACE** constraint defined as the composition of the **INCREASING_TERRACE** pattern, the feature **SURF**, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>t</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_{i+1})</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_{i+1})</td>
</tr>
</tbody>
</table>

Table 4.330: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the simplified automaton of the **SUM_SURF_INCREASING_TERRACE** constraint defined as the composition of the **INCREASING_TERRACE** pattern, the feature **SURF**, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>0</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_{i+1})</td>
<td></td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_{i+1})</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_{i+1})</td>
<td></td>
</tr>
</tbody>
</table>
Based on the \textsc{inflexion} pattern.

\textbf{Constraint} \hspace{1cm} \textsc{sum\_surf\_inflexion}(\textsc{value}, \textsc{variables})

\textbf{Arguments} \hspace{1cm} \textsc{value} : dvar
\hspace{1cm} \textsc{variables} : collection(var−dvar)

\textbf{Restrictions} \hspace{1cm} \textsc{sv} \leq 2 \lor \textsc{rv} \leq 1 \Rightarrow \textsc{value} = 0
\hspace{1cm} \textsc{value} = 0 \lor \textsc{value} \geq \min(\textsc{minv}, (\textsc{sv} − 2) \star \textsc{minv})
\hspace{1cm} \textsc{value} = 0 \lor \textsc{value} \leq \max(\textsc{maxv}, (\textsc{sv} − 2) \star \textsc{maxv})
\hspace{1cm} \text{among}(\textsc{n1}, \textsc{variables}[2, \textsc{sv} − 1], (\textsc{maxv}))
\hspace{1cm} \textsc{n1} \geq \textsc{value} − \max(0, (\textsc{sv} − 2) \star (\textsc{maxv} − 1))
\hspace{1cm} \textsc{n2} \geq \min(0, (\textsc{sv} − 2) \star (\textsc{minv} + 1)) − \textsc{value}
\hspace{1cm} \text{among}(\textsc{n2}, \textsc{variables}[2, \textsc{sv} − 1], (\textsc{minv}))
\hspace{1cm} \text{required}(\textsc{variables}, \textsc{var})
\hspace{1cm} \text{where}
\hspace{1cm} \textsc{minv} = \text{minval}(\textsc{variables}.\textsc{var})
\hspace{1cm} \textsc{maxv} = \text{maxval}(\textsc{variables}.\textsc{var})
\hspace{1cm} \textsc{sv} = |\textsc{variables}|
\hspace{1cm} \textsc{rv} = \text{range}(\textsc{variables}.\textsc{var})

VALUE is the sum of the surface of occurrences of the \textsc{inflexion} pattern in the time-series given by the \textsc{variables} collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern \textsc{inflexion} is the maximal subsequence which matches the regular expression ‘\textless (\textless | =)* \textgreater | \textgreater (\textless | =)* \textless’.
Assume that the occurrence of the pattern \textsc{inflexion} starts at position \textit{i} and ends at position \textit{j}. The feature \textsc{surf} computes the sum of the values from index \textit{i} + 1 to index \textit{j}.

\textbf{Example} \hspace{1cm} (49, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])

Figure 4.1371 provides an example where the \textsc{sum\_surf\_inflexion} (49, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4]) constraint holds.

\textbf{Typical} \hspace{1cm} |\textsc{variables}| > 2
\hspace{1cm} \text{range}(\textsc{variables}.\textsc{var}) > 1
Figure 4.1371: Illustrating the SUM_SURF_INFLEXION constraint of the Example slot

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figures 4.1372 and 4.1373 respectively depict the automaton associated with the constraint SUM_SURF_INFLEXION and its simplified form.

Figure 4.1372: Automaton for the SUM_SURF_INFLEXION constraint obtained by applying decoration Table 3.37 to the seed transducer of the INFLEXION pattern where default is 0 (transition \( r \rightarrow t \) has the same registers updates as transition \( t \rightarrow r \))

Figure 4.1373: Simplified automaton for the SUM_SURF_INFLEXION constraint obtained by applying decoration Table 3.26 to the seed transducer of the INFLEXION pattern where default is 0 (transition \( r \rightarrow t \) has the same registers updates as transition \( t \rightarrow r \))
SUM_SURF_PEAK

Origin

Based on the PEAK pattern.

Constraint

SUM_SURF_PEAK(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ min(minv + 1, (sv−2) * (minv + 1))
VALUE = 0 ∨ VALUE ≤ max(maxv, (sv−2) * maxv)
among(n1, VARIABLES[2, sv−1], {maxv})
n1 ≥ VALUE − max(0, (sv−2) * (maxv − 1))
among(n2, VARIABLES[2, sv−1], {minv + 1})
n2 ≥ min(0, (sv−2) * (minv + 2)) − VALUE
required(VARIABLES, var)

where

minv = minval(VARIABLES, var)
maxv = maxval(VARIABLES, var)
sv = |VARIABLES|
rv = range(VARIABLES, var)

VALUE is the sum of the surface of occurrences of the PEAK pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression ‘< (|=)< (>|=)∗ ’.
Assume that the occurrence of the pattern PEAK starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example

(32, ⟨7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1⟩)

Figure 4.1374 provides an example where the SUM_SURF_PEAK (32, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1]) constraint holds.

Typical

|VARIABLES| > 2
range(VARIABLES, var) > 1
Figure 4.1374: Illustrating the \texttt{SUM\_SURF\_PEAK} constraint of the \textbf{Example} slot

\textbf{Symmetry} \\
Items of \texttt{VARIABLES} can be \textbf{reversed}.

\textbf{Arg. properties} \\
\textbf{Functional dependency}: \texttt{VALUE} determined by \texttt{VARIABLES}.
Automaton

Figures 4.1375 and 4.1376 respectively depict the automaton associated with the constraint SUM_SURF_PEAK and its simplified form.

Figure 4.1375: Automaton for the SUM_SURF_PEAK constraint obtained by applying decoration Table 3.37 to the seed transducer of the PEAK pattern where default is 0

Figure 4.1376: Simplified automaton for the SUM_SURF_PEAK constraint obtained by applying decoration Table 3.26 to the seed transducer of the PEAK pattern where default is 0
Table 4.331: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the SUM_SURF_PEAK constraint defined as the composition of the PEAK pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>t</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
</tbody>
</table>

Table 4.332: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the simplified automaton of the SUM_SURF_PEAK constraint defined as the composition of the PEAK pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1})</td>
<td>0</td>
</tr>
</tbody>
</table>
SUM_SURF_PLAIN

Origin
Based on the PLAIN pattern.

Constraint
SUM_SURF_PLAIN(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ min(minv, minv * (sv - 2) * 2)
VALUE = 0 ∨ VALUE ≤ max(maxv - 1, (maxv - 1) * (sv - 2) * 2)
among(n1, VARIABLES[sv - 1], (maxv - 1))
n1 ≥ VALUE - max(0, (sv - 2 - c) * (maxv - 2))
among(n2, VARIABLES[sv - 1], (minv))
n2 ≥ min(0, (sv - 2 - c) * (minv + 1)) - VALUE
required(VARIABLES, var)

where
minv = minval(VARIABLES, var)
maxv = maxval(VARIABLES, var)
sv = |VARIABLES|
rv = range(VARIABLES, var)
c = min(1, VALUE mod (sv - 2))

VALUE is the sum of the surface of occurrences of the PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose
An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘>’='<'.
Assume that the occurrence of the pattern PLAIN starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
(15, (2, 3, 6, 5, 7, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3))

Figure 4.1377 provides an example where the SUM_SURF_PLAIN (15, [2, 3, 6, 5, 7, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES, var) > 1
Figure 4.1377: Illustrating the SUM_SURF.PLAIN constraint of the **Example** slot

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1378 and 4.1379 respectively depict the automaton associated with the constraint `SUM_SURFPLAIN` and its simplified form.

Figure 4.1378: Automaton for the `SUM_SURFPLAIN` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `PLAIN` pattern where `default` is 0.

Figure 4.1379: Simplified automaton for the `SUM_SURFPLAIN` constraint obtained by applying decoration Table 3.29 to the seed transducer of the `PLAIN` pattern where `default` is 0.
### Table 4.333: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the SUM_SURF_PLAIN constraint defined as the composition of the PLAIN pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>t</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>

### Table 4.334: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the simplified automaton of the SUM_SURF_PLAIN constraint defined as the composition of the PLAIN pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1}$</td>
</tr>
</tbody>
</table>
SUM_SURF_PLATEAU

DESCRIPTION

Origin
Based on the PLATEAU pattern.

Constraint
SUM_SURF_PLATEAU(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var-dvar)

Restrictions

sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ min(minv + 1, (sv − 2) * (minv + 1))
VALUE = 0 ∨ VALUE ≤ max(maxv, (sv − 2) * maxv)

among(n1, VARIABLES[2, sv − 1], [maxv])
n1 ≥ VALUE − max(0, (sv − 2 − c) * (maxv − 1))

among(n2, VARIABLES[2, sv − 1], [minv + 1])
n2 ≥ min(0, (sv − 2 − c) * (minv + 2)) − VALUE

required(VARIABLES, var)

where
minv = minval(VARIABLES, var)
maxv = maxval(VARIABLES, var)
sv = |VARIABLES|
rv = range(VARIABLES, var)
c = min(1, VALUE mod (sv − 2))

VALUE is the sum of the surface of occurrences of the PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose
An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression '<='.'
Assume that the occurrence of the pattern PLATEAU starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
(17, (7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5))

Figure 4.1380 provides an example where the SUM_SURF_PLATEAU (17, [7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES, var) > 1
Figure 4.1380: Illustrating the SUM_SURF_PLATEAU constraint of the Example slot

<table>
<thead>
<tr>
<th>Symmetry</th>
<th>Items of VARIABLES can be reversed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arg. properties</td>
<td>Functional dependency: VALUE determined by VARIABLES.</td>
</tr>
</tbody>
</table>
Automaton

Figures 4.1381 and 4.1382 respectively depict the automaton associated with the constraint SUM_SURF_PLATEAU and its simplified form.

Figure 4.1381: Automaton for the SUM_SURF_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PLATEAU pattern where default is 0

Figure 4.1382: Simplified automaton for the SUM_SURF_PLATEAU constraint obtained by applying decoration Table 3.29 to the seed transducer of the PLATEAU pattern where default is 0
Table 4.335: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the \texttt{SUM\_SURF\_PLATEAU} constraint defined as the composition of the \texttt{PLATEAU} pattern, the feature \texttt{SURF}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
 & \texttt{SUM\_SURF\_PLATEAU} & \\
\hline
\texttt{s} & \texttt{\textcolor{red}{C} + \textcolor{red}{C}} & \texttt{\textcolor{red}{C} + \textcolor{red}{C}} & \texttt{\textcolor{red}{C} + \textcolor{red}{C}} \\
\texttt{r} & \texttt{\textcolor{red}{C} + \textcolor{red}{C}} & \texttt{\textcolor{red}{D} + \textcolor{red}{D} + \texttt{VAR}_{i+1}} & \texttt{\textcolor{red}{D} + \textcolor{red}{D} + \texttt{VAR}_{i+1}} \\
\texttt{t} & \texttt{\textcolor{red}{C} + \textcolor{red}{C}} & \texttt{\textcolor{red}{D} + \textcolor{red}{D} + \texttt{VAR}_{i+1}} & \texttt{\textcolor{red}{D} + \textcolor{red}{D} + \texttt{VAR}_{i+1}} \\
\hline
\end{tabular}
\end{table}

Table 4.336: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the simplified automaton of the \texttt{SUM\_SURF\_PLATEAU} constraint defined as the composition of the \texttt{PLATEAU} pattern, the feature \texttt{SURF}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
 & \texttt{SUM\_SURF\_PLATEAU} & \\
\hline
\texttt{s} & 0 & 0 & 0 \\
\texttt{r} & 0 & \texttt{\textcolor{red}{D} + \textcolor{red}{D} + \texttt{VAR}_{i+1}} & \texttt{\textcolor{red}{D} + \textcolor{red}{D} + \texttt{VAR}_{i+1}} \\
\texttt{t} & 0 & \texttt{\textcolor{red}{D} + \textcolor{red}{D} + \texttt{VAR}_{i+1}} & \texttt{\textcolor{red}{D} + \textcolor{red}{D} + \texttt{VAR}_{i+1}} \\
\hline
\end{tabular}
\end{table}
SUM_SURF_PROPERPLAIN

Origin
Based on the PROPERPLAIN pattern.

Constraint
SUM_SURF_PROPERPLAIN(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ min(2 * minv1, (sv − 2) * minv2)
VALUE = 0 ∨ VALUE ≤ max(2 * (maxv − 1) ∗ (sv − 2) * (maxv − 1) ∗ 6)
among(n1, VARIABLES[2, sv − 1], (maxv − 1))
n1 ≥ VALUE − max(0, (sv − 2 − c) ∗ (maxv − 2))
among(n2, VARIABLES[2, sv − 1], (minv))
n2 ≥ min(0, (sv − 2 − c) ∗ (minv + 1)) − VALUE
required(VARIABLES, var)

where
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES| 
rv = range(VARIABLES.var)
c = min(1, VALUE mod (sv − 2))

VALUE is the sum of the surface of occurrences of the PROPERPLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose
An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression ‘> = + <’.
Assume that the occurrence of the pattern PROPERPLAIN starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
(27, [2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5])

Figure 4.1383 provides an example where the SUM_SURF_PROPERPLAIN (27, [2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5]) constraint holds.

Typical
|VARIABLES| > 3
range(VARIABLES.var) > 1
Figure 4.1383: Illustrating the SUM_SURF_PROPERPLAIN constraint of the Example slot

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1384 and 4.1385 respectively depict the automaton associated with the constraint SUM_SURF_PROPER_PLAIN and its simplified form.

Figure 4.1384: Automaton for the SUM_SURF_PROPER_PLAIN constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER_PLAIN pattern where default is 0

Figure 4.1385: Simplified automaton for the SUM_SURF_PROPER_PLAIN constraint obtained by applying decoration Table 3.29 to the seed transducer of the PROPER_PLAIN pattern where default is 0
Table 4.337: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the `SUM_SURF_PROPER_PLAIN` constraint defined as the composition of the `PROPER_PLAIN` pattern, the feature `SURF`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>s</code></td>
<td><code>r</code></td>
<td><code>t</code></td>
<td></td>
</tr>
<tr>
<td><code>s</code></td>
<td><code>C</code></td>
<td><code>C</code></td>
<td><code>C</code></td>
</tr>
<tr>
<td><code>r</code></td>
<td><code>C</code></td>
<td><code>C</code></td>
<td><code>D</code></td>
</tr>
<tr>
<td><code>t</code></td>
<td><code>D</code></td>
<td><code>D</code></td>
<td><code>D</code></td>
</tr>
</tbody>
</table>

Table 4.338: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the simplified automaton of the `SUM_SURF_PROPER_PLAIN` constraint defined as the composition of the `PROPER_PLAIN` pattern, the feature `SURF`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>s</code></td>
<td><code>r</code></td>
<td><code>t</code></td>
<td></td>
</tr>
<tr>
<td><code>s</code></td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><code>r</code></td>
<td>0</td>
<td>0</td>
<td><code>D</code></td>
</tr>
<tr>
<td><code>t</code></td>
<td><code>D</code></td>
<td><code>D</code></td>
<td><code>D</code></td>
</tr>
</tbody>
</table>
SUM_SURF_PROPER_PLATEAU

Origin
Based on the PROPER_PLATEAU pattern.

Constraint
SUM_SURF_PROPER_PLATEAU(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var-dvar)

Restrictions
\( sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
VALUE = 0 \lor VALUE \geq \min(2 \times (\text{minv} + 1), (sv - 2) \times (\text{minv} + 1))
\text{among}(n1, \text{VARIABLES}[2, sv - 1], \text{(maxv)})
n1 \geq \text{VALUE} - \max(0, (sv - 2 - c) \times (\text{maxv} - 1))
\text{among}(n2, \text{VARIABLES}[2, sv - 1], (\text{minv} + 1))
n2 \geq \min(0, (sv - 2 - c) \times (\text{minv} + 2)) - \text{VALUE}
\text{required}(\text{VARIABLES}.\text{var})

where
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
sv = |\text{VARIABLES}|
rv = \text{range}(\text{VARIABLES}.\text{var})
c = \min(1, \text{VALUE} \mod (sv - 2))

Purpose
VALUE is the sum of the surface of occurrences of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression ‘< =+ >’.
Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

Example
(29, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))

Figure 4.1386 provides an example where the SUM_SURF_PROPER_PLATEAU (29, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3]) constraint holds.

Typical
|\text{VARIABLES}| > 3
\text{range}(\text{VARIABLES}.\text{var}) > 1
Figure 4.1386: Illustrating the SUM_SURF_PROPER_PLATEAU constraint of the Example slot

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1387 and 4.1388 respectively depict the automaton associated with the constraint \textit{SUM\_SURF\_PROPER\_PLATEAU} and its simplified form.

Figure 4.1387: Automaton for the \textit{SUM\_SURF\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textit{PROPER\_PLATEAU} pattern where default is 0

Figure 4.1388: Simplified automaton for the \textit{SUM\_SURF\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 3.29 to the seed transducer of the \textit{PROPER\_PLATEAU} pattern where default is 0
Table 4.339: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the SUM_SURF_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
<th>( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
</tr>
<tr>
<td>( t )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>

Table 4.340: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the simplified automaton of the SUM_SURF_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
<th>( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( r )</td>
<td>0</td>
<td>0</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
</tr>
<tr>
<td>( t )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
<td>( \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_{i+1} )</td>
</tr>
</tbody>
</table>
SUM_SURF_STEADY

Purpose

VALUE is the sum of the surface of occurrences of the STEADY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern STEADY is the subsequence which matches the regular expression ‘=’.

Assume that the occurrence of the pattern STEADY starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

Example

Figure 4.1389 provides an example where the SUM_SURF_STEADY (60, [1, 1, 7, 3, 3, 5, 5, 5, 6, 5, 5, 7, 2, 6, 6]) constraint holds.

Typical

$|\text{VARIABLES}| > 1$
SUM_SURF_STEADY

Figure 4.1389: Illustrating the SUM_SURF_STEADY constraint of the Example slot

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figures 4.1390 and 4.1391 respectively depict the automaton associated with the constraint `SUM_SURF_STEADY` and its simplified form.

Figure 4.1390: Automaton for the `SUM_SURF_STEADY` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `STEADY` pattern where `default` is 0.

Figure 4.1391: Simplified automaton for the `SUM_SURF_STEADY` constraint obtained by applying decoration Table 3.40 to the seed transducer of the `STEADY` pattern where `default` is 0.

Table 4.341: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the `SUM_SURF_STEADY` constraint defined as the composition of the `STEADY` pattern, the feature `SURF`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.342: Concrete glue matrix, derived from the parametrised glue matrix 3.16, for the simplified automaton of the SUM_SURF_STEADY constraint defined as the composition of the STEADY pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
</tr>
</tbody>
</table>
Origin
Based on the \texttt{STEADY\_SEQUENCE} pattern.

Constraint
\texttt{SUM\_SURF\_STEADY\_SEQUENCE(VALUE, VARIABLES)}

Arguments
\begin{verbatim}
VALUE : dvar
VARIABLES : collection(var--dvar)
\end{verbatim}

Restrictions
\begin{verbatim}
sv \leq 1 \Rightarrow VALUE = 0
rv = 1 \land sv \geq 2 \Rightarrow VALUE \geq \minv \ast sv
rv \geq 2 \Rightarrow VALUE = 0 \lor VALUE \geq \min(2 \ast \minv, \minv \ast sv)
rv = 1 \land sv \geq 2 \Rightarrow VALUE \leq \maxv \ast sv
rv \geq 2 \Rightarrow VALUE = 0 \lor VALUE \leq \max(2 \ast \maxv, \maxv \ast sv)
\end{verbatim}

\begin{verbatim}
\texttt{among}(n1, VARIABLES[1, sv], (\maxv))
n1 \geq VALUE - \max(0, sv \ast (\maxv - 1))
\texttt{among}(n2, VARIABLES[2, sv], (\minv))
n2 \geq \min(0, sv \ast (\minv + 1)) - VALUE
\end{verbatim}

\texttt{required}(VARIABLES, var)

\begin{verbatim}
where
\minv = \minval(VARIABLES.var)
\maxv = \maxval(VARIABLES.var)
sv = |VARIABLES|
rv = \texttt{range}(VARIABLES.var)
\end{verbatim}

Purpose
\begin{itemize}
\item VALUE is the sum of the surface of occurrences of the \texttt{STEADY\_SEQUENCE} pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.
\item An occurrence of the pattern \texttt{STEADY\_SEQUENCE} is the maximal subsequence which matches the regular expression ‘\texttt{+=}’.
\item Assume that the occurrence of the pattern \texttt{STEADY\_SEQUENCE} starts at position $i$ and ends at position $j$. The feature \texttt{SURF} computes the sum of the values from index $i$ to index $j + 1$.
\end{itemize}

Example
\texttt{(31, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1])}

Figure 4.1392 provides an example where the \texttt{SUM\_SURF\_STEADY\_SEQUENCE (31, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1])} constraint holds.

Typical
\begin{verbatim}
|VARIABLES| > 1
\end{verbatim}
Figure 4.1392: Illustrating the SUM_SURF_STEADY_SEQUENCE constraint of the Example slot

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1393 and 4.1394 respectively depict the automaton associated with the constraint \texttt{SUM\_SURF\_STEADY\_SEQUENCE} and its simplified form.

\begin{align*}
\begin{cases}
  C \leftarrow \text{default} \\
  D \leftarrow 0 \\
  R \leftarrow \text{default}
\end{cases}
\end{align*}

\begin{align*}
\begin{cases}
  C \leftarrow \text{default} \\
  D \leftarrow 0 \\
  R \leftarrow R + C
\end{cases}
\end{align*}

Figure 4.1393: Automaton for the \texttt{SUM\_SURF\_STEADY\_SEQUENCE} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{STEADY\_SEQUENCE} pattern where \texttt{default} is 0

\begin{align*}
\begin{cases}
  \{ R \leftarrow \text{default} \}
\end{cases}
\end{align*}

\begin{align*}
\begin{cases}
  \{ R \leftarrow R + \text{VAR}_i + \text{VAR}_{i+1} \}
\end{cases}
\end{align*}

Figure 4.1394: Simplified automaton for the \texttt{SUM\_SURF\_STEADY\_SEQUENCE} constraint obtained by applying decoration Table 3.40 to the seed transducer of the \texttt{STEADY\_SEQUENCE} pattern where \texttt{default} is 0
Table 4.343: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the SUM_SURF_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\[
\begin{array}{cc}
  s & r \\
  \overrightarrow{C} + \overrightarrow{C} & \overrightarrow{C} + \overrightarrow{C} \\
  \overrightarrow{C} + \overrightarrow{C} & \overrightarrow{C} + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} - \VAR_{i+1}
\end{array}
\]

Table 4.344: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the simplified automaton of the SUM_SURF_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\[
\begin{array}{cc}
  s & r \\
  0 & 0 \\
  0 & -\VAR_{i+1}
\end{array}
\]
SUM_SURF_STRICTLY_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>AGGREGATOR</th>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUM_SURF_STRICTLY_DECREASING_SEQUENCE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DESCRIPTION**

**Origin**

Based on the `SUM_SURF_STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**

`SUM_SURF_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)`

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow VALUE = 0 \\
rv = 2 & \Rightarrow VALUE = 0 \lor VALUE \geq \min(2 \times \text{minv} + 1, (2 \times \text{minv} + 1) \times \text{np}) \\
rv \geq 3 & \Rightarrow
\begin{cases}
VALUE = 0, \\
VALUE \geq \min \left( 2 \times \text{minv} + 1, (2 \times \text{minv} + 1) \times \text{np} + \min(0, sv \mod 2 \times (\text{minv} + 2)) \right)
\end{cases}
\end{align*}
\]

**Purpose**

**VALUE** is the sum of the surface of occurrences of the `SUM_SURF_STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, **VALUE** takes the default value 0.

An occurrence of the pattern `SUM_SURF_STRICTLY_DECREASING_SEQUENCE` is the **maximal** subsequence which matches the regular expression `'>+'`. Assume that the occurrence of the pattern `SUM_SURF_STRICTLY_DECREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i\) to index \(j + 1\).
**Example**

\[(31, \{4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3\})\]

Figure 4.1395 provides an example where the \(\text{SUM}_{\text{SURF}}_{\text{STRICTLY DECREASING SEQUENCE}}\) constraint holds.

![Diagram](image)

Figure 4.1395: Illustrating the \(\text{SUM}_{\text{SURF}}_{\text{STRICTLY DECREASING SEQUENCE}}\) constraint of the **Example** slot

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range} (\text{VARIABLES.var}) > 1\)

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figures 4.1396 and 4.1397 respectively depict the automaton associated with the constraint SUM_SURF.Strictly_DECREASINGSEQUENCE and its simplified form.

\[
\begin{align*}
&\{ C \leftarrow \text{default}, D \leftarrow 0, R \leftarrow \text{default} \} \\
&\begin{array}{c}
\leq S \\
\leq \\
\leq \{ C \leftarrow \text{default}, D \leftarrow 0, R \leftarrow R + C \} \\
\leq \{ R \leftarrow \text{default} \} \\
\leq \{ R \leftarrow R + \text{VAR}_i + \text{VAR}_{i+1} \} \\
\leq \{ R \leftarrow R + \text{VAR}_{i+1} \} \\
\end{array}
\end{align*}
\]

Figure 4.1396: Automaton for the SUM_SURF.Strictly_DECREASINGSEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICTLY_DECREASINGSEQUENCE pattern where default is 0

\[
\begin{align*}
&\{ R \leftarrow \text{default} \} \\
&\begin{array}{c}
\leq S \\
\leq \{ R \leftarrow R + \text{VAR}_i + \text{VAR}_{i+1} \} \\
\leq \{ R \leftarrow R + \text{VAR}_{i+1} \} \\
\end{array}
\end{align*}
\]

Figure 4.1397: Simplified automaton for the SUM_SURF.Strictly_DECREASINGSEQUENCE constraint obtained by applying decoration Table 3.40 to the seed transducer of the STRICTLY_DECREASINGSEQUENCE pattern where default is 0
SUM_SURF_STRICTLY_DECREASING_SEQUENCE

Table 4.345: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the SUM_SURF_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICLY_DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>C + C</td>
<td>C + C</td>
</tr>
<tr>
<td>r</td>
<td>C + C</td>
<td>C + C + D + D - VAR_{i+1}</td>
</tr>
</tbody>
</table>

Table 4.346: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the SUM_SURF_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICLY_DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>-VAR_{i+1}</td>
</tr>
</tbody>
</table>
Origin
Based on the \texttt{STRICTLY\_INCREASING\_SEQUENCE} pattern.

Constraint
\texttt{SUM\_SURF\_STRICTLY\_INCREASING\_SEQUENCE}(\texttt{VALUE, VARIABLES})

Arguments
\begin{itemize}
\item \texttt{VALUE} : \texttt{dvar}
\item \texttt{VARIABLES} : \texttt{collection(var−dvar)}
\end{itemize}

Restrictions
\begin{align*}
sv &\leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \notag \\
rv &= 2 \Rightarrow VALUE = 0 \lor VALUE \geq \min (2 \times \texttt{minv} + 1, (2 \times \texttt{minv} + 1) \times \texttt{np}) \notag \\
rv &\geq 3 \Rightarrow \\
&\quad \left( VALUE = 0, \\
&\quad \sqrt{ \texttt{VALUE} \geq \min \left( 2 \times \texttt{minv} + 1, \\
&\quad \left( 2 \times \texttt{minv} + 1 \right) \times \texttt{np} + \min (0, sv \mod 2 \times (\texttt{minv} + 2)) \right) } \right) \\
rv &= 2 \Rightarrow VALUE = 0 \lor VALUE \leq \max (2 \times \texttt{maxv} - 1, (2 \times \texttt{maxv} - 1) \times \texttt{np}) \notag \\
rv &\geq 3 \Rightarrow \\
&\quad \left( VALUE = 0, \\
&\quad \sqrt{ \texttt{VALUE} \leq \max \left( 2 \times \texttt{maxv} - 1, \\
&\quad \left( 2 \times \texttt{maxv} - 1 \right) \times \texttt{np} + \max (0, sv \mod 2 \times (\texttt{maxv} - 2)) \right) } \right) \\
\texttt{among}(n1, \texttt{VARIABLES}[1, sv], (\texttt{maxv} - 1, \texttt{maxv})) \notag \\
n1 &\geq VALUE - \max (0, \texttt{np} \times (\texttt{maxv} - 2)) - \texttt{np} \notag \\
\texttt{among}(n2, \texttt{VARIABLES}[1, sv], (\texttt{minv}, \texttt{minv} + 1)) \notag \\
n2 &\geq \min (0, \texttt{np} \times (\texttt{minv} + 2)) - \texttt{np} - VALUE \notag \\
\texttt{required}(\texttt{VARIABLES, var}) \notag \\
\end{align*}

where
\begin{itemize}
\item \texttt{sv} = |\texttt{VARIABLES}|
\item \texttt{np} = |\texttt{sv}/2|
\item \texttt{minv} = \texttt{minval}(\texttt{VARIABLES, var})
\item \texttt{maxv} = \texttt{maxval}(\texttt{VARIABLES, var})
\item \texttt{rv} = \texttt{range}(\texttt{VARIABLES, var})
\end{itemize}

Purpose
\begin{itemize}
\item \texttt{VALUE} is the sum of the surface of occurrences of the \texttt{STRICTLY\_INCREASING\_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection. If the pattern does not occur, \texttt{VALUE} takes the default value 0.
\item An occurrence of the pattern \texttt{STRICTLY\_INCREASING\_SEQUENCE} is the \textit{maximal} sub-sequence which matches the regular expression ‘<\texttt{+}’.
\item Assume that the occurrence of the pattern \texttt{STRICTLY\_INCREASING\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{SURF} computes the sum of the values from index \(i\) to index \(j+1\).
\end{itemize}
Example

$$(30, \{4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3\})$$

Figure 4.1398 provides an example where the \texttt{SUM\_SURF\_STRICTLY\_INCREASING\_SEQUENCE} constraint holds.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{example.png}
\caption{Illustrating the \texttt{SUM\_SURF\_STRICTLY\_INCREASING\_SEQUENCE} constraint of the \textbf{Example} slot}
\end{figure}

Typical

$$|\text{VARIABLES}| > 1$$

$$\text{range(}\text{VARIABLES.var}) > 1$$

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 4.1399 and 4.1400 respectively depict the automaton associated with the constraint SUM_SURF_STRICTLY_INCREASING_SEQUENCE and its simplified form.

Figure 4.1399: Automaton for the SUM_SURF_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0

Figure 4.1400: Simplified automaton for the SUM_SURF_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.40 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0
Table 4.347: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the `SUM_SURF_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the feature `SURF`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\[
\begin{array}{c|c|c}
 s & r \\
 \hline
 s & C\overrightarrow{+}C & \overrightarrow{+}C \\
 r & \overrightarrow{+}C & \overrightarrow{+}C+D+D-\text{VAR}_{i+1}
\end{array}
\]

Table 4.348: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the simplified automaton of the `SUM_SURF_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the feature `SURF`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\[
\begin{array}{c|c|c}
 s & r \\
 \hline
 s & 0 & 0 \\
 r & 0 & -\text{VAR}_{i+1}
\end{array}
\]
SUM_SURF_SUMMIT

**Description**

Based on the SUMMIT pattern.

**Arguments**

- VALUE : dvar
- VARIABLES : collection(var–dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \\
rv = 2 & \Rightarrow VALUE = 0 \lor VALUE \geq \min(minv + 1 \& (minv + 1) \times np) \\
rv \geq 3 & \Rightarrow \\
& \quad (VALUE = 0, VALUE \geq \min(minv + 1 \& np, \min(minv + 1 \& (sv - 2 \times (minv + 1) + 1))) \\
rv = 2 & \Rightarrow VALUE = 0 \lor VALUE \leq \max(maxv, maxv + np) \\
rv \geq 3 & \Rightarrow \\
& \quad (VALUE = 0, VALUE \leq \max(maxv + np, \max(maxv, (sv - 2 \times (maxv - 1)))
\end{align*}
\]

\[
\begin{align*}
& \text{among}(n1, VARIABLES; 2, sv - 1, (maxv - 1, maxv)) \\
rv = 2 & \lor maxv = 1 \Rightarrow n1 \geq VALUE - np \times \max(0, maxv - 1) \\
rv > 2 & \land maxv > 1 \Rightarrow n1 \geq VALUE - (sv - 2 \times (maxv - 2) - 1)
\end{align*}
\]

\[
\begin{align*}
& \text{among}(n2, VARIABLES; 2, sv - 1, (minv + 1)) \\
rv = 2 & \lor minv = -1 \Rightarrow n2 \geq np \times \min(0, \minv + 2) - VALUE \\
rv > 2 & \land minv < 1 \Rightarrow n2 \geq (sv - 2 \times (\minv + 2) - VALUE
\end{align*}
\]

**Purpose**

VALUE is the sum of the surface of occurrences of the SUMMIT pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression \("(< | < (= | <) ^ <)(> | > (|= | >)^ >)\)." Arrive that the occurrence of the pattern SUMMIT starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.
Example

\[(23, \langle 7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1 \rangle)\]

Figure 4.1401 provides an example where the SUM_SURF_SUMMIT constraint holds.

Figure 4.1401: Illustrating the SUM_SURF_SUMMIT constraint of the Example slot

Typical

- \(|\text{VARIABLES}| > 2\)
- range(\text{VARIABLES}.var) > 1

Symmetry

- Items of VARIABLES can be reversed.

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
Figures 4.1402 and 4.1403 respectively depict the automaton associated with the constraint \texttt{SUM\_SURF\_SUMMIT} and its simplified form.

Figure 4.1402: Automaton for the \texttt{SUM\_SURF\_SUMMIT} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{SUMMIT} pattern where \texttt{default} is 0 (transition \(u \rightarrow r\) has the same register update as transition \(r \rightarrow u\))
Figure 4.1403: Simplified automaton for the SUM_SURF_SUMMIT constraint obtained by applying decoration Table 3.26 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
Table 4.349: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the \texttt{SUM\_SURF\_SUMMIT} constraint defined as the composition of the \texttt{SUMMIT} pattern, the feature \texttt{SURF}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
<th>(u)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\bar{c} + \bar{c})</td>
<td>(\bar{c} + \bar{c})</td>
<td>(\bar{c} + \bar{c})</td>
<td>(\bar{c} + \bar{c})</td>
</tr>
<tr>
<td>(r)</td>
<td>(\bar{c} + \bar{c})</td>
<td>(\bar{d} + \bar{d} + \text{VAR}_{i+1})</td>
<td>(\bar{c} + \bar{d} + \text{VAR}_{i+1})</td>
<td>(\bar{c} + \bar{c})</td>
</tr>
<tr>
<td>(t)</td>
<td>(\bar{c} + \bar{c})</td>
<td>(\bar{c} + \bar{d} + \text{VAR}_{i+1})</td>
<td>(\bar{c} + \bar{c})</td>
<td>(\bar{c} + \bar{d} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>(u)</td>
<td>(\bar{c} + \bar{c})</td>
<td>(\bar{c} + \bar{c})</td>
<td>(\bar{c} + \bar{d} + \text{VAR}_{i+1})</td>
<td>(\bar{c} + \bar{c})</td>
</tr>
</tbody>
</table>

Table 4.350: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the simplified automaton of the \texttt{SUM\_SURF\_SUMMIT} constraint defined as the composition of the \texttt{SUMMIT} pattern, the feature \texttt{SURF}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
<th>(u)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(r)</td>
<td>0</td>
<td>(\bar{d} + \bar{d} + \text{VAR}_{i+1})</td>
<td>(\bar{d} + \bar{d} + \text{VAR}_{i+1})</td>
<td>0</td>
</tr>
<tr>
<td>(t)</td>
<td>0</td>
<td>(\bar{d} + \bar{d} + \text{VAR}_{i+1})</td>
<td>0</td>
<td>(\bar{d} + \bar{d} + \text{VAR}_{i+1})</td>
</tr>
<tr>
<td>(u)</td>
<td>0</td>
<td>0</td>
<td>(\bar{d} + \bar{d} + \text{VAR}_{i+1})</td>
<td>0</td>
</tr>
</tbody>
</table>
Origin: Based on the VALLEY pattern.

Constraint: $\text{SUM\_SURF\_VALLEY}(\text{VALUE}, \text{VARIABLES})$

Arguments:
- \text{VALUE} : dvar
- \text{VARIABLES} : collection(var\_dvar)

Restrictions:
\[
\begin{align*}
sv \leq 2 \lor rv \leq 1 & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 \lor \text{VALUE} \geq \min(\text{minv}, \text{minv} \cdot (sv - 2)) & \\
\text{VALUE} = 0 \lor \text{VALUE} \leq \max(\text{maxv} - 1, (\text{maxv} - 1) \cdot (sv - 2)) & \\
\text{among}(n1, \text{VARIABLES}[2, sv - 1], (\text{maxv} - 1)) & \\
n1 \geq \text{VALUE} - \max(0, (sv - 2) \cdot (\text{maxv} - 2)) & \\
\text{among}(n2, \text{VARIABLES}[2, sv - 1], (\text{minv})) & \\
n2 \geq \min(0, (sv - 2) \cdot (\text{minv} + 1)) - \text{VALUE} & \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) & \\
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) & \\
sv = |\text{VARIABLES}| & \\
rv = \text{range}(\text{VARIABLES}.\text{var}) &
\end{align*}
\]

Purpose: An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression ‘$>$ (= | >)’$^*$’$<$. Assume that the occurrence of the pattern VALLEY starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

Example: $(35, (1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7))$

Figure 4.1404 provides an example where the SUM\_SURF\_VALLEY $(35, [1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7])$ constraint holds.

Typical: $|\text{VARIABLES}| > 2$
$\text{range}(\text{VARIABLES}.\text{var}) > 1$
Figure 4.1404: Illustrating the SUM_SURF_VALLEY constraint of the Example slot

**Symmetry**

- Items of VARIABLES can be reversed.

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
Figures 4.1405 and 4.1406 respectively depict the automaton associated with the constraint SUM_SURF_VALLEY and its simplified form.

Figure 4.1405: Automaton for the SUM_SURF_VALLEY constraint obtained by applying decoration Table 3.37 to the seed transducer of the VALLEY pattern where default is 0

Figure 4.1406: Simplified automaton for the SUM_SURF_VALLEY constraint obtained by applying decoration Table 3.26 to the seed transducer of the VALLEY pattern where default is 0
Table 4.351: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the SUM_SURF_VALLEY constraint defined as the composition of the VALLEY pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>C + C</td>
<td>C + C</td>
<td>C + C</td>
</tr>
<tr>
<td>r</td>
<td>C + C</td>
<td>D + D + VAR_i+1</td>
<td>C + D + D + VAR_i+1</td>
</tr>
<tr>
<td>t</td>
<td>C + C</td>
<td>C + D + D + VAR_i+1</td>
<td>C + C</td>
</tr>
</tbody>
</table>

Table 4.352: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the simplified automaton of the SUM_SURF_VALLEY constraint defined as the composition of the VALLEY pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>D + D + VAR_i+1</td>
<td>D + D + VAR_i+1</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>D + D + VAR_i+1</td>
<td>0</td>
</tr>
</tbody>
</table>
SUM_SURF_ZIGZAG

DESCRIPTION

Origin
Based on the ZIGZAG pattern.

Constraint
SUM_SURF_ZIGZAG(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = 0

VALUE = 0,

\[ \text{VALUE} \geq \min \left( 2 \cdot \text{minv} + 1 \cdot \left[ (sv - 1)/2 \right] + \text{minv} + \left[ (sv - 2)/2 \right] + (\text{minv} + 1) \cdot \right) \]

VALUE = 0,

\[ \text{VALUE} \leq \max \left( 2 \cdot \text{maxv} - 1 \cdot \left[ (sv - 1)/2 \right] + \text{maxv} + \left[ (sv - 2)/2 \right] + (\text{maxv} - 1) \cdot \right) \]

among(n1, VARIABLES[2, sv - 1], (maxv - 1, maxv))

n1 ≥ VALUE − [(sv - 1)/2] − max(0, (sv - 2) ∗ (maxv - 2))

n2 ≥ min(0, (sv - 2) ∗ (minv + 2)) − [(sv - 1)/2] − VALUE

among(n2, VARIABLES[2, sv - 1], (minv, minv + 1))

required(VARIABLES, var)

where

\[ \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \]

\[ \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \]

sv = |VARIABLES|

rv = range(VARIABLES.var)

VALUE is the sum of the surface of occurrences of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘(<>)+(< | <>) | (<>)+ (> | >>)’. Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

Example

\( (33, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1)) \)

Figure 4.1407 provides an example where the SUM_SURF_ZIGZAG (33, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1]) constraint holds.
Figure 4.1407: Illustrating the SUM_SURF_ZIGZAG constraint of the Example slot

Typical

\[|\text{VARIABLES}| > 3\]

\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

Symmetry

Items of \text{VARIABLES} can be reversed.

Arg. properties

Functional dependency: \text{VALUE} determined by \text{VARIABLES}. 
Automaton

Figures 4.1408 and 4.1409 respectively depict the automaton associated with the constraint SUM_SURF_ZIGZAG and its simplified form.
Figure 4.1408: Automaton for the SUM_SURF_ZIGZAG constraint obtained by applying decoration Table 3.37 to the seed transducer of the ZIGZAG pattern where default is 0; (1) missing transitions from a, b, c, d, e, f to s are labelled by $=$; (2) on transitions from b, c, e, f to s the register $D$ is reset to its initial value; (3) on transitions from c, f to s the register $R$ is updated wrt $C$ and the register $C$ is reset to its initial value.
Figure 4.1409: Simplified automaton for the SUM_SURF_ZIGZAG constraint obtained by applying decoration Table 3.29 to the seed transducer of the ZIGZAG pattern where default is 0; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the register $D$ is reset to its initial value.
Table 4.353: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the SUM_SURF_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.354: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the simplified automaton of the `SUM_SURF_ZIGZAG` constraint defined as the composition of the `ZIGZAG` pattern, the feature `SURF`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_SURF_ZIGZAG

2935
SUM_WIDTH_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the DECREASING_SEQUENCE pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>SUM_WIDTH_DECREASING_SEQUENCE(VALUE, VARIABLES)</td>
</tr>
<tr>
<td>Arguments</td>
<td>VALUE : dvar, VARIABLES : collection(var−dvar)</td>
</tr>
</tbody>
</table>
| Restrictions | sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0  
VALUE = 0 ∨ VALUE ≥ 2  
rv = 2 ⇒ VALUE ≤ 2 + np  
rv ≥ 3 ⇒ VALUE ≤ sv  
required(VARIABLES, var)  
where  
rv = range(VARIABLES, var)  
sv = |VARIABLES|  
np = ⌊sv/2⌋  
VALUE is the sum of the width of occurrences of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.  
Purpose | An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘>(>||=)’>(|’.
Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position i and ends at position j. The feature WIDTH computes the value j − i + 2. |
| Example     | (9, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) |
| Typical     | |VARIABLES| > 1  
range(VARIABLES, var) > 1 |
| Symmetry    | One and the same constant can be added to the var attribute of all items of VARIABLES. |
| Arg. properties | Functional dependency: VALUE determined by VARIABLES. |
Figure 4.1410: Illustrating the SUM_WIDTH_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.1411 and 4.1412 respectively depict the automaton associated with the constraint `SUM_WIDTH_DECREASING_SEQUENCE` and its simplified form.

\[
\begin{align*}
\text{Table 4.355: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the } & \text{SUM_WIDTH_DECREASING_SEQUENCE constraint defined as the composition of the } \text{DECREASING_SEQUENCE pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.} \\
\end{align*}
\]

<table>
<thead>
<tr>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td>t</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
</tbody>
</table>

Figure 4.1411: Automaton for the SUM_WIDTH_DECREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING_SEQUENCE pattern where default is 0.
Figure 4.1412: Simplified automaton for the \texttt{SUM\_WIDTH\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 3.30 to the seed transducer of the \texttt{DECREASING\_SEQUENCE} pattern where \texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-1} + D_{i-1} + 1 \geq 0 \) are linear invariants.

Table 4.356: Concrete glue matrix, derived from the parametrised glue matrix 3.5, for the simplified automaton of the \texttt{SUM\_WIDTH\_DECREASING\_SEQUENCE} constraint defined as the composition of the \texttt{DECREASING\_SEQUENCE} pattern, the feature \texttt{WIDTH}, and the aggregator \texttt{sum}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.1413: (top) Automata without registers for the \texttt{SUM\_WIDTH\_DECREASING\_SEQUENCE\_EQ\_UP\_WHEN\_RANGE\_EQ\_2} and the \texttt{SUM\_WIDTH\_DECREASING\_SEQUENCE\_EQ\_UP} constraints; they describe all sequences maximising the sum of the widths of all the occurrences of the \texttt{DECREASING\_SEQUENCE} pattern of a sequence of \texttt{sv} variables when the difference between the maximum and the minimum of the variables plus one of the sequence of variables is (A) equal to 2, i.e. \( 2 \times \left\lfloor \frac{\texttt{sv}}{2} \right\rfloor \) of the \texttt{Restrictions} slot (see (1)); (B) strictly greater than 2, i.e. \texttt{sv} (see (2)). Within (A) states \( s, s' \) and \( t' \) are accepting when \( sv \mod 2 = 1 \), while state \( t \) is accepting when \( sv \mod 2 = 0 \). (bottom) All corresponding solutions for \( sv - 1 \in \{1, 2, 3, 4\} \).
Figure 4.1414: Automata without registers for the (A) \textsc{sum\_width\_decreasing\_sequence\_is\_even} and the (B) \textsc{sum\_width\_decreasing\_sequence\_is\_odd} constraints; they respectively achieve an even/odd sum of width of the \textsc{decreasing\_sequence} pattern on a sequence of $n$ variables; the second symbol represents whether the register $D$ is even or odd, while the third symbol denotes whether the register $R$ is even or odd.
SUM_WIDTH_DECREEASING_TERRACE

**Origin**
Based on the DECREASING_TERRACE pattern.

**Constraint**
SUM_WIDTH_DECREEASING_TERRACE(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
sv ≤ 3 ∨ rv ≤ 2 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 2
VALUE ≤ max(0, sv − 2v)
required(VARIABLES, var)

where
sv = |VARIABLES|
rv = range(VARIABLES.var)

VALUE is the sum of the width of occurrences of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

**Purpose**
An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression ‘>= +’.
Assume that the occurrence of the pattern DECREASING_TERRACE starts at position i and ends at position j. The feature WIDTH computes the value j − i.

**Example**
(4, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3))

Figure 4.1415 provides an example where the SUM_WIDTH_DECREEASING_TERRACE (4, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3]) constraint holds.

**Typical**
|VARIABLES| > 3
range(VARIABLES.var) > 2

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1415: Illustrating the SUM_WIDTH_DECREASING_TERRACE constraint of the Example slot
Figures 4.1416 and 4.1417 respectively depict the automaton associated with the constraint SUM_WIDTH_DECREASING_TERRACE and its simplified form.

Figure 4.1416: Automaton for the SUM_WIDTH_DECREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the DECREASING_TERRACE pattern where default is 0.

Table 4.357: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the SUM_WIDTH_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.1417: Simplified automaton for the `SUM_WIDTH_DECREASING_TERRACE` constraint obtained by applying decoration Table 3.29 to the seed transducer of the `DECREASING_TERRACE` pattern where `default` is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_{i-1} + 1 \geq 0 \) are linear invariants.

Table 4.358: Concrete glue matrix, derived from the parametrised glue matrix 3.6, for the simplified automaton of the `SUM_WIDTH_DECREASING_TERRACE` constraint defined as the composition of the `DECREASING_TERRACE` pattern, the feature `WIDTH`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Specialisation

Figure 4.1418: (left) Automaton without registers for the `SUM_WIDTH_DECREASING_TERRACE_EQ_UP` constraint; it describes all sequences maximising the sum of the widths of all the occurrences of the `DECREASING_TERRACE` pattern of a sequence of `sv` variables, i.e. `max(0, sv - 2)` of the `Restrictions` slot (see 1). (right) All corresponding solutions for `sv - 1 ∈ {1, 2, 3, 4}`.

Figure 4.1419: Automata without registers for the (A) `SUM_WIDTH_DECREASING_TERRACE_IS_EVEN` and the (B) `SUM_WIDTH_DECREASING_TERRACE_IS_ODD` constraints; they respectively achieve an even/odd sum of width of the `DECREASING_TERRACE` pattern on a sequence of `n` variables; the second symbol represents whether the register `D` is even or odd, while the third symbol denotes whether the register `R` is even or odd.
SUM_WIDTH_DECREASING_TERRACE

2947
Description

Origin
Based on the GORGE pattern.

Constraint
\[ \text{SUM}_{\text{WIDTH}}_{\text{GORGE}}(\text{VALUE}, \text{VARIABLES}) \]

Arguments
- VALUE : dvar
- VARIABLES : collection(var−dvar)

Restrictions
- \[ \text{sv} \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \]
- \[ \text{VALUE} = 0 \lor \text{VALUE} \geq 1 \]
- \[ \text{rv} = 2 \Rightarrow \text{VALUE} \leq \text{np} \]
- \[ \text{rv} \geq 3 \Rightarrow \text{VALUE} \leq \max(0, \text{sv} - 2) \]

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

where
- \[ \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \]
- \[ \text{sv} = |\text{VARIABLES}| \]
- \[ \text{np} = \max(0, |(\text{sv} - 1)/2|) \]

Purpose
An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \((> | > (= | >)^* >)(< | < (= | <)^* <)\).

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example
\[(6, (1, 7, 3, 4, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7))\]

Figure 4.1420 provides an example where the SUM_{WIDTH}_{GORGE} \((6, [1, 7, 3, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7])\) constraint holds.

Typical
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

Symmetries
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1420: Illustrating the \texttt{SUM\_WIDTH\_GORGE} constraint of the \textbf{Example} slot
Automaton

Figures 4.1421 and 4.1422 respectively depict the automaton associated with the constraint SUM_WIDTH_GORGE and its simplified form.

Figure 4.1421: Automaton for the SUM_WIDTH_GORGE constraint obtained by applying decoration Table 3.37 to the seed transducer of the GORGE pattern where default is 0 (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
Figure 4.122: Simplified automaton for the \texttt{SUM\_WIDTH\_GORGE} constraint obtained by applying decoration Table 3.26 to the seed transducer of the \texttt{GORGE} pattern where \texttt{default} is 0 (transition \texttt{u} $\rightarrow$ \texttt{r} has the same register update as transition \texttt{r} $\rightarrow$ \texttt{u}); $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_{i-1} + 1 \geq 0$ are linear invariants.
Table 4.359: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the SUM_WIDTH_GORGE constraint defined as the composition of the GORGE pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>t</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>u</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
</tbody>
</table>

Table 4.360: Concrete glue matrix, derived from the parametrised glue matrix 3.7, for the simplified automaton of the SUM_WIDTH_GORGE constraint defined as the composition of the GORGE pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{C}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>u</td>
<td>0</td>
<td>0</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>0</td>
</tr>
</tbody>
</table>
Specialisation

Figure 4.1423: (top) Automata without registers for the SUM_WIDTH_GORGE_EQ_UP_WHEN_RANGE_EQ_2 and the SUM_WIDTH_GORGE_EQ_UP constraints; they describe all sequences maximising the sum of the widths of all the occurrences of the GORGE pattern of a sequence of $sv$ variables when the difference between the maximum and the minimum of the variables plus one of the sequence of variables is (A) equal to 2, i.e. $\max(0, \frac{sv - 1}{2})$ of the Restrictions slot (see 1), (B) strictly greater than 2, i.e. $\max(0, sv - 2)$ (see 2). Within (A) states $s$ and $t$ are accepting when $sv \mod 2 = 1$, while states $r$ and $t'$ are accepting when $sv \mod 2 = 0$. (bottom) All corresponding solutions for $sv - 1 \in \{1, 2, 3, 4\}$. 
Figure 4.1424: Automata without registers for the (A) SUM_WIDTH_GORGE_IS_EVEN and the (B) SUM_WIDTH_GORGE_IS_ODD constraints; they respectively achieve an even/odd sum of width of the GORGE pattern on a sequence of $n$ variables; the second symbol represents whether the register D is even or odd, while the third symbol denotes whether the register R is even or odd.
SUM_WIDTH_INCREASING_SEQUENCE

Origin
Based on the INCREASING_SEQUENCE pattern.

Constraint
SUM_WIDTH_INCREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 2
rv = 2 ⇒ VALUE ≤ 2 + np
rv ≥ 3 ⇒ VALUE ≤ sv

where
rv = range(VARIABLES.var)
sv = |VARIABLES|
np = ⌊sv/2⌋

Purpose
An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘< (< | =)* < | ’.
Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position i and ends at position j. The feature WIDTH computes the value j − i + 2.

Example
(9, ⟨4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3⟩)

Figure 4.1425 provides an example where the SUM_WIDTH_INCREASING_SEQUENCE (9, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES.var) > 1

Symmetry
One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1425: Illustrating the SUM_WIDTH_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.1426 and 4.1427 respectively depict the automaton associated with the constraint SUM_WIDTH_INCREASING_SEQUENCE and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*}
\]  
\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow R + C
\end{align*}
\]  
\[
\begin{align*}
C &\leftarrow D + 2 \\
D &\leftarrow 0
\end{align*}
\]  
\[
\begin{align*}
C &\leftarrow C + D + 1 \\
D &\leftarrow 0
\end{align*}
\]

Figure 4.1426: Automaton for the SUM_WIDTH_INCREASING_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0.

\[
\begin{array}{|c|c|}
\hline
s & t \\
\hline
s & \overline{C} + \overline{C} & \overline{C} + \overline{C} \\
\hline
\overline{C} + C & \overline{C} + \overline{C} + \overline{D} + \overline{D} - 1^M \\
\hline
\end{array}
\]

Table 4.361: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the SUM_WIDTH_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.127: Simplified automaton for the `SUM_WIDTH_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 3.30 to the seed transducer of the `INCREASINGSEQUENCE` pattern where `default` is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_{i-1} + 1 \geq 0 \) are linear invariants.

Table 4.362: Concrete glue matrix, derived from the parametrised glue matrix 3.9, for the simplified automaton of the `SUM_WIDTH_INCREASING_SEQUENCE` constraint defined as the composition of the `INCREASINGSEQUENCE` pattern, the feature `WIDTH`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Specialisation

Figure 4.1428: (top) Automata without registers for the SUM_WIDTH_INCREASING_SEQUENCE_EQ_UP_WHEN_RANGE_EQ_2 and the SUM_WIDTH_INCREASING_SEQUENCE_EQ_UP constraints; they describe all sequences maximising the sum of the widths of all the occurrences of the INCREASING_SEQUENCE pattern of a sequence of sv variables when the difference between the maximum and the minimum of the variables plus one of the sequence of variables is (A) equal to 2, i.e. $2 \cdot \left\lfloor \frac{sv}{2} \right\rfloor$ of the Restrictions slot (see 1), (B) strictly greater than 2, i.e. $sv$ (see 2). Within (A) states $s$, $s'$ and $t'$ are accepting when $sv \mod 2 = 1$, while state $t$ is accepting when $sv \mod 2 = 0$. (bottom) All corresponding solutions for $sv - 1 \in \{1, 2, 3, 4\}$. 

\begin{align*}
\text{States} & \quad \text{Conditions} \\
< & \quad s, s', t' \\
\geq & \quad \geq s'' \leq t' \\
> & \quad > s' \\
= & \quad = \\
\text{(A)} & \\
\text{(B)}
\end{align*}
SUM_WIDTH_INCREASING_SEQUENCE

Figure 4.1429: Automata without registers for the (A) SUM_WIDTH_INCREASING_SEQUENCE_IS_EVEN and the (B) SUM_WIDTH_INCREASING_SEQUENCE_IS_ODD constraints; they respectively achieve an even/odd sum of width of the INCREASING_SEQUENCE pattern on a sequence of n variables; the second symbol represents whether the register D is even or odd, while the third symbol denotes whether the register R is even or odd.
SUM_WIDTH_INCREASING_TERRACE

Origin
Based on the INCREASING_TERRACE pattern.

Constraint
SUM_WIDTH_INCREASING_TERRACE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 2 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 2
VALUE ≤ max(0, sv − 2v)
required(VARIABLES, var)
where
sv = |VARIABLES|
rv =range(VARIABLES.var)

Purpose
An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression ‘<\(+<\)’. Assume that the occurrence of the pattern INCREASING_TERRACE starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example
(5, (1, 3, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4))

Figure 4.1430 provides an example where the SUM_WIDTH_INCREASING_TERRACE (5, [1, 3, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4]) constraint holds.

Typical
|VARIABLES| > 3
range(VARIABLES.var) > 2

Symmetry
One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1430: Illustrating the SUM_WIDTH_INCREASING_TERRACE constraint of the Example slot
Automaton

Figures 4.1431 and 4.1432 respectively depict the automaton associated with the constraint SUM_WIDTH_INCREASING_TERRACE and its simplified form.

\[
\begin{align*}
  &\{ \begin{array}{l}
    C \leftarrow \text{default} \\
    D \leftarrow 0 \\
    R \leftarrow \text{default}
  \end{array} \} \\
  &\{ D \leftarrow 0 \} \\
  \end{align*}
\]

Figure 4.1431: Automaton for the SUM_WIDTH_INCREASING_TERRACE constraint obtained by applying decoration Table 3.37 to the seed transducer of the INCREASING_TERRACE pattern where default is 0.

Table 4.363: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the SUM_WIDTH_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.1432: Simplified automaton for the SUM_WIDTH_INCREASING_TERRACE constraint obtained by applying decoration Table 3.29 to the seed transducer of the INCREASING_TERRACE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_{i-1} + 1 \geq 0 \) are linear invariants.

Table 4.364: Concrete glue matrix, derived from the parametrised glue matrix 3.10, for the simplified automaton of the SUM_WIDTH_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature WIDTH, and the aggregator sum: cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Specialisation

Figure 4.1433: (left) Automaton without registers for the SUM_WIDTH_INCREASING_TERRACE_EQ_UP constraint; it describes all sequences maximising the sum of the widths of all the occurrences of the INCREASING_TERRACE pattern of a sequence of \( sv \) variables, i.e. \( \max(0, sv - 2) \) of the Restrictions slot (see 1). (right) All corresponding solutions for \( sv - 1 \in \{1, 2, 3, 4\} \).

Figure 4.1434: Automata without registers for the SUM_WIDTH_INCREASING_TERRACE_IS_EVEN and the (B) SUM_WIDTH_INCREASING_TERRACE_IS_ODD constraints; they respectively achieve an even/odd sum of width of the INCREASING_TERRACE pattern on a sequence of \( n \) variables; the second symbol represents whether the register \( D \) is even or odd, while the third symbol denotes whether the register \( R \) is even or odd.
**SUM_WIDTH_INFLEXION**

### Origin
Based on the **INFLEXION** pattern.

### Constraint
\[
\text{SUM_WIDTH_INFLEXION}(\text{VALUE, VARIABLES})
\]

### Arguments
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

### Restrictions
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \)
- \( \text{VALUE} = 0 \lor \text{VALUE} \geq 1 \)
- \( \text{VALUE} \leq \max(0, sv - 2v) \)
- \( \text{required}(\text{VARIABLES, var}) \)

where
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES.var}) \)

### Purpose
An occurrence of the pattern **INFLEXION** is the maximal subsequence which matches the regular expression '\(< (< | =) > | > (> | =) > \)'.
Assume that the occurrence of the pattern **INFLEXION** starts at position \( i \) and ends at position \( j \). The feature **WIDTH** computes the value \( j - i \).

### Example
\[
(13, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))
\]

Figure 4.1435 provides an example where the **SUM_WIDTH_INFLEXION** constraint holds.

### Typical
- \( |\text{VARIABLES}| > 2 \)
- \( \text{range}(\text{VARIABLES.var}) > 1 \)

### Symmetries
- Items of **VARIABLES** can be reversed.
- One and the same constant can be added to the **var** attribute of all items of **VARIABLES**.

### Arg. properties
- Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 4.1435: Illustrating the `SUM_WIDTH_INFLEXION` constraint of the Example slot
Automaton

Figures 4.1436 and 4.1437 respectively depict the automaton associated with the constraint \texttt{SUM_WIDTH_INFLEXION} and its simplified form.

Figure 4.1436: Automaton for the \texttt{SUM_WIDTH_INFLEXION} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \texttt{INFLEXION} pattern where \texttt{default} is 0 (transition \(r \rightarrow t\) has the same registers updates as transition \(t \rightarrow r\)).

Figure 4.1437: Simplified automaton for the \texttt{SUM_WIDTH_INFLEXION} constraint obtained by applying decoration Table 3.26 to the seed transducer of the \texttt{INFLEXION} pattern where \texttt{default} is 0 (transition \(r \rightarrow t\) has the same registers updates as transition \(t \rightarrow r\)); \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-1} + D_{i-1} + 1 \geq 0\) are linear invariants.
Specialisation

Figure 4.1438:  (top) Automaton without registers for the SUM_WIDTH_INFLEXION_EQ_UP constraint; it describes all sequences maximising the sum of the widths of all the occurrences of the INFLEXION pattern of a sequence of $sv$ variables, i.e. $\max(0, sv - 2)$ of the Restrictions slot (see 3). (bottom) All corresponding solutions for $sv - 1 \in \{1, 2, 3\}$. 
Figure 4.1439: Automaton without registers for the $\text{SUM}_\text{WIDTH}_\text{INFLEXION}_\text{IS}_\text{EVEN}$ constraint; it achieves an even sum of width of the $\text{INFLEXION}$ pattern on a sequence of $n$ variables; within the name of each state the second symbol represents whether the register $D$ is even or odd, while the third symbol denotes whether the register $R$ is even or odd. The automaton for the $\text{SUM}_\text{WIDTH}_\text{INFLEXION}_\text{IS}_\text{ODD}$ constraint achieves an odd sum of width of the $\text{INFLEXION}$ pattern on a sequence of $n$ variables; it is obtained by switching the accepting and the non-accepting states.
**SUM_WIDTH_PEAK**

**DESCRIPTION**

**Origin**  
Based on the **PEAK** pattern.

**Constraint**  
**SUM_WIDTH_PEAK**(VALUE, VARIABLES)

**Arguments**  
VALUE : dvar  
VARIABLES : collection(var−dvar)

**Restrictions**  
\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE = 0 \lor VALUE \geq 1 \]
\[ VALUE \leq \max(0, sv - 2v) \]
\[ required(VARIABLES, var) \]
where
\[ sv = |VARIABLES| \]
\[ rv = range(VARIABLES, var) \]

**Purpose**  
An occurrence of the pattern **PEAK** is the maximal subsequence which matches the regular expression `'<(=|<)*(>||=)*>'`.  
Assume that the occurrence of the pattern **PEAK** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i\).

**Example**  
\((8, (7,5,5,1,4,5,2,2,3,5,6,2,3,3,1))\)

Figure 4.1440 provides an example where the **SUM_WIDTH_PEAK** \((8, [7,5,5,1,4,5,2,2,3,5,6,2,3,3,1])\) constraint holds.

**Typical**  
\[ |VARIABLES| > 2 \]
\[ range(VARIABLES, var) > 1 \]

**Symmetries**  
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**  
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1440: Illustrating the SUM_WIDTH_PEAK constraint of the Example slot
Automaton

Figures 4.1441 and 4.1442 respectively depict the automaton associated with the constraint SUMWIDTH_PEAK and its simplified form.

Figure 4.1441: Automaton for the SUMWIDTH_PEAK constraint obtained by applying decoration Table 3.37 to the seed transducer of the PEAK pattern where default is 0;

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*}
\]

≥ s

≥

\{D \leftarrow D + 1\}

Figure 4.1442: Simplified automaton for the SUMWIDTH_PEAK constraint obtained by applying decoration Table 3.26 to the seed transducer of the PEAK pattern where default is 0; \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-1} + D_{i-1} + 1 \geq 0\) are linear invariants.
Table 4.365: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the \textit{SUM}_\textit{WIDTH}_\textit{PEAK} constraint defined as the composition of the \textit{PEAK} pattern, the \textit{feature WIDTH}, and the \textit{aggregator sum}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\mathcal{C} + \mathcal{C}$</td>
<td>$\mathcal{C} + \mathcal{C}$</td>
<td>$\mathcal{C} + \mathcal{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\mathcal{C} + \mathcal{C}$</td>
<td>$\mathcal{D} + \mathcal{D} + 1$</td>
<td>$\mathcal{C} + \mathcal{D} + \mathcal{D} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>$\mathcal{C} + \mathcal{C}$</td>
<td>$\mathcal{C} + \mathcal{D} + \mathcal{D} + 1$</td>
<td>$\mathcal{C} + \mathcal{C}$</td>
</tr>
</tbody>
</table>

Table 4.366: Concrete glue matrix, derived from the parametrised glue matrix 3.11, for the simplified automaton of the \textit{SUM}_\textit{WIDTH}_\textit{PEAK} constraint defined as the composition of the \textit{PEAK} pattern, the \textit{feature WIDTH}, and the \textit{aggregator sum}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>$\mathcal{D} + \mathcal{D} + 1$</td>
<td>$\mathcal{D} + \mathcal{D} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>$\mathcal{D} + \mathcal{D} + 1$</td>
<td>0</td>
</tr>
</tbody>
</table>
Specialisation

\[
\begin{align*}
\text{s} &< \text{r} & \text{r} &< \text{t} \\
\geq &< &< \leq &< \leq &< \leq &< \\
\geq &< &< &< &< &< &< \\
\end{align*}
\]

Figure 4.1443: (top) Automaton without registers for the SUM_WIDTH_PEAK_EQ_UP constraint; it describes all sequences maximising the sum of the widths of all the occurrences of the PEAK pattern of a sequence of \(s v\) variables, i.e. \(\max(0, sv - 2)\) of the Restrictions slot (see \(\text{(1)}\)). (bottom) All corresponding solutions for \(sv - 1 \in \{4, 5\}\).

\[
\begin{align*}
\text{sEE} &< \text{rEE} & \text{rEO} &< \text{rEO} \\
\geq &< &< &< &< &< &< \\
\geq &< &< &< &< &< &< \\
\end{align*}
\]

Figure 4.1444: Automata without registers for the (A) SUM_WIDTH_PEAK_IS_EVEN and the (B) SUM_WIDTH_PEAK_IS_ODD constraints; they respectively achieve an even/odd sum of width of the PEAK pattern on a sequence of \(n\) variables; the second symbol represents whether the register D is even or odd, while the third symbol denotes whether the register R is even or odd.
SUM_WIDTH_PEAK 2979
SUM_WIDTHPLAIN

DESCRIPTION

Origin
Based on the PLAIN pattern.

Constraint
SUM_WIDTHPLAIN(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var − dvar)

Restrictions
\( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
VALUE = 0 \lor VALUE \geq 1
VALUE \leq \max(0, sv - 2k)
required(VARIABLES, var)
where
\( sv = |\text{VARIABLES}| \)
\( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

Purpose
An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘\(>\)\(\ast\)<’.
Assume that the occurrence of the pattern PLAIN starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example
\[(4, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3))\]

Figure 4.1445 provides an example where the SUM_WIDTHPLAIN (4, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3]) constraint holds.

Typical
\[|\text{VARIABLES}| > 2 \]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Symmetries
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1445: Illustrating the SUM_WIDTH.PLAIN constraint of the Example slot
Figures 4.1446 and 4.1447 respectively depict the automaton associated with the constraint \textsc{SUM\_WIDTH\_PLAIN} and its simplified form.

**Automaton**

Figures 4.1446 and 4.1447 respectively depict the automaton associated with the constraint \textsc{SUM\_WIDTH\_PLAIN} and its simplified form.

**Figure 4.1446:** Automaton for the \textsc{SUM\_WIDTH\_PLAIN} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsc{PLAIN} pattern where \texttt{default} is 0.

**Figure 4.1447:** Simplified automaton for the \textsc{SUM\__WIDTH\_PLAIN} constraint obtained by applying decoration Table 3.29 to the seed transducer of the \textsc{PLAIN} pattern where \texttt{default} is 0; \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-1} + D_{i-1} + 1 \geq 0\) are linear invariants.
Table 4.367: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the \texttt{SUM\_WIDTH\_PLAIN} constraint defined as the composition of the \texttt{PLAIN} pattern, the feature \texttt{WIDTH}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
  & \(s\) & \(r\) & \(t\) \\
\hline
\(s\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) \\
\(r\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{D} + \overrightarrow{D} + 1\) & \(\overrightarrow{C} + \overrightarrow{D} + 1\) \\
\(t\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{D} + \overrightarrow{D} + 1\) & \(\overrightarrow{C} + \overrightarrow{D} + 1\) \\
\hline
\end{tabular}
\caption{Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the \texttt{SUM\_WIDTH\_PLAIN} constraint defined as the composition of the \texttt{PLAIN} pattern, the feature \texttt{WIDTH}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.}
\end{table}

Table 4.368: Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the simplified automaton of the \texttt{SUM\_WIDTH\_PLAIN} constraint defined as the composition of the \texttt{PLAIN} pattern, the feature \texttt{WIDTH}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
  & \(s\) & \(r\) & \(t\) \\
\hline
\(s\) & 0 & 0 & 0 \\
\(r\) & 0 & \(\overrightarrow{D} + \overrightarrow{D} + 1\) & \(\overrightarrow{C} + \overrightarrow{D} + 1\) \\
\(t\) & 0 & \(\overrightarrow{D} + \overrightarrow{D} + 1\) & \(\overrightarrow{C} + \overrightarrow{D} + 1\) \\
\hline
\end{tabular}
\caption{Concrete glue matrix, derived from the parametrised glue matrix 3.12, for the simplified automaton of the \texttt{SUM\_WIDTH\_PLAIN} constraint defined as the composition of the \texttt{PLAIN} pattern, the feature \texttt{WIDTH}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.}
\end{table}
Specialisation

\[ s \preceq s' \]

\[ s < r \]

\[ s = r \]

\[ s > r \]

\[ > < \]

\[ > = < \]

\[ > = = < \]

Figure 4.1448: **(left)** Automaton without registers for the `SUM_WIDTH_PLAIN_EQ_UP` constraint; it describes all sequences maximising the sum of the widths of all the occurrences of the `PLAIN` pattern of a sequence of `sv` variables, i.e. \( \max(0, sv - 2) \) of the **Restrictions** slot (see 1). **(right)** All corresponding solutions for `sv - 1` \( \in \{1, 2, 3, 4\} \).

Figure 4.1449: Automata without registers for the **(A)** `SUM_WIDTH_PLAIN_IS_EVEN` and the **(B)** `SUM_WIDTH_PLAIN_IS_ODD` constraints; they respectively achieve an even/odd sum of width of the `PLAIN` pattern on a sequence of `n` variables; the second symbol represents whether the register `D` is even or odd, while the third symbol denotes whether the register `R` is even or odd.
SUM_WIDTH_PLATEAU

**Origin**
Based on the PLATEAU pattern.

**Constraint**
SUM_WIDTH_PLATEAU(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**
- $sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0$
- $VALUE = 0 \lor VALUE \geq 1$
- $VALUE \leq \text{max}(0, sv − 2d)$
- $\text{required}(\text{VARIABLES}, \text{var})$

where
- $sv = |\text{VARIABLES}|$
- $rv = \text{range}(\text{VARIABLES}.\text{var})$

VALUE is the sum of the width of occurrences of the PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

**Purpose**
An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression ‘< = * >’.
Assume that the occurrence of the pattern PLATEAU starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j − i$.

**Example**
$(7, (1,3,3,5,5,5,2,4,4,4,3,3,1,5,5))$

Figure 4.1450 provides an example where the SUM_WIDTH_PLATEAU $(7, [1,3,3,5,5,5,2,4,4,4,3,3,1,5,5])$ constraint holds.

**Typical**
- $|\text{VARIABLES}| > 2$
- $\text{range}(\text{VARIABLES}.\text{var}) > 1$

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1450: Illustrating the SUM_WIDTH_PLATEAU constraint of the Example slot
Automaton

Figures 4.1451 and 4.1452 respectively depict the automaton associated with the constraint \textsc{sum\_width\_plateau} and its simplified form.

Figure 4.1451: Automaton for the \textsc{sum\_width\_plateau} constraint obtained by applying decoration Table 3.37 to the seed transducer of the \textsc{plateau} pattern where \texttt{default} is 0.

Figure 4.1452: Simplified automaton for the \textsc{sum\_width\_plateau} constraint obtained by applying decoration Table 3.29 to the seed transducer of the \textsc{plateau} pattern where \texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-1} + D_{i-1} + 1 \geq 0\) are linear invariants.
Table 4.369: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the `SUM_WIDTH_PLATEAU` constraint defined as the composition of the `PLATEAU` pattern, the feature `WIDTH`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
<tr>
<td>t</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
</tbody>
</table>

Table 4.370: Concrete glue matrix, derived from the parametrised glue matrix 3.13, for the simplified automaton of the `SUM_WIDTH_PLATEAU` constraint defined as the composition of the `PLATEAU` pattern, the feature `WIDTH`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
</tbody>
</table>
Specialisation

Figure 4.1453: (left) Automaton without registers for the SUM_WIDTH_PLATEAU_EQ_UP constraint; it describes all sequences maximising the sum of the widths of all the occurrences of the PLATEAU pattern of a sequence of sv variables, i.e. $\max(0, sv - 2)$ of the Restrictions slot (see 4). (right) All corresponding solutions for $sv - 1 \in \{1, 2, 3, 4\}$.

Figure 4.1454: Automata without registers for the (A) SUM_WIDTH_PLATEAU_IS_EVEN and the (B) SUM_WIDTH_PLATEAU_IS_ODD constraints; they respectively achieve an even/odd sum of width of the PLATEAU pattern on a sequence of n variables; the second symbol represents whether the register D is even or odd, while the third symbol denotes whether the register R is even or odd.
SUM_WIDTH_PLATEAU

2991
**SUM_WIDTH_PROPERPLAIN**

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <code>PROPERPLAIN</code> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>( \text{SUM_WIDTH_PROPERPLAIN}(\text{VALUE, VARIABLES}) )</td>
</tr>
</tbody>
</table>
| **Arguments** | \[
| VALUE : dvar |
| VARIABLES : \( \text{collection}(\text{var} \rightarrow \text{dvar}) \) |
| **Restrictions** | \[
| \( sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = 0 \) |
| \( VALUE = 0 \lor VALUE \geq 2 \) |
| \( VALUE \leq \max(0, sv - 2) \) |
| required(VARIABLES, var) |
| where \[
| sv = |\text{VARIABLES}| |
| rv = \text{range}(\text{VARIABLES}.\text{var}) |
| **Purpose** | An occurrence of the pattern `PROPERPLAIN` is the maximal subsequence which matches the regular expression ‘\( > = + < \)’. Assume that the occurrence of the pattern `PROPERPLAIN` starts at position \( i \) and ends at position \( j \). The feature `WIDTH` computes the value \( j - i \). |
| **Example** | \((7,\langle 2,7,5,6,3,7,4,4,5,6,5,3,3,3,5 \rangle)\) |

Figure 4.1455 provides an example where the `SUM_WIDTH_PROPERPLAIN` (\((7,\langle 2,7,5,6,3,7,4,4,5,6,5,3,3,3,5 \rangle)\)) constraint holds.

**Typical**
- \( |\text{VARIABLES}| > 3 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

**Symmetries**
- Items of `VARIABLES` can be reversed.
- One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**
- **Functional dependency**: `VALUE` determined by `VARIABLES`. 
Figure 4.1455: Illustrating the SUM_WIDTH_PROPER_PLAIN constraint of the Example slot
Automaton

Figures 4.1456 and 4.1457 respectively depict the automaton associated with the constraint `SUM_WIDTH_PROPER_PLAIN` and its simplified form.

Figure 4.1456: Automaton for the `SUM_WIDTH_PROPER_PLAIN` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `PROPER_PLAIN` pattern where `default` is 0.

Figure 4.1457: Simplified automaton for the `SUM_WIDTH_PROPER_PLAIN` constraint obtained by applying decoration Table 3.29 to the seed transducer of the `PROPER_PLAIN` pattern where `default` is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_{i-1} + 1 \geq 0 \) are linear invariants.
Table 4.371: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the SUM_WIDTH_PROPER_PLAIN constraint defined as the composition of the PROPER_PLAIN pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\hat{C} + \hat{C})</td>
<td>(\hat{C} + \hat{C})</td>
<td>(\hat{C} + \hat{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\hat{C} + \hat{C})</td>
<td>(\hat{C} + \hat{C})</td>
<td>(\hat{D} + \hat{D} + 1)</td>
</tr>
<tr>
<td>t</td>
<td>(\hat{C} + \hat{C})</td>
<td>(\hat{D} + \hat{D} + 1)</td>
<td>(\hat{D} + \hat{D} + 1)</td>
</tr>
</tbody>
</table>

Table 4.372: Concrete glue matrix, derived from the parametrised glue matrix 3.14, for the simplified automaton of the SUM_WIDTH_PROPER_PLAIN constraint defined as the composition of the PROPER_PLAIN pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>0</td>
<td>(\hat{D} + \hat{D} + 1)</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>(\hat{D} + \hat{D} + 1)</td>
<td>(\hat{D} + \hat{D} + 1)</td>
</tr>
</tbody>
</table>
Specialisation

Figure 4.1458: (left) Automaton without registers for the SUM_WIDTH_PROPERPLAIN_EQ_UP constraint; it describes all sequences maximising the sum of the widths of all the occurrences of the PROPERPLAIN pattern of a sequence of $sv$ variables, i.e. $\max(0, sv - 2)$ of the Restrictions slot (see ①). (right) All corresponding solutions for $sv - 1 \in \{1, 2, 3, 4\}$.

Figure 4.1459: Automata without registers for the (A) SUM_WIDTH_PROPERPLAIN_IS_EVEN and the (B) SUM_WIDTH_PROPERPLAIN_IS_ODD constraints; they respectively achieve an even/odd sum of width of the PROPERPLAIN pattern on a sequence of $n$ variables; the second symbol represents whether the register $D$ is even or odd, while the third symbol denotes whether the register $R$ is even or odd.
SUM_WIDTH_PROPER_PLATEAU

Description

Origin
Based on the PROPER_PLATEAU pattern.

Constraint
SUM_WIDTH_PROPER_PLATEAU(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 2
VALUE ≤ max(0, sv − 2\(v\))
required(VARIABLES, var)

where
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which
matches the regular expression '<=+>'.
Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \(i\) and
ends at position \(j\). The feature WIDTH computes the value \(j − i\).

Example
(7, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))

Figure 4.1460 provides an example where the SUM_WIDTH_PROPER_PLATEAU
(7, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3]) constraint holds.

Typical
|VARIABLES| > 3
range(VARIABLES.var) > 1

Symmetries
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 4.1460: Illustrating the SUM_WIDTH_PROPER_PLATEAU constraint of the Example slot
Automaton

Figures 4.1461 and 4.1462 respectively depict the automaton associated with the constraint SUM_WIDTH_PROPER_PLATEAU and its simplified form.

Figure 4.1461: Automaton for the SUM_WIDTH_PROPER_PLATEAU constraint obtained by applying decoration Table 3.37 to the seed transducer of the PROPER_PLATEAU pattern where default is 0.

Figure 4.1462: Simplified automaton for the SUM_WIDTH_PROPER_PLATEAU constraint obtained by applying decoration Table 3.29 to the seed transducer of the PROPER_PLATEAU pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_{i-1} + 1 \geq 0$ are linear invariants.
Table 4.373: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the SUM_WIDTH_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{D} + \overline{D} + 1)</td>
</tr>
<tr>
<td>t</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{D} + \overline{D} + 1)</td>
<td>(\overline{D} + \overline{D} + 1)</td>
</tr>
</tbody>
</table>

Table 4.374: Concrete glue matrix, derived from the parametrised glue matrix 3.15, for the simplified automaton of the SUM_WIDTH_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>0</td>
<td>(\overline{D} + \overline{D} + 1)</td>
</tr>
<tr>
<td>t</td>
<td>(\overline{D} + \overline{D} + 1)</td>
<td>(\overline{D} + \overline{D} + 1)</td>
<td>(\overline{D} + \overline{D} + 1)</td>
</tr>
</tbody>
</table>
Specialisation

Figure 4.1463: (left) Automaton without registers for the `SUM_WIDTH_PROPER_PLATEAU_EQ_UP` constraint; it describes all sequences maximising the sum of the widths of all the occurrences of the `PROPER_PLATEAU` pattern of a sequence of `sv` variables, i.e. \( \max(0, sv - 2) \) of the `Restrictions` slot (see 1). (right) All corresponding solutions for `sv - 1 \in \{1, 2, 3, 4\}`.

Figure 4.1464: Automata without registers for the (A) `SUM_WIDTH_PROPER_PLATEAU_IS_EVEN` and the (B) `SUM_WIDTH_PROPER_PLATEAU_IS_ODD` constraints; they respectively achieve an even/odd sum of width of the `PROPER_PLATEAU` pattern on a sequence of `n` variables; the second symbol represents whether the register \( D \) is even or odd, while the third symbol denotes whether the register \( R \) is even or odd.
**SUM_WIDTH_STEADY_SEQUENCE**

**Origin**
Based on the `STEADY_SEQUENCE` pattern.

**Constraint**
`SUM_WIDTH_STEADY_SEQUENCE(VALUE, VARIABLES)`

**Arguments**
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

**Restrictions**
- $sv \leq 1 \Rightarrow VALUE = 0$
- $rv = 1 \Rightarrow VALUE \geq sv$
- $rv \geq 2 \Rightarrow VALUE = 0 \lor VALUE \geq 2$
- $VALUE \leq sv$
  - `required(VARIABLES, var)`

**Purpose**
An occurrence of the pattern `STEADY_SEQUENCE` is the *maximal* subsequence which matches the regular expression `$'=+'$.
Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `WIDTH` computes the value $j - i + 2$.

**Example**
- `(11, ⟨3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1⟩)`

Figure 4.1465 provides an example where the `SUM_WIDTH_STEADY_SEQUENCE` constraint holds.

**Typical**
- $|VARIABLES| > 1$

**Symmetries**
- Items of `VARIABLES` can be reversed.
- One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**
- Functional dependency: `VALUE` determined by `VARIABLES`. 

---

VALUE is the sum of the width of occurrences of the `STEADY_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value 0.
Figure 4.1465: Illustrating the SUM_WIDTH_STEADY_SEQUENCE constraint of the Example slot
Automaton

Figures 4.1466 and 4.1467 respectively depict the automaton associated with the constraint SUM_WIDTH_STEADY_SEQUENCE and its simplified form.

Figure 4.1466: Automaton for the SUM_WIDTH_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.37 to the seed transducer of the STEADY_SEQUENCE pattern where default is 0.

Figure 4.1467: Simplified automaton for the SUM_WIDTH_STEADY_SEQUENCE constraint obtained by applying decoration Table 3.40 to the seed transducer of the STEADY_SEQUENCE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + 2 \geq 0 \) are linear invariants.
Table 4.375: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the \texttt{SUM\_WIDTH\_STEADY\_SEQUENCE} constraint defined as the composition of the \texttt{STEADY\_SEQUENCE} pattern, the feature \texttt{WIDTH}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{tabular}{|c|c|}
  \hline
  \textbf{s} & \textbf{r} \\
  \hline
  \textbf{s} & \texttt{C} + \texttt{C} & \texttt{C} + \texttt{C} \\
  \textbf{r} & \texttt{C} + \texttt{C} & \texttt{C} + \texttt{C} + \texttt{D} + \texttt{D} - 1 \text{M} \\
  \hline
\end{tabular}

Table 4.376: Concrete glue matrix, derived from the parametrised glue matrix 3.17, for the simplified automaton of the \texttt{SUM\_WIDTH\_STEADY\_SEQUENCE} constraint defined as the composition of the \texttt{STEADY\_SEQUENCE} pattern, the feature \texttt{WIDTH}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

\begin{tabular}{|c|c|}
  \hline
  \textbf{s} & \textbf{r} \\
  \hline
  \textbf{s} & 0 & 0 \\
  \textbf{r} & 0 & -1 \text{M} \\
  \hline
\end{tabular}
Specialisation

\[ s = r \approx s' = \geq s' \]

Figure 4.1468: (top) Automaton without registers for the `SUM_WIDTH_STEADY_SEQUENCE_EQ_UP` constraint; it describes all sequences maximising the sum of the widths of all the occurrences of the `STEADY_SEQUENCE` pattern of a sequence of `sv` variables, i.e. `sv` of the `Restrictions` slot (see ①). (bottom) All corresponding solutions for `sv - 1 \in \{1, 2, 3, 4\}`.

Figure 4.1469: Automaton without registers for the (A) `SUM_WIDTH_STEADY_SEQUENCE_IS_EVEN` and the (B) `SUM_WIDTH_STEADY_SEQUENCE_IS_ODD` constraints; they respectively achieve an even/odd sum of width of the `STEADY_SEQUENCE` pattern on a sequence of `n` variables; within the name of each state the second symbol represents whether the register `R` is even or odd.
SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE

**Description**

**Origin**

Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**

```
SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

**Restrictions**

```
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 2
rv = 2 ⇒ VALUE ≤ sv − sv mod 2
rv ≥ 3 ⇒ VALUE ≤ sv
required(VARIABLES, var)
```

where

```
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

**Purpose**

VALUE is the sum of the width of occurrences of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the maximal sub-sequence which matches the regular expression ‘>\+’.

Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position i and ends at position j. The feature WIDTH computes the value $j - i + 2$.

**Example**

```
(8, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))
```

Figure 4.1470 provides an example where the `SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE` (8,[4,4,6,4,1,1,3,4,4,6,6,5,2,2,4,3]) constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```

**Symmetry**

One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.1470: Illustrating the SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 4.1471 and 4.1472 respectively depict the automaton associated with the constraint `SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE` and its simplified form.

Figure 4.1471: Automaton for the `SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern where `default` is 0

<table>
<thead>
<tr>
<th></th>
<th><code>s</code></th>
<th><code>r</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>s</code></td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td><code>r</code></td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} - 1)</td>
</tr>
</tbody>
</table>

Table 4.377: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the `SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_DECREASING_SEQUENCE` pattern, the feature `WIDTH`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.1472: Simplified automaton for the \texttt{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE} constraint obtained by applying decoration Table 3.40 to the seed transducer of the \texttt{STRICTLY_DECREASING_SEQUENCE} pattern where \texttt{default} is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + 2 \geq 0$ are linear invariants.

Table 4.378: Concrete glue matrix, derived from the parametrised glue matrix 3.18, for the simplified automaton of the \texttt{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE} constraint defined as the composition of the \texttt{STRICTLY_DECREASING_SEQUENCE} pattern, the feature \texttt{WIDTH}, and the aggregator \texttt{sum}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Specialisation

Figure 4.1473: (top) Automata without registers for the \textsc{sum\_width\_strictly\_decreasing\_sequence\_eq\_up\_when\_range\_eq\_2} and the \textsc{sum\_width\_strictly\_decreasing\_sequence\_eq\_up} constraints; they describe all sequences maximising the sum of the widths of all the occurrences of the \textsc{strictly\_decreasing\_sequence} pattern of a sequence of \text{sv} variables when the difference between the maximum and the minimum of the variables plus one of the sequence of variables is (A) equal to 2, i.e. \text{sv} mod 2 of the \textbf{Restrictions} slot (see (1)), (B) strictly greater than 2, i.e. \text{sv} (see (2)). Within (A) states \( s, s' \) and \( t' \) are accepting when \text{sv} mod 2 = 1, while state \( t \) is accepting when \text{sv} mod 2 = 0. (bottom) All corresponding solutions for \text{sv} − 1 ∈ \{1, 2, 3, 4\}. 
Figure 4.1474: Automata without registers for the (A) SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE_IS_EVEN and the (B) SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE_IS_ODD constraints; they respectively achieve an even/odd sum of width of the STRICTLY_DECREASING_SEQUENCE pattern on a sequence of \( n \) variables; the second symbol represents whether the register \( R \) is even or odd.
SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE

**Origin**
Based on the \texttt{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE} pattern.

**Constraint**
\texttt{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)}

**Arguments**
- \texttt{VALUE} : dvar
- \texttt{VARIABLES} : collection variability\texttt{−dvar}

**Restrictions**
- \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE = 0 \lor VALUE \geq 2 \)
- \( rv = 2 \Rightarrow VALUE \leq sv - sv \mod 2 \)
- \( rv \geq 3 \Rightarrow VALUE \leq sv \)
- \( \text{required(VARIABLES, var)} \)
  where
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range(VARIABLES.var)} \)

**Purpose**
VALUE is the sum of the width of occurrences of the \texttt{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern \texttt{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE} is the maximal sub-sequence which matches the regular expression \texttt{<+}. Assume that the occurrence of the pattern \texttt{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE} starts at position \( i \) and ends at position \( j \). The feature \texttt{WIDTH} computes the value \( j - i + 2 \).

**Example**
(10, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))

Figure 4.1475 provides an example where the \texttt{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE} (10, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) constraint holds.

**Typical**
- \( |\text{VARIABLES}| > 1 \)
- \( \text{range(VARIABLES.var)} > 1 \)

**Symmetry**
One and the same constant can be added to the \texttt{var} attribute of all items of \texttt{VARIABLES}.

**Arg. properties**
- Functional dependency: VALUE determined by \texttt{VARIABLES}.
Figure 4.1475: Illustrating the SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Figures 4.1476 and 4.1477 respectively depict the automaton associated with the constraint \( \text{SUM}_{\text{WIDTH}} \text{}_\text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE} \) and its simplified form.

Figure 4.1476: Automaton for the \( \text{SUM}_{\text{WIDTH}} \text{}_\text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE} \) constraint obtained by applying decoration Table 3.37 to the seed transducer of the \( \text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE} \) pattern where \text{default} is 0

Table 4.379: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the \( \text{SUM}_{\text{WIDTH}} \text{}_\text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE} \) constraint defined as the composition of the \( \text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE} \) pattern, the feature \( \text{WIDTH} \), and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.1477: Simplified automaton for the \texttt{SUM WIDTH STRICTLY INCREASING SEQUENCE} constraint obtained by applying decoration Table 3.40 to the seed transducer of the \texttt{STRICTLY INCREASING SEQUENCE} pattern where \texttt{default} is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + 2 \geq 0$ are linear invariants.

Table 4.380: Concrete glue matrix, derived from the parametrised glue matrix 3.19, for the simplified automaton of the \texttt{SUM WIDTH STRICTLY INCREASING SEQUENCE} constraint defined as the composition of the \texttt{STRICTLY INCREASING SEQUENCE} pattern, the feature \texttt{WIDTH}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Specialisation

Figure 4.14.78:  (top) Automata without registers for the \texttt{SUM\_WIDTH\_STRICLY\_INCREASING\_SEQUENCE\_EQ\_UP WHEN RANGE EQ 2} and the \texttt{SUM\_WIDTH\_STRICLY\_INCREASING\_SEQUENCE\_EQ\_UP} constraints; they describe all sequences maximising the sum of the widths of all the occurrences of the \texttt{STRICLY\_INCREASING\_SEQUENCE} pattern of a sequence of $sv$ variables when the difference between the maximum and the minimum of the variables plus one of the sequence of variables is (A) equal to 2 of the Restrictions slot (see \texttt{1}), i.e. $sv - sv \mod 2$, (B) strictly greater than 2, i.e. $sv$ (see \texttt{2}). Within (A) states $s$, $s'$ and $t'$ are accepting when $sv \mod 2 = 1$, while state $t$ is accepting when $sv \mod 2 = 0$. (bottom) All corresponding solutions for $sv - 1 \in \{1, 2, 3, 4\}$. 

\begin{align*}
1 < s' & \quad 1 < s'' \leq t' \\
1 < s' & \quad 1 < s'' \leq t' \\
1 < s' & \quad 1 < s'' \leq t' \\
1 < s' & \quad 1 < s'' \leq t' \\
1 < s' & \quad 1 < s'' \leq t' \\
1 < s' & \quad 1 < s'' \leq t' \\
1 < s' & \quad 1 < s'' \leq t'
\end{align*}
Figure 4.1479: Automata without registers for the (A) SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE_IS_EVEN and the (B) SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE_IS_ODD constraints; they respectively achieve an even/odd sum of width of the STRICTLY_INCREASING_SEQUENCE pattern on a sequence of \( n \) variables; the second symbol represents whether the register \( R \) is even or odd.
**SUM_WIDTH_SUMMIT**

### Description

**Origin**
Based on the SUMMIT pattern.

**Constraint**

\[ \text{SUM_WIDTH_SUMMIT}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`

**Restrictions**

\[ sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \]
\[ \text{VALUE} = 0 \lor \text{VALUE} \geq 1 \]
\[ rv = 2 \Rightarrow \text{VALUE} \leq np \]
\[ rv \geq 3 \Rightarrow \text{VALUE} \leq \max(0, sv - 2\#) \]

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

where

\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]
\[ np = \max(0, (sv - 1)/2) \]

**Purpose**

An occurrence of the pattern `SUMMIT` is the maximal subsequence which matches the regular expression ‘\(\langle | < | (= | <)^* \rangle | > | (= | >)^* \rangle\)’.

Assume that the occurrence of the pattern `SUMMIT` starts at position \(i\) and ends at position \(j\). The feature `WIDTH` computes the value \(j - i\).

**Example**

\((6, \langle 7, 1, 5, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1 \rangle)\)

Figure 4.1480 provides an example where the `SUM_WIDTH_SUMMIT` constraint holds.

**Typical**

\(|\text{VARIABLES}| > 2\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Symmetries**

- Items of `VARIABLES` can be **reversed**.
- One and the same constant can be **added** to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`.

**PL**
Figure 4.1480: Illustrating the SUM_WIDTH_SUMMIT constraint of the Example slot
Automaton

Figures 4.1481 and 4.1482 respectively depict the automaton associated with the constraint `SUM_WIDTH_SUMMIT` and its simplified form.

Figure 4.1481: Automaton for the `SUM_WIDTH_SUMMIT` constraint obtained by applying decoration Table 3.37 to the seed transducer of the `SUMMIT` pattern where `default` is 0 (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$)
Figure 4.1482: Simplified automaton for the SUM_WIDTH_SUMMIT constraint obtained by applying decoration Table 3.26 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same register update as transition $r \rightarrow u$); $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_{i-1} + 1 \geq 0$ are linear invariants.
Table 4.381: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the `SUM_WIDTH_SUMMIT` constraint defined as the composition of the `SUMMIT` pattern, the `feature WIDTH`, and the `aggregator sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>ĺ + ĺ</td>
<td>ĺ + ĺ</td>
<td>ĺ + ĺ</td>
<td>ĺ + ĺ</td>
</tr>
<tr>
<td>r</td>
<td>ĺ + ĺ</td>
<td>ĺ + ĺ + 1</td>
<td>ĺ + ĺ + 1</td>
<td>ĺ + ĺ + 1</td>
</tr>
<tr>
<td>t</td>
<td>ĺ + ĺ</td>
<td>ĺ + ĺ + 1</td>
<td>ĺ + ĺ</td>
<td>ĺ + ĺ + 1</td>
</tr>
<tr>
<td>u</td>
<td>ĺ + ĺ</td>
<td>ĺ + ĺ</td>
<td>ĺ + ĺ + 1</td>
<td>ĺ + ĺ + 1</td>
</tr>
</tbody>
</table>

Table 4.382: Concrete glue matrix, derived from the parametrised glue matrix 3.20, for the simplified automaton of the `SUM_WIDTH_SUMMIT` constraint defined as the composition of the `SUMMIT` pattern, the `feature WIDTH`, and the `aggregator sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>r</td>
<td>0</td>
<td>ĺ + ĺ + 1</td>
<td>ĺ + ĺ + 1</td>
<td>ĺ + ĺ</td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>ĺ + ĺ + 1</td>
<td>0</td>
<td>ĺ + ĺ + 1</td>
</tr>
<tr>
<td>u</td>
<td>0</td>
<td>0</td>
<td>ĺ + ĺ + 1</td>
<td>0</td>
</tr>
</tbody>
</table>
Figure 4.1483:  

(top) Automata without registers for the SUM_WIDTH_SUMMIT_EQ_UP_WHEN_RANGE_EQ_2 and the SUM_WIDTH_SUMMIT_EQ_UP constraints; they describe all sequences maximising the sum of the widths of all the occurrences of the SUMMIT pattern of a sequence of sv variables when the difference between the maximum and the minimum of the variables plus one of the sequence of variables is (A) equal to 2, i.e. $\max(0, \lfloor \frac{sv-1}{2} \rfloor)$ of the Restrictions slot (see \(1\)), (B) strictly greater than 2, i.e. $\max(0, sv - 2)$ (see \(2\)). Within (A) states $s$ and $t$ are accepting when $sv \mod 2 = 1$, while states $r$ and $t'$ are accepting when $sv \mod 2 = 0$. (bottom) All corresponding solutions for $sv - 1 \in \{1, 2, 3, 4\}$. 


[Diagram of automata with states and transitions labeled with inequalities involving $s$, $r$, $t$, and $t'$.]
Figure 4.1484: Automata without registers for the (A) SUM_WIDTH_SUMMIT_IS_EVEN and the (B) SUM_WIDTH_SUMMIT_IS_ODD constraints; they respectively achieve an even/odd sum of width of the SUMMIT pattern on a sequence of $n$ variables; the second symbol represents whether the register D is even or odd, while the third symbol denotes whether the register R is even or odd.
**SUM_WIDTH_VALLEY**

**Origin**
Based on the VALLEY pattern.

**Constraint**

\[
\text{SUM_WIDTH_VALLEY}(\text{VALUE, VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & = 0 \lor \text{VALUE} \geq 1 \\
\text{VALUE} & \leq \max(0, sv - 2v) \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \( > (= | >) (< | =)^* < \).

Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\[(9, (1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7))\]

Figure 4.1485 provides an example where the SUM_WIDTH_VALLEY (9, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7]) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 4.1485: Illustrating the SUM_WIDTH_VALLEY constraint of the Example slot
Figures 4.1486 and 4.1487 respectively depict the automaton associated with the constraint SUM_WIDTH_VALLEY and its simplified form.

Figure 4.1486: Automaton for the SUM_WIDTH_VALLEY constraint obtained by applying decoration Table 3.37 to the seed transducer of the VALLEY pattern where default is 0

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\overline{c} + \overline{c}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\overline{d} + \overline{d} + 1$</td>
<td>$\overline{c} + \overline{d} + \overline{d} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\overline{c} + \overline{d} + \overline{d} + 1$</td>
<td>$\overline{c} + \overline{c}$</td>
</tr>
</tbody>
</table>

Table 4.383: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the SUM_WIDTH_VALLEY constraint defined as the composition of the VALLEY pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 4.1487: Simplified automaton for the SUM_WIDTH_VALLEY constraint obtained by applying decoration Table 3.26 to the seed transducer of the VALLEY pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_{i-1} + 1 \geq 0$ are linear invariants.

Table 4.384: Concrete glue matrix, derived from the parametrised glue matrix 3.21, for the simplified automaton of the SUM_WIDTH_VALLEY constraint defined as the composition of the VALLEY pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Specialisation

Figure 4.1488: (top) Automaton without registers for the SUM_WIDTH_VALLEY_EQ_UP constraint; it describes all sequences maximising the sum of the widths of all the occurrences of the VALLEY pattern of a sequence of $sv$ variables, i.e. $\max(0, sv - 2)$ of the Restrictions slot (see (1)). (bottom) All corresponding solutions for $sv - 1 \in \{4, 5\}$.

Figure 4.1489: Automata without registers for the SUM_WIDTH_VALLEY_IS_EVEN and the (B) SUM_WIDTH_VALLEY_IS_ODD constraints; they respectively achieve an even/odd sum of width of the VALLEY pattern on a sequence of $n$ variables; the second symbol represents whether the register D is even or odd, while the third symbol denotes whether the register R is even or odd.
SUM_WIDTH VALLEY

3035
SUM_WIDTH_ZIGZAG

DESCRIPTION

Based on the ZIGZAG pattern.

Constraint

SUM_WIDTH_ZIGZAG(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

\[sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = 0\]
\[VALUE = 0 \lor VALUE \geq 2\]
\[VALUE \leq \max (0, sv - 2)\]
\[\text{required}(\text{VARIABLES}, \text{var})\]

where

\[sv = |\text{VARIABLES}|\]
\[rv = \text{range}(\text{VARIABLES}.\text{var})\]

Purpose

VALUE is the sum of the width of occurrences of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ‘(<>)^+(< | <> | (>)^+(> | <<)’.
Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example

\((11, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\)

Figure 4.1490 provides an example where the SUM_WIDTH_ZIGZAG constraint holds.

Typical

|VARIABLES| > 3
range(VARIABLES.var) > 1

Symmetries

• Items of VARIABLES can be reversed.
• One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 4.1490: Illustrating the SUM_WIDTH_ZIGZAG constraint of the Example slot
Automaton

Figures 4.1491 and 4.1492 respectively depict the automaton associated with the constraint \texttt{SUM\_WIDTH\_ZIGZAG} and its simplified form.
Figure 4.1491: Automaton for the SUM_WIDTH_ZIGZAG constraint obtained by applying decoration Table 3.37 to the seed transducer of the ZIGZAG pattern where default is 0; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the register D is reset to its initial value; (3) on transitions from c, f to s the register R is updated wrt C and the register C is reset to its initial value.
Figure 4.1492: Simplified automaton for the SUM_WIDTH_ZIGZAG constraint obtained by applying decoration Table 3.41 to the seed transducer of the ZIGZAG pattern where default is 0; missing transitions from a, b, c, d, e, f to s are labelled by =; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + 2 \geq 0$ are linear invariants.
Table 4.385: Concrete glue matrix, derived from the parametrised glue matrix 3.22, for the `SUM_WIDTH_ZIGZAG` constraint defined as the composition of the `ZIGZAG` pattern, the feature `WIDTH`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Table 4.386: Concrete glue matrix, not directly derived from the parametrised glue matrix 3.22, for the simplified automaton of the `SUM_WIDTH_ZIGZAG` constraint defined as the composition of the `ZIGZAG` pattern, the feature `WIDTH`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Specialisation

Figure 4.1493: (left) Automaton without registers for the `SUM_WIDTH_ZIGZAG_EQ_UP` constraint; it describes all sequences maximising the sum of the widths of all the occurrences of the ZIGZAG pattern of a sequence of sv variables, i.e. \( \max(0, sv - 2) \) of the `Restrictions` slot (see 1). (right) All corresponding solutions for \( sv - 1 \in \{2, 3, 4, 5\} \).
Figure 4.1494: Automata without registers for the (A) SUM_WIDTH_ZIGZAG_IS_EVEN and the (B) SUM_WIDTH_ZIGZAG_IS_ODD constraints; they respectively achieve an even/odd sum of width of the ZIGZAG pattern on a sequence of $n$ variables; the second symbol represents whether the register $R$ is even or odd.
SUM_WIDTH_ZIGZAG

3045
SURF_BUMP_ON_DECREASING_SEQUENCE

### Description

**Origin**

Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**

SURF_BUMP_ON_DECREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)

**Arguments**

- VARIABLES : collection(var–dvar)
- FEATURES : collection(var–dvar)
- DEFAULT : int

**Restrictions**

required(VARIABLES.var)
required(FEATURES.var)
|VARIABLES| = |FEATURES|

**Purpose**

Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of BUMP_ON_DECREASING_SEQUENCE is identified then FEATURES\([i]\) is the default value DEFAULT; otherwise FEATURES\([i]\) gives the feature value of the corresponding occurrence of BUMP_ON_DECREASING_SEQUENCE.

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression ‘>>><’.

Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+2\) to index \(j\).

**Example**

Figure 4.1495 provides an example where the SURF_BUMP_ON_DECREASING_SEQUENCE ([7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3], [0, 0, 0, 0, 16, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 11, 0, 0], 0) constraint holds.

**Typical**

|VARIABLES| > 5
range(VARIABLES.var) > 2

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1495: Illustrating the SURF_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
SURF_BUMP_ON_DECREASING_SEQUENCE
**SURF_DECREASING**

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <strong>DECREASING</strong> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><strong>SURF_DECREASING</strong>(VARIABLES, FEATURES, DEFAULT)</td>
</tr>
</tbody>
</table>
| **Arguments** | VARIABLES : collection(var—dvar)  
FEATURES : collection(var—dvar)  
DEFAULT : int |
| **Restrictions** | required(VARIABLES, var)  
required(FEATURES, var)  
|VARIABLES| = |FEATURES| |
| **Purpose** | Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of **DECREASING** is identified then FEATURES[$i$] is the default value DEFAULT; otherwise FEATURES[$i$] gives the feature value of the corresponding occurrence of **DECREASING**.  
An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression ‘$>$’.  
Assume that the occurrence of the pattern **DECREASING** starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$. |
| **Example** | Figure 4.1496 provides an example where the SURF_DECREASING $(\langle 3, 4, 2, 5, 6, 4, 4, 4, 3, 1, 1, 4, 6, 4, 4, \rangle, \langle 0, 6, 0, 0, 0, 10, 0, 7, 4, 0, 0, 0, 10, 0, 0, 0, 0 \rangle, 0)$ constraint holds. |
| **Typical** | $|\text{VARIABLES}| > 1$  
range(\text{VARIABLES.var}) > 1 |
| **Arg. properties** | Functional dependency: FEATURES determined by VARIABLES and DEFAULT. |
Figure 4.1496: Illustrating the SURF_DECREASING constraint of the Example slot
Automaton  Use the decoration table 3.32 to synthesise the corresponding register automaton.
SURF_DECREASING

3053
**SURF_DECREASING_SEQUENCE**

<table>
<thead>
<tr>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>SURF_DECREASING_SEQUENCE</td>
<td>▶ ▶ ✈ ✈ DESCRIPTION AUTOMATON</td>
</tr>
</tbody>
</table>

**Origin**
Based on the **DECREASING_SEQUENCE** pattern.

**Constraint**

SURF_DECREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)

**Arguments**

| VARIABLES : collection(var−dvar) |
| FEATURES : collection(var−dvar) |
| DEFAULT : int |

**Restrictions**

required(VARIABLES.var)  
required(FEATURES.var)  
|VARIABLES| = |FEATURES|

**Purpose**

Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of **DECREASING_SEQUENCE** is identified (even if this occurrence of pattern is not complete) then FEATURES[$i$] is the default value DEFAULT; otherwise FEATURES[$i$] gives the feature value of the corresponding occurrence of **DECREASING_SEQUENCE**.

An occurrence of the pattern **DECREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression ‘$> (> | =)^* > | >$’.

Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

**Example**

Figure 4.1497 provides an example where the SURF_DECREASING_SEQUENCE ([3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 4, 6, 4, 4], [0, 6, 0, 0, 0, 0, 18, 0, 0, 0, 0, 0, 10, 0, 0, 0]) constraint holds.

**Typical**

|VARIABLES| > 1  
range(VARIABLES.var) > 1

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1497: Illustrating the SURF DECREASING SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>SURF_DECREASING_TERRACE</td>
<td></td>
</tr>
</tbody>
</table>

**DESCRIPTION AUTOMATON**

- **Origin**: Based on the `DECREASING_TERRACE` pattern.
- **Constraint**: `SURF_DECREASING_TERRACE(VARIABLES, FEATURES, DEFAULT)`
- **Arguments**:
  - `VARIABLES` : `collection(var−dvar)`
  - `FEATURES` : `collection(var−dvar)`
  - `DEFAULT` : `int`
- **Restrictions**:
  - `required(VARIABLES.var)`
  - `required(FEATURES.var)`
  - `|VARIABLES| = |FEATURES|`
- **Purpose**:
  - Consider the time-series given by the `VARIABLES` collection. If position `i` does not correspond to the first position in `VARIABLES` where an occurrence of `DECREASING_TERRACE` is identified then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `DECREASING_TERRACE`.
  - An occurrence of the pattern `DECREASING_TERRACE` is the *maximal* subsequence which matches the regular expression `'>=+>'`.
  - Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 1` to index `j`.

**Example**

Figure 4.1498 provides an example where the `SURF_DECREASING_TERRACE` 
```
([6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 3, 3, 3], [0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 8, 0, 0, 0], 0)
```
constraint holds.

**Typical**

```
|VARIABLES| > 3
range(VARIABLES.var) > 2
```

**Arg. properties**

- Functional dependency: `FEATURES` determined by `VARIABLES` and `DEFAULT`.
Figure 4.1498: Illustrating the SURF\_DECREASING\_TERRACE constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
### Origin
Based on the **DIP_ON_INCREASING_SEQUENCE** pattern.

### Constraint

<table>
<thead>
<tr>
<th><strong>SURF_DIP_ON_INCREASING_SEQUENCE</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VARIABLES</strong> : <strong>collection(var−dvar)</strong></td>
</tr>
<tr>
<td><strong>FEATURES</strong> : <strong>collection(var−dvar)</strong></td>
</tr>
<tr>
<td><strong>DEFAULT</strong> : <strong>int</strong></td>
</tr>
</tbody>
</table>

### Restrictions

- `required(VARIABLES.var)`
- `required(FEATURES.var)`
- `|VARIABLES| = |FEATURES|`

### Purpose
Consider the time-series given by the **VARIABLES** collection. If position \(i\) does not correspond to the first position in **VARIABLES** where an occurrence of **DIP_ON_INCREASING_SEQUENCE** is identified then **FEATURES**[\(i\)] is the default value of **DEFAULT**; otherwise **FEATURES**[\(i\)] gives the feature value of the corresponding occurrence of **DIP_ON_INCREASING_SEQUENCE**.

An occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** is the subsequence which matches the regular expression ‘<<><<’.

Assume that the occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i + 2\) to index \(j\).

### Example
Figure 4.1499 provides an example where the **SURF_DIP_ON_INCREASING_SEQUENCE** constraint holds.

### Typical

- `|VARIABLES| > 5`
- `range(VARIABLES.var) > 2`

### Arg. properties

- **Functional dependency:** **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.1499: Illustrating the SURF_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
SURF_DIP_ON_INCREASING_SEQUENCE
**SURF_GORGE**

**Feature Pattern**

**DESCRIPTION AUTOMATON**

$\text{(} > | > (= | >)^* >) < | < (= | <)^* < \text{)}$

**Origin**

Based on the GORGE pattern.

**Constraint**

$\text{SURF}_G\text{OR}_G(\text{VARIABLES, FEATURES, DEFAULT})$

**Arguments**

- VARIABLES: $\text{collection(var \text{- dvar})}$
- FEATURES: $\text{collection(var \text{- dvar})}$
- DEFAULT: $\text{int}$

**Restrictions**

- $\text{required}(\text{VARIABLES, var})$
- $\text{required}(\text{FEATURES, var})$
- $|\text{VARIABLES}| = |\text{FEATURES}|$

Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of GORGE is identified (even if this occurrence of pattern is not complete) then FEATURES[$i$] is the default value DEFAULT; otherwise FEATURES[$i$] gives the feature value of the corresponding occurrence of GORGE.

**Purpose**

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘$\text{(} > | > (= | >)^* >) < | < (= | <)^* < \text{)}$’. Assume that the occurrence of the pattern GORGE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Example**

Figure 4.1500 provides an example where the SURF_GORGE constraint holds.

**Typical**

- $|\text{VARIABLES}| > 2$
- $\text{range}(\text{VARIABLES, var}) > 1$

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1500: Illustrating the SURF_GORGE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
### Description

**Origin**
Based on the **INCREASING** pattern.

**Constraint**
\[
\text{SURF\_INCREASING(VARIABLES, FEATURES, DEFAULT)}
\]

**Arguments**
- **VARIABLES**: `collection(var−dvar)
- **FEATURES**: `collection(var−dvar)
- **DEFAULT**: `int`

**Restrictions**
- `required(VARIABLES.var)
- `required(FEATURES.var)
- `|VARIABLES| = |FEATURES|

**Purpose**
Consider the time-series given by the **VARIABLES** collection. If position \(i\) does not correspond to the first position in **VARIABLES** where an occurrence of **INCREASING** is identified then **FEATURES\[i\]** is the default value **DEFAULT**; otherwise **FEATURES\[i\]** gives the feature value of the corresponding occurrence of **INCREASING**.

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression \'<\'.

Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**
Figure 4.1501 provides an example where the **SURF\_INCREASING**
\([4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 3, 1, 3, 3, 3], [0, 8, 0, 0, 0, 0, 4, 0, 7, 10, 0, 0, 0, 4, 0, 0], 0\)
constraint holds.

**Typical**
- `|VARIABLES| > 1
- `range(VARIABLES.var) > 1`

**Arg. properties**
Functional dependency: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.1501: Illustrating the SURF_INCREASING constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
**SURF_INCREASING_SEQUENCE**

**Origin**
Based on the **INCREASING_SEQUENCE** pattern.

**Constraint**

```
SURF_INCREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)
```

**Arguments**

- VARIABLES: `collection(var−dvar)`
- FEATURES: `collection(var−dvar)`
- DEFAULT: `int`

**Restrictions**

- `required(VARIABLES.var)`
- `required(FEATURES.var)`
- `|VARIABLES| = |FEATURES|`

**Purpose**
Consider the time-series given by the VARIABLES collection. If position \( i \) does not correspond to the first position in VARIABLES where an occurrence of **INCREASING_SEQUENCE** is identified (even if this occurrence of pattern is not complete) then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of **INCREASING_SEQUENCE**.

An occurrence of the pattern **INCREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression ‘\(< (< | =)* < | <'.

Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i \) to index \( j + 1 \).

**Example**

Figure 4.1502 provides an example where the **SURF_INCREASING_SEQUENCE** ([4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 3, 1, 3, 3], [0, 8, 0, 0, 0, 0, 17, 0, 0, 0, 0, 0, 4, 0, 0], 0) constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1502: Illustrating the SURF_INCREASING_SEQUENCE constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
**SURF_INCREASING_TERRACE**

**Origin**
Based on the **INCREASING_TERRACE** pattern.

**Constraint**
```
SURF_INCREASING_TERRACE(VARIABLES, FEATURES, DEFAULT)
```

**Arguments**
- VARIABLES: `collection(var−dvar)`
- FEATURES: `collection(var−dvar)`
- DEFAULT: `int`

**Restrictions**
- `required(VARIABLES.var)`
- `required(FEATURES.var)`
- `|VARIABLES| = |FEATURES|`

**Purpose**
Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of **INCREASING_TERRACE** is identified then FEATURES[$i$] is the default value DEFAULT; otherwise FEATURES[$i$] gives the feature value of the corresponding occurrence of **INCREASING_TERRACE**.

An occurrence of the pattern **INCREASING_TERRACE** is the *maximal* subsequence which matches the regular expression ‘$<=$+$<=$’.

Assume that the occurrence of the pattern **INCREASING_TERRACE** starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i+1$ to index $j$.

**Example**
Figure 4.1503 provides an example where the SURF_INCREASING_TERRACE ([1, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 4, 4], [0, 0, 0, 0, 0, 10, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0], 0) constraint holds.

**Typical**
- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 2`

**Arg. properties**
- Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1503: Illustrating the SURF_INCREASING_TERRACE constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
**SURF_INFLEXION**

**Origin**
Based on the INFLEXION pattern.

**Constraint**

\[
\text{SURF\_INFLEXION(VARIABLES, FEATURES, DEFAULT)}
\]

**Arguments**

- **VARIABLES**: \text{collection(var\_dvar)}
- **FEATURES**: \text{collection(var\_dvar)}
- **DEFAULT**: int

**Restrictions**

\[
\text{required(VARIABLES, var)}\]
\[
\text{required(FEATURES, var)}\]
\[
|\text{VARIABLES}| = |\text{FEATURES}|
\]

**Purpose**

Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of INFLEXION is identified then FEATURES\([i]\) is the default value DEFAULT; otherwise FEATURES\([i]\) gives the feature value of the corresponding occurrence of INFLEXION.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(<(< \mid =)\ast > | > (\mid =)\ast <\)'s.

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

Figure 4.1504 provides an example where the SURF_INFLEXION ([1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4], [0, 0, 0, 14, 0, 0, 11, 5, 2, 5, 1, 5, 0, 6, 0, 0, 0]) constraint holds.

**Typical**

- |VARIABLES| \(> 2\)
- range(VARIABLES.var) \(> 1\)

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1504: Illustrating the SURF_INFLEXION constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of PEAK is identified (even if this occurrence of pattern is not complete) then FEATURES\([i]\) is the default value DEFAULT; otherwise FEATURES\([i]\) gives the feature value of the corresponding occurrence of PEAK.

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression '\(< (\leq | <) (> | =) (> | =) >\)'.

Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+1\) to index \(j\).

**Example**

Figure 4.1505 provides an example where the SURFPEAT \(([7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1], [0, 0, 0, 0, 0, 9, 0, 0, 0, 0, 0, 9, 0, 0, 9, 0], 0)\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 2\]
\[\text{range(VARIABLES.var)} > 1\]

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1505: Illustrating the SURF_PEAK constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
SURF.PLAIN

DESCRIPTION

Origin
Based on the PLAIN pattern.

Constraint
SURF.PLAIN(VARIABLES, FEATURES, DEFAULT)

Arguments
VARIABLES : collection(var – dvar)
FEATURES : collection(var – dvar)
DEFAULT : int

Restrictions
required(VARIABLES.var)
required(FEATURES.var)
|VARIABLES| = |FEATURES|

Purpose
Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of PLAIN is identified then FEATURES[\(i\)] is the default value DEFAULT; otherwise FEATURES[\(i\)] gives the feature value of the corresponding occurrence of PLAIN.

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘\(> = * <\)’.

Assume that the occurrence of the pattern PLAIN starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example
Figure 4.1506 provides an example where the SURF.PLAIN (\([2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 6, 6, 3]\), \([0, 0, 0, 5, 0, 0, 0, 4, 0, 0, 0, 6, 0, 0, 0, 6, 0, 0, 0]\), 0) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1

Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1506: Illustrating the SURF.PLAIN constraint of the Example slot
Use the decoration table 3.32 to synthesise the corresponding register automaton.
SURF_PLATEAU

DESCRIPTON  AUTOMATON

Origin  Based on the PLATEAU pattern.

Constraint  \text{SURF\_PLATEAU(VARIABLES, FEATURES, DEFAULT)}

Arguments  
\begin{align*}
\text{VARIABLES} & : \text{collection}\(\text{var-dvar}\) \\
\text{FEATURES} & : \text{collection}\(\text{var-dvar}\) \\
\text{DEFAULT} & : \text{int}
\end{align*}

Restrictions  
\begin{align*}
\text{required(VARIABLES, var)} \\
\text{required(FEATURES, var)} \\
|\text{VARIABLES}| = |\text{FEATURES}|
\end{align*}

Purpose  Consider the time-series given by the \text{VARIABLES} collection. If position \(i\) does not correspond to the first position in \text{VARIABLES} where an occurrence of \text{PLATEAU} is identified then \text{FEATURES}[i] is the default value \text{DEFAULT}; otherwise \text{FEATURES}[i] gives the feature value of the corresponding occurrence of \text{PLATEAU}.

Assume that the occurrence of the pattern \text{PLATEAU} starts at position \(i\) and ends at position \(j\). The feature \text{SURF} computes the sum of the values from index \(i + 1\) to index \(j\).

Example  Figure 4.1507 provides an example where the \text{SURF\_PLATEAU} \(([7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5], [0, 0, 0, 3, 0, 0, 0, 4, 0, 0, 0, 10, 0, 0, 0, 0])\) constraint holds.

Typical  
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range(VARIABLES, var)} & > 1
\end{align*}

Arg. properties  Functional dependency: \text{FEATURES} determined by \text{VARIABLES} and \text{DEFAULT}.
Figure 4.1507: Illustrating the SURF_PLATEAU constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
Feature Pattern

SURF_PROPER.PLAIN

Description Automaton

Origin
Based on the PROPER.PLAIN pattern.

Constraint
SURF_PROPER.PLAIN(VARIABLES, FEATURES, DEFAULT)

Arguments
- VARIABLES : collection(var-dvar)
- FEATURES : collection(var-dvar)
- DEFAULT : int

Restrictions
- required(VARIABLES.var)
- required(FEATURES.var)
- |VARIABLES| = |FEATURES|

Consider the time-series given by the VARIABLES collection. If position i does not correspond to the first position in VARIABLES where an occurrence of PROPER.PLAIN is identified then FEATURES[i] is the default value DEFAULT; otherwise FEATURES[i] gives the feature value of the corresponding occurrence of PROPER.PLAIN.

Purpose
An occurrence of the pattern PROPER.PLAIN is the maximal subsequence which matches the regular expression ‘> =+ <’. Assume that the occurrence of the pattern PROPER.PLAIN starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
Figure 4.1508 provides an example where the SURF_PROPER.PLAIN ([2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0]) constraint holds.

Typical
- |VARIABLES| > 3
- range(VARIABLES.var) > 1

Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1508: Illustrating the SURF_PROPER.PLAIN constraint of the Example slot
Automaton  Use the decoration table 3.32 to synthesise the corresponding register automaton.
### Description

**Origin**
Based on the `PROPER_PLATEAU` pattern.

**Constraint**

```latex
SURF_PROPER_PLATEAU(VARIABLES, FEATURES, DEFAULT)
```

**Arguments**

- **VARIABLES**: `collection(var−dvar)`
- **FEATURES**: `collection(var−dvar)`
- **DEFAULT**: `int`

**Restrictions**

- **required(VARIABLES.var)**
- **required(FEATURES.var)**
- `|VARIABLES| = |FEATURES|`

Consider the time-series given by the VARIABLES collection. If position `i` does not correspond to the first position in VARIABLES where an occurrence of `PROPER_PLATEAU` is identified then FEATURES[`i`] is the default value DEFAULT; otherwise FEATURES[`i`] gives the feature value of the corresponding occurrence of `PROPER_PLATEAU`.

**Purpose**

An occurrence of the pattern `PROPER_PLATEAU` is the maximal subsequence which matches the regular expression `'< =+ >'`.

Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 1` to index `j`.

**Example**

Figure 4.1509 provides an example where the `SURF_PROPER_PLATEAU` constraint holds.

**Typical**

- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 1`

**Arg. properties**

- Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
SURF_PROPER_PLATEAU

Figure 4.1509: Illustrating the SURF_PROPER_PLATEAU constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
SURF_PROPER_PLATEAU 3105
**SURF_STEADY**

<table>
<thead>
<tr>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>SURF_STEADY</td>
<td></td>
</tr>
</tbody>
</table>

### Description

- **Origin**: Based on the **STEADY** pattern.

- **Constraint**: `SURF_STEADY(VARIABLES, FEATURES, DEFAULT)`

- **Arguments**:
  - `VARIABLES`: `collection(var−dvar)`
  - `FEATURES`: `collection(var−dvar)`
  - `DEFAULT`: `int`

- **Restrictions**:
  - `required(VARIABLES, var)`
  - `required(FEATURES, var)`
  - `|VARIABLES| = |FEATURES|`

**Purpose**

Consider the time-series given by the **VARIABLES** collection. If position $i$ does not correspond to the first position in **VARIABLES** where an occurrence of **STEADY** is identified then $\text{FEATURES}[i]$ is the default value **DEFAULT**; otherwise $\text{FEATURES}[i]$ gives the feature value of the corresponding occurrence of **STEADY**.

An occurrence of the pattern **STEADY** is the subsequence which matches the regular expression `='`. Assume that the occurrence of the pattern **STEADY** starts at position $i$ and ends at position $j$. The feature **SURF** computes the sum of the values from index $i$ to index $j + 1$.

### Example

Figure 4.1510 provides an example where the **SURF_STEADY** ($\{[1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6], [2, 0, 0, 6, 0, 10, 0, 0, 10, 0, 0, 0, 12, 0], 0\}$) constraint holds.

### Typical

$|\text{VARIABLES}| > 1$

### Arg. properties

Functional dependency: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.1510: Illustrating the SURF_STEADY constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
**SURF_STEADY_SEQUENCE**

**DESCRIPTION**

**Origin**
Based on the `STEADY_SEQUENCE` pattern.

**Constraint**

```
SURF_STEADY_SEQUENCE(VARIABLES, FEATURES, DEFAULT)
```

**Arguments**

- **VARIABLES**: `collection(var-dvar)`
- **FEATURES**: `collection(var-dvar)`
- **DEFAULT**: `int`

**Restrictions**

```text
required(VARIABLES.var)
required(FEATURES.var)
|VARIABLES| = |FEATURES|
```

Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of `STEADY_SEQUENCE` is identified (even if this occurrence of pattern is not complete) then FEATURES\([i]\) is the default value DEFAULT; otherwise FEATURES\([i]\) gives the feature value of the corresponding occurrence of `STEADY_SEQUENCE`.

**Purpose**

An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression `=+*`.

Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j+1\).

**Example**

Figure 4.1511 provides an example where the `SURF_STEADY_SEQUENCE` ([3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 3, 2, 1, 1], [0, 2, 0, 0, 15, 0, 0, 0, 4, 0, 8, 0, 0, 0, 0, 2, 0, 0]) constraint holds.

**Typical**

```
|VARIABLES| > 1
```

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1511: Illustrating the `SURF_STEADY_SEQUENCE` constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
SURF\_\_DECREASING\_SEQUENCE

### Description

**Origin**
Based on the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern.

**Constraint**
\texttt{SURF\_\_DECREASING\_SEQUENCE(VARIABLES, FEATURES, DEFAULT)}

**Arguments**
- \texttt{VARIABLES} : \texttt{collection(var\_dvar)}
- \texttt{FEATURES} : \texttt{collection(var\_dvar)}
- \texttt{DEFAULT} : \texttt{int}

**Restrictions**
- \texttt{required(VARIABLES.var)}
- \texttt{required(FEATURES.var)}
- \texttt{|VARIABLES| = |FEATURES|}

**Purpose**
Consider the time-series given by the \texttt{VARIABLES} collection. If position \(i\) does not correspond to the first position in \texttt{VARIABLES} where an occurrence of \texttt{STRICTLY\_DECREASING\_SEQUENCE} is identified (even if this occurrence of pattern is not complete) then \texttt{FEATURES[i]} is the default value \texttt{DEFAULT}; otherwise \texttt{FEATURES[i]} gives the feature value of the corresponding occurrence of \texttt{STRICTLY\_DECREASING\_SEQUENCE}.

An occurrence of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} is the maximal subsequence which matches the regular expression ‘\texttt{\textgreater \textgreater}’.

Assume that the occurrence of the pattern \texttt{STRICTLY\_DECREASING\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{SURF} computes the sum of the values from index \(i\) to index \(j+1\).

**Example**
Figure 4.1512 provides an example where the \texttt{SURF\_\_DECREASING\_SEQUENCE ([4, 4, 6, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3], [0, 0, 11, 0, 0, 0, 0, 0, 0, 0, 13, 0, 0, 0, 7, 0], 0)} constraint holds.

**Typical**
- \texttt{|VARIABLES| > 1}
- \texttt{range(VARIABLES.var) > 1}

**Arg. properties**
Functional dependency: \texttt{FEATURES} determined by \texttt{VARIABLES} and \texttt{DEFAULT}.
Figure 4.1512: Illustrating the SURF.Strictly-Decreasing-Sequence constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SURF</strong></td>
<td><strong>STRICTLY_INCREASING_SEQUENCE</strong></td>
</tr>
</tbody>
</table>

**Description**

- **Origin**: Based on the **STRICTLY_INCREASING_SEQUENCE** pattern.
- **Constraint**: `SURF_STRICTLY_INCREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)`
- **Arguments**: 
  - `VARIABLES`: `collection(var−dvar)`
  - `FEATURES`: `collection(var−dvar)`
  - `DEFAULT`: `int`
- **Restrictions**: 
  - `required(VARIABLES.var)`
  - `required(FEATURES.var)`
  - `|VARIABLES| = |FEATURES|`

**Purpose**

Consider the time-series given by the `VARIABLES` collection. If position `i` does not correspond to the first position in `VARIABLES` where an occurrence of `STRICTLY_INCREASING_SEQUENCE` is identified (even if this occurrence of pattern is not complete) then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `STRICTLY_INCREASING_SEQUENCE`.

An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `'<+'`. Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.

**Example**

Figure 4.1513 provides an example where the `SURF_STRICTLY_INCREASING_SEQUENCE` `([4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 3, 1, 2, 3], [0, 8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Arg. properties**

Functional dependency: `FEATURES` determined by `VARIABLES` and `DEFAULT`.
Figure 4.1513: Illustrating the SURF\_STRICTLY\_INCREASING\_SEQUENCE constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
SURF_STRICTLY_INCREASING_SEQUENCE
**SURF_SUMMIT**

**FEATURE PATTERN**

**SURF_SUMMIT**

▶ ▷ ◄ ◄

**DESCRIPTION AUTOMATON**

(≪ | ≪ (≪ | ≪)* ≪)(≫ | ≫ (≫ | ≫)* ≫)

**Origin**
Based on the **SUMMIT** pattern.

**Constraint**

\[
\text{SURF\_SUMMIT}(\text{VARIABLES, FEATURES, DEFAULT})
\]

**Arguments**

\[
\begin{align*}
\text{VARIABLES} & : \text{collection}(\text{var} \rightarrow \text{dvar}) \\
\text{FEATURES} & : \text{collection}(\text{var} \rightarrow \text{dvar}) \\
\text{DEFAULT} & : \text{int}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{required}(\text{VARIABLES.var}) \\
\text{required}(\text{FEATURES.var}) \\
|\text{VARIABLES}| = |\text{FEATURES}|
\end{align*}
\]

Consider the time-series given by the **VARIABLES** collection. If position \( i \) does not correspond to the first position in **VARIABLES** where an occurrence of **SUMMIT** is identified (even if this occurrence of pattern is not complete) then **FEATURES**[\( i \)] is the default value **DEFAULT**; otherwise **FEATURES**[\( i \)] gives the feature value of the corresponding occurrence of **SUMMIT**.

**Purpose**

An occurrence of the pattern **SUMMIT** is the maximal subsequence which matches the regular expression ’((≪ | ≪ (≪ | ≪)* ≪)(≫ | ≫ (≫ | ≫)* ≫)’.

Assume that the occurrence of the pattern **SUMMIT** starts at position \( i \) and ends at position \( j \). The feature **SURF** computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**

Figure 4.1514 provides an example where the **SURF_SUMMIT** ([7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 3], [0, 0, 13, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 3, 0], 0) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range} (\text{VARIABLES.var}) > 1
\]

**Arg. properties**

Functional dependency: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.1514: Illustrating the SURF_SUMMIT constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
SURF_SUMMIT 3125
### Origin
Based on the **VALLEY** pattern.

### Constraint
\[
\text{SURF\_VALLEY}((\text{VARIABLES}, \text{FEATURES}, \text{DEFAULT}))
\]

### Arguments
\[
\begin{align*}
\text{VARIABLES} & : \text{collection}(\text{var-dvar}) \\
\text{FEATURES} & : \text{collection}(\text{var-dvar}) \\
\text{DEFAULT} & : \text{int}
\end{align*}
\]

### Restrictions
\[
\begin{align*}
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{required}(\text{FEATURES}, \text{var}) \\
|\text{VARIABLES}| = |\text{FEATURES}|
\end{align*}
\]

Consider the time-series given by the **VARIABLES** collection. If position \(i\) does not correspond to the first position in **VARIABLES** where an occurrence of **VALLEY** is identified (even if this occurrence of pattern is not complete) then \(\text{FEATURES}[i]\) is the default value **DEFAULT**; otherwise \(\text{FEATURES}[i]\) gives the feature value of the corresponding occurrence of **VALLEY**.

### Purpose
An occurrence of the pattern **VALLEY** is the *maximal* subsequence which matches the regular expression \(\>(= | >)^* (< | =)^* <\).

Assume that the occurrence of the pattern **VALLEY** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i + 1\) to index \(j\).

### Example
Figure 4.1515 provides an example where the **SURF\_VALLEY** \(([1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 13, 0, 0, 0, 0, 0, 15, 0], 0)\) constraint holds.

### Typical
\[
\begin{align*}
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES.var}) > 1
\end{align*}
\]

### Arg. properties
Functional dependency: **FEATURES** determined by **VARIABLES** and **DEFAULT**.
Figure 4.1515: Illustrating the SURF_VALLEY constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
<table>
<thead>
<tr>
<th>FEATURE</th>
<th>PATTERN</th>
</tr>
</thead>
<tbody>
<tr>
<td>SURF_ZIGZAG</td>
<td></td>
</tr>
</tbody>
</table>

**Description Automaton**

```
(<>)+(< | <>)| (<>)+ (> | ><)
```

**Origin**
Based on the ZIGZAG pattern.

**Constraint**

```
SURF_ZIGZAG(VARIABLES, FEATURES, DEFAULT)
```

**Arguments**

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>FEATURES</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>collection(var−dvar)</td>
<td>collection(var−dvar)</td>
<td>int</td>
</tr>
</tbody>
</table>

**Restrictions**

- required(VARIABLES, var)
- required(FEATURES, var)
- |VARIABLES| = |FEATURES|

Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of ZIGZAG is identified (even if this occurrence of pattern is not complete) then FEATURES\([i]\) is the default value DEFAULT; otherwise FEATURES\([i]\) gives the feature value of the corresponding occurrence of ZIGZAG.

**Purpose**
An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression `'(<>)+(< | <>)| (<>)+ (> | ><)'`. Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

Figure 4.1516 provides an example where the SURF_ZIGZAG (\([4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 5, 0, 0, 0, 7, 0, 0, 0, 21, 0, 0, 0, 0]\)) constraint holds.

**Typical**

- |VARIABLES| > 3
- range(VARIABLES.var) > 1

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1516: Illustrating the SURF_ZIGZAG constraint of the **Example** slot
| Automaton | Use the decoration table 3.32 to synthesise the corresponding register automaton. |
WIDTH_DECREASING_SEQUENCE

**Origin** Based on the DECREASING_SEQUENCE pattern.

**Constraint**

\[
\text{WIDTH\_DECREASING\_SEQUENCE}(\text{VARIABLES, FEATURES, DEFAULT})
\]

**Arguments**

- **VARIABLES**: \(\text{collection} (\text{var} - \text{dvar})\)
- **FEATURES**: \(\text{collection} (\text{var} - \text{dvar})\)
- **DEFAULT**: \(\text{int} > (\geq | =) \ast > | >\)

**Restrictions**

- \(\text{required}(\text{VARIABLES}.\text{var})\)
- \(\text{required}(\text{FEATURES}.\text{var})\)
- \(\text{sv} = |\text{FEATURES}|\)
- \(\text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{FEATURES}.\text{var} = 0\)
- \(\text{FEATURES}.\text{var} = 0 \lor \text{FEATURES}.\text{var} \geq 2\)
- \(\text{rv} = 2 \Rightarrow \text{FEATURES}.\text{var} \leq 2\)
- \(\text{rv} \geq 3 \Rightarrow \text{FEATURES}.\text{var} \leq \text{sv}\)
- \(\text{DEFAULT} = 0\)
  where
- \(\text{sv} = |\text{VARIABLES}|\)
- \(\text{rv} = \text{range}(\text{VARIABLES}.\text{var})\)

**Purpose**

Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of DECREASING_SEQUENCE is identified (even if this occurrence of pattern is not complete) then FEATURES\([i]\) is the default value DEFAULT; otherwise FEATURES\([i]\) gives the feature value of the corresponding occurrence of DECREASING_SEQUENCE.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression ‘\(> (> | =)^\ast > | >\)’.

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i + 2\).

**Example**

Figure 4.1517 provides an example where the WIDTH\_DECREASING\_SEQUENCE ([3, 4, 2, 5, 6, 4, 3, 1, 4, 6, 4, 4], [0, 2, 0, 0, 0, 5, 0, 0, 0, 0, 0, 2, 0, 0]) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 1\)
\(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1517: Illustrating the WIDTH_DECREASING_SEQUENCE constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
WIDTH_DECREASING_SEQUENCE

3137
<table>
<thead>
<tr>
<th>Feature</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>WIDTH</td>
<td>DECREASING_TERRACE</td>
</tr>
</tbody>
</table>

**Description**

**Origin**

Based on the `DECREASING_TERRACE` pattern.

**Constraint**

`WIDTH_DECREASING_TERRACE(VARIABLES, FEATURES, DEFAULT)`

**Arguments**

- `VARIABLES` : collection(var\_dvar)
- `FEATURES` : collection(var\_dvar)
- `DEFAULT` : int

**Restrictions**

- `required(VARIABLES.var)`
- `required(FEATURES.var)`
- `sv = |FEATURES|`
- `sv ≤ 3 ∨ rv ≤ 2 ⇒ FEATURES.var = 0`
- `FEATURES.var = 0 ∨ FEATURES.var ≥ 2`
- `FEATURES.var ≤ max(0, sv - 2)`
- `DEFAULT = 0`

where

- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**

Consider the time-series given by the `VARIABLES` collection. If position `i` does not correspond to the first position in `VARIABLES` where an occurrence of `DECREASING_TERRACE` is identified then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `DECREASING_TERRACE`.

An occurrence of the pattern `DECREASING_TERRACE` is the *maximal* subsequence which matches the regular expression `">=+">`.

Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j - i`

**Example**

Figure 4.1518 provides an example where the `WIDTH_DECREASING_TERRACE` ([6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3], [0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 2, 0, 0, 0, 2, 0, 0, 0]) constraint holds.

**Typical**

- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 2`

**Arg. properties**

Functional dependency: `FEATURES` determined by `VARIABLES` and `DEFAULT`. 
Figure 4.1518: Illustrating the WIDTH_DECREASING_TERRACE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
**WIDTH_GATE**

**DESCRIPTION**

Origin: Based on the GORGE pattern.

**Constraint**

\[
\text{WIDTH_GATE} (\text{VARIABLES, FEATURES, DEFAULT})
\]

**Arguments**

- VARIABLES: \(\text{collection(var\_dvar)}\)
- FEATURES: \(\text{collection(var\_dvar)}\)
- DEFAULT: \(\text{int}\)

**Restrictions**

- \(\text{required} (\text{VARIABLES}.\text{var})\)
- \(\text{required} (\text{FEATURES}.\text{var})\)
- \(sv = |\text{FEATURES}|\)
- \(\text{FEATURES}.\text{var} \leq 2 \lor rv \leq 1 \Rightarrow \text{FEATURES}.\text{var} = 0\)
- \(\text{FEATURES}.\text{var} \geq 0\)
- \(rv = 2 \Rightarrow \text{FEATURES}.\text{var} \leq 1\)
- \(rv \geq 3 \Rightarrow \text{FEATURES}.\text{var} \leq \max(0, sv - 2)\)
- \(\text{DEFAULT} = 0\)

where

- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range}(\text{VARIABLES}.\text{var})\)

**Purpose**

Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of GORGE is identified (even if this occurrence of pattern is not complete) then FEATURES\([i]\) is the default value DEFAULT; otherwise FEATURES\([i]\) gives the feature value of the corresponding occurrence of GORGE.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression ‘\((>= | > | >)^* > | < | (<= | <)^* <\)’.

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

Figure 4.1519 provides an example where the WIDTH_GATE ([1, 7, 3, 4, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 4, 6, 5, 7], [0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 1, 0], 0) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1519: Illustrating the WIDTH_GORGE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
FEATURE  
WIDTH_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th></th>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
</table>

**Origin**

Based on the `INCREASING_SEQUENCE` pattern.

**Constraint**

`WIDTH_INCREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)`

**Arguments**

- VARIABLES: `collection(var−dvar)`
- FEATURES: `collection(var−dvar)`
- DEFAULT: `int`

**Restrictions**

```plaintext
required(VARIABLES, var)
required(FEATURES, var)
sv = |FEATURES|
sv ≤ 1 ∨ rv ≤ 1 ⇒ FEATURES.var = 0
FEATURES.var = 0 ∨ FEATURES.var ≥ 2
rv = 2 ⇒ FEATURES.var ≤ 2
rv ≥ 3 ⇒ FEATURES.var ≤ sv
DEFAULT = 0
where
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

Consider the time-series given by the VARIABLES collection. If position `i` does not correspond to the first position in VARIABLES where an occurrence of `INCREASING_SEQUENCE` is identified (even if this occurrence of pattern is not complete) then FEATURES[`i`] is the default value DEFAULT; otherwise FEATURES[`i`] gives the feature value of the corresponding occurrence of `INCREASING_SEQUENCE`.

**Purpose**

An occurrence of the pattern `INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression ‘`<(<|=)*<|`’.

Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature WIDTH computes the value `j − i + 2`.

**Example**

Figure 4.1520 provides an example where the `WIDTH_INCREASING_SEQUENCE` ([4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 3, 1, 3, 3], [0, 2, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 2, 0, 0], 0) constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1520: Illustrating the \texttt{WIDTH\_INCREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Use the decoration table 3.32 to synthesise the corresponding register automaton.
**DESCRIPTION**

**Origin**
Based on the INCREASING_TERRACE pattern.

**Constraint**
WIDTH_INCREASING_TERRACE(VARIABLES, FEATURES, DEFAULT)

**Arguments**
- VARIABLES : collection(var−dvar)
- FEATURES : collection(var−dvar)
- DEFAULT : int

**Restrictions**

- required(VARIABLES.var)
- required(FEATURES.var)
- sv = |FEATURES|
- sv ≤ 3 ∨ rv ≤ 2 ⇒ FEATURES.var = 0
- FEATURES.var = 0 ∨ FEATURES.var ≥ 2
- FEATURES.var ≤ max(0, sv − 2)
- DEFAULT = 0

where
- sv = |VARIABLES|
- rv = range(VARIABLES.var)

**Purpose**
Consider the time-series given by the VARIABLES collection. If position \( i \) does not correspond to the first position in VARIABLES where an occurrence of INCREASING_TERRACE is identified then FEATURES[\( i \)] is the default value DEFAULT; otherwise FEATURES[\( i \)] gives the feature value of the corresponding occurrence of INCREASING_TERRACE.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression '\(<=+<\)'.

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**
Figure 4.1521 provides an example where the WIDTH_INCREASING_TERRACE ([1, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 4, 4], [0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 3, 0, 0]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 3\)
- range(VARIABLES.var) > 2

**Arg. properties**
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1521: Illustrating the WIDTH_INCREASING_TERRACE constraint of the Example slot
Use the decoration table 3.32 to synthesise the corresponding register automaton.
WIDTH_INCREASING_TERRACE

3153
Based on the INFLEXION pattern.

**Constraint**
- \(\text{WIDTH\_INFLEXION}(\text{VARIABLES}, \text{FEATURES}, \text{DEFAULT})\)

**Arguments**
- \(\text{VARIABLES} : \text{collection}(\text{var} \ldots \text{dvar})\)
- \(\text{FEATURES} : \text{collection}(\text{var} \ldots \text{dvar})\)
- \(\text{DEFAULT} : \text{int}\)

**Restrictions**
- \(\text{required}(\text{VARIABLES}, \text{var})\)
- \(\text{required}(\text{FEATURES}, \text{var})\)
- \(\text{sv} = |\text{FEATURES}|\)
- \(\text{sv} \leq 2 \vee \text{rv} \leq 1 \Rightarrow \text{FEATURES}\.\text{var} = 0\)
- \(\text{FEATURES}\.\text{var} \geq 0\)
- \(\text{FEATURES}\.\text{var} \leq \max(0, \text{sv} - 2)\)
- \(\text{DEFAULT} = 0\)

where
- \(\text{sv} = |\text{VARIABLES}|\)
- \(\text{rv} = \text{range}(\text{VARIABLES}\.\text{var})\)

Consider the time-series given by the \(\text{VARIABLES}\) collection. If position \(i\) does not correspond to the first position in \(\text{VARIABLES}\) where an occurrence of \(\text{INFLEXION}\) is identified then \(\text{FEATURES}[i]\) is the default value \(\text{DEFAULT}\); otherwise \(\text{FEATURES}[i]\) gives the feature value of the corresponding occurrence of \(\text{INFLEXION}\).

An occurrence of the pattern \(\text{INFLEXION}\) is the maximal subsequence which matches the regular expression \(\langle (< | =) | > | > | =) \rangle^* <\).

Assume that the occurrence of the pattern \(\text{INFLEXION}\) starts at position \(i\) and ends at position \(j\). The feature \(\text{WIDTH}\) computes the value \(j - i\).

**Example**
Figure 4.1522 provides an example where the \(\text{WIDTH\_INFLEXION}\) \(([1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4], [0, 0, 0, 3, 0, 0, 3, 1, 1, 1, 1, 0, 2, 0, 0], 0)\) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}\.\text{var}) > 1\)

**Arg. properties**
- Functional dependency: \(\text{FEATURES}\) determined by \(\text{VARIABLES}\) and \(\text{DEFAULT}\).
Figure 4.1522: Illustrating the WIDTH_INFLEXION constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
WIDTH_INFLEXION

3157
Based on the \textit{PEAK} pattern.

Consider the time-series given by the \textit{VARIABLES} collection. If position \(i\) does not correspond to the first position in \textit{VARIABLES} where an occurrence of \textit{PEAK} is identified (even if this occurrence of pattern is not complete) then \textit{FEATURES}[i] is the default value \textit{DEFAULT}; otherwise \textit{FEATURES}[i] gives the feature value of the corresponding occurrence of \textit{PEAK}.

An occurrence of the pattern \textit{PEAK} is the maximal subsequence which matches the regular expression '<\(\text{\textbackslash =}\)\(\text{\textbackslash <}\)\(\text{\textbackslash >}\)\(\text{\textbackslash |}\)\(\text{\textbackslash =}\)\(\text{\textbackslash >}\)\(\text{\textbackslash >}\)'. Assume that the occurrence of the pattern \textit{PEAK} starts at position \(i\) and ends at position \(j\). The feature \textit{WIDTH} computes the value \(j - i\).

\textbf{Example}\texttt{Figure 4.1523} provides an example where the \textit{WIDTH\_PEAK} constraint holds.

\textbf{Typical}\n\[|\textit{VARIABLES}| > 2\]
\[\text{range}(\textit{VARIABLES}.\textit{var}) > 1\]

\textbf{Arg. properties}\nFunctional dependency: \textit{FEATURES} determined by \textit{VARIABLES} and \textit{DEFAULT}. 
Figure 4.1523: Illustrating the WIDTH\_PEAK constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
WIDTH_PEAK

3161
DESCRIPTION AUTOMATON

Origin
Based on the PLAIN pattern.

Constraint
WIDTH_PLAIN(VARIABLES, FEATURES, DEFAULT)

Arguments
VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int

Restrictions
required(VARIABLES, var)
required(FEATURES, var)
sv = |FEATURES|
sv ≤ 2 ∨ rv ≤ 1 ⇒ FEATURES, var = 0
FEATURES, var ≥ 0
FEATURES, var ≤ max(0, sv − 2)
DEFAULT = 0
where
sv = |VARIABLES|
rv = range(VARIABLES, var)

Consider the time-series given by the VARIABLES collection. If position i does not correspond to the first position in VARIABLES where an occurrence of PLAIN is identified then FEATURES[i] is the default value DEFAULT; otherwise FEATURES[i] gives the feature value of the corresponding occurrence of PLAIN.

Purpose
An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression ‘> =\* <’. Assume that the occurrence of the pattern PLAIN starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example
Figure 4.1524 provides an example where the WIDTH_PLAIN ([2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3], [0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES, var) > 1

Arg. properties
Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1524: Illustrating the WIDTH_PLAIN constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
### WIDTH_PLATEAU

**Pattern**

 Origin

Based on the PLATEAU pattern.

 Constraint

```
WIDTH_PLATEAU(VARIABLES, FEATURES, DEFAULT)
```

 Arguments

```
VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int
```

 Restrictions

```
required(VARIABLES.var)
required(FEATURES.var)
sv = |FEATURES|
sv ≤ 2 ∨ rv ≤ 1 ⇒ FEATURES.var = 0
FEATURES.var ≥ 0
FEATURES.var ≤ max(0, sv − 2)
DEFAULT = 0
```

where

```
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

Purpose

Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of PLATEAU is identified then FEATURES[$i$] is the default value DEFAULT; otherwise FEATURES[$i$] gives the feature value of the corresponding occurrence of PLATEAU.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression ‘< =∗ >’. Assume that the occurrence of the pattern PLATEAU starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j − i$.

Example

Figure 4.1525 provides an example where the WIDTH_PLATEAU ([1, 3, 3, 5, 5, 5, 2, 4, 4, 4, 3, 3, 1, 5, 5], [0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical

```
|VARIABLES| > 2
range(VARIABLES.var) > 1
```

Arg. properties

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1525: Illustrating the WIDTH_PLATEAU constraint of the Example slot
Automaton  Use the decoration table 3.32 to synthesise the corresponding register automaton.
Origin

Based on the \texttt{PROPER\_PLAIN} pattern.

Constraint

\texttt{WIDTH\_PROPER\_PLAIN(VARIABLES, FEATURES, DEFAULT)}

Arguments

\begin{itemize}
  \item \texttt{VARIABLES} : \texttt{collection(var–dvar)}
  \item \texttt{FEATURES} : \texttt{collection(var–dvar)}
  \item \texttt{DEFAULT} : \texttt{int}
\end{itemize}

Restrictions

\begin{itemize}
  \item \texttt{required(VARIABLES.var)}
  \item \texttt{required(FEATURES.var)}
  \item \texttt{sv = |FEATURES|}
  \item \texttt{sv \leq 3 \lor rv \leq 1} \Rightarrow \texttt{FEATURES.var} = 0
  \item \texttt{FEATURES.var} = 0 \lor \texttt{FEATURES.var} \geq 2
  \item \texttt{FEATURES.var} \leq \texttt{max(0, sv – 2)}
  \item \texttt{DEFAULT} = 0
\end{itemize}

\texttt{where}

\texttt{sv = |VARIABLES|}

\texttt{rv = range(VARIABLES.var)}

Purpose

Consider the time-series given by the \texttt{VARIABLES} collection. If position \(i\) does not correspond to the first position in \texttt{VARIABLES} where an occurrence of \texttt{PROPER\_PLAIN} is identified then \texttt{FEATURES}[i] is the default value \texttt{DEFAULT}; otherwise \texttt{FEATURES}[i] gives the feature value of the corresponding occurrence of \texttt{PROPER\_PLAIN}.

An occurrence of the pattern \texttt{PROPER\_PLAIN} is the \textit{maximal} subsequence which matches the regular expression \('\textgreater =^{+} \textless\').

Assume that the occurrence of the pattern \texttt{PROPER\_PLAIN} starts at position \(i\) and ends at position \(j\). The feature \texttt{WIDTH} computes the value \(j - i\).

Example

Figure 4.1526 provides an example where the \texttt{WIDTH\_PROPER\_PLAIN} ([2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5], [0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0], 0) constraint holds.

Typical

\begin{itemize}
  \item \texttt{|VARIABLES|} \geq 3
  \item \texttt{range(VARIABLES.var)} \geq 1
\end{itemize}

Arg. properties

\begin{itemize}
  \item Functional dependency: \texttt{FEATURES} determined by \texttt{VARIABLES} and \texttt{DEFAULT}.
\end{itemize}
Figure 4.1526: Illustrating the WIDTH_PROPER.PLAIN constraint of the Example slot
**Automaton**

Use the decoration table 3.32 to synthesise the corresponding register automaton.
WIDTH_PROPER_PLATEAU

FEATURE

PATTERN

WIDTH_PROPER_PLATEAU

▶ ◄ ◄  DESCRIPTION  AUTOMATON

Origin

Based on the PROPER_PLATEAU pattern.

Constraint

WIDTH_PROPER_PLATEAU(VARIABLES, FEATURES, DEFAULT)

Arguments

VARIABLES : collection(var−dvar)
FEATURES : collection(var−dvar)
DEFAULT : int

Restrictions

required(VARIABLES.var)
required(FEATURES.var)
sv = |FEATURES|
sv ≤ 3 ∨ rv ≤ 1 ⇒ FEATURES.var = 0
FEATURES.var = 0 ∨ FEATURES.var ≥ 2
FEATURES.var ≤ max(0, sv − 2)
DEFAULT = 0
where
sv = |VARIABLES|
rv = range(VARIABLES.var)

Consider the time-series given by the VARIABLES collection. If position i does not correspond to the first position in VARIABLES where an occurrence of PROPER_PLATEAU is identified then FEATURES[i] is the default value DEFAULT; otherwise FEATURES[i] gives the feature value of the corresponding occurrence of PROPER_PLATEAU.

Purpose

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression '< =+ >'.
Assume that the occurrence of the pattern PROPER_PLATEAU starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example

Figure 4.1527 provides an example where the WIDTH_PROPER_PLATEAU ([7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3], [0, 0, 0, 2, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 3, 0], 0) constraint holds.

Typical

|VARIABLES| > 3
range(VARIABLES.var) > 1

Arg. properties

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1527: Illustrating the WIDTH_PROPER_PLATEAU constraint of the Example slot
| Automaton       | Use the decoration table 3.32 to synthesise the corresponding register automaton. |
WIDTH_STEADY_SEQUENCE

- **Origin**: Based on the *STEADY_SEQUENCE* pattern.
- **Constraint**: `WIDTH_STEADY_SEQUENCE(VARIABLES, FEATURES, DEFAULT)

- **Arguments**
  - `VARIABLES`: collection(var−dvar)
  - `FEATURES`: collection(var−dvar)
  - `DEFAULT`: int

- **Restrictions**
  - `required(VARIABLES, var)`
  - `required(FEATURES, var)`
  - `sv = |FEATURES|`
  - `FEATURES.var ≥ 0`
  - `FEATURES.var ≤ sv`
  - `sv ≤ 1 ⇒ FEATURES.var = 0`
  - `FEATURES.var = 0 ∨ FEATURES.var ≥ 2`
  - `rv = 1 ∧ sv ≥ 2 ⇒ maxval(FEATURES.var) = sv`
  - `DEFAULT = 0`
  - where
    - `sv = |VARIABLES|`
    - `rv = range(VARIABLES.var)`

- **Purpose**
  Consider the time-series given by the `VARIABLES` collection. If position `i` does not correspond to the first position in `VARIABLES` where an occurrence of `STEADY_SEQUENCE` is identified (even if this occurrence of pattern is not complete) then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `STEADY_SEQUENCE`.
  An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression ‘=*’.
  Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

- **Example**
  Figure 4.1528 provides an example where the `WIDTH_STEADY_SEQUENCE` constraint holds.

- **Typical**
  - `|VARIABLES| > 1`

- **Arg. properties**
  - Functional dependency: `FEATURES` determined by `VARIABLES` and `DEFAULT`.
Figure 4.1528: Illustrating the WIDTH_STEADY SEQUENCE constraint of the Example slot.
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
WIDTH_STRICTLY_DECREASING_SEQUENCE

**Origin**
Based on the **STRICTLY_DECREASING_SEQUENCE** pattern.

**Constraint**

| WIDTH_STRICTLY_DECREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT) |

**Arguments**

- **VARIABLES**: collection(var−dvar)
- **FEATURES**: collection(var−dvar)
- **DEFAULT**: int

**Restrictions**

- `required(VARIABLES,var)`
- `required(FEATURES,var)`
- `sv = |FEATURES|`
- `sv ≤ 1 \lor rv ≤ 1 \Rightarrow FEATURES.var = 0`
- `FEATURES.var = 0 \lor FEATURES.var ≥ 2`
- `FEATURES.var ≤ min(sv,rv)`
- `DEFAULT = 0`

where

- `sv = |VARIABLES|`
- `rv=range(VARIABLES.var)`

**Purpose**
Consider the time-series given by the **VARIABLES** collection. If position `i` does not correspond to the first position in **VARIABLES** where an occurrence of **STRICTLY_DECREASING_SEQUENCE** is identified (even if this occurrence of pattern is not complete) then FEATURES[i] is the default value `DEFAULT`; otherwise FEATURES[i] gives the feature value of the corresponding occurrence of **STRICTLY_DECREASING_SEQUENCE**.

An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the `maximal` sub-sequence which matches the regular expression `'>+'`.

Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position `i` and ends at position `j`. The feature **WIDTH** computes the value `j − i + 2`.

**Example**

Figure 4.1529 provides an example where the **WIDTH_STRICTLY_DECREASING_SEQUENCE** ([4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3], [0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 2, 0], 0) constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Arg. properties**
Functional dependency: FEATURES determined by **VARIABLES** and **DEFAULT**.
Figure 4.1529: Illustrating the WIDTH_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
**Origin**
Based on the `WIDTH_STRICTLY_INCREASING_SEQUENCE` pattern.

**Constraint**
`WIDTH_STRICTLY_INCREASING_SEQUENCE(VARIABLES, FEATURES, DEFAULT)`

**Arguments**
- `VARIABLES` : `collection(var−dvar)`
- `FEATURES` : `collection(var−dvar)`
- `DEFAULT` : `int`

**Restrictions**
- `required(VARIABLES.var)`
- `required(FEATURES.var)`
- `sv = |FEATURES|`
- `sv ≤ 1 ∨ rv ≤ 1 ⇒ FEATURES.var = 0`
- `FEATURES.var = 0 ∨ FEATURES.var ≥ 2`
- `FEATURES.var ≤ min(sv, rv)`
- `DEFAULT = 0`

where
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**
Consider the time-series given by the `VARIABLES` collection. If position `i` does not correspond to the first position in `VARIABLES` where an occurrence of `STRICTLY_INCREASING_SEQUENCE` is identified (even if this occurrence of pattern is not complete) then `FEATURES[i]` is the default value `DEFAULT`; otherwise `FEATURES[i]` gives the feature value of the corresponding occurrence of `STRICTLY_INCREASING_SEQUENCE`. An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `'<+`'.

Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

**Example**
Figure 4.1530 provides an example where the `WIDTH_STRICTLY_INCREASING_SEQUENCE` `([4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 3, 1, 2, 3], [0, 2, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 3, 0, 0], 0)` constraint holds.

**Typical**
- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Arg. properties**
Functional dependency: `FEATURES` determined by `VARIABLES` and `DEFAULT`. 
Figure 4.1530: Illustrating the `WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint of the Example slot
Use the decoration table 3.32 to synthesise the corresponding register automaton.
WIDTH\_STRICTLY\_INCREASING\_SEQUENCE

3189
Based on the \textsc{summit} pattern.

\textbf{Constraint} \hspace{1cm} \textsc{width\_summit}(\textsc{variables}, \textsc{features}, \textsc{default})

\textbf{Arguments} \hspace{1cm} \begin{align*} 
\textsc{variables} & : \text{collection}(\text{var} - \text{dvar}) \\
\textsc{features} & : \text{collection}(\text{var} - \text{dvar}) \\
\textsc{default} & : \text{int} 
\end{align*}

\textbf{Restrictions} \hspace{1cm} \begin{align*} 
\text{required}(\text{variables}.\text{var}) \\
\text{required}(\text{features}.\text{var}) \\
sv & = |\text{features}| \\
sv & \leq 2 \lor rv \leq 1 \Rightarrow \text{features}.\text{var} = 0 \\
\text{features}.\text{var} & \geq 0 \\
rv & = 2 \Rightarrow \text{features}.\text{var} \leq 1 \\
rv & \geq 3 \Rightarrow \text{features}.\text{var} \leq \max(0, sv - 2) \\
\text{default} & = 0 \\
\text{where} \\
sv & = |\text{variables}| \\
rv & = \text{range}(\text{variables}.\text{var})
\end{align*}

Consider the time-series given by the \textsc{variables} collection. If position \(i\) does not correspond to the first position in \textsc{variables} where an occurrence of \textsc{summit} is identified (even if this occurrence of pattern is not complete) then \text{features}[i] is the default value \text{default}; otherwise \text{features}[i] gives the feature value of the corresponding occurrence of \textsc{summit}.

\textbf{Purpose} \hspace{1cm} An occurrence of the pattern \textsc{summit} is the \textit{maximal} subsequence which matches the regular expression \((< | < (= | <) < > | > (= | >) >)^*\).

Assume that the occurrence of the pattern \textsc{summit} starts at position \(i\) and ends at position \(j\). The feature \textsc{width} computes the value \(j - i\).

\textbf{Example} \hspace{1cm} Figure 4.1531 provides an example where the \textsc{width\_summit} \([7, 1, 5, 4, 4, 3, 4, 6, 2, 3, 2, 3, 1], \{0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 1, 0\}, 0\) constraint holds.

\textbf{Typical} \hspace{1cm} \begin{align*} 
|\text{variables}| & > 2 \\
\text{range}(\text{variables}.\text{var}) & > 1 
\end{align*}

\textbf{Arg. properties} \hspace{1cm} Functional dependency: \textsc{features} determined by \textsc{variables} and \textsc{default}.
Figure 4.1531: Illustrating the WIDTH_SUMMIT constraint of the Example slot
Automaton

Use the decoration table 3.32 to synthesise the corresponding register automaton.
| WIDTH_SUMMIT | 3193 |
**WIDTH_VALLEY**

**DESCRIPTION AUTOMATON**

| **Origin** | Based on the VALLEY pattern. |
| **Constraint** | WIDTH_VALLEY(VARIABLES, FEATURES, DEFAULT) |
| **Arguments** | VARIABLES : collection(var−dvar) |
| | FEATURES : collection(var−dvar) |
| | DEFAULT : int |
| **Restrictions** | required(VARIABLES.var) |
| | required(FEATURES.var) |
| | sv = |FEATURES| |
| | sv ≤ 2 ∨ rv ≤ 1 ⇒ FEATURES.var = 0 |
| | FEATURES.var ≥ 0 |
| | FEATURES.var ≤ max(0, sv − 2) |
| | DEFAULT = 0 |
| | where |
| | sv = |VARIABLES| |
| | rv =range(VARIABLES.var) |

Consider the time-series given by the VARIABLES collection. If position $i$ does not correspond to the first position in VARIABLES where an occurrence of VALLEY is identified (even if this occurrence of pattern is not complete) then FEATURES[i] is the default value DEFAULT; otherwise FEATURES[i] gives the feature value of the corresponding occurrence of VALLEY.

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression '$(= | >)^* (| <) =$'.

Assume that the occurrence of the pattern VALLEY starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j − i$.

**Purpose**

**Example**

Figure 4.1532 provides an example where the WIDTH_VALLEY ($[1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7], [0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 4, 0, 0, 0, 3, 0], 0$) constraint holds.

**Typical**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Arg. properties**

Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1532: Illustrating the WIDTH_Valley constraint of the Example slot
Use the decoration table 3.32 to synthesise the corresponding register automaton.
Origin: Based on the ZIGZAG pattern.

Constraint: 

\[
\text{WIDTH\_ZIGZAG}(\text{VARIABLES}, \text{FEATURES}, \text{DEFAULT})
\]

Arguments:

- VARIABLES: \text{collection}(\text{var} - \text{dvar})
- FEATURES: \text{collection}(\text{var} - \text{dvar})
- DEFAULT: \text{int}

Restrictions:

\[\text{required}(\text{VARIABLES}.)\]
\[\text{required}(\text{FEATURES}.)\]
\[sv = |\text{FEATURES}|\]
\[sv \leq 3 \lor rv \leq 1 \Rightarrow \text{FEATURES}.\text{var} = 0\]
\[\text{FEATURES}.\text{var} = 0 \lor \text{FEATURES}.\text{var} \geq 2\]
\[\text{FEATURES}.\text{var} \leq \max(0, sv - 2)\]
\[\text{DEFAULT} = 0\]

where:

\[sv = |\text{VARIABLES}|\]
\[rv = \text{range}(\text{VARIABLES}.\text{var})\]

Purpose:

Consider the time-series given by the VARIABLES collection. If position \(i\) does not correspond to the first position in VARIABLES where an occurrence of ZIGZAG is identified (even if this occurrence of pattern is not complete) then FEATURES[\(i\)] is the default value DEFAULT; otherwise FEATURES[\(i\)] gives the feature value of the corresponding occurrence of ZIGZAG.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression '\((<<)\)*\((<>|<<)\)|\((<>|>)\)*\((>|<<)\)'.

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example:

Figure 4.153 provides an example where the WIDTH\_ZIGZAG \([4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 3, 0, 0, 2, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 0, 0]\) constraint holds.

Typical:

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

Arg. properties: Functional dependency: FEATURES determined by VARIABLES and DEFAULT.
Figure 4.1533: Illustrating the WIDTH_ZIGZAG constraint of the Example slot
Automaton Use the decoration table 3.32 to synthesise the corresponding register automaton.
A Database of Parameterised Invariants

This chapter presents a database of parameterised invariants where the parameter is the sequence length. We currently have five types of invariants:

- For a single time-series constraint, linear invariants are obtained by applying the Farkas Lemma [13] to the automaton with registers that was synthesised [17] for a given time-series constraint. These invariants link consecutive register values. For those time-series constraints for which two register automata are available, we only generate invariants for the simplified automaton, i.e. the one that uses a specialised decoration table.

- For the conjunction of two time-series constraints on the same sequence of variables of length $n$, linear invariants of the form $e + e_0 n + e_1 x + e_2 y \geq 0$, with $e, e_0, e_1, e_2 \in \mathbb{Z}$, linking the result variables $x$ and $y$ of both time-series constraints are obtained by using the method described in [5]:

  1. Compute the intersection of the register automata corresponding to the two time-series constraints. For building this intersection we use, when available, the simplified versions of the register automata, but note that the resulting invariants can be used both with the non-simplified and the simplified register automata.

  2. Generate all possible combinations of coefficient signs of the invariants to generate, and for each combination:

     (a) Build from the intersection automaton an invariant-weighted digraph, where the weight of each arc represents the variation of the left-hand side $e + e_0 n + e_1 x + e_2 y$ of the desired linear invariant when the corresponding transition of the intersection automaton is triggered.

     (b) Construct an optimisation problem where the constraints prevent negative cycles in the invariant-weighted digraph, and where the optimisation criterion promotes the generation of sharp invariants. The optimal
5. A DATABASE OF PARAMETERISED INVARIANTS

solution to this optimisation problem are the values of the coefficients $e_0, e_1, e_2$.

(c) Solve a shortest-path problem from the initial state to the accepting states of an automaton derived from the intersection automaton and from the invariant-weighted digraph, giving the constant term $e$ of the linear invariant.

3. Using a heuristic, filter out linear invariants that do not remove any infeasible points.

For the majority of linear invariants we prove their sharpness by providing two distinct points whose coordinates are possibly parameterised by the sequence length $n$, which both are located on the corresponding linear equality constraint and are feasible with respect to the conjunction of the two considered time-series constraints. Feasibility of a point is proved by showing that the intersection of two finite automata that accept only time series whose characteristics correspond to the coordinates of the point is not empty. Each of the two finite automata corresponds to the automata provided in the **Specialisation** slot of a time-series constraint.

- Applying the same technique used for generating linear invariants linking the result variables of a conjunction of two time-series constraints, we also generate conditional linear constraints that are valid provided the result variables are not assigned to their default value.

- For the conjunction of two time-series constraints on the same sequence of variables of length $n$, non-linear invariants corresponding to disjunction of linear constraints.

- For a single time-series constraint or for the conjunction of two time-series constraints on the same sequence of variables, we derive linear invariants involving the result variables $x$ and $y$, as well as one of the three variables $o<, o=, o>$, which respectively denote the numbers of occurrences of signature variables that are assigned to ‘$<$’, ‘$=$’ and ‘$>$’. This last category of invariants is derived manually.

The database of parameterised invariants contains 917 entries: 176 entries corresponding to a single constraint and 741 entries corresponding to pairs of constraints. The total number of invariants is 2347: 759 linear invariants, 899 conditional linear invariants, 666 non-linear invariants, and 23 manually derived invariants.

There exist two complementary versions of this database of parameterised invariants, namely a machine-readable version where each invariant is represented as a Prolog fact, and a human-readable version where these Prolog facts are pretty-printed as **\LaTeX** formulae. The machine-readable version is available from the electronic constraint catalogue given in Appendix A of this document as a stand-alone Prolog attached file, while the human-readable version is given in this chapter. Within the human-readable version, invariants are sorted by lexicographically increasing order on the tuples of lexicographically sorted constraint names. Figures that display the
invariants linking the result variables of two time-series constraints use the following conventions, where Figure 5.1 illustrates how the various types of invariants are depicted. For this illustrative purpose we use the following set of invariants, where after a linear invariant we possibly provide the coordinates of two distinct feasible points lying on the line corresponding to the linear invariant.

\[ 2x \leq y \]
\[ n \geq 9: \ (1, 2) \quad (2, 4) \]
\[ x > 0 \land y > 0 \Rightarrow 4x + y \leq 2n - 6 \]
\[ x > 0 \land y > 0 \Rightarrow x + y \leq n - 1 \]
\[ n \geq 9: \ (1, n - 2) \quad (2, n - 3) \]
\[ y \neq 2x + 1 \]
\[ 2x + y \leq 2n - 2 \]
\[ \left( \begin{array}{c}
    x \neq \left\lfloor \frac{n}{2} \right\rfloor , \\
    y < 1, \\
    y > n \cdot \min(1, \max(0, n - 1)) - 3, \\
    n \mod 2 = 1, \\
    y \mod 2 = 0
\end{array} \right) \]
\[ y \neq 1 \]
\[ y \leq 3x \]
\[ n \geq 5: \ (0, 0) \quad (1, 3) \]
\[ y = \max(0, n - 2) \land n > 1 \Rightarrow 2x \geq n - 2 - n \mod 2 \]
\[ n > 1 \Rightarrow 2x + y \leq 2n - 3 \]
\[ n \geq 7: \ (\max(0, n - 2) - 1, 3) \quad (\max(0, n - 2) - 2, 5) \]
\[ y > 0 \Rightarrow x \leq n - 3 \]
\[ n \geq 5: \ (\max(0, n - 2) - 1, 2) \quad (\max(0, n - 2) - 1, 3) \]
\[ y \neq 1 \]

where:

- the invariants \[ 1 \], \[ 2 \], \[ 3 \], and \[ 4 \] are associated with the pair (NB_DIP_ON_INCREASING_SEQUENCE, SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE),
- the invariants \[ 5 \], \[ 6 \], and \[ 7 \] are associated with the pair (NB_STEADY_SEQUENCE, SUM_WIDTH_DECREASING_SEQUENCE),
- the invariants \[ 8 \], \[ 9 \], \[ 10 \], \[ 11 \], and \[ 12 \] are associated with the pair (SUM_WIDTH_SUMMIT, SUM_WIDTH_ZIGZAG).

In our figures, e.g. Figure 5.1, we use the following colour and shape scheme:

- A feasible pair of values of result variables is shown with a blue circle: light blue for a pair for which at least one of the values corresponds to the default value of one of the two time-series constraints, and blue otherwise. The convex hull of all feasible pairs of values is shown as an enclosing blue dotted line.
• A yellow circle denotes an infeasible point that is located outside the convex hull of feasible points and that is not discarded by any invariants; e.g. see the yellow point of coordinates $(3, 9)$ of Part (C) of Figure 5.1.

• A red circle denotes an infeasible point inside the convex hull that is not discarded by any non-linear invariants, but that may be discarded by some conditional linear invariants; e.g. see the red point of coordinates $(5, 8)$ for the pair $(\text{SUM}_-\text{WIDTH}_-\text{PLATEAU}, \text{SUM}_-\text{WIDTH}_-\text{ZIGZAG})$ and the sequence size 11.

• A diamond-shaped point denotes an infeasible point outside the convex hull, that belongs to a set of infeasible points discarded by some non-linear invariant. An illustrative example is the point of coordinates $(3, 7)$ in Part (A) of Figure 5.1, which is located outside the convex hull of feasible points and is associated with the disequality $\lozenge y \neq 2 \cdot x + 1$.

• A violet-coloured point denotes an infeasible point discarded by an invariant corresponding to a disequality between a result variable and a natural number. For instance, the violet points in Part (B) of Figure 5.1 correspond to the set of infeasible points associated with the disequality $\lozenge y \neq 1$.

• An orange-coloured point denotes an infeasible point discarded by an invariant corresponding to a diagonal, horizontal line, or vertical line that starts at a specific point and extends to the border in one direction, or to a horizontal or vertical line that excludes points for which one of the coordinates is odd or even. For instance, the orange points in Part (A) of Figure 5.1 correspond to the set of infeasible points associated with $\ket{y \neq 2 \cdot x + 1}$.

• A brown-coloured point denotes an infeasible point discarded by a disjunction. For instance, the brown points in Part (B) of Figure 5.1 correspond to the set of infeasible points associated with $\lozenge$.

• A cyan-coloured (respectively pink-coloured) half-space corresponds to infeasible points that are forbidden by linear inequalities (respectively by conditional linear inequalities). In Part (A) of Figure 5.1, the set of points that are forbidden by the linear inequality $\lozenge 2 \cdot x \leq y$ is coloured in cyan, while the set of points that are discarded by the conditional linear inequality $\lozenge x > 0 \land y > 0 \Rightarrow x + y \leq n - 1$ is coloured in pink.

• A thick light violet line represents a set of infeasible points that are discarded by an implication of the form $x = v \Rightarrow \ldots$ (or $y = v \Rightarrow \ldots$), where $v$ is a natural number. For instance, the thick light violet line in Part (C) of Figure 5.1 corresponds to the set of infeasible points associated with $\lozenge$.

• For each linear invariant for which we prove sharpness, we provide two distinct points that are both feasible and are located on the line corresponding to the linear invariant. Note that the coordinates of such points may be parameterised by the sequence length $n$. For instance, given the conjunction $\text{NB}_-\text{DIP}_-\text{ON}_-\text{INCREASING}_-\text{SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM}_-\text{WIDTH}_-\text{STRICLTY}_-\text{DECREASING}_-\text{SEQUENCE}(y, \text{VARIABLES})$, we give
Figure 5.1: Illustration of how the different types of invariants are depicted

for the sharp invariant \( x > 0 \land y > 0 \Rightarrow x + y \leq n - 1 \) (sharp when \( n \geq 11 \)) the two points \((1, n - 2)\) and \((2, n - 3)\), which (1) satisfy the equality \( x + y = n - 1 \) and (2) are feasible wrt the mentioned conjunction of time-series constraints, provided \( n \geq 11 \). Such points are graphically depicted by a coloured squared box; e.g. see the points of coordinates \((1, 10)\) and \((2, 9)\) in Part (A) of Figure 5.1, where \( n = 12 \).
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{MAX}_\text{MAX}_\text{BUMP}_\text{ON}_\text{DECREASING}\text{SEQUENCE} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

\[ \text{MAX}_\text{MAX}_\text{DECREASING} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

\[ \text{MAX}_\text{MAX}_\text{DECREASING}_\text{SEQUENCE} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

\[ \text{MAX}_\text{MAX}_\text{DIP}_\text{ON}_\text{INCREASING}_\text{SEQUENCE} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

\[ \text{MAX}_\text{MAX}_\text{INCREASING} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

\[ \text{MAX}_\text{MAX}_\text{INCREASING}_\text{SEQUENCE} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

\[ \text{MAX}_\text{MAX}_\text{INFLEXION} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

\[ \text{MAX}_\text{MAX}_\text{PEAK} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

\[ \text{MAX}_\text{MAX}_\text{STRICTLY}_{\text{DECREASING}}\text{\_SEQUENCE} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

\[ \text{MAX}_\text{MAX}_\text{STRICTLY}_{\text{INCREASING}}\text{\_SEQUENCE} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

\[ \text{MAX}_\text{MAX}_\text{SUMMIT} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

\[ \text{MAX}_\text{MAX}_\text{ZIGZAG} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

\[ \text{MAX}_\text{MIN}_\text{BUMP}_\text{ON}_\text{DECREASING}_\text{SEQUENCE} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]
\[
\forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1}
\]

\[
\forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1}
\]

\[
\forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1}
\]

\[
\forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1}
\]

\[
\forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1}
\]

\[
\forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1}
\]

\[
\forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1}
\]

\[
\forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1}
\]

\[
\forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1}
\]

\[
\forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1}
\]

\[
\forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1}
\]

\[
\forall i \in [3, |\text{VARIABLES}|] : r_i \geq \text{var}_{i-2} - \text{var}_{i-1}
\]
$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [3, |\text{VARIABLES}|]: r_i \geq \text{var}_{i-1} - \text{var}_{i-2}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$

$\forall i \in [2, |\text{VARIABLES}|]: r_i \geq r_{i-1}$
MAX_SURF_INCREASING_SEQUENCE
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]

MAX_SURF_INCREASING_TERRACE
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]

MAX_SURF_INFLEXION
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]

MAX_SURF_PEAK
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]

MAX_SURF_PLAIN
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]

MAX_SURF_PLATEAU
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]

MAX_SURF_PROPER_PLAIN
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]

MAX_SURF_PROPER_PLATEAU
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]

MAX_SURF_STEADY
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]

MAX_SURF_STEADY_SEQUENCE
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]

MAX_SURF.Strictly_DECREASING_SEQUENCE
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]

MAX_SURF.Strictly_INCREASING_SEQUENCE
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]

MAX_SURF.Summit
\[ \forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \]
5. A DATABASE OF PARAMETERISED INVARIANTS

**MAX_SURF_VALLEY**
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

**MAX_SURF_ZIGZAG**
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]

**MAX_WIDTH_DECREASING_SEQUENCE**
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} + d_i \]

**MAX_WIDTH_DECREASING_TERRACE**
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} + d_i \]

**MAX_WIDTH_GORGE**
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} + d_i \]

**MAX_WIDTH_INCREASING_SEQUENCE**
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} + d_i \]

**MAX_WIDTH_INCREASING_TERRACE**
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} + d_i \]

**MAX_WIDTH_INFLEXION**
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} + d_i \]

**MAX_WIDTH PEAK**
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} + d_i \]

**MAX_WIDTH_plain**
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \]
\[ \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} + d_i \]
∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \geq \lfloor r_{i-1} \rfloor

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \leq \lfloor r_{i-1} \rfloor + d_i

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \geq \lfloor r_{i-1} \rfloor

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \leq \lfloor r_{i-1} \rfloor + d_i

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \geq \lfloor r_{i-1} \rfloor

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \leq \lfloor r_{i-1} \rfloor + d_i

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \geq \lfloor r_{i-1} \rfloor

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \leq \lfloor r_{i-1} \rfloor + d_i

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \geq \lfloor r_{i-1} \rfloor

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \leq \lfloor r_{i-1} \rfloor + d_i

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \geq \lfloor r_{i-1} \rfloor

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \leq \lfloor r_{i-1} \rfloor + d_i

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \geq \lfloor r_{i-1} \rfloor

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \leq \lfloor r_{i-1} \rfloor + d_i

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \geq \lfloor r_{i-1} \rfloor

∀ i ∈ [2, \lbrack \text{VARIABLES} \rbrack] : \lfloor r_i \rfloor \leq \lfloor r_{i-1} \rfloor + d_i
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{MIN\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE} \\
\forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1}
\]

\[
\text{MIN\_MAX\_DECREASING} \\
\forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1}
\]

\[
\text{MIN\_MAX\_DECREASING\_SEQUENCE} \\
\forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1}
\]

\[
\text{MIN\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE} \\
\forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1}
\]

\[
\text{MIN\_MAX\_INCREASING} \\
\forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1}
\]

\[
\text{MIN\_MAX\_INCREASING\_SEQUENCE} \\
\forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1}
\]

\[
\text{MIN\_MAX\_INFLEXION} \\
\forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1}
\]

\[
\text{MIN\_MAX\_PEAK} \\
\forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1}
\]

\[
\text{MIN\_MAX\_SUMMIT} \\
\forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1}
\]

\[
\text{MIN\_MAX\_ZIGZAG} \\
\forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1}
\]

\[
\text{MIN\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE} \\
\forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1}
\]
\( \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \)

**MIN_MIN_DECREASING**

**MIN_MIN_DECREASING_SEQUENCE**

**MIN_MIN_DIP_ON_INCREASING_SEQUENCE**

**MIN_MIN_GORGE**

**MIN_MIN_INCREASING**

**MIN_MIN_INCREASING_SEQUENCE**

**MIN_MIN_INFLEXION**

**MIN_MIN_LEAST_DECREASINGSEQUENCE**

**MIN_MIN_LEAST_INCREASINGSEQUENCE**

**MIN_MIN_Valley**

**MIN_MIN_ZIGZAG**

**MIN_RANGE_DECREASING**

**MIN_RANGE_DECREASING_SEQUENCE**
5. A DATABASE OF PARAMETERISED INVARIANTS

\textbf{MIN\_RANGE\_INCREASING}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]

\textbf{MIN\_RANGE\_INCREASING\_SEQUENCE}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]

\textbf{MIN\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]

\textbf{MIN\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]

\textbf{MIN\_SURF\_BUMP\_ON\_DECREASING\_SEQUENCE}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]

\textbf{MIN\_SURF\_DECREASING}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]

\textbf{MIN\_SURF\_DECREASING\_SEQUENCE}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]

\textbf{MIN\_SURF\_DECREASING\_TERRACE}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]

\textbf{MIN\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]

\textbf{MIN\_SURF\_GORGE}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]

\textbf{MIN\_SURF\_INCREASING}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]

\textbf{MIN\_SURF\_INCREASING\_SEQUENCE}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]

\textbf{MIN\_SURF\_INCREASING\_TERRACE}
\[ \forall i \in [2, |\text{VARIABLES}|]: r_i \leq r_{i-1} \]
\[
\begin{align*}
\text{MINSURF\_INFLUXION} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \\
\text{MINSURF\_PEAK} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \\
\text{MINSURF\_PLAIN} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \\
\text{MINSURF\_PLATEAU} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \\
\text{MINSURF\_PROPER\_PLAIN} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \\
\text{MINSURF\_PROPER\_PLATEAU} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \\
\text{MINSURF\_STEADY} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \\
\text{MINSURF\_STEADY\_SEQUENCE} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \\
\text{MINSURF\_STRICTLY\_DECREASING\_SEQUENCE} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \\
\text{MINSURF\_STRICTLY\_INCREASING\_SEQUENCE} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \\
\text{MINSURF\_SUMMIT} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \\
\text{MINSURF\_VALLEY} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1} \\
\text{MINSURF\_ZIGZAG} & \quad \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-1}
\end{align*}
\]
\textbf{MIN\_WIDTH\_DECREASING\_SEQUENCE}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]

\textbf{MIN\_WIDTH\_DECREASING\_TERRACE}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]

\textbf{MIN\_WIDTH\_GORGE}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]

\textbf{MIN\_WIDTH\_INCREASING\_SEQUENCE}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]

\textbf{MIN\_WIDTH\_INCREASING\_TERRACE}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]

\textbf{MIN\_WIDTH\_INPLEXION}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]

\textbf{MIN\_WIDTH\_PEAK}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]

\textbf{MIN\_WIDTH\_PLAIN}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]

\textbf{MIN\_WIDTH\_PLATEAU}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]

\textbf{MIN\_WIDTH\_PROPER\_PLAIN}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]

\textbf{MIN\_WIDTH\_PROPER\_PLATEAU}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]

\textbf{MIN\_WIDTH\_STEADY\_SEQUENCE}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]

\textbf{MIN\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}\[ \forall i \in [2,|\text{VARIABLES}|] : r_i \leq r_{i-1} \]
∀ i ∈ [2, |VARIABLES|] : r_i ≤ r_{i-1}

MIN_WIDTH_SUMMIT
∀ i ∈ [2, |VARIABLES|] : r_i ≤ r_{i-1}

MIN_WIDTH_VALLEY
∀ i ∈ [2, |VARIABLES|] : r_i ≤ r_{i-1}

MIN_WIDTH_ZIGZAG
∀ i ∈ [2, |VARIABLES|] : r_i ≤ r_{i-1}

NB_DECREASING
∀ i ∈ [2, |VARIABLES|] : r_i ≥ r_{i-1}
∀ i ∈ [2, |VARIABLES|] : r_i ≤ r_{i-1} + 1
∀ i ∈ [1, |VARIABLES|] : o_i > r_i

NB_DECREASING_SEQUENCE
∀ i ∈ [2, |VARIABLES|] : r_i ≥ r_{i-1}
∀ i ∈ [3, |VARIABLES|] : r_i ≤ r_{i-2} + 1
∀ i ∈ [1, |VARIABLES|] : o_i ≥ r_i

NB_DECREASING_TERRACE
∀ i ∈ [2, |VARIABLES|] : r_i ≥ r_{i-1}
∀ i ∈ [3, |VARIABLES|] : r_i ≤ r_{i-2} + 1
∀ i ∈ [1, |VARIABLES|] : o_i ≥ r_i + r_i > 0
∀ i ∈ [1, |VARIABLES|] : o_i ≥ r_i

NB_GORGE
∀ i ∈ [2, |VARIABLES|] : r_i ≥ r_{i-1}
∀ i ∈ [3, |VARIABLES|] : r_i ≤ r_{i-2} + 1
∀ i ∈ [1, |VARIABLES|] : o_i ≥ r_i
∀ i ∈ [1, |VARIABLES|] : o_i ≥ r_i

NB_INCREASING
∀ i ∈ [2, |VARIABLES|] : r_i ≥ r_{i-1}
∀ i ∈ [2, |VARIABLES|] : r_i ≤ r_{i-1} + 1
∀ i ∈ [1, |VARIABLES|] : o_i ≥ r_i
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \forall i \in [2, \text{VARIABLES}] : r_i \geq r_{i-1} \]
\[ \forall i \in [3, \text{VARIABLES}] : r_i \leq r_{i-2} + 1 \]
\[ \forall i \in [1, \text{VARIABLES}] : o_i^{<} \geq r_i \]

\[ \forall i \in [2, \text{VARIABLES}] : r_i \geq r_{i-1} \]
\[ \forall i \in [3, \text{VARIABLES}] : r_i \leq r_{i-2} + 1 \]
\[ \forall i \in [1, \text{VARIABLES}] : o_i^{<} \geq r_i + r_1 > 0 \]
\[ \forall i \in [1, \text{VARIABLES}] : o_i^{=} \geq r_i \]

\[ \forall i \in [2, \text{VARIABLES}] : r_i \geq r_{i-1} \]
\[ \forall i \in [2, \text{VARIABLES}] : r_i \leq r_{i-1} + 1 \]

\[ \forall i \in [2, \text{VARIABLES}] : r_i \geq r_{i-1} \]
\[ \forall i \in [3, \text{VARIABLES}] : r_i \leq r_{i-2} + 1 \]
\[ \forall i \in [1, \text{VARIABLES}] : o_i^{<} \geq r_i \]
\[ \forall i \in [1, \text{VARIABLES}] : o_i^{=} \geq r_i \]

\[ \forall i \in [2, \text{VARIABLES}] : r_i \geq r_{i-1} \]
\[ \forall i \in [3, \text{VARIABLES}] : r_i \leq r_{i-2} + 1 \]
\[ \forall i \in [1, \text{VARIABLES}] : o_i^{<} \geq r_i \]
\[ \forall i \in [1, \text{VARIABLES}] : o_i^{=} \geq r_i \]

\[ \forall i \in [2, \text{VARIABLES}] : r_i \geq r_{i-1} \]
\[ \forall i \in [4, \text{VARIABLES}] : r_i \leq r_{i-3} + 1 \]
\[ \forall i \in [1, \text{VARIABLES}] : o_i^{<} \geq r_i \]
\[ \forall i \in [1, \text{VARIABLES}] : o_i^{=} \geq r_i \]
\[ \forall i \in [1, \text{VARIABLES}] : o_i^{=} \geq r_i \]
\( \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \)
\( \forall i \in [4, |\text{VARIABLES}|] : r_i \leq r_{i-3} + 1 \)
\( \forall i \in [1, |\text{VARIABLES}|] : o_i^\neq \geq r_i \)
\( \forall i \in [1, |\text{VARIABLES}|] : o_i^> \geq r_i \)
\( \forall i \in [1, |\text{VARIABLES}|] : o_i^\leq \geq r_i \)

\textbf{NB\_STORM}
\( \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \)
\( \forall i \in [2, |\text{VARIABLES}|] : r_i \leq r_{i-3} + 1 \)
\( \forall i \in [1, |\text{VARIABLES}|] : o_i^\neq = r_i \)

\textbf{NB\_STORM\_SEQUENCE}
\( \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \)
\( \forall i \in [3, |\text{VARIABLES}|] : r_i \leq r_{i-2} + 1 \)
\( \forall i \in [1, |\text{VARIABLES}|] : o_i^\neq \geq r_i \)

\textbf{NB\_STORM\_DECREASING\_SEQUENCE}
\( \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \)
\( \forall i \in [3, |\text{VARIABLES}|] : r_i \leq r_{i-2} + 1 \)
\( \forall i \in [1, |\text{VARIABLES}|] : o_i^> \geq r_i \)

\textbf{NB\_STORM\_INCREASING\_SEQUENCE}
\( \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \)
\( \forall i \in [3, |\text{VARIABLES}|] : r_i \leq r_{i-2} + 1 \)
\( \forall i \in [1, |\text{VARIABLES}|] : o_i^< \geq r_i \)

\textbf{NB\_SUMMIT}
\( \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \)
\( \forall i \in [3, |\text{VARIABLES}|] : r_i \leq r_{i-2} + 1 \)
\( \forall i \in [1, |\text{VARIABLES}|] : o_i^\neq \geq r_i \)
\( \forall i \in [1, |\text{VARIABLES}|] : o_i^> \geq r_i \)

\textbf{NB\_VALLEY}
\( \forall i \in [2, |\text{VARIABLES}|] : r_i \geq r_{i-1} \)
\( \forall i \in [3, |\text{VARIABLES}|] : r_i \leq r_{i-2} + 1 \)
\( \forall i \in [1, |\text{VARIABLES}|] : o_i^\neq \geq r_i \)
\( \forall i \in [1, |\text{VARIABLES}|] : o_i^< \geq r_i \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{NB\_ZIGZAG} & \\
\forall i \in [2, |\text{VARIABLES}|] & : r_i \geq r_{i-1} \\
\forall i \in [4, |\text{VARIABLES}|] & : r_i \leq r_{i-3} + 1 \\
\forall i \in [1, |\text{VARIABLES}|] & : o_i^\leq \geq r_i \\
\forall i \in [1, |\text{VARIABLES}|] & : o_i^> \geq r_i \\
\forall i \in [1, |\text{VARIABLES}|] & : o_i^\leq \leq |\text{VARIABLES}| - i - 3 \times r_i \\
\text{SUM\_RANGE\_DECREASING} & \\
\forall i \in [3, |\text{VARIABLES}|] & : r_i - r_{i-2} \geq \text{var}_i - \text{var}_{i-2} \\
\forall i \in [2, |\text{VARIABLES}|] & : r_i - r_{i-1} \geq \text{var}_i - \text{var}_{i-1} \\
\text{SUM\_WIDTH\_DECREASING\_SEQUENCE} & \\
\forall i \in [2, |\text{VARIABLES}|] & : r_i \geq r_{i-1} \\
\forall i \in [2, |\text{VARIABLES}|] & : r_i \leq r_{i-1} + d_{i-1} + 1 \\
\forall i \in [1, |\text{VARIABLES}|] & : o_i^\leq \geq \min(2, r_i - 1) \\
\text{SUM\_WIDTH\_DECREASING\_TERRACE} & \\
\forall i \in [2, |\text{VARIABLES}|] & : r_i \geq r_{i-1} \\
\forall i \in [2, |\text{VARIABLES}|] & : r_i \leq r_{i-1} + d_{i-1} + 1 \\
\forall i \in [1, |\text{VARIABLES}|] & : o_i^\leq \geq \min(2, r_i) \\
\forall i \in [1, |\text{VARIABLES}|] & : o_i^> \geq \lfloor (r_i + 1)/2 \rfloor \\
\text{SUM\_WIDTH\_GORGE} & \\
\forall i \in [2, |\text{VARIABLES}|] & : r_i \geq r_{i-1} \\
\forall i \in [2, |\text{VARIABLES}|] & : r_i \leq r_{i-1} + d_{i-1} + 1 \\
\forall i \in [1, |\text{VARIABLES}|] & : o_i^\leq \geq \min(1, r_i) \\
\forall i \in [1, |\text{VARIABLES}|] & : o_i^> \geq \min(1, r_i) \\
\text{SUM\_WIDTH\_INCREASING\_SEQUENCE} & \\
\forall i \in [2, |\text{VARIABLES}|] & : r_i \geq r_{i-1} \\
\forall i \in [2, |\text{VARIABLES}|] & : r_i \leq r_{i-1} + d_{i-1} + 1 \\
\forall i \in [1, |\text{VARIABLES}|] & : o_i^\leq \geq \min(2, r_i - 1) \\
\text{SUM\_WIDTH\_INCREASING\_TERRACE} & \\
\forall i \in [2, |\text{VARIABLES}|] & : r_i \geq r_{i-1} \\
\forall i \in [2, |\text{VARIABLES}|] & : r_i \leq r_{i-1} + d_{i-1} + 1 \\
\forall i \in [1, |\text{VARIABLES}|] & : o_i^\leq \geq \min(2, r_i) \\
\forall i \in [1, |\text{VARIABLES}|] & : o_i^> \geq \lfloor (r_i + 1)/2 \rfloor \\
\end{align*}
\[
\forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \\
\forall i \in [2, |VARIABLES|] : r_i \leq r_{i-1} + d_{i-1} + 1
\]

**SUM_WIDTH_PEAK**

\[
\forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \\
\forall i \in [2, |VARIABLES|] : r_i \leq r_{i-1} + d_{i-1} + 1 \\
\forall i \in [1, |VARIABLES|] : o_i^x \geq \min(1, r_i) \\
\forall i \in [1, |VARIABLES|] : o_i^\circ \geq \min(1, r_i)
\]

**SUM_WIDTHPLAIN**

\[
\forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \\
\forall i \in [2, |VARIABLES|] : r_i \leq r_{i-1} + d_{i-1} + 1 \\
\forall i \in [1, |VARIABLES|] : o_i^x \geq \min(1, r_i) \\
\forall i \in [1, |VARIABLES|] : o_i^\circ \geq \min(1, r_i) \\
\forall i \in [1, |VARIABLES|] : o_i^\circ \geq r_i - \lfloor \max(0, |VARIABLES| - i)/2 \rfloor
\]

**SUM_WIDTH_PLATEAU**

\[
\forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \\
\forall i \in [2, |VARIABLES|] : r_i \leq r_{i-1} + d_{i-1} + 1 \\
\forall i \in [1, |VARIABLES|] : o_i^x \geq \min(1, r_i) \\
\forall i \in [1, |VARIABLES|] : o_i^\circ \geq \min(1, r_i) \\
\forall i \in [1, |VARIABLES|] : o_i^\circ \geq r_i - \lfloor \max(0, |VARIABLES| - i)/2 \rfloor
\]

**SUM_WIDTH_PROPERPLAIN**

\[
\forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \\
\forall i \in [2, |VARIABLES|] : r_i \leq r_{i-1} + d_{i-1} + 1 \\
\forall i \in [1, |VARIABLES|] : o_i^x \geq \min(1, r_i) \\
\forall i \in [1, |VARIABLES|] : o_i^\circ \geq \min(1, r_i) \\
\forall i \in [1, |VARIABLES|] : o_i^\circ \geq \lceil (r_i + 1)/2 \rceil
\]

**SUM_WIDTH_PROPERPLATEAU**

\[
\forall i \in [2, |VARIABLES|] : r_i \geq r_{i-1} \\
\forall i \in [2, |VARIABLES|] : r_i \leq r_{i-1} + d_{i-1} + 1 \\
\forall i \in [1, |VARIABLES|] : o_i^x \geq \min(1, r_i) \\
\forall i \in [1, |VARIABLES|] : o_i^\circ \geq \min(1, r_i) \\
\forall i \in [1, |VARIABLES|] : o_i^\circ \geq \lceil (r_i + 1)/2 \rceil
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

**SUM\_WIDTH\_STEADY\_SEQUENCE**

\[\forall i \in [2, \text{VARIABLES}] : r_i \geq r_{i-1}\]
\[\forall i \in [2, \text{VARIABLES}] : r_i \leq r_{i-1} + 2\]
\[o_i^w \geq \lceil (r_i + 1)/2 \rceil\]
\[o_i^w \leq \max(0, r_i - 1)\]

**SUM\_WIDTH\_STRICKLY\_DECREASING\_SEQUENCE**

\[\forall i \in [2, \text{VARIABLES}] : r_i \geq r_{i-1}\]
\[\forall i \in [2, \text{VARIABLES}] : r_i \leq r_{i-1} + 2\]
\[\forall i \in [1, \text{VARIABLES}] : o_i^> \geq \lceil (r_i + 1)/2 \rceil\]
\[\forall i \in [1, \text{VARIABLES}] : o_i^< \leq \max(0, r_i - 1)\]

**SUM\_WIDTH\_STRICKLY\_INCREASING\_SEQUENCE**

\[\forall i \in [2, \text{VARIABLES}] : r_i \geq r_{i-1}\]
\[\forall i \in [2, \text{VARIABLES}] : r_i \leq r_{i-1} + 2\]
\[\forall i \in [1, \text{VARIABLES}] : o_i^> \geq \lceil (r_i + 1)/2 \rceil\]
\[\forall i \in [1, \text{VARIABLES}] : o_i^< \leq \max(0, r_i - 1)\]

**SUM\_WIDTH\_SUMMIT**

\[\forall i \in [2, \text{VARIABLES}] : r_i \geq r_{i-1}\]
\[\forall i \in [2, \text{VARIABLES}] : r_i \leq r_{i-1} + d_{i-1} + 1\]
\[\forall i \in [1, \text{VARIABLES}] : o_i^> \geq \min(1, r_i)\]
\[\forall i \in [1, \text{VARIABLES}] : o_i^< \geq \min(1, r_i)\]

**SUM\_WIDTH\_VALLEY**

\[\forall i \in [2, \text{VARIABLES}] : r_i \geq r_{i-1}\]
\[\forall i \in [2, \text{VARIABLES}] : r_i \leq r_{i-1} + d_{i-1} + 1\]
\[\forall i \in [1, \text{VARIABLES}] : o_i^> \geq \min(1, r_i)\]
\[\forall i \in [1, \text{VARIABLES}] : o_i^< \geq \min(1, r_i)\]

**SUM\_WIDTH\_ZIGZAG**

\[\forall i \in [2, \text{VARIABLES}] : r_i \geq r_{i-1}\]
\[\forall i \in [2, \text{VARIABLES}] : r_i \leq r_{i-1} + 2\]
\[\forall i \in [1, \text{VARIABLES}] : o_i^> \geq \lceil (r_i + 1)/2 \rceil\]
\[\forall i \in [1, \text{VARIABLES}] : o_i^< \geq \lceil (r_i + 1)/2 \rceil\]
\[ \text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_DECREASING}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 9: \ (1, sv - 2) \ (2, sv - 3) \)
2. \( y = 0 \Rightarrow x = 0 \)
3. \( x > 0 \land y > 0 \Rightarrow 2 \times x \leq y - 2 \)
   - \( sv \geq 9: \ (1, 4) \ (2, 6) \)
4. \( x < 1 \lor y \neq 3 \)

\[ \text{SV} = 12 \]

\[ \text{SV} = 11 \]

\[ \text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \]
\[ \text{NB\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y = 0 \Rightarrow x = 0 \)
2. \( x > 0 \land y > 0 \Rightarrow x + 2 \times y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 12: \ (1, \lfloor sv/2 \rfloor - 1) \ (3, \lfloor sv/2 \rfloor - 2) \)
   - \( sv \mod 2 = 2 \land sv \geq 15: \ (2, \lfloor sv/2 \rfloor - 1) \ (4, \lfloor sv/2 \rfloor - 2) \)
3. \( x > 0 \land y > 0 \Rightarrow x \leq y - 1 \)
   - \( sv \geq 9: \ (1, 2) \ (2, 3) \)

\[ \text{SV} = 12 \]
\[ \text{SV} = 11 \]
A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(sv > 1 \Rightarrow 3 \times x + 2 \times y \leq sv - 2 - (sv - 2) \mod 2\)
2. \(x > 0 \land y > 0 \Rightarrow 3 \times x + 2 \times y \leq sv - 3\)

\[
\begin{align*}
\text{sv mod 2} = 1 \land sv \geq 9: & \quad (0, \lfloor (sv - 2)/2 \rfloor) \quad (2, \lfloor (sv - 2)/2 \rfloor - 3) \\
\text{sv mod 2} = 0 \land sv \geq 12: & \quad (1, \lfloor (sv - 2)/2 \rfloor - 2) \quad (3, \lfloor (sv - 2)/2 \rfloor - 5)
\end{align*}
\]
\text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x,\text{VARIABLES}) \land \\
\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(y,\text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( sv > 2 \Rightarrow 3 \cdot x + 3 \cdot y \leq sv - 3 - (sv - 3) \mod 3 \)
   - \( sv \mod 3 = 0 \land sv \geq 6 \): (\(\lfloor (sv - 3)/3 \rfloor\), 0) (0, \(\lfloor (sv - 3)/3 \rfloor\))
   - \( sv \mod 3 = 1 \land sv \geq 6 \): (\(\lfloor (sv - 3)/3 \rfloor\), 0) (0, \(\lfloor (sv - 3)/3 \rfloor\))
   - \( sv \mod 3 = 2 \land sv \geq 8 \): (\(\lfloor (sv - 3)/3 \rfloor\), 0) (0, \(\lfloor (sv - 3)/3 \rfloor - 1\))

2. \( x > 0 \land y > 0 \Rightarrow 3 \cdot x + 3 \cdot y \leq sv - 5 - (sv - 5) \mod 3 \)
   - \( (sv - 2) \mod 3 = 0 \land sv \geq 11 \): (\(\lfloor (sv - 3)/3 \rfloor - 1\), 1) (\(\lfloor (sv - 3)/3 \rfloor - 2\), 2)
   - \( (sv - 2) \mod 3 = 1 \land sv \geq 12 \): (\(\lfloor (sv - 3)/3 \rfloor - 2\), 1) (\(\lfloor (sv - 3)/3 \rfloor - 3\), 2)
   - \( (sv - 2) \mod 3 = 2 \land sv \geq 13 \): (\(\lfloor (sv - 3)/3 \rfloor - 2\), 1) (\(\lfloor (sv - 3)/3 \rfloor - 3\), 2)

3. \( \lor \begin{cases} 
   x \neq \max(0, \lfloor sv/3 \rfloor) - 1, \\
   y < 1, \\
   y > \max(0, \lfloor sv/3 \rfloor) - 1, \\
   1 = sv \mod 2, \\
   0 = y \mod 2 
\end{cases} \)

4. \( \lor \begin{cases} 
   x \neq \max(0, \lfloor sv/3 \rfloor) - 1, \\
   x < 1, \\
   x > \max(0, \lfloor sv/3 \rfloor) - 1, \\
   1 = sv \mod 2, \\
   0 = x \mod 2 
\end{cases} \)

\( sv = 12 \)

\( sv = 11 \)
A DATABASE OF PARAMETERISED INVARIANTS

$\text{NB}_\text{BUMP}_\text{ON}_\text{DECREASING}_\text{SEQUENCE}(x, \text{VARIABLES}) \land \text{NB}_\text{GORGE}(y, \text{VARIABLES})$ with $sv = |\text{VARIABLES}|$

1. $x \leq y$
   - $sv \geq 6$: (0, 0) (1, 1)

2. $x > 0 \land y > 0 \Rightarrow x + 2 \cdot y \leq sv - 2$
   - $sv \mod 2 = 0 \land sv \geq 16$: (2, $\lceil (sv - 1)/2 \rceil - 1$) (4, $\lceil (sv - 1)/2 \rceil - 2$)
   - $sv \mod 2 = 1 \land sv \geq 13$: (1, $\lceil (sv - 1)/2 \rceil - 1$) (3, $\lceil (sv - 1)/2 \rceil - 2$)

$sv = 12$

$sv = 11$

$\text{NB}_\text{BUMP}_\text{ON}_\text{DECREASING}_\text{SEQUENCE}(x, \text{VARIABLES}) \land \text{NB}_\text{INCREASING}(y, \text{VARIABLES})$ with $sv = |\text{VARIABLES}|$

1. $x \leq y$
   - $sv \geq 9$: (1, 1) (2, 2)

2. $x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq sv - 3$
   - $sv \geq 9$: (1, $sv - 5$) (2, $sv - 7$)

3. $y \neq \max(0, sv - 1) - 3$
   - $x < 1$
   - $x > \max(0, \lfloor (sv - 3)/2 \rfloor) - 1$
   - $0 = x \mod 2$

$sv = 12$

$sv = 11$
\( \text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x \leq y \)
   - \( sv \geq 9: \) (1, 1) (2, 2)
2. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq sv - 3 \)
3. \( x > 0 \land y > 0 \Rightarrow x + 2 \cdot y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 14: \) (1, \( \lfloor sv/2 \rfloor - 1 \)) (3, \( \lfloor sv/2 \rfloor - 2 \))
   - \( sv \mod 2 = 1 \land sv \geq 17: \) (2, \( \lfloor sv/2 \rfloor - 1 \)) (4, \( \lfloor sv/2 \rfloor - 2 \))
A DATABASE OF PARAMETERISED INVARIANTS

\[\text{NB}_\text{BUMP ON DECREASING SEQUENCE}(x, \text{VARIABLES}) \land \text{NB}_\text{INCREASING TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

\[\begin{align*}
\text{1} & : sv > 1 \Rightarrow 3x + 2y \leq sv - 2 - (sv - 2) \mod 2 \\
\text{2} & : x > 0 \land y > 0 \Rightarrow 3x + 2y \leq sv - 4 \\
\text{3} & : sv \mod 2 = 1 \land sv \geq 13: \\
& \quad \begin{cases} 
        y \neq \max(0, \lfloor sv - 2/2 \rfloor - 2), \\
        x < 1, \\
        x > \max(0, \lceil (sv - 3)/3 \rceil) - 1, \\
        1 = sv \mod 2, \\
        0 = x \mod 2 
    \end{cases}
\end{align*}\]

\[sv = 12\]

\[sv = 11\]
NB_BUMP_ON_DECREASING_SEQUENCE(\(x\), VARIABLES) ∧
NB_INFLEXION(\(y\), VARIABLES) with \(sv = |\text{VARIABLES}|\)

1. \(2 \cdot x \leq y\)
   \(\quad sv \geq 6: (0, 0) \quad (1, 2)\)
2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
3. \(x > 0 \land y > 0 \Rightarrow x + y \leq sv - 3\)

\(sv \geq 9: (1, sv - 4) \quad (2, sv - 5)\)

\(sv = 12\)

\(sv = 11\)

NB_BUMP_ON_DECREASING_SEQUENCE(\(x\), VARIABLES) ∧
NB_PEAK(\(y\), VARIABLES) with \(sv = |\text{VARIABLES}|\)

1. \(x \leq y\)
   \(\quad sv \geq 6: (0, 0) \quad (1, 1)\)
2. \(x > 0 \land y > 0 \Rightarrow x + 2 \cdot y \leq sv - 2\)
   \(\quad sv \mod 2 = 0 \land sv \geq 16: (2, \lfloor sv - 1 \rfloor / 2 - 1) \quad (4, \lfloor sv - 1 \rfloor / 2 - 2)\)
   \(\quad sv \mod 2 = 1 \land sv \geq 13: (1, \lfloor sv - 1 \rfloor / 2 - 1) \quad (3, \lfloor sv - 1 \rfloor / 2 - 2)\)

\(sv = 12\)

\(sv = 11\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ NB_{\text{bump on decreasing sequence}}(x, \text{VARIABLES}) \land NB_{\text{plateau}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \geq 6 \): (0, 0) (1, 1)

2. \( x > 0 \land y > 0 \Rightarrow x + 2 \times y \leq sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 16 \): (2, \( \lceil \frac{(sv - 1)}{2} \rceil - 1 \)) (4, \( \lceil \frac{(sv - 1)}{2} \rceil - 2 \))
   - \( sv \mod 2 = 1 \land sv \geq 13 \): (1, \( \lceil \frac{(sv - 1)}{2} \rceil - 1 \)) (3, \( \lceil \frac{(sv - 1)}{2} \rceil - 2 \))
\text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

\[ x > 0 \land y > 0 \Rightarrow 3 \times x + 3 \times y \leq sv - 2 - (sv - 2) \mod 3 \]

- \((sv - 2) \mod 3 = 0 \land sv \geq 8: \quad ((\lfloor \frac{sv - 3}{3} \rfloor, 1) \quad ((\lfloor \frac{sv - 3}{3} \rfloor - 1, 2)\]
- \((sv - 2) \mod 3 = 1 \land sv \geq 9: \quad ((\lfloor \frac{sv - 3}{3} \rfloor - 1, 1) \quad ((\lfloor \frac{sv - 3}{3} \rfloor - 2, 2)\]
- \((sv - 2) \mod 3 = 2 \land sv \geq 10: \quad ((\lfloor \frac{sv - 3}{3} \rfloor - 1, 1) \quad ((\lfloor \frac{sv - 3}{3} \rfloor - 2, 2)\]

\[ sv = 12 \]

\[ sv = 11 \]

\text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

\[ x > 0 \land y > 0 \Rightarrow 3 \times x + 3 \times y \leq sv - 2 - (sv - 2) \mod 3 \]

- \((sv - 2) \mod 3 = 0 \land sv \geq 8: \quad ((\lfloor \frac{sv - 3}{3} \rfloor, 1) \quad ((\lfloor \frac{sv - 3}{3} \rfloor - 1, 2)\]
- \((sv - 2) \mod 3 = 1 \land sv \geq 9: \quad ((\lfloor \frac{sv - 3}{3} \rfloor - 1, 1) \quad ((\lfloor \frac{sv - 3}{3} \rfloor - 2, 2)\]
- \((sv - 2) \mod 3 = 2 \land sv \geq 10: \quad ((\lfloor \frac{sv - 3}{3} \rfloor - 1, 1) \quad ((\lfloor \frac{sv - 3}{3} \rfloor - 2, 2)\]

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}_{\text{BUMP ON DECREASING SEQUENCE}}(x, \text{VARIABLES}) \land \\
\text{NB}_{\text{STEADY SEQUENCE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x > 0 \land y > 0 \Rightarrow 3 \ast x + y \leq sv - 3 \)
2. \( x < 1 \lor y \neq \max(0, sv - 1) - 4 \)

\( sv = 12 \) \hspace{1cm} \( sv = 11 \)

\[
\text{NB}_{\text{BUMP ON DECREASING SEQUENCE}}(x, \text{VARIABLES}) \land \\
\text{NB}_{\text{STEADY SEQUENCE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x > 0 \land y > 0 \Rightarrow 3 \ast x + 2 \ast y \leq sv - 1 \)
2. \( x > 0 \land y > 0 \Rightarrow 3 \ast x \leq y + sv - 5 \)

\( sv = 12 \) \hspace{1cm} \( sv = 11 \)
\[ \text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y = 0 \Rightarrow x = 0 \)
2. \( x > 0 \land y > 0 \Rightarrow x + 2 \cdot y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 12: (1, \lfloor sv/2 \rfloor - 1) \) \( (3, \lfloor sv/2 \rfloor - 2) \)
   - \( sv \mod 2 = 1 \land sv \geq 15: (2, \lfloor sv/2 \rfloor - 1) \) \( (4, \lfloor sv/2 \rfloor - 2) \)
3. \( x > 0 \land y > 0 \Rightarrow x \leq y - 1 \)
   - \( sv \geq 9: (1, 2) \) \( (2, 3) \)

\[ \text{sv} = 12 \]

\[ \text{nv} = 11 \]

\[ \text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_STRRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \geq 9: (1, 1) \) \( (2, 2) \)
2. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq sv - 3 \)
3. \( x > 0 \land y > 0 \Rightarrow x + 2 \cdot y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 14: (1, \lfloor sv/2 \rfloor - 1) \) \( (3, \lfloor sv/2 \rfloor - 2) \)
   - \( sv \mod 2 = 1 \land sv \geq 17: (2, \lfloor sv/2 \rfloor - 1) \) \( (4, \lfloor sv/2 \rfloor - 2) \)

\[ \text{sv} = 12 \]

\[ \text{nv} = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{BUMP}_\text{ON}_\text{DECREASING}_\text{SEQUENCE}(x,\text{VARIABLES}) \land \text{NB}_\text{SUMMIT}(y,\text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \geq 6 \): \((0, 0)\) (1, 1)
2. \( x > 0 \land y > 0 \Rightarrow x + 2 \cdot y \leq sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 16 \): \((2, \lfloor (sv - 1)/2 \rfloor - 1)\) (4, \(\lfloor (sv - 1)/2 \rfloor - 2\))
   - \( sv \mod 2 = 1 \land sv \geq 13 \): \((1, \lfloor (sv - 1)/2 \rfloor - 1)\) (3, \(\lfloor (sv - 1)/2 \rfloor - 2\))

\[ sv = 12 \]
\[ sv = 11 \]
\[ \text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \geq 6 \): (0, 0) (1, 1)
2. \( x > 0 \land y > 0 \Rightarrow x + 2 \cdot y \leq sv - 3 \)

\[ \text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y = 0 \Rightarrow x = 0 \)
2. \( x > 0 \land y > 0 \Rightarrow 3 \cdot x \leq y - 3 \)
   - \( sv \geq 9 \): (1, 6) (2, 9)
3. \( x < 1 \lor y \neq 4 \)
4. \( x < 1 \lor y \neq 5 \)
5. \( x \neq 0 \lor y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\begin{enumerate}
\item $sv > 1 \Rightarrow 3 * x + y \leq sv - 2$
\item $x > 0 \land y > 0 \Rightarrow 3 * x + y \leq sv - 3$
\end{enumerate}

\[ \text{sv} \geq 11: (1, sv - 6) \quad (2, sv - 9) \]

\[ y \neq 1 \]

\[ nb\_bump\_on\_decreasing\_sequence \]

\[ \text{sum\_width\_decreasing\_terrace} \]

\[ sv = 12 \]

\[ sv = 11 \]

\[ nb\_bump\_on\_decreasing\_sequence \]

\[ \text{sum\_width\_decreasing\_terrace} \]

\[ \text{nb\_bump\_on\_decreasing\_sequence} \]

\[ \text{sum\_width\_gorge} \]

\[ sv = 12 \]

\[ sv = 11 \]

\[ nb\_bump\_on\_decreasing\_sequence \]

\[ \text{sum\_width\_gorge} \]

\[ \text{nb\_bump\_on\_decreasing\_sequence} \]

\[ \text{nb\_bump\_on\_decreasing\_sequence} \]

\[ \text{sum\_width\_gorge} \]

\[ \text{nb\_bump\_on\_decreasing\_sequence} \]
NB_BUMP_ON_DECREASING_SEQUENCE (x, VARIABLES) ∧
SUM_WIDTH_INCREASING_SEQUENCE (y, VARIABLES) with sv = |VARIABLES|

1. $2 \times x \leq y$
   - $sv \geq 9$: (1, 2) (2, 4)
2. $x > 0 \land y > 0 \Rightarrow 4 \times x + y \leq 2 \times sv - 6$
3. $x > 0 \land y > 0 \Rightarrow x + y \leq sv - 1$
   - $sv \geq 11$: (1, sv - 2) (2, sv - 3)
4. $y \neq 2 \times x + 1$

sv = 12

sv = 11

---

NB_BUMP_ON_DECREASING_SEQUENCE (x, VARIABLES) ∧
SUM_WIDTH_INCREASING_TERRACE (y, VARIABLES) with sv = |VARIABLES|

1. $sv > 1 \Rightarrow 3 \times x + y \leq sv - 2$
2. $x > 0 \land y > 0 \Rightarrow 3 \times x + y \leq sv - 4$
   - $sv \geq 12$: (1, sv - 7) (2, sv - 10)
3. $x \neq 1 \lor y \neq (sv - 2) \times \min(1, \max(0, sv - 3)) - 4 \lor sv < 7$
4. $y \neq 1$

sv = 12

sv = 11
5. A DATABASE OF PARAMETERISED INVARIANTS

NB_BUMP_ON_DECREASING_SEQUENCE(x, VARIABLES) \land
SUM_WIDTH_INFLEXION(y, VARIABLES) with sv = |VARIABLES|

1. \(3 \cdot x \leq y\)
   - sv \geq 6: (0, 0) (1, 3)

2. \(x > 0 \land y > 0 \Rightarrow 3 \cdot x + 3 \cdot y \leq 4 \cdot sv - 10 - (4 \cdot sv - 10) \mod 3\)
   - \((sv - 1) \mod 3 = 2 \land sv \geq 6: ((sv - 3) / 3, sv - 3) \land ((sv - 3) / 3 - 1, sv - 2)\)

---

\(sv = 12\)

\(sv = 11\)

---

NB_BUMP_ON_DECREASING_SEQUENCE(x, VARIABLES) \land
SUM_WIDTH_PEAK(y, VARIABLES) with sv = |VARIABLES|

1. \(2 \cdot x \leq y\)
   - sv \geq 6: (0, 0) (1, 2)

2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - sv \geq 7: (0, sv - 2) (1, sv - 3)

3. \(x > 0 \land y > 0 \Rightarrow 4 \cdot x + y \leq 2 \cdot sv - 6\)

---

\(sv = 12\)

\(sv = 11\)
\[
\text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x \leq y \)
   - \( sv \geq 6 \): (0, 0) (1, 1)
2. \( sv > 1 \Rightarrow 2 \times x + y \leq sv - 2 \)
3. \( x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq sv - 3 \)

\[
\text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x \leq y \)
   - \( sv \geq 6 \): (0, 0) (1, 1)
2. \( sv > 1 \Rightarrow 2 \times x + y \leq sv - 2 \)
3. \( x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq sv - 3 \)

\( sv \geq 9 \): (1, \( sv - 5 \)) (2, \( sv - 7 \))
5. A DATABASE OF PARAMETERISED INVARIANTS

\[\text{NB}_{\text{BUMP\_ON\_DECREASING\_SEQUENCE}}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(sv > 1 \Rightarrow 3 \times x + y \leq sv - 2\)
2. \(x > 0 \land y > 0 \Rightarrow 3 \times x + y \leq sv - 3\)

\(sv \geq 11: (1, sv - 6) \land (2, sv - 9)\)

3. \(y \neq 1\)

\[\text{NB}_{\text{BUMP\_ON\_DECREASING\_SEQUENCE}}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(sv > 1 \Rightarrow 3 \times x + y \leq sv - 2\)
2. \(x > 0 \land y > 0 \Rightarrow 3 \times x + y \leq sv - 3\)

\(sv \geq 11: (1, sv - 6) \land (2, sv - 9)\)

3. \(y \neq 1\)
\text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x,\text{VARIABLES}) \land \text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y,\text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x > 0 \land y > 0 \Rightarrow 3 * x + y \leq sv - 1 \)
2. \( x > 0 \land y > 0 \Rightarrow 3 * x \leq sv - 4 - (sv - 4) \mod 3 \)
   - \( (sv - 1) \mod 3 = 1 \land sv \geq 5: ([sv - 3]/3, 2) \)
   - \( (sv - 1) \mod 3 = 2 \land sv \geq 6: ([sv - 3]/3 - 1, 2) \) \( (sv - 3)/3 - 1, 3) \)
3. \( y \neq 1 \)

\text{nb\_bump\_on\_decreasing\_sequence}

\text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x,\text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(y,\text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( y = 0 \Rightarrow x = 0 \)
2. \( x > 0 \land y > 0 \Rightarrow 3 * x \leq y - 3 \)
   - \( sv \geq 9: (1, 6) \) \( (2, 9) \)
3. \( x < 1 \lor y \neq 4 \)
4. \( x < 1 \lor y \neq 5 \)
5. \( x \neq 0 \lor y \neq 1 \)

\text{nb\_bump\_on\_decreasing\_sequence}
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \cdot x \leq y\)
   - \(sv \geq 9\): (1, 2) (2, 4)
2. \(x > 0 \land y > 0 \Rightarrow 4 \cdot x + y \leq 2 \cdot sv - 6\)
3. \(x > 0 \land y > 0 \Rightarrow x + y \leq sv - 1\)
   - \(sv \geq 11\): (1, \(sv - 2\)) (2, \(sv - 3\))
4. \(y \neq 2 \cdot x + 1\)

\[ \text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \cdot x \leq y\)
   - \(sv \geq 6\): (0, 0) (1, 2)
2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 7\): (0, \(sv - 2\)) (1, \(sv - 3\))
3. \(x > 0 \land y > 0 \Rightarrow 4 \cdot x + y \leq 2 \cdot sv - 6\)

\[ \text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \cdot x \leq y\)
   - \(sv \geq 6\): (0, 0) (1, 2)
2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 7\): (0, \(sv - 2\)) (1, \(sv - 3\))
3. \(x > 0 \land y > 0 \Rightarrow 4 \cdot x + y \leq 2 \cdot sv - 6\)
\[ \text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \cdot x \leq y\)
   - \(sv \geq 6: \ (0, 0) \quad (1, 2)\)
2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 7: \ (0, sv - 2) \quad (1, sv - 3)\)
3. \(x > 0 \land y > 0 \Rightarrow 4 \cdot x + y \leq 2 \cdot sv - 6\)

\[\begin{align*}
&\text{nb\_bump\_on\_decreasing\_sequence} \\
&\text{sum\_width\_valley}
\end{align*}\]

\[\begin{align*}
&\text{nb\_bump\_on\_decreasing\_sequence} \\
&\text{sum\_width\_valley}
\end{align*}\]
5. A DATABASE OF PARAMETERISED INVARIANTS

**NB DECREASING**(x, VARIABLES) ∧
**NB DECREASING_SEQUENCE**(y, VARIABLES) with \( sv = |VARIABLES| \)

1. \( x + y \leq sv \)
   - \( sv \geq 6: (sv - 2, 2) \) (sv - 3, 3)
2. \( y \leq x \)
   - \( sv \geq 4: (1, 1) \) (2, 2)
3. \( sv > 1 \Rightarrow x \leq y + sv - 2 \)
4. \( x > 0 \Rightarrow y \geq 1 \)
   - \( sv \geq 4: (sv - 2, 1) \) (sv - 3, 1)

**NB DECREASING**(x, VARIABLES) ∧ **NB DECREASING_TERRACE**(y, VARIABLES) with \( sv = |VARIABLES| \)

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 6: (sv - 2, 1) \) (sv - 3, 2)
2. \( x = 0 \Rightarrow y = 0 \)
3. \( x > 0 \land y > 0 \Rightarrow y \leq x - 1 \)
   - \( sv \geq 6: (2, 1) \) (3, 2)
\(\text{NB\_DECREASING}(x, \text{VARIABLES}) \land \\
\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\)

1. \(y \leq x\)
   - \(sv \geq 9: \ (1, 1) \quad (2, 2)\)
2. \(x > 0 \land y > 0 \Rightarrow x + 2 \ast y \leq sv - 3\)
   - \(sv \geq 9: \ (sv - 5, 1) \quad (sv - 7, 2)\)
3. \(\bigvee \begin{pmatrix} 
    x \neq \max(0, sv - 1) - 3, \\
y < 1, \\
y > \max(0, [(sv - 3)/3]) - 1, \\
0 = y \mod 2 
\end{pmatrix}\)

\(\text{NB\_DECREASING}(x, \text{VARIABLES}) \land \\
\text{NB\_GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\)

1. \(x + y \leq sv - 1\)
   - \(sv \geq 5: \ (sv - 2, 1) \quad (sv - 3, 2)\)
2. \(y \leq x\)
   - \(sv \geq 5: \ (1, 1) \quad (2, 2)\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \text{NB\_INCREASING}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 3: (sv - 2, 1) \ (sv - 3, 2) \)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \text{NB\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 4: (sv - 2, 1) \ (sv - 3, 2) \)

2. \( y \leq x + 1 \)
   - \( sv \geq 4: (0, 1) \ (1, 2) \)
\textbf{NB\_DECREASING}(x, \text{VARIABLES}) \land \textbf{NB\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \(x > 0 \land y > 0 \implies x + 2 \ast y \leq sv - 2\)
   - \(sv > 0: (sv - 4, 1) \quad (sv - 6, 2)\)

\[\begin{array}{c}
\text{\textbf{sv} = 12} \\
\text{\textbf{sv} = 11}
\end{array}\]

\textbf{NB\_DECREASING}(x, \text{VARIABLES}) \land \textbf{NB\_INPLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \(2 \ast x + y \leq 2 \ast sv - 2\)
   - \(sv > 0: (sv - 2, 2) \quad (sv - 3, 4)\)
2. \(y \leq 2 \ast x\)
   - \(sv > 0: (1, 2) \quad (2, 4)\)

\[\begin{array}{c}
\text{\textbf{sv} = 12} \\
\text{\textbf{sv} = 11}
\end{array}\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \text{NB\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x + y \leq sv - 1 \]
   - \[ sv \geq 5 : (sv - 2, 1) \quad (sv - 3, 2) \]
2. \[ y \leq x \]
   - \[ sv \geq 5 : (1, 1) \quad (2, 2) \]

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \text{NB\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x + y \leq sv - 1 \]
   - \[ sv \geq 5 : (sv - 2, 1) \quad (sv - 3, 2) \]
2. \[ y \leq x \]
   - \[ sv \geq 5 : (1, 1) \quad (2, 2) \]

\[ sv = 12 \]

\[ sv = 11 \]
\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \text{NB\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 5 \): \((sv - 2, 1)\) \( (sv - 3, 2)\)
2. \( y \leq x \)
   - \( sv \geq 5 \): \((1, 1)\) \((2, 2)\)

\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \text{NB\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + 2 \cdot y \leq sv - 1 \)
   - \( sv \geq 7 \): \((sv - 3, 1)\) \((sv - 5, 2)\)
2. \( y \leq x \)
   - \( sv \geq 7 \): \((1, 1)\) \((2, 2)\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}_\text{DECREASING}(x, \text{VARIABLES}) \land \\
\text{NB}_\text{PROPER\_PLATEAU}(y, \text{VARIABLES}) \quad \text{with } sv = |\text{VARIABLES}|
\]

1. \( x + 2 \cdot y \leq sv - 1 \)
   - \( sv \geq 7: (sv - 3, 1) \quad (sv - 5, 2) \)
2. \( y \leq x \)
   - \( sv \geq 7: (1, 1) \quad (2, 2) \)

\[
\text{NB}_\text{DECREASING}(x, \text{VARIABLES}) \land \\
\text{NB}_\text{STEADY}(y, \text{VARIABLES}) \quad \text{with } sv = |\text{VARIABLES}|
\]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 3: (sv - 2, 1) \quad (sv - 3, 2) \)
\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 4: (sv - 2, 1) \quad (sv - 3, 2) \)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv \)
   - \( sv \geq 0: (sv - 2, 2) \quad (sv - 3, 3) \)
2. \( y \leq x \)
   - \( sv \geq 4: (1, 1) \quad (2, 2) \)
3. \( sv > 1 \Rightarrow x \leq y + sv - 2 \)
4. \( x > 0 \Rightarrow y \geq 1 \)
   - \( sv \geq 4: (sv - 2, 1) \quad (sv - 3, 1) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\textbf{NB}_\text{DECREASING}(x, \text{VARIABLES}) \land \\
\textbf{NB}_\text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv - 1 \)
2. \( sv \geq 4: (sv - 2, 1) \quad (sv - 3, 2) \)

\textbf{NB}_\text{DECREASING}(x, \text{VARIABLES}) \land \\
\textbf{NB}_\text{SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv - 1 \)
2. \( sv \geq 5: (sv - 2, 1) \quad (sv - 3, 2) \)
3. \( y \leq x \)
4. \( sv \geq 5: (1, 1) \quad (2, 2) \)
\textbf{NB\_DECREASING}(x, \text{VARIABLES}) \land \textbf{NB\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 5: (sv - 2, 1) \quad (sv - 3, 2) \)
2. \( y \leq x \)
   - \( sv \geq 5: (1, 1) \quad (2, 2) \)

\textbf{NB\_DECREASING}(x, \text{VARIABLES}) \land \textbf{NB\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 7: (sv - 2, 1) \quad (sv - 3) \)
2. \( y \leq x \)
   - \( sv \geq 7: (1, 1) \quad (2, 2) \)
   - \( y \neq \max(0, \lfloor (sv - 1)/3 \rfloor) - 0, \quad x < \max(0, \lfloor (sv - 1)/3 \rfloor) + 1, \quad x > 1 \ast \max(0, sv - 1) - \max(0, \lfloor (sv - 1)/3 \rfloor) - 1, \quad 1 \neq sv \mod 3 \)

\textbf{NB\_DECREASING}(x, \text{VARIABLES}) \land \textbf{NB\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 7: (sv - 2, 1) \quad (sv - 3) \)
2. \( y \leq x \)
   - \( sv \geq 7: (1, 1) \quad (2, 2) \)
   - \( y \neq \max(0, \lfloor (sv - 1)/3 \rfloor) - 0, \quad x < \max(0, \lfloor (sv - 1)/3 \rfloor) + 1, \quad x > 1 \ast \max(0, sv - 1) - \max(0, \lfloor (sv - 1)/3 \rfloor) - 1, \quad 1 \neq sv \mod 3 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x = 0 \Rightarrow y = 0 \)
2. \( y = 0 \Rightarrow x = 0 \)
3. \( sv > 1 \Rightarrow 2 \times x \leq y + sv - 2 \)
4. \( sv > 1 \Rightarrow y \leq 2 \times x + sv - 2 \)
5. \( x > 0 \land y > 0 \Rightarrow x \leq y - 1 \)
6. \( sv \geq 4 : \ (sv - 2, sv - 1) \ (sv - 3, sv - 2) \)
7. \( y > 0 \Rightarrow x \geq 1 \)
8. \( x \neq 1 \lor y < 3 \)
\text{NB\_DECREASING}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \(2 \times x + y \leq 2 \times sv - 2\)
2. \(sv \geq 0: \ (sv - 2, 2) \quad (sv - 3, 4)\)
3. \(x = 0 \Rightarrow y = 0\)
4. \(x > 0 \land y > 0 \Rightarrow x \geq 2\)
5. \(sv \geq 5: \ (2, sv - 2) \quad (2, sv - 3)\)
6. \(x \neq 1 \lor y < 1\)
7. \(y \neq 1\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB \textunderscore DECREASING}(x, \text{VARIABLES}) \land \\
\text{SUM\textunderscore WIDTH\textunderscore GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 2 \)
   - \( sv \geq 3): (sv - 2, sv - 3) \land (sv - 2, sv - 3) \)

2. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 5): (1, sv - 2) \land (1, sv - 3) \)

\[ \text{NB \textunderscore DECREASING}(x, \text{VARIABLES}) \land \\
\text{SUM\textunderscore WIDTH\textunderscore INCREASING\textunderscore SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2 \times x + y \leq 2 \times sv - 2 \)
   - \( sv \geq 4): (sv - 2, 2) \land (sv - 3, 4) \)

2. \( y \neq 1 \)
\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow x + y \leq sv - 2 \)
2. \( sv \geq 6: (sv - 4, 2) \text{ or } (1, sv - 3) \)
3. \( y > 0 \Rightarrow x \leq sv - 4 \)
4. \( y \neq 1 \)

\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 2 \)
2. \( sv \geq 6: (sv - 2, sv - 2) \text{ or } (sv - 2, sv - 3) \)
3. \( y > 0 \Rightarrow x \geq 1 \)
4. \( sv \geq 3: (1, sv - 2) \text{ or } (1, sv - 3) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}_\text{DECREASING}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{itemize}
\item 1 \( x > 0 \land y > 0 \Rightarrow x \leq sv - 2 \)
\item \( sv \geq 4: (sv - 2, 1) (sv - 2, 2) \)
\item 2 \( y > 0 \Rightarrow x \geq 1 \)
\item \( sv \geq 3: (1, sv - 2) (1, sv - 3) \)
\end{itemize}

\[
\text{NB}_\text{DECREASING}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{itemize}
\item 1 \( x + y \leq sv - 1 \)
\item \( sv \geq 4: (sv - 2, 1) (sv - 3, 2) \)
\item 2 \( y > 0 \Rightarrow x \geq 1 \)
\item \( sv \geq 3: (1, sv - 2) (1, sv - 3) \)
\end{itemize}
\textbf{NB\_DECREASING}(x, \text{VARIABLES}) \land \textbf{SUM\_WIDTH\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 4: (sv - 2, 1) \quad (sv - 3, 2) \)
2. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 3: (1, sv - 2) \quad (1, sv - 3) \)

\textbf{NB\_DECREASING}(x, \text{VARIABLES}) \land \textbf{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 5: (sv - 3, 2) \quad (sv - 4, 3) \)
2. \( y > 0 \Rightarrow x \leq sv - 3 \)
3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 5: (1, sv - 2) \quad (1, sv - 3) \)
4. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}_\text{DECREASING}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{PROPER}_\text{PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x + y \leq sv - 1\)
   - \(sv \geq 5\): (\(sv - 3, 2\), (\(sv - 4, 3\))
2. \(y > 0 \Rightarrow x \leq sv - 3\)
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 5\): (1, \(sv - 2\), (1, \(sv - 3\))
4. \(y \neq 1\)

\[
\text{nb}_\text{decreasing}
\]

\[
\text{sum}_\text{width}_\text{proper}_\text{plateau}
\]

\[
sv = 12
\]

\[
sv = 11
\]

\[
\text{NB}_\text{DECREASING}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{STEADY}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2x + y \leq 2sv - 2\)
   - \(sv \geq 4\): (\(sv - 2, 2\), (\(sv - 3, 4\))
2. \(y \neq 1\)

\[
\text{nb}_\text{decreasing}
\]

\[
\text{sum}_\text{width}_\text{steady}_\text{sequence}
\]

\[
sv = 12
\]

\[
sv = 11
\]
\text{DECREASING}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( y \leq 2 \times x \)
2. \( x = 0 \Rightarrow y = 0 \)
3. \( y = 0 \Rightarrow x = 0 \)
4. \( sv > 1 \Rightarrow 2 \times x \leq y + sv - 2 \)
5. \( x > 0 \land y > 0 \Rightarrow x \leq y - 1 \)
6. \( sv \geq 4: (sv - 2, sv - 1), (sv - 3, sv - 2) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{itemize}
\item \texttt{NB\_DECREASING}(x, VARIABLES) \land \texttt{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, VARIABLES) with \( sv = |\text{VARIABLES}| \)
\begin{enumerate}
\item \( 2 \times x + y \leq 2 \times sv - 2 \)
\item \( sv \geq 4: \ (sv - 2, 2) \ (sv - 3, 4) \)
\item \( y \neq 1 \)
\end{enumerate}
\end{itemize}

\begin{itemize}
\item \texttt{NB\_DECREASING}(x, VARIABLES) \land \texttt{SUM\_WIDTH\_SUMMIT}(y, VARIABLES) with \( sv = |\text{VARIABLES}| \)
\begin{enumerate}
\item \( x > 0 \land y > 0 \Rightarrow x \leq sv - 2 \)
\item \( sv \geq 4: \ (sv - 2, 1) \ (sv - 2, 2) \)
\item \( y > 0 \Rightarrow x \geq 1 \)
\item \( sv \geq 3: \ (1, sv - 2) \ (1, sv - 3) \)
\end{enumerate}
\end{itemize}
\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 2 \)
   - \( sv \geq 3: (sv - 2, sv - 2) \quad (sv - 2, sv - 3) \)
2. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 4: (1, sv - 2) \quad (1, sv - 3) \)

\[ \text{NB\_DECREASING}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2 \cdot x + y \leq 2 \cdot sv - 2 \)
   - \( sv \geq 0: (sv - 2, 2) \quad (sv - 3, 4) \)
2. \( y \leq 2 \cdot x \)
   - \( sv \geq 0: (1, 2) \quad (2, 4) \)
3. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB} \_ \text{DECREASING \_} \text{SEQUENCE}(x, \text{VARIABLES}) \land \text{NB} \_ \text{DECREASING \_} \text{TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\begin{enumerate}
\item \(2 \times x + 2 \times y \leq sv - sv \mod 2\)
  \begin{itemize}
  \item \(sv \mod 2 = 0 \land sv \geq 4: ([sv/2], 0) \quad ([sv/2] - 1, 1)\)
  \item \(sv \mod 2 = 1 \land sv \geq 5: ([sv/2], 0) \quad ([sv/2] - 1, 1)\)
  \end{itemize}
\item \(y > 0 \Rightarrow x \geq 1\)
  \begin{itemize}
  \item \(sv \geq 4: (1, [(sv - 2)/2]) \quad (1, [(sv - 2)/2] - 1)\)
  \end{itemize}
\end{enumerate}

\[ \text{NB} \_ \text{DECREASING \_} \text{SEQUENCE}(x, \text{VARIABLES}) \land \text{NB} \_ \text{DIP \_} \text{ON \_} \text{INCREASING \_} \text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\begin{enumerate}
\item \(y \leq x\)
  \begin{itemize}
  \item \(sv \geq 9: (1, 1) \quad (2, 2)\)
  \end{itemize}
\item \(x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq sv - 1\)
  \begin{itemize}
  \item \(sv \mod 2 = 0 \land sv \geq 14: ([sv/2] - 1, 1) \quad ([sv/2] - 2, 3)\)
  \item \(sv \mod 2 = 1 \land sv \geq 17: ([sv/2] - 1, 2) \quad ([sv/2] - 2, 4)\)
  \end{itemize}
\item \(x > 0 \land y > 0 \Rightarrow x + 2 \times y \leq sv - 3\)
\end{enumerate}
\textsc{nb_decreasing_sequence}(x, \text{VARIABLES}) \land 
\textsc{nb_gorge}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( 3 \cdot x \leq y + sv + 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 8: \) \( \left\lceil \frac{sv}{2} \right\rceil, \left\lfloor \frac{(sv - 1)/2}{2} \right\rfloor \)
   - \( sv \mod 2 = 1 \land sv \geq 11: \) \( \left\lceil \frac{sv}{2} \right\rceil, \left\lfloor \frac{(sv - 1)/2}{2} - 2 \right\rfloor \)

2. \( y \leq x \)
   - \( sv \geq 5: \) \((1, 1)\) \((2, 2)\)

\textsc{nb_decreasing_sequence}(x, \text{VARIABLES}) \land 
\textsc{nb_increasing}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 4: \) \((1, sv - 2)\) \((2, sv - 3)\)

2. \( x \leq y + 1 \)
   - \( sv \geq 4: \) \((1, 0)\) \((2, 1)\)

\textsc{nb_decreasing_sequence}(x, \text{VARIABLES}) \land 
\textsc{nb_increasing}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 4: \) \((1, sv - 2)\) \((2, sv - 3)\)

2. \( x \leq y + 1 \)
   - \( sv \geq 4: \) \((1, 0)\) \((2, 1)\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\( \text{NB}_\text{DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB}_\text{INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x + y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 2: (\lfloor sv/2 \rfloor, \lfloor sv/2 \rfloor - 1) \quad (\lfloor sv/2 \rfloor - 1, \lfloor sv/2 \rfloor) \)
2. \( x \leq y + 1 \)
   - \( sv \geq 4: (1, 0) \quad (2, 1) \)
3. \( y \leq x + 1 \)
   - \( sv \geq 4: (0, 1) \quad (1, 2) \)

\( sv = 12 \)

\( sv = 11 \)

\( \text{NB}_\text{DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB}_\text{INCREASING\_TERRACE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2x + 2y \leq sv - sv \mod 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 6: (\lfloor sv/2 \rfloor, 0) \quad (\lfloor sv/2 \rfloor - 1, 1) \)
   - \( sv \mod 2 = 1 \land sv \geq 5: (\lfloor sv/2 \rfloor, 0) \quad (\lfloor sv/2 \rfloor - 1, 1) \)
2. \( x > 0 \land y > 0 \Rightarrow x + 2y \leq sv - 2 \)

\( sv = 12 \)

\( sv = 11 \)
\[ \text{NB}_{\text{DECREASING\_SEQUENCE}}(x, \text{VARIABLES}) \land \text{NB}_{\text{INFLEXION}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \cdot x \leq y + 2 \]
   - \( sv \geq 4 \): (1, 0) (2, 2)
2. \[ y \leq 2 \cdot x \]
   - \( sv \geq 6 \): (1, 2) (2, 4)

\[ \text{NB}_{\text{DECREASING\_SEQUENCE}}(x, \text{VARIABLES}) \land \text{NB}_{\text{PEAK}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x \leq y + 1 \]
   - \( sv \geq 4 \): (1, 0) (2, 1)
2. \[ y \leq x \]
   - \( sv \geq 5 \): (1, 1) (2, 2)
5. A DATABASE OF PARAMETRIZED INVARIANTS

\[ \text{NB}_\text{DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land 
\text{NB}_\text{PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y + 1 \)
   - \( sv \geq 4: (1, 0) \text{ (2, 1)} \)
2. \( y \leq x \)
   - \( sv \geq 5: (1, 1) \text{ (2, 2)} \)

\[ \text{NB}_\text{DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land 
\text{NB}_\text{PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y + 1 \)
   - \( sv \geq 4: (1, 0) \text{ (2, 1)} \)
2. \( y \leq x \)
   - \( sv \geq 5: (1, 1) \text{ (2, 2)} \)
\[ \text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

### Case 1: \( sv \mod 2 = 0 \land sv \geq 8 \)
- \( (\lfloor sv/2 \rfloor, 0) \)  \( (\lfloor sv/2 \rfloor - 1, 2) \)

### Case 2: \( sv \mod 2 = 1 \land sv \geq 11 \)
- \( (\lfloor sv/2 \rfloor, 1) \)  \( (\lfloor sv/2 \rfloor - 1, 3) \)

\[ y \leq x \]
- \( sv \geq 7: (1, 1) \)  \( (2, 2) \)

---

\[ \text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

### Case 1: \( sv \mod 2 = 0 \land sv \geq 8 \)
- \( (\lfloor sv/2 \rfloor, 0) \)  \( (\lfloor sv/2 \rfloor - 1, 2) \)

### Case 2: \( sv \mod 2 = 1 \land sv \geq 11 \)
- \( (\lfloor sv/2 \rfloor, 1) \)  \( (\lfloor sv/2 \rfloor - 1, 3) \)

\[ y \leq x \]
- \( sv \geq 7: (1, 1) \)  \( (2, 2) \)
5. A DATABASE OF PARAMETERISED INVARIENTS

\( \text{NB}_\text{DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB}_\text{STEADY}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \cdot x + y \leq sv \)
   - \( sv \geq 4: (1, sv - 2) \quad (2, sv - 4) \)

\( sv = 12 \)

\( sv = 11 \)

\( \text{NB}_\text{DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB}_\text{STEADY\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \cdot x + y \leq sv \)
   - \( sv \mod 2 = 0 \land sv \geq 4: (\lfloor sv/2 \rfloor, 0) \quad (\lfloor sv/2 \rfloor - 2, 2) \)
   - \( sv \mod 2 = 1 \land sv \geq 7: (\lfloor sv/2 \rfloor, 1) \quad (\lfloor sv/2 \rfloor - 1, 3) \)

\( sv = 12 \)

\( sv = 11 \)
\( \text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x \leq y \)
   - \( sv \geq 4: (1, 1) \quad (2, 2) \)
2. \( sv > 1 \Rightarrow 2 \cdot y \leq 2 \cdot x + sv - 2 - (sv - 2) \mod 2 \)
3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 4: (1, \lfloor sv/2 \rfloor) \quad (1, \lfloor sv/2 \rfloor - 1) \)

\( sv = 12 \)

\( sv = 11 \)

\( \text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x + y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 2: (\lfloor sv/2 \rfloor, \lfloor sv/2 \rfloor - 1) \quad (\lfloor sv/2 \rfloor - 1, \lfloor sv/2 \rfloor) \)
2. \( x \leq y + 1 \)
   - \( sv \geq 4: (1, 0) \quad (2, 1) \)

\( sv = 12 \)

\( sv = 11 \)
A DATABASE OF PARAMETERISED INVARIANTS

\[ 3 \cdot x \leq y + sv + 1 \]

\[ sv \mod 2 = 0 \land sv \geq 8: \left( \lfloor sv/2 \rfloor, \lfloor (sv - 1)/2 \rfloor \right) \]
\[ sv \mod 2 = 1 \land sv \geq 11: \left( \lfloor sv/2 \rfloor, \lfloor (sv - 1)/2 \rfloor - 3 \right) \]

\[ y \leq x \]
\[ sv \geq 5: (1, 1) (2, 2) \]

\[ 3 \cdot x \leq y + sv + 1 \]

\[ sv \mod 2 = 0 \land sv \geq 8: \left( \lfloor sv/2 \rfloor, \lfloor (sv - 1)/2 \rfloor \right) \]
\[ sv \mod 2 = 1 \land sv \geq 11: \left( \lfloor sv/2 \rfloor, \lfloor (sv - 1)/2 \rfloor - 3 \right) \]

\[ y \leq x \]
\[ sv \geq 5: (1, 1) (2, 2) \]
\[ \text{NB}_{\text{DECREASING\_SEQUENCE}}(x, \text{VARIABLES}) \land \text{NB}_{\text{ZIGZAG}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x + y \leq sv + 1\)
   - \(sv \mod 2 = 0 \land sv \geq 10: (\lfloor sv/2 \rfloor, 1) (\lfloor sv/2 \rfloor - 1, 3)\)
   - \(sv \mod 2 = 1 \land sv \geq 13: (\lfloor sv/2 \rfloor, 2) (\lfloor sv/2 \rfloor - 1, 4)\)

2. \(y \leq x\)
   - \(sv \geq 7: (1, 1) (2, 2)\)

3. \(y = 0 \Rightarrow 3 \times x \leq sv + 1 - (sv + 1) \mod 3\)

\[ \text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x \leq y\)
   - \(sv \geq 4: (1, 2) (2, 4)\)

2. \(sv > 1 \Rightarrow y \leq 2 \times x + sv - 2\)

3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 4: (1, sv) (1, sv - 1)\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \cdot x + y \leq sv\)
   - \(sv \geq 6\): (1, \(sv - 2\)) (2, \(sv - 4\))
2. \(y > 0 \Rightarrow 2 \cdot x \leq sv - 2 - (sv - 2) \mod 2\)
   - \(sv \mod 2 = 1 \land sv \geq 5\): (\(\lfloor sv/2 \rfloor - 1, 2\)) (\(\lfloor sv/2 \rfloor - 1, 3\))
3. \(y > 0 \Rightarrow x \geq 1\)
4. \(y \neq 1\)

\[ \text{SV} = 12 \]

\[ \text{SV} = 11 \]

\[ \text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(3 \cdot x \leq y + sv + 1\)
2. \(x + y \leq sv - 1\)
   - \(sv \geq 4\): (1, \(sv - 2\)) (2, \(sv - 3\))
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 3\): (1, \(sv - 2\)) (1, \(sv - 3\))

\[ \text{SV} = 12 \]

\[ \text{SV} = 11 \]
\[ \text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \cdot x + y \leq 2 \cdot sv - 2 \]
   \[ sv \text{ mod } 2 = 0 \land sv \geq 2 \Rightarrow ([sv/2], sv - 2) \land ([sv/2] - 1, sv) \]
2. \[ 2 \cdot x \leq y + 2 \]
   \[ sv \geq 4: (1, 0) \land (2, 2) \]
3. \[ y \neq 1 \]

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \cdot x + y \leq sv \]
   \[ sv \geq 8: (2, sv - 4) \land (3, sv - 6) \]
2. \[ x > 0 \land y > 0 \Rightarrow x + y \leq sv - 2 \]
   \[ sv \geq 6: (1, sv - 3) \land (2, sv - 4) \]
3. \[ y > 0 \Rightarrow 2 \cdot x \leq sv - 2 - (sv - 2) \text{ mod } 2 \]
   \[ sv \text{ mod } 2 = 1 \land sv \geq 7 \Rightarrow ([sv/2] - 1, 2) \land ([sv/2] - 1, 3) \]
4. \[ y \neq 1 \]

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

**NB\_DECREASING\_SEQUENCE** (x, VARIABLES) ∧ **SUM\_WIDTH\_INFLEXION** (y, VARIABLES) with \( sv = |\text{VARIABLES}| \)

1. \( 2x \leq y + 2 \)
   - \( sv \geq 4: (1, 0) \ (2, 2) \)
2. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 3: (1, sv - 2) \ (1, sv - 3) \)

**NB\_DECREASING\_SEQUENCE** (x, VARIABLES) ∧ **SUM\_WIDTH\_PEAK** (y, VARIABLES) with \( sv = |\text{VARIABLES}| \)

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 4: (1, sv - 2) \ (2, sv - 3) \)
2. \( x \leq y + 1 \)
   - \( sv \geq 4: (1, 0) \ (2, 1) \)
3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 3: (1, sv - 2) \ (1, sv - 3) \)
NB_DECREASING_SEQUENCE(x, VARIABLES) \land SUM_WIDTH_PLAIN(y, VARIABLES) with sv = |VARIABLES|

1. $x + y \leq sv - 1$
   - sv $\geq 4$: (1, sv - 2) (2, sv - 3)

2. $x \leq y + 1$
   - sv $\geq 4$: (1, 0) (2, 1)

3. $y > 0 \Rightarrow x \geq 1$
   - sv $\geq 3$: (1, sv - 2) (1, sv - 3)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(4 \ast x + y \leq 2 \ast sv\)
   - \(sv \text{ mod } 2 = 0 \land sv \geq 8\): \(([\lfloor sv/2 \rfloor], 0)\) \( ([\lfloor sv/2 \rfloor] - 1, 4) \)
   - \(sv \text{ mod } 2 = 1 \land sv \geq 11\): \(([\lfloor sv/2 \rfloor], 2)\) \( ([\lfloor sv/2 \rfloor] - 1, 6)\)
2. \(x + y \leq sv - 1\)
3. \(y \geq 5\): \((1, sv - 2)\) \( (2, sv - 3)\)
4. \(y > 0 \Rightarrow x \geq 1\)
5. \(y \geq 5\): \((1, sv - 2)\) \( (1, sv - 3)\)
6. \(y \neq 1\)

\[ s v = 12 \]

\[ s v = 11 \]
3279

\[ \text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(4 \times x + y \leq 2 \times sv\)

- \(sv \mod 2 = 0 \land sv \geq 8:\) \(([sv/2], 0) \land ([sv/2] - 1, 4)\)
- \(sv \mod 2 = 1 \land av \geq 11:\) \(([sv/2], 2) \land ([sv/2] - 1, 6)\)

2. \(x + y \leq sv - 1\)

- \(sv \geq 5:\) \((1, sv - 2) \land (2, sv - 3)\)

3. \(y > 0 \Rightarrow x \geq 1\)

- \(sv \geq 5:\) \((1, sv - 2) \land (1, sv - 3)\)

4. \(y \neq 1\)

\[ \text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(4 \times x + y \leq 2 \times sv\)

- \(sv \mod 2 = 0 \land av \geq 4:\) \(([sv/2], 0) \land ([sv/2] - 1, 4)\)
- \(sv \mod 2 = 1 \land av \geq 7:\) \(([sv/2], 2) \land ([sv/2] - 1, 6)\)

2. \(y \neq 1\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( 2 \times x \leq y \)
   - \( sv \geq 4: \ (1, 2) \ (2, 4) \)
2. \( sv > 1 \Rightarrow y \leq 2 \times x + sv - 2 \)
3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 4: \ (1, sv - 1) \ (1, sv - 2) \)

\[
\text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( 2 \times x + y \leq 2 \times sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 2: \ ([sv/2], sv - 2) \ ([sv/2] - 1, sv) \)
2. \( 2 \times x \leq y + 2 \)
   - \( sv \geq 4: \ (1, 0) \ (2, 2) \)
3. \( y \neq 1 \)
\(\text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES})\) with \(sv = |\text{VARIABLES}|\)

1. \(3 \times x \leq y + sv + 1\)
2. \(x + y \leq sv - 1\)
3. \(sv \geq 4: (1, sv - 2) \quad (2, sv - 3)\)
   - \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 3: (1, sv - 2) \quad (1, sv - 3)\)

\(sv = 12\)

\(sv = 11\)

\(\text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES})\) with \(sv = |\text{VARIABLES}|\)

1. \(x + y \leq sv - 1\)
2. \(x \leq y + 1\)
3. \(sv \geq 4: (1, sv - 2) \quad (2, sv - 3)\)
4. \(y > 0 \Rightarrow x \geq 1\)
5. \(sv \geq 3: (1, sv - 2) \quad (1, sv - 3)\)

\(sv = 12\)

\(sv = 11\)
A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq 2 \times x \)
   - \( sv \geq 6 \): (1, 2) (2, 4)
2. \( y = 0 \Rightarrow 3 \times x \leq sv + 1 - (sv + 1) \mod 3 \)
3. \( x > 0 \land y > 0 \Rightarrow 3 \times x \leq y + sv \)
4. \( y > 0 \Rightarrow 6 \times x \leq y + 2 \times sv + 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 8 \): ([sv/2], sv - 2) ([sv/2] - 1, sv - 8)
   - \( sv \mod 2 = 1 \land sv \geq 11 \): ([sv/2], sv - 5) ([sv/2] - 1, sv - 11)
5. \( y \neq 1 \)
**NB_DECREASING_TERRACE** (x, VARIABLES) \( \land \)
**NB_DIP_ON_INCREASING_SEQUENCE** (y, VARIABLES) with \( sv = |\text{VARIABLES}| \)

1. \( sv > 1 \Rightarrow 2 \cdot x + 3 \cdot y \leq sv - 2 - (sv - 2) \mod 2 \)
2. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x + 3 \cdot y \leq sv - 4 \)

\( sv \mod 2 = 1 \land sv \geq 13: \)

\[
\begin{align*}
x & \neq \max(0, \lfloor (sv - 2) / 2 \rfloor - 1, 1 - (sv - 2) / 2 - 2) \\
y & < 1, \\
y & > \max(0, \lfloor (sv - 3) / 3 \rfloor) - 1, \\
1 & = sv \mod 2, \\
0 & = y \mod 2
\end{align*}
\]

\( sv = 12 \)

\( sv = 11 \)

**NB_DECREASING_TERRACE** (x, VARIABLES) \( \land \)
**NB_GORGE** (y, VARIABLES) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \cdot x + 2 \cdot y \leq sv - 1 - (sv - 1) \mod 2 \)
2. \( (sv - 1) \mod 2 = 0 \land sv \geq 5: \)
3. \( (sv - 1) \mod 2 = 1 \land sv \geq 4: \)

\( sv = 12 \)

\( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[\text{NB\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{NB\_INCREASING}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq sv - 2\)

- \(sv \geq 6: (1, sv - 4) \quad (2, sv - 6)\)

\[\begin{align*}
&\text{sv = 12} \\
&\text{sv = 11}
\end{align*}\]

\[\text{NB\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{NB\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(2 \cdot x + 2 \cdot y \leq sv - sv \mod 2\)

- \(sv \mod 2 = 0 \land sv \geq 6: (\lfloor (sv - 2)/2 \rfloor - 1, 2) \quad (\lfloor (sv - 2)/2 \rfloor - 2, 3)\)

- \(sv \mod 2 = 1 \land sv \geq 5: (\lfloor (sv - 2)/2 \rfloor, 1) \quad (\lfloor (sv - 2)/2 \rfloor - 1, 2)\)

2. \(x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq sv - 2\)

\[\begin{align*}
&\text{sv = 12} \\
&\text{sv = 11}
\end{align*}\]
\(\text{NB}_{\text{DECREASING_TERRACE}}(x, \text{VARIABLES}) \land \text{NB}_{\text{INCREASING_TERRACE}}(y, \text{VARIABLES})\) with \(sv = |\text{VARIABLES}|\)

\[
\begin{align*}
&1.
sv > 1 \Rightarrow 2 \times x + 2 \times y \leq sv - 2 - (sv - 2) \mod 2 \\
&2.
sv \mod 2 = 0 \land sv \geq 4: (((sv - 2)/2), 0) \quad (0, ((sv - 2)/2)) \\
&3.
sv \mod 2 = 1 \land sv \geq 5: (((sv - 2)/2), 0) \quad (0, ((sv - 2)/2) - 1, 1) \\
\end{align*}
\[
\begin{align*}
&4.
x > 0 \land y > 0 \Rightarrow 2 \times x + 2 \times y \leq sv - 3 - (sv - 3) \mod 2 \\
&5.
(sv - 1) \mod 2 = 0 \land sv \geq 7: (((sv - 2)/2) - 1, 1) \quad (((sv - 2)/2) - 2, 2) \\
&6.
(sv - 1) \mod 2 = 1 \land sv \geq 8: (((sv - 2)/2) - 1, 1) \quad (((sv - 2)/2) - 3, 3) \\
\end{align*}
\]

\[
\begin{align*}
&7.
x ^{\circ} \geq x + y \\
&8.
\begin{cases}
 x \neq \max(0, \lfloor(sv - 2)/2\rfloor) - 3, \\
 y < 3, \\
 y > \max(0, \lfloor(sv - 2)/2\rfloor) - 1, \\
 1 = sv \mod 2, \\
 0 = y \mod 2 \\
 y \neq \max(0, \lfloor(sv - 2)/2\rfloor) - 3, \\
 x < 3, \\
 x > \max(0, \lfloor(sv - 2)/2\rfloor) - 1, \\
 1 = sv \mod 2, \\
 0 = x \mod 2 \\
 y \neq \max(0, \lfloor(sv - 2)/2\rfloor) - 2, \\
 y < 2, \\
 y > \max(0, \lfloor(sv - 2)/2\rfloor) - 1, \\
 1 = sv \mod 2, \\
 1 = y \mod 2 \\
 y \neq \max(0, \lfloor(sv - 2)/2\rfloor) - 2, \\
 x < 2, \\
 x > \max(0, \lfloor(sv - 2)/2\rfloor) - 1, \\
 1 = sv \mod 2, \\
 1 = x \mod 2 \\
 x \neq \max(0, \lfloor(sv - 2)/2\rfloor) - 1, \\
 y < 1, \\
 y > \max(0, \lfloor(sv - 2)/2\rfloor) - 1, \\
 1 = sv \mod 2, \\
 0 = y \mod 2 \\
 y \neq \max(0, \lfloor(sv - 2)/2\rfloor) - 1, \\
 x < 1, \\
 x > \max(0, \lfloor(sv - 2)/2\rfloor) - 1, \\
 1 = sv \mod 2, \\
 0 = x \mod 2 \\
\end{cases}
\end{align*}
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{NB\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( sv > 1 \implies 2 \times x + y \leq sv - 2 \)
2. \( sv \geq 4: (0, sv - 2) \quad (1, sv - 4) \)
\( \text{NB}_\text{DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{NB}_\text{PEAK}(y, \text{VARIABLES}) \) with \( \text{sv} = |\text{VARIABLES}| \)

\[ 2 \times x + 2 \times y \leq \text{sv} - 1 - (\text{sv} - 1) \mod 2 \]

- \( (\text{sv} - 1) \mod 2 = 0 \land \text{sv} \geq 5: \ (\lfloor (\text{sv} - 2)/2 \rfloor, 1) \)
- \( (\text{sv} - 1) \mod 2 = 1 \land \text{sv} \geq 4: \ (\lfloor (\text{sv} - 2)/2 \rfloor, 0) \)

\[ \text{sv} \geq 5: \ (\lfloor (\text{sv} - 2)/2 \rfloor, 1) \]

\( \text{sv} \geq 4: \ (\lfloor (\text{sv} - 2)/2 \rfloor, 0) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ NB_{DECREASING\_TERRACE}(x, VARIABLES) \land \]
\[ NB_{PLATEAU}(y, VARIABLES) \text{ with } sv = |VARIABLES| \]

1. \[ 2 * x + 2 * y \leq sv - 1 - (sv - 1) \text{ mod } 2 \]
   - \((sv - 1) \text{ mod } 2 = 0 \land sv \geq 5: ((sv - 2)/2, 1)\]
   - \((sv - 1) \text{ mod } 2 = 1 \land sv \geq 4: ((sv - 2)/2, 0)\]

\[ sv = 12 \]

\[ (sv - 1) \text{ mod } 2 = 0 \land sv \geq 5: ((sv - 2)/2, 1) \]
\[ (sv - 1) \text{ mod } 2 = 1 \land sv \geq 4: ((sv - 2)/2, 0) \]

\[ sv = 11 \]

\[ sv \mod 2 = 0 \land sv \geq 10: ((sv - 2)/2 - 1, 1) \]
\[ sv \mod 2 = 1 \land sv \geq 9: ((sv - 2)/2 - 1, 1) \]

\[ sv = 12 \]

\[ sv = 11 \]
\( \text{NB} \_\text{DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{NB} \_\text{PROPER\_PLATEAU}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x + 3 \times y \leq sv - 1 \)
2. \( sv \mod 2 = 0 \land sv \geq 10: (\lfloor (sv - 2)/2 \rfloor - 1, 1) \) \( (\lfloor (sv - 2)/2 \rfloor - 4, 3) \)

\( sv = 12 \)

\( x \leq y \)

1. \( sv \geq 6: (1, 1) \) \( (2, 2) \)
2. \( x > 0 \land y > 0 \Rightarrow x + y \leq sv - 2 \)

\( sv = 11 \)

\( sv \geq 6: (1, sv - 3) \) \( (2, sv - 4) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB} \text{\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{NB} \text{\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \geq 6: (1, 1) \ (2, 2) \)
2. \( x > 0 \land y > 0 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 4: \left( \left\lfloor \frac{sv - 2}{2} \right\rfloor, \left\lfloor \frac{sv}{2} \right\rfloor - 1 \right) \left( \left\lfloor \frac{sv - 2}{2} \right\rfloor - 1, \left\lfloor \frac{sv}{2} \right\rfloor \right) \)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB} \text{\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{NB} \text{\_STRITCLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y = 0 \Rightarrow x = 0 \)
2. \( x > 0 \land y > 0 \Rightarrow x \leq y - 1 \)
   - \( sv \geq 6: (1, 2) \ (2, 3) \)

\[ sv = 12 \]

\[ sv = 11 \]
\textsc{nb\_decreasing\_terrace}(x, \textsc{variables}) \land \\
\textsc{nb\_strictly\_increasing\_sequence}(y, \textsc{variables}) \text{ with } sv = |\textsc{variables}|

1. \[ x > 0 \land y > 0 \implies 2 \times x + y \leq sv - 2 \]

\begin{itemize}
  \item \( sv \mod 2 = 0 \land \sv \geq 6: \text{\(((sv - 2)/2) - 1, 2\)} \text{\(((sv - 2)/2) - 2, 3\)} \)
  \item \( sv \mod 2 = 1 \land \sv \geq 5: \text{\(((sv - 2)/2) - 2, 1\)} \text{\(((sv - 2)/2) - 1, 2\)} \)
\end{itemize}

\textsc{nb\_decreasing\_terrace}(x, \textsc{variables}) \land \\
\textsc{nb\_summit}(y, \textsc{variables}) \text{ with } sv = |\textsc{variables}|

\begin{itemize}
  \item \( 2 \times x + 2 \times y \leq sv - 1 - (sv - 1) \mod 2 \)
  \item \( (sv - 1) \mod 2 = 0 \land \sv \geq 5: \text{\(((sv - 2)/2) - 1, 2\)} \text{\(((sv - 2)/2) - 1, 2\)} \)
  \item \( (sv - 1) \mod 2 = 1 \land \sv \geq 4: \text{\(((sv - 2)/2) - 1, 2\)} \text{\(((sv - 2)/2) - 1, 2\)} \)
\end{itemize}

\text{{\small sv = 12}}

\text{{\small sv = 11}}
5. A DATABASE OF PARAMETERISED INVARIANTS

NB\_DECREASING\_TERRACE(\(x\), VARIABLES) \land \nNB\_VALLEY(\(y\), VARIABLES) with \(sv = \lvert\text{VARIABLES}\rvert\)

1. \(2\times x + 2\times y \leq sv - 1 - (sv - 1) \mod 2\)
   - \((sv - 1) \mod 2 = 0 \land sv \geq 5: (\lfloor (sv - 2)/2 \rfloor, 1)\)
   - \((sv - 1) \mod 2 = 1 \land sv \geq 4: (\lfloor (sv - 2)/2 \rfloor, 0)\)

\(sv = 12\)

\(sv = 11\)

NB\_DECREASING\_TERRACE(\(x\), VARIABLES) \land \nNB\_ZIGZAG(\(y\), VARIABLES) with \(sv = \lvert\text{VARIABLES}\rvert\)

1. \(x + 3\times y \leq sv - 1\)
   - \(sv \mod 3 = 0 \land sv \geq 24: (2, \lfloor (sv - 1)/3 \rfloor)\)
   - \(sv \mod 3 = 1 \land sv \geq 16: (6, \lfloor (sv - 1)/3 \rfloor)\)
   - \(sv \mod 3 = 2 \land sv \geq 20: (1, \lfloor (sv - 1)/3 \rfloor)\)

\(sv > 1 \Rightarrow 2\times x + 2\times y \leq sv - 2 - (sv - 2) \mod 2\)

\(sv = 12\)

\(sv = 11\)
\[ \text{NB\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y = 0 \Rightarrow x = 0 \)
2. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x \leq y - 2 \)
3. \( sv \geq 0: (1, 4) \quad (2, 6) \)
4. \( x < 2 \lor y \neq 5 \)
5. \( x < 1 \lor y \neq 3 \)
6. \( x \neq 0 \lor y \neq 1 \)

\begin{align*}
\text{sv} & = 12 \\
\text{sv} & = 11
\end{align*}
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}\_\text{DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \times x \leq y\)
   - \(sv \geq 4: (0, 0) \quad (1, 2)\)

2. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 5: (1, sv - 2) \quad (1, sv - 3)\)
\( \text{NB}\_\text{DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_GORGE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

\begin{align*}
1. & \quad x > 0 \land y > 0 \implies 2x \leq sv - 3 - (sv - 3) \mod 2 \\
2. & \quad (sv - 1) \mod 2 = 1 \land sv \geq 4: \quad ((sv - 2)/2 - 1, sv - 3) \\
& \quad \begin{cases} 
\quad x \neq \max(0, ([sv - 2]/2)) - 1, \\
\quad y < 2, \\
\quad y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 2, \\
\quad 1 = sv \mod 2, \\
\quad 1 = y \mod 2 
\end{cases} \\
3. & \quad \begin{cases} 
\quad x \neq \max(0, ([sv - 2]/2)), \\
\quad y < 1, \\
\quad y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 1, \\
\quad 0 = sv \mod 2 
\end{cases}
\end{align*}

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\text{NB\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \(2 \times x + y \leq sv\)
2. \(x = \max(0, \lfloor (sv - 2)/2 \rfloor) \land sv \mod 2 = 1 \land sv > 3 \Rightarrow y \leq 2\)
3. \(x > 0 \land y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2\)
4. \((sv - 1) \mod 2 = 1 \land sv \geq 4: (\lfloor (sv - 2)/2 \rfloor - 1, 2) \land (\lfloor (sv - 2)/2 \rfloor - 1, 3)\)
5. \(y \neq 1\)
\[ \text{NB}_{\text{DECREASING\_TERRACE}}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( sv > 1 \Rightarrow 2 \times x + y \leq sv - 2 \)
2. \( sv \mod 2 = 0 \land sv \geq 4: \left( \left( \frac{(sv - 2)}{2} \right), 0 \right) \land (0, sv - 2) \)
3. \( x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq sv - 3 \)
4. \( sv \geq 9: (1, sv - 5) \land (2, sv - 7) \)
5. \( o = x + \left( \frac{(y + 1)}{2} \right) \land x \neq \text{max}(0, \left( \frac{(sv - 2)}{2} \right)) - 1, \land \quad y \neq (sv - 2) \times \text{min}(1, \text{max}(0, sv - 3)) - 4, \land \quad sv < 7 \)
6. \( x \neq 2 \lor y \neq (sv - 2) \times \text{min}(1, \text{max}(0, sv - 3)) - 4 \lor sv < 7 \)
7. \( x \neq \text{max}(0, \left( \frac{(sv - 2)}{2} \right)) - 2, \land y < 3, \land \quad x > (sv - 2) \times \text{min}(1, \text{max}(0, sv - 3)) - 2, \land 1 = sv \mod 2, \land 1 = y \mod 2 \)
8. \( x > \text{max}(0, \left( \frac{(sv - 2)}{2} \right)) - 1, \land 1 = sv \mod 2, \land 0 = x \mod 2 \land x \neq \text{max}(0, \left( \frac{(sv - 2)}{2} \right)) - 1, \land y < 1, \land y > (sv - 2) \times \text{min}(1, \text{max}(0, sv - 3)) - 2, \land 1 = sv \mod 2 \)
9. \( y \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_{\text{DECREASING TERRACE}}(x, \text{VARIABLES}) \wedge \text{SUM}_{\text{WIDTH INFLEXION}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \wedge y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2 \)

2. \( (sv - 1) \mod 2 = 0 \wedge sv \geq 4 \):
   \[
   \begin{align*}
   & 2 \times (sv - 2) \times [0, (sv - 2) / 2] - 1, sv - 2) - 3, \\
   & y < 3,
   \end{align*}
   \]

3. \( (sv - 1) \mod 2 = 1 \wedge sv \geq 4 \):
   \[
   \begin{align*}
   & 2 \times (sv - 2) \times [0, (sv - 2) / 2] - 1, sv - 3) - 3, \\
   & y < 3,
   \end{align*}
   \]

\[ \left\{ \begin{array}{l}
   x \neq \max(0, [(sv - 2) / 2]) - 1, \\
   y \neq \max(0, [(sv - 2) / 2]), \\
   1 = sv \mod 2, \\
   0 = y \mod 2 \\
   \end{array} \right. \]

\( y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 3, \)

\( y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 1, \)

\( 0 = sv \mod 2 \)

\( s = sv \mod 2 \)

\( 0 = y \mod 2 \)

\( y < 3, \)

\( y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 3, \)

\( y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 1, \)

\( 0 = sv \mod 2 \)
\(\text{NB\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES})\) with \(sv = |\text{VARIABLES}|\)

1. \(x > 0 \land y > 0 \Rightarrow 2x \leq sv - 3 - (sv - 3) \mod 2\)
2. \((sv - 1) \mod 2 = 1 \land sv \geq 4: ((sv - 2)/2 - 1, sv - 2)\)
3. \(x > 0 \land y > 0 \Rightarrow 2x \leq y + sv - 5\)
   \[x \neq \max(0, [(sv - 2)/2]) - 1,\]
   \[y < 2,\]
   \[y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 2,\]
   \[1 = sv \mod 2,\]
   \[1 = y \mod 2\]
4. \(x \neq \max(0, [(sv - 2)/2]),\)
   \[y < 1,\]
   \[y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 1,\]
   \[0 = sv \mod 2\]

\(sv = 12\)

\(sv = 11\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB}_\text{DECREASING_\text{TERRACE}}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH_\text{PLAIN}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \( sv > 1 \Rightarrow 2 \times x + y \leq sv - 2 \)
2. \( sv \geq 4: (0, sv - 2) \) \( (1, sv - 4) \)

\[
\begin{align*}
\text{NB}_\text{DECREASING_\text{TERRACE}}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH_\text{PLATEAU}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \( sv > 1 \Rightarrow 2 \times x + y \leq sv - 2 \)
2. \( sv \geq 4: (0, sv - 2) \) \( (1, sv - 4) \)
\[
\text{NB\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(sv > 1 \Rightarrow 2 \times x + y \leq sv - 2\)
2. \(sv \geq 6: (0, sv - 2) \land (1, sv - 4)\)
3. \(y > 0 \Rightarrow 2 \times x \leq sv - 4 - (sv - 4) \mod 2\)
4. \(sv \mod 2 = 1 \land sv \geq 5: (\lfloor (sv - 2)/2 \rfloor - 1, 2) \land (\lfloor (sv - 2)/2 \rfloor - 1, 3)\)
5. \(sv \mod 2 = 1 \land sv \geq 5: (\lfloor (sv - 2)/2 \rfloor - 1, 3)\)

\[
\text{NB\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(sv > 1 \Rightarrow 2 \times x + y \leq sv - 2\)
2. \(sv \geq 6: (0, sv - 2) \land (1, sv - 4)\)
3. \(y > 0 \Rightarrow 2 \times x \leq sv - 4 - (sv - 4) \mod 2\)
4. \(sv \mod 2 = 1 \land sv \geq 5: (\lfloor (sv - 2)/2 \rfloor - 1, 2) \land (\lfloor (sv - 2)/2 \rfloor - 1, 3)\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\text{NB\_DECREASING\_TERRACE}(x, VARIABLES) \land \text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, VARIABLES) \text{ with } sv = |VARIABLES|

\begin{enumerate}
\item $2 \times x \leq y$
\begin{itemize}
\item $sv \geq 6$: (1, 2) (2, 4)
\end{itemize}
\item $x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq 2 \times sv - 4$
\begin{itemize}
\item $sv \mod 2 = 0 \land sv \geq 4$: $\lfloor (sv - 2)/2 \rfloor, sv - 2$ $\lfloor (sv - 2)/2 \rfloor - 1, sv$ \end{itemize}
\item $x \neq 0 \lor y \neq 1$
\end{enumerate}

\begin{tikzpicture}[scale=0.8]
\begin{axis}[
axis lines=left,
tickwidth=0.1pt,
ticklen=2pt,
xlabel={nb\_decreasing\_terrace},
ylabel={sum\_width\_steady\_sequence},
xmin=0, xmax=12,
ymin=0, ymax=12,
]
\addplot [only marks, mark size=2pt, mark=square] coordinates {
(1, 2) (2, 4) (11, 11) (10, 10) (9, 9) (8, 8) (7, 7) (6, 6) (5, 5) (4, 4) (3, 3) (2, 2) (1, 1)
};
\end{axis}
\end{tikzpicture}
\( \text{NB}\_\text{DECREASING}\_\text{TERRACE}(x, \text{VARIABLES}) \land \text{SUM}\_\text{WIDTH}\_\text{STRICTLY}\_\text{DECREASING}\_\text{SEQUENCE}(y, \text{VARIABLES}) \) with $sv = |\text{VARIABLES}|$

1. $y = 0 \Rightarrow x = 0$
2. $x > 0 \land y > 0 \Rightarrow 2 \ast x \leq y - 2$
3. $x < 2 \land y \geq 5$
4. $y < 1,$
5. $y > sv \ast \max(1, \max(0, sv - 1)) - 3,$
6. $x = 0 \land y \mod 2$
7. $x < 1 \lor y \neq 3$
8. $x \neq 0 \lor y \neq 1$
5. A DATABASE OF PARAMETERISED INVARIANTS

\( \text{NB\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = \text{|VARIABLES|} \)

1. \( 2 \times x + y \leq sv \)
   - \( sv \geq 8: (1, sv - 2) \land (2, sv - 4) \)
2. \( x = \max(0, \lfloor (sv - 2)/2 \rfloor) \land sv \mod 2 = 1 \land sv > 3 \Rightarrow y \leq 2 \)
3. \( x > 0 \land y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2 \)
4. \( (sv - 1) \mod 2 = 1 \land sv \geq 4: (\lfloor (sv - 2)/2 \rfloor - 1, 2) \land (\lfloor (sv - 2)/2 \rfloor - 1, 3) \)
5. \( y \neq 1 \)
\[\text{NB}_\text{DECREASING}_\text{TERRACE}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(x > 0 \land y > 0 \Rightarrow 2 \cdot x \leq sv - 3 - (sv - 3) \mod 2\)
2. \((sv - 1) \mod 2 = 1 \land sv \geq 4: ((sv - 2)/2 - 1, sv - 2)\) \(((sv - 2)/2 - 1, sv - 3)\)

3. \(x > 0 \land y > 0 \Rightarrow 2 \cdot x \leq y + sv - 5\)
   \(x \neq \max(0, [(sv - 2)/2]) - 1, \quad y < 2,\)
   \(y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 2,\)
   \(1 = sv \mod 2,\)
   \(1 = y \mod 2,\)

4. \(x \neq \max(0, [(sv - 2)/2]),\)
   \(y < 1,\)
   \(y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 1,\)
   \(0 = sv \mod 2\)

\[sv = 12\]
\[sv = 11\]
A DATABASE OF PARAMETERISED INVARIANTS

\[ NB_{DECREASING\_TERRACE}(x, \text{VARIABLES}) \land SUM\_WIDTH\_VALLEY(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2 \)
2. \( x > 0 \land y > 0 \Rightarrow 2 \times x \leq y + sv - 5 \)
3. \( x \neq \max(0, [(sv - 2)/2]) - 1, y < 2, y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 2, 1 = sv \mod 2, 1 = y \mod 2 \)
4. \( x \neq \max(0, [(sv - 2)/2]), y < 1, y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 1, 0 = sv \mod 2 \)

\( sv = 12 \)

\( sv = 11 \)
\( \text{NB\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( sv > 1 \Rightarrow 2 \cdot x + y \leq sv - 2 \)
   - \( sv \geq 6: \ (0, sv - 2) \quad (1, sv - 4) \)

2. \( y > 0 \Rightarrow 2 \cdot x \leq sv - 4 - (sv - 4) \mod 2 \)
   - \( sv \mod 2 = 1 \land sv \geq 5: \ ((sv - 2)/2 - 1, 2) \quad ((sv - 2)/2 - 1, 3) \)

3. \( y \neq 1 \)

\( \text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_GORGE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x \leq y \)
   - \( sv \geq 6: \ (0, 0) \quad (1, 1) \)

2. \( x > 0 \land y > 0 \Rightarrow x + 2 \cdot y \leq sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 16: \ (2, [(sv - 1)/2] - 1) \quad (4, [(sv - 1)/2] - 2) \)
   - \( sv \mod 2 = 1 \land sv \geq 13: \ (1, [(sv - 1)/2] - 1) \quad (3, [(sv - 1)/2] - 2) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \]
\[ \text{NB\_INCREASING}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 9: (1, sv - 2) \quad (2, sv - 3) \)
2. \( y = 0 \Rightarrow x = 0 \)
3. \( x > 0 \land y > 0 \Rightarrow 2 * x \leq y - 2 \)
   - \( sv \geq 9: (1, 4) \quad (2, 6) \)
4. \( x < 1 \lor y \neq 3 \)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \]
\[ \text{NB\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y = 0 \Rightarrow x = 0 \)
2. \( x > 0 \land y > 0 \Rightarrow x + 2 * y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 12: (1, \lfloor sv/2 \rfloor - 1) \quad (3, \lfloor sv/2 \rfloor - 2) \)
   - \( sv \mod 2 = 1 \land sv \geq 15: (2, \lfloor sv/2 \rfloor - 1) \quad (4, \lfloor sv/2 \rfloor - 2) \)
3. \( x > 0 \land y > 0 \Rightarrow x \leq y - 1 \)
   - \( sv \geq 9: (1, 2) \quad (2, 3) \)

\[ sv = 12 \]

\[ sv = 11 \]
\[\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(sv > 1 \Rightarrow 3 \times x + 2 \times y \leq sv - 2 - (sv - 2) \mod 2\)
2. \(sv \mod 2 = 1 \land sv \geq 9: (0, \lfloor (sv - 2)/2 \rfloor) \quad (2, \lfloor (sv - 2)/2 \rfloor - 3)\)
3. \(x > 0 \land y > 0 \Rightarrow 3 \times x + 2 \times y \leq sv - 3\)
4. \(sv \mod 2 = 0 \land sv \geq 12: (1, \lfloor (sv - 2)/2 \rfloor - 2) \quad (3, \lfloor (sv - 2)/2 \rfloor - 5)\)

\[\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(2 \times x \leq y\)
2. \(sv \geq 6: (0, 0) \quad (1, 2)\)
3. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
4. \(x > 0 \land y > 0 \Rightarrow x + y \leq sv - 3\)
5. \(sv \geq 9: (1, sv - 4) \quad (2, sv - 5)\)
\( \text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_PEAK}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x \leq y \)
   - \( sv \geq 6 \): \((0, 0) (1, 1)\)

2. \( x > 0 \land y > 0 \Rightarrow x + 2 \cdot y \leq sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 16 \): \((2, \lceil (sv - 1)/2 \rceil - 1) (4, \lceil (sv - 1)/2 \rceil - 2)\)
   - \( sv \mod 2 = 1 \land sv \geq 13 \): \((1, \lceil (sv - 1)/2 \rceil - 1) (3, \lceil (sv - 1)/2 \rceil - 2)\)

\( sv = 12 \)

\( sv = 11 \)
$\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|$

1. $x \leq y$
   - $sv \geq 6$: $(0, 0) \land (1, 1)$

2. $x > 0 \land y > 0 \Rightarrow x + 2 \ast y \leq sv - 2$
   - $sv \mod 2 = 0 \land sv \geq 16$: $(2, \lfloor(sv - 1)/2\rfloor - 1) \land (4, \lfloor(sv - 1)/2\rfloor - 2)$
   - $sv \mod 2 = 1 \land sv \geq 13$: $(1, \lfloor(sv - 1)/2\rfloor - 1) \land (3, \lfloor(sv - 1)/2\rfloor - 2)$

$sv = 12$

$sv = 11$

$\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|$

1. $x > 0 \land y > 0 \Rightarrow 3 \ast x + 3 \ast y \leq sv - 2 - (sv - 2) \mod 3$
   - $(sv - 2) \mod 3 = 0 \land sv \geq 8$: $(\lfloor(sv - 3)/3\rfloor, 1) \land (\lfloor(sv - 3)/3\rfloor - 1, 2)$
   - $(sv - 2) \mod 3 = 1 \land sv \geq 9$: $(\lfloor(sv - 3)/3\rfloor - 1, 1) \land (\lfloor(sv - 3)/3\rfloor - 2, 2)$
   - $(sv - 2) \mod 3 = 2 \land sv \geq 10$: $(\lfloor(sv - 3)/3\rfloor - 1, 1) \land (\lfloor(sv - 3)/3\rfloor - 2, 2)$

$sv = 12$

$sv = 11$
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow 3 \cdot x + 3 \cdot y \leq sv - 2 - (sv - 2) \mod 3 \)
   - \( (sv - 2) \mod 3 = 0 \land sv \geq 8: \) \((\lfloor sv/3 \rfloor, 1)\)
   - \( (sv - 2) \mod 3 = 1 \land sv \geq 9: \) \((\lfloor sv/3 \rfloor - 1, 1)\)
   - \( (sv - 2) \mod 3 = 2 \land sv \geq 10: \) \((\lfloor sv/3 \rfloor - 1, 2)\)

\[ \text{sv} = 12 \]

\[ \text{sv} = 11 \]

\[ \text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_STEADY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow 3 \cdot x + y \leq sv - 3 \)
   - \( sv \geq 9: \) \((1, sv - 6)\)
   - \( x < 1 \lor y \neq \max(0, sv - 1) - 4 \)

\[ \text{sv} = 12 \]

\[ \text{sv} = 11 \]
\[ \text{NB_DIP_ON_INCREASING_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB_STEADY_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow 3x + 2y \leq sv - 1 \)
2. \( sv \mod 2 = 0 \land sv \geq 14: (1, \lfloor sv/2 \rfloor - 2) \quad (3, \lfloor sv/2 \rfloor - 5) \)
3. \( x > 0 \land y > 0 \Rightarrow 3x \leq y + sv - 5 \)

\[ \text{NB_DIP_ON_INCREASING_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB_STRICTLY_DECREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
2. \( sv \geq 9: (1, 1) \quad (2, 2) \)
3. \( x > 0 \land y > 0 \Rightarrow 2x + y \leq sv - 3 \)
4. \( x > 0 \land y > 0 \Rightarrow x + 2y \leq sv - 1 \)
5. \( sv \mod 2 = 0 \land sv \geq 14: (1, \lfloor sv/2 \rfloor - 1) \quad (3, \lfloor sv/2 \rfloor - 2) \)
6. \( sv \mod 2 = 1 \land sv \geq 17: (2, \lfloor sv/2 \rfloor - 1) \quad (4, \lfloor sv/2 \rfloor - 2) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{DIP}_\text{ON}\_\text{INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB}_\text{STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y = 0 \Rightarrow x = 0 \)
2. \( x > 0 \land y > 0 \Rightarrow x + 2 \times y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 12: (1, \lfloor sv/2 \rfloor - 1) \quad (3, \lfloor sv/2 \rfloor - 2) \)
   - \( sv \mod 2 = 1 \land sv \geq 15: (2, \lfloor sv/2 \rfloor - 1) \quad (4, \lfloor sv/2 \rfloor - 2) \)
3. \( x > 0 \land y > 0 \Rightarrow x \leq y - 1 \)
   - \( sv \geq 9: (1, 2) \quad (2, 3) \)

\[ \text{sv} = 12 \]

\[ \text{sv} = 11 \]

\[ \text{NB}_\text{DIP}_\text{ON}\_\text{INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB}_\text{SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \geq 6: (0, 0) \quad (1, 1) \)
2. \( x > 0 \land y > 0 \Rightarrow x + 2 \times y \leq sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 16: (2, \lfloor (sv - 1)/2 \rfloor - 1) \quad (4, \lfloor (sv - 1)/2 \rfloor - 2) \)
   - \( sv \mod 2 = 1 \land sv \geq 13: (1, \lfloor (sv - 1)/2 \rfloor - 1) \quad (3, \lfloor (sv - 1)/2 \rfloor - 2) \)

\[ \text{sv} = 12 \]

\[ \text{sv} = 11 \]
\( \text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \wedge \text{NB\_VALLEY}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x \leq y \)
   - \( sv \geq 6 \): (0, 0) (1, 1)
2. \( x > 0 \land y > 0 \Rightarrow x + 2 \ast y \leq sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 16 \): (2, \( \lfloor (sv - 1)/2 \rfloor - 1 \)) (4, \( \lfloor (sv - 1)/2 \rfloor - 2 \))
   - \( sv \mod 2 = 1 \land sv \geq 13 \): (1, \( \lfloor (sv - 1)/2 \rfloor - 1 \)) (3, \( \lfloor (sv - 1)/2 \rfloor - 2 \))

\( sv = 12 \) \hspace{1cm} \( sv = 11 \)

\( \text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \wedge \text{NB\_ZIGZAG}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x \leq y \)
   - \( sv \geq 6 \): (0, 0) (1, 1)
2. \( x > 0 \land y > 0 \Rightarrow x + 2 \ast y \leq sv - 3 \)

\( sv = 12 \) \hspace{1cm} \( sv = 11 \)

\( \text{nb\_dip\_on\_increasing\_sequence} \)

\( \text{nb\_valley} \)

\( \text{nb\_ziggag} \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{DIP}_{\text{ON INCREASING SEQUENCE}}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_{\text{DECREASING SEQUENCE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x \leq y\)
   - \(sv \geq 9: (1, 2) (2, 4)\)
2. \(x > 0 \land y > 0 \Rightarrow 4 \times x + y \leq 2 \times sv - 6\)
3. \(x > 0 \land y > 0 \Rightarrow x + y \leq sv - 1\)
   - \(sv \geq 11: (1, sv - 2) (2, sv - 3)\)
4. \(y \neq 2 \times x + 1\)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]

\[ \text{nb}_\text{dip_on_increasing_sequence} \]

\[ \text{sum}_\text{width_decreasing_sequence} \]
\[\text{NB_DIP_ON_INCREASING_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(2 \times x \leq y\)
   - \(sv \geq 6\) \((0, 0)\) (1, 2)
2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 7\) \((0, sv - 2)\) (1, sv - 3)
3. \(x > 0 \land y > 0 \Rightarrow 4 \times x + y \leq 2 \times sv - 6\)

\[\text{NB_DIP_ON_INCREASING_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_INCREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

0. \(y = 0 \Rightarrow x = 0\)
1. \(x > 0 \land y > 0 \Rightarrow 3 \times x \leq y - 3\)
   - \(sv \geq 9\) \((1, 6)\) (2, 9)
2. \(x < 1 \lor y \neq 4\)
3. \(x < 1 \lor y \neq 5\)
4. \(x \neq 0 \lor y \neq 1\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \\
1. sv > 1 \Rightarrow 3 \times x + y \leq sv - 2 \\
2. x > 0 \land y > 0 \Rightarrow 3 \times x + y \leq sv - 3 \\
\quad sv \geq 11: (1, sv - 6) (2, sv - 9) \\
3. y \neq 1
\end{align*}

\begin{align*}
\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \\
1. 3 \times x \leq y \\
\quad sv \geq 6: (0, 0) (1, 3) \\
2. x > 0 \land y > 0 \Rightarrow 3 \times x + 3 \times y \leq 4 \times sv - 10 - (4 \times sv - 10) \mod 3 \\
\quad (sv - 1) \mod 3 = 2 \land sv \geq 6: (\left\lfloor (sv - 3) / 3 \right\rfloor, sv - 3) (\left\lfloor (sv - 3) / 3 \right\rfloor - 1, sv - 2)
\end{align*}
\( \text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x \leq y \)
   - \( sv \geq 6: (0, 0) \) \( (1, 2) \)
2. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 7: (0, sv - 2) \) \( (1, sv - 3) \)
3. \( x > 0 \land y > 0 \Rightarrow 4 \times x + y \leq 2 \times sv - 6 \)

\( sv = 12 \)

\( sv = 11 \)

\( \text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x \leq y \)
   - \( sv \geq 6: (0, 0) \) \( (1, 1) \)
2. \( sv > 1 \Rightarrow 2 \times x + y \leq sv - 2 \)
3. \( x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq sv - 3 \)
   - \( sv \geq 9: (1, sv - 5) \) \( (2, sv - 7) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. $x \leq y$
   - $sv \geq 6$: $(0, 0)$ $(1, 1)$
2. $sv > 1 \Rightarrow 2 \cdot x + y \leq sv - 2$
3. $x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq sv - 3$
   - $sv \geq 9$: $(1, sv - 5)$ $(2, sv - 7)$

\begin{align*}
\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. $sv > 1 \Rightarrow 3 \cdot x + y \leq sv - 2$
2. $x > 0 \land y > 0 \Rightarrow 3 \cdot x + y \leq sv - 3$
   - $sv \geq 11$: $(1, sv - 6)$ $(2, sv - 9)$
3. $y \neq 1$
\(NB\_DIP\_ON\_INCREASING\_SEQUENCE(x, VARIABLES) \land \) 
\(SUM\_WIDTH\_PROPER\_PLATEAU(y, VARIABLES) \) with \(sv = |VARIABLES|\)

1. \(sv > 1 \Rightarrow 3 \times x + y \leq sv - 2\)
2. \(x > 0 \land y > 0 \Rightarrow 3 \times x + y \leq sv - 3\)

\(sv \geq 11: (1, sv - 6) \quad (2, sv - 9)\)

3. \(y \neq 1\)

\(nb\_dip\_on\_increasing\_sequence\)

---

\(NB\_DIP\_ON\_INCREASING\_SEQUENCE(x, VARIABLES) \land \) 
\(SUM\_WIDTH\_STEADY\_SEQUENCE(y, VARIABLES) \) with \(sv = |VARIABLES|\)

1. \(x > 0 \land y > 0 \Rightarrow 3 \times x + y \leq sv - 1\)

\(sv \geq 11: (1, sv - 4) \quad (2, sv - 7)\)

2. \(x > 0 \land y > 0 \Rightarrow 3 \times x \leq sv - 4 - (sv - 4) \mod 3\)

\(sv - 1 \mod 3 = 1 \land sv \geq 5: (\lfloor (sv - 3) / 3 \rfloor, 2) \quad (\lfloor (sv - 3) / 3 \rfloor, 3)\)

\(sv - 1 \mod 3 = 2 \land sv \geq 6: (\lfloor (sv - 3) / 3 \rfloor - 1, 2) \quad (\lfloor (sv - 3) / 3 \rfloor - 1, 3)\)

3. \(y \neq 1\)

\(nb\_dip\_on\_increasing\_sequence\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \times x \leq y\)
   - \(sv \geq 9: \ (1, 2) \ (2, 4)\)
2. \(x > 0 \land y > 0 \Rightarrow 4 \times x + y \leq 2 \times sv - 6\)
3. \(x > 0 \land y > 0 \Rightarrow x + y \leq sv - 1\)
   - \(sv \geq 11: \ (1, sv - 2) \ (2, sv - 3)\)
4. \(y \neq 2 \times x + 1\)
\(\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICKLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES})\) with \(sv = |\text{VARIABLES}|\)

1. \(y = 0 \Rightarrow x = 0\)
2. \(x > 0 \land y > 0 \Rightarrow 3 \times x \leq y - 3\)
3. \(sv \geq 9: (1, 6)\) \((2, 9)\)
4. \(x < 1 \lor y \neq 4\)
5. \(x < 1 \lor y \neq 5\)
6. \(x \neq 0 \lor y \neq 1\)

\(sv = 12\)
\(sv = 11\)

\(\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES})\) with \(sv = |\text{VARIABLES}|\)

1. \(2 \times x \leq y\)
2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
3. \(sv \geq 7: (0, sv - 2)\) \((1, sv - 3)\)
4. \(x > 0 \land y > 0 \Rightarrow 4 \times x + y \leq 2 \times sv - 6\)

\(sv = 12\)
\(sv = 11\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB~DIP~ON~INCREASING~SEQUENCE}(x, \text{VARIABLES}) \wedge \\
\text{SUM~WIDTH~VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \times x \leq y\)
   - \(sv \geq 6: (0, 0) \ (1, 2)\)
2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 7: (0, sv - 2) \ (1, sv - 3)\)
3. \(x > 0 \wedge y > 0 \Rightarrow 4 \times x + y \leq 2 \times sv - 6\)

\[
\text{NB~DIP~ON~INCREASING~SEQUENCE}(x, \text{VARIABLES}) \wedge \\
\text{SUM~WIDTH~ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \times x \leq y\)
   - \(sv \geq 6: (0, 0) \ (1, 2)\)
2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
3. \(x > 0 \wedge y > 0 \Rightarrow x + y \leq sv - 3\)
   - \(sv \geq 11: (1, sv - 4) \ (2, sv - 5)\)
4. \(y \neq 2 \times x + 1\)
\[ \text{NB}_\text{GORGE}(x, \text{VARIABLES}) \land \\
\text{NB}_\text{INCREASING}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
2. \( sv \geq 5: \ (1, sv - 2) \quad (2, sv - 3) \)
3. \( x \leq y \)
   - \( sv \geq 5: \ (1, 1) \quad (2, 2) \)

\[ sv = 12 \]
\[ sv = 11 \]

\[ \text{NB}_\text{GORGE}(x, \text{VARIABLES}) \land \\
\text{NB}_\text{INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \geq 5: \ (1, 1) \quad (2, 2) \)
2. \( 3 \cdot y \leq x + sv + 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 8: \ ((sv - 1)/2, \lfloor sv/2 \rfloor) \quad (\lfloor (sv - 1)/2 \rfloor + 3, \lfloor sv/2 \rfloor - 1) \)
   - \( sv \mod 2 = 1 \land sv \geq 11: \ ((sv - 1)/2 - 2, \lfloor sv/2 \rfloor) \quad (\lfloor (sv - 1)/2 \rfloor - 5, \lfloor sv/2 \rfloor - 1) \)

\[ sv = 12 \]
\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\textbf{NB\_GORGE}(x, \text{VARIABLES}) \land \textbf{NB\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } \text{sv} = |\text{VARIABLES}|

1) \(2 \ast x + 2 \ast y \leq \text{sv} - 1 - (\text{sv} - 1) \mod 2\)
   - \( (\text{sv} - 1) \mod 2 = 0 \land \text{sv} \geq 7: (\lceil (\text{sv} - 1)/2 \rceil - 1, 1) \quad (\lceil (\text{sv} - 1)/2 \rceil - 2, 2)\)
   - \( (\text{sv} - 1) \mod 2 = 1 \land \text{sv} \geq 4: (\lceil (\text{sv} - 1)/2 \rceil, 0) \quad (\lceil (\text{sv} - 1)/2 \rceil - 1, 1)\)

\textbf{NB\_GORGE}(x, \text{VARIABLES}) \land \textbf{NB\_INFLEXION}(y, \text{VARIABLES}) \text{ with } \text{sv} = |\text{VARIABLES}|

\(2 \ast x \leq y + 1\)
   - \( \text{sv} \geq 5: (1, 1) \quad (2, 3)\)

\(\text{sv} > 1 \Rightarrow 3 \ast y \leq 2 \ast x + 2 \ast \text{sv} - 3\)
   - \( \text{sv} \mod 2 = 0 \land \text{sv} \geq 12: (\lceil (\text{sv} - 1)/2 \rceil - 2, \text{sv} - 3) \quad (\lceil (\text{sv} - 1)/2 \rceil - 5, \text{sv} - 5)\)
   - \( \text{sv} \mod 2 = 1 \land \text{sv} \geq 9: (\lceil (\text{sv} - 1)/2 \rceil - 1, \text{sv} - 2) \quad (\lceil (\text{sv} - 1)/2 \rceil - 4, \text{sv} - 4)\)
\[ \begin{align*}
NB\_GORGE(\mathbf{x}, \text{VARIABLES}) \land \\
NB\_PEAK(\mathbf{y}, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \\
\text{1. } \mathbf{x} \leq \mathbf{y} + 1 \\
\text{2. } sv \geq 5: \ (1, 0) \quad (2, 1) \\
\text{3. } 3 \times \mathbf{y} \leq \mathbf{x} + sv \land \text{sv mod 2 = 0} \land sv \geq 12: \ ([((\text{sv} - 1)/2] - 2, [(\text{sv} - 1)/2]) \quad ([((\text{sv} - 1)/2] - 5, [(\text{sv} - 1)/2] - 1) \\
\text{4. } \text{sv mod 2 = 1} \land \text{sv mod 2 = 0} \land sv \geq 9: \ ([((\text{sv} - 1)/2] - 1, [(\text{sv} - 1)/2]) \quad ([((\text{sv} - 1)/2] - 4, [(\text{sv} - 1)/2] - 1) \\
\text{5. } sv > 1 \Rightarrow \mathbf{x} + \mathbf{y} \leq sv - 2 \\
\text{6. } sv \mod 2 = 1 \land sv \geq 3: \ ([((\text{sv} - 1)/2], [(\text{sv} - 1)/2] - 1) \quad ([((\text{sv} - 1)/2] - 1, [(\text{sv} - 1)/2]) \\
\end{align*} \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\( \text{NB\_GORGE}(x, \text{VARIABLES}) \land \text{NB\_PLATEAU}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x \leq y + 1 \)
   - \( sv \geq 5 \): (1, 0) (2, 1)

2. \( 3 \times y \leq x + sv \)
   - \( sv \mod 2 = 0 \land sv \geq 12 \): \((\lfloor (sv - 1)/2 \rfloor - 2, \lfloor (sv - 1)/2 \rfloor)\) \((\lfloor (sv - 1)/2 \rfloor - 5, \lfloor (sv - 1)/2 \rfloor - 1)\)
   - \( sv \mod 2 = 1 \land sv \geq 9 \): \((\lfloor (sv - 1)/2 \rfloor - 1, \lfloor (sv - 1)/2 \rfloor - 1)\) \((\lfloor (sv - 1)/2 \rfloor, \lfloor (sv - 1)/2 \rfloor - 1)\)

3. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 8 \): \((\lfloor (sv - 1)/2 \rfloor - 1, \lfloor (sv - 1)/2 \rfloor - 1)\) \((\lfloor (sv - 1)/2 \rfloor - 1, \lfloor (sv - 1)/2 \rfloor - 1)\)

\( \text{NB\_GORGE}(x, \text{VARIABLES}) \land \text{NB\_PROPER\_PLAIN}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x + 3 \times y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 10 \): \((\lfloor (sv - 1)/2 \rfloor - 1, 1)\) \((\lfloor (sv - 1)/2 \rfloor - 4, 3)\)

\( sv = 12 \)

\( sv = 11 \)
\[
\text{NB\_GORGE}(x, \text{VARIABLES}) \land \\
\text{NB\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \cdot x + y \leq sv - 1\)
2. \(sv \mod 2 = 0 \land sv \geq 12: ((sv - 1)/2, 1) \land ((sv - 1)/2 - 1, 3)\)
3. \(sv \mod 2 = 1 \land sv \geq 15: ((sv - 1)/2 - 1, 2) \land ((sv - 1)/2 - 2, 4)\)
4. \(4 \cdot y \leq x + sv\)
5. \(sv > 1 \Rightarrow x + 2 \cdot y \leq sv - 2\)

\[
\text{NB\_GORGE}(x, \text{VARIABLES}) \land \\
\text{NB\_STEADY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \cdot x + y \leq sv - 1\)
2. \(sv \geq 5: (1, sv - 3) \land (2, sv - 5)\)

\[
\text{NB\_STEEADY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \cdot x + y \leq sv - 1\)
2. \(sv \geq 5: (1, sv - 3) \land (2, sv - 5)\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{GORGE}(x, \text{VARIABLES}) \land \text{NB}_\text{STABLE}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2x + y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 8 \): \((\lfloor sv - 1 \rfloor / 2, 1)\) \((\lfloor sv - 1 \rfloor / 2 - 1, 3)\)
   - \( sv \mod 2 = 1 \land sv \geq 11 \): \((\lfloor sv - 1 \rfloor / 2 - 1, 2)\) \((\lfloor sv - 1 \rfloor / 2 - 2, 4)\)

2. \( x + 2y \leq sv \)
   - \( sv \mod 2 = 0 \land sv \geq 8 \): \((0, \lfloor sv / 2 \rfloor)\) \((2, \lfloor sv / 2 \rfloor - 1)\)
   - \( sv \mod 2 = 1 \land sv \geq 11 \): \((1, \lfloor sv / 2 \rfloor)\) \((3, \lfloor sv / 2 \rfloor - 1)\)

\[ \text{NB}_\text{GORGE}(x, \text{VARIABLES}) \land \text{NB}_\text{STRICLY}_\text{DECREASING}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \geq 5 \): \((1, 1)\) \((2, 2)\)
\[ \text{NB\_GORGE}(x, \text{VARIABLES}) \land \]
\[ \text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \geq 5 \): \((1, 1) \) \((2, 2)\)

2. \( 3 \times x \leq y + sv \)
   - \( sv \mod 2 = 0 \land sv \geq 12 \): \([(\text{sv} - 1)/2, [(\text{sv} - 1)/2] - 2)\) \([(\text{sv} - 1)/2] - 1, [(\text{sv} - 1)/2] - 5)\)
   - \( sv \mod 2 = 1 \land sv \geq 9 \): \([(\text{sv} - 1)/2, [(\text{sv} - 1)/2] - 1)\) \([(\text{sv} - 1)/2] - 1, [(\text{sv} - 1)/2] - 4)\)

3. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \mod 2 = 1 \land sv \geq 3 \): \([(\text{sv} - 1)/2, [(\text{sv} - 1)/2] - 1)\) \([(\text{sv} - 1)/2] - 1, [(\text{sv} - 1)/2]\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_GORGE}(x, \text{VARIABLES}) \land \text{NB\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \geq 3 \): (0, 0) (1, 1)
2. \( 3 \cdot y \leq x + sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 10 \):
     - \((((sv - 1)/2) - 1, (sv - 1)/2))
     - \(((sv - 1)/2) - 4, [(sv - 1)/2] - 1)\)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_GORGE}(x, \text{VARIABLES}) \land \text{NB\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2 \cdot x + y \leq sv \)
   - \( sv \mod 2 = 0 \land sv \geq 14 \):
     - \((((sv - 1)/2), 2))
     - \(((sv - 1)/2) - 1, 4)\)
   - \( sv \mod 2 = 1 \land sv \geq 11 \):
     - \((((sv - 1)/2), 1))
     - \(((sv - 1)/2) - 1, 3)\)
2. \( y \leq x \)
   - \( sv \geq 4 \): (0, 0) (1, 1)
3. \( y = 0 \Rightarrow 3 \cdot x \leq sv - sv \mod 3 \)

\[ sv = 12 \]

\[ sv = 11 \]
\[ \text{NB\_Gorge}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \times x + y \leq 2 \times sv - 2 \]
2. \[ 2 \times x \leq y \]
3. \[ sv \geq 5: (1, 2) \quad (2, 4) \]
4. \[ x \neq 0 \lor y \neq 1 \]

\[ sv = 12 \]

\[ \text{NB\_Gorge}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \times x + y \leq sv - 1 \]
2. \[ y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2 \]
3. \[ (sv - 1) \mod 2 = 1 \land sv \geq 6: ((sv - 1)/2 - 1, 2) \quad ((sv - 1)/2 - 1, 3) \]
4. \[ y \neq 1 \]

\[ sv = 12 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_GORGE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 5: (1, sv - 2) \quad (2, sv - 3) \)

2. \( x \leq y \)
   - \( sv \geq 3: (0, 0) \quad (1, 1) \)

3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 4: (1, sv - 2) \quad (1, sv - 3) \)

4. \( sv = 12 \)

5. \( sv = 11 \)

\[
\text{NB\_GORGE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( 2 \cdot x + y \leq 2 \cdot sv - 2 \)

2. \( 2 \cdot x \leq y \)
   - \( sv \geq 5: (1, 2) \quad (2, 4) \)

3. \( x \neq 0 \lor y \neq 1 \)

4. \( sv = 12 \)

5. \( sv = 11 \)
\[ \text{NB}_\text{GORGE}(x, \text{VARIABLES}) \wedge \text{SUM}_\text{WIDTH}_\text{INCREASING}_\text{TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \times x + y \leq sv - 1 \]
   - \( sv \geq 7 \): \((1, sv - 3) \quad (2, sv - 5)\)

2. \[ y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2\]
   - \((sv - 1) \mod 2 = 1 \wedge sv \geq 6\): \(([(sv - 1)/2] - 1, 2) \quad ([(sv - 1)/2] - 1, 3)\)
   - \( y \neq 1\)

\[ \begin{array}{c}
\text{sv = 12} \\
\text{sv = 11}
\end{array} \]

\[ \begin{array}{c}
\text{nb_gorge} \\
\text{sum_width_increasing_terrace}
\end{array} \]

\[ \text{NB}_\text{GORGE}(x, \text{VARIABLES}) \wedge \text{SUM}_\text{WIDTH}_\text{INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \times x \leq y + 1 \]
   - \( sv \geq 5 \): \((1, 1) \quad (2, 3)\)

\[ \begin{array}{c}
\text{sv = 12} \\
\text{sv = 11}
\end{array} \]

\[ \begin{array}{c}
\text{nb_gorge} \\
\text{sum_width_inflexion}
\end{array} \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_GORGE}(x, \text{VARIABLES}) \land \sum\text{-\textsc{width\_peak}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y + 1 \)
   - \( sv \geq 5 \): (1, 0) (2, 1)
2. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 3 \): (0, \( sv - 2 \)) (1, \( sv - 3 \))

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_GORGE}(x, \text{VARIABLES}) \land \sum\text{-\textsc{width\_plain}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 - (sv - 1) \mod 2 \)
   - \((sv - 1) \mod 2 = 1 \land sv \geq 6 \): (0, \( sv - 2 \)) (1, \( sv - 3 \))
2. \( x \leq y \)
   - \( sv \geq 3 \): (0, 0) (1, 1)
3. \( x < \lfloor (sv - 1)/2 \rfloor \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 6 \): (0, \( sv - 2 \)) (1, \( sv - 3 \))
4. \( y \neq x + 1 \)

\[ sv = 12 \]

\[ sv = 11 \]
\[ \text{NB_GORGE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y + 1 \)
   - \( sv \geq 5 \): (1, 0) (2, 1)
2. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 3 \): (0, \( sv - 2 \)) (1, \( sv - 3 \))

---

\[ \text{NB_GORGE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_PROPER_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow 2 * x + y \leq sv - 2 \)
   - \( sv \geq 8 \): (1, \( sv - 4 \)) (2, \( sv - 6 \))
2. \( y > 0 \Rightarrow 2 * x \leq sv - 4 - (sv - 4) \mod 2 \)
   - \( sv \mod 2 = 1 \land sv \geq 5 \): (\( ((sv - 1)/2) \) - 2, 2) (\( ((sv - 1)/2) \) - 2, 3)
3. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{GORGE}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{PROPER}_\text{PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(4 \times x + y \leq 2 \times sv - 2\)
   - \(sv \mod 2 = 0 \land sv \geq 8\): \(\lceil (sv - 1)/2 \rceil, 2\) \(\lceil (sv - 1)/2 \rceil - 1, 6\)
   - \(sv \mod 2 = 1 \land sv \geq 11\): \(\lceil (sv - 1)/2 \rceil - 1, 4\) \(\lceil (sv - 1)/2 \rceil - 2, 8\)

2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 5\): \((0, sv - 2)\) \((1, sv - 3)\)

3. \(y \neq 1\)

\[ \text{NB}_\text{GORGE}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{STEADY}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(4 \times x + y \leq 2 \times sv - 2\)
   - \(sv \mod 2 = 0 \land sv \geq 8\): \(\lceil (sv - 1)/2 \rceil, 2\) \(\lceil (sv - 1)/2 \rceil - 1, 6\)
   - \(sv \mod 2 = 1 \land sv \geq 11\): \(\lceil (sv - 1)/2 \rceil - 1, 4\) \(\lceil (sv - 1)/2 \rceil - 2, 8\)

2. \(x + y \leq sv\)
   - \(sv \geq 8\): \((1, sv - 1)\) \((2, sv - 2)\)

3. \(y \neq 1\)
\( \text{NB}_\text{GORGE}(x, \text{VARIABLES}) \land \\
\text{SUM WIDTH STRICLY DECREASING SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \\
\begin{align*}
\text{1. } & 2x + 2y \leq 3sv - 2 - (3sv - 2) \mod 2 \\
\text{2. } & sv \mod 2 = 1 \land sv \geq 3: ([sv - 1]/2, sv - 1), ([sv - 1]/2 - 1, sv) \\
\text{3. } & 2x \leq y \\
\text{4. } & sv \geq 5: (1, 2), (2, 4) \\
\text{5. } & x \neq 0 \lor y \neq 1
\end{align*}
\)

\( \text{sv} = 12 \)

\( \text{sv} = 11 \)

\( \text{NB}_\text{GORGE}(x, \text{VARIABLES}) \land \\
\text{SUM WIDTH STRICTLY INCREASING SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \\
\begin{align*}
\text{1. } & 2x + y \leq 2sv - 2 \\
\text{2. } & 2x \leq y \\
\text{3. } & sv \geq 5: (1, 2), (2, 4) \\
\text{4. } & x \neq 0 \lor y \neq 1
\end{align*}
\)

\( \text{sv} = 12 \)

\( \text{sv} = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}_\text{GORGE}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{enumerate}
\item \(3 \cdot x \leq y + sv\)
\item \(sv > 1 \Rightarrow x + y \leq sv - 2\)
\end{enumerate}

\begin{itemize}
\item \(sv \geq 3\): \((0, sv - 2)\) \quad \((1, sv - 3)\)
\end{itemize}

\(sv = 12\)

\begin{figure}
\centering
\includegraphics[width=0.4\textwidth]{figure1}
\end{figure}

\(sv = 11\)

\begin{figure}
\centering
\includegraphics[width=0.4\textwidth]{figure2}
\end{figure}

\[
\text{NB}_\text{GORGE}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{enumerate}
\item \(x + y \leq sv - 1\)
\item \(x \leq y\)
\item \(x = 0 \lor y = 1\)
\end{enumerate}

\begin{itemize}
\item \(sv \geq 5\): \((1, sv - 2)\) \quad \((2, sv - 3)\)
\item \(sv \geq 3\): \((0, 0)\) \quad \((1, 1)\)
\end{itemize}

\(sv = 12\)

\begin{figure}
\centering
\includegraphics[width=0.4\textwidth]{figure3}
\end{figure}

\(sv = 11\)

\begin{figure}
\centering
\includegraphics[width=0.4\textwidth]{figure4}
\end{figure}
\[ \text{NB}_\text{GORGE}(x, \text{VARIABLES}) \land \text{SUM WIDTH}_\text{ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq 3 \times x \)
2. \( sv \geq 5: (0, 0) \quad (1, 1) \)
3. \( 3 \times y \leq 4 \times x + sv \)
4. \( y = 0 \Rightarrow 3 \times x \leq sv - sv \mod 3 \)
5. \( x > 0 \land y > 0 \Rightarrow 3 \times x \leq y + sv - 2 \)
6. \( y > 0 \Rightarrow 6 \times x \leq y + 2 \times sv - 1 \)

\[
\begin{align*}
\text{sv mod } 2 = 0 \land sv \geq 14: \left(\left\lfloor (sv - 1)/2\right\rfloor, sv - 5\right) & \quad \left(\left\lfloor (sv - 1)/2\right\rfloor - 1, sv - 11\right) \\
\text{sv mod } 2 = 1 \land sv \geq 17: \left(\left\lfloor (sv - 1)/2\right\rfloor - 1, sv - 8\right) & \quad \left(\left\lfloor (sv - 1)/2\right\rfloor - 2, sv - 14\right)
\end{align*}
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\textbf{NB\_INCREASING}(x,\text{VARIABLES}) \land \\
\text{NB\_INCREASING\_SEQUENCE}(y,\text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv \)
   \- \( sv \geq 6: (sv - 2, 2) \quad (sv - 3, 3) \)

2. \( y \leq x \)
   \- \( sv \geq 4: (1, 1) \quad (2, 2) \)

3. \( sv > 1 \Rightarrow x \leq y + sv - 2 \)

4. \( x > 0 \Rightarrow y \geq 1 \)
   \- \( sv \geq 4: (sv - 2, 1) \quad (sv - 3, 1) \)

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{diagram1}
\end{figure}

\textbf{NB\_INCREASING}(x,\text{VARIABLES}) \land \\
\text{NB\_INCREASING\_TERRACE}(y,\text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv - 1 \)
   \- \( sv \geq 6: (sv - 2, 1) \quad (sv - 3, 2) \)

2. \( x = 0 \Rightarrow y = 0 \)

3. \( x > 0 \land y > 0 \Rightarrow y \leq x - 1 \)
   \- \( sv \geq 6: (2, 1) \quad (3, 2) \)

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{diagram2}
\end{figure}
\[ \text{NB\_INCREASING}(x, \text{VARIABLES}) \land \text{NB\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \cdot x + y \leq 2 \cdot sv - 2 \]
   - \[ sv \geq 6: (sv - 2, 2) \quad (sv - 3, 4) \]
2. \[ y \leq 2 \cdot x \]
   - \[ sv \geq 6: (1, 2) \quad (2, 4) \]

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_INCREASING}(x, \text{VARIABLES}) \land \text{NB\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x + y \leq sv - 1 \]
   - \[ sv \geq 5: (sv - 2, 1) \quad (sv - 3, 2) \]
2. \[ y \leq x \]
   - \[ sv \geq 5: (1, 1) \quad (2, 2) \]

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\texttt{NB\_INCREASING}(x, VARIABLES) \land \\
\texttt{NB\_PLAIN}(y, VARIABLES) \text{ with } sv = |VARIABLES|

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 5: (sv - 2, 1) \ (sv - 3, 2) \)
2. \( y \leq x \)
   - \( sv \geq 5: (1, 1) \ (2, 2) \)

\texttt{NB\_INCREASING}(x, VARIABLES) \land \\
\texttt{NB\_PLATEAU}(y, VARIABLES) \text{ with } sv = |VARIABLES|

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 5: (sv - 2, 1) \ (sv - 3, 2) \)
2. \( y \leq x \)
   - \( sv \geq 5: (1, 1) \ (2, 2) \)
\[ \text{NB}_\text{INCREASING}(x, \text{VARIABLES}) \land \\
\text{NB}_\text{PROPER}_\text{PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x + 2 \cdot y \leq sv - 1 \]
   - \( sv \geq 7 \): \((sv - 3, 1)\) \((sv - 5, 2)\)
2. \[ y \leq x \]
   - \( sv \geq 7 \): \((1, 1)\) \((2, 2)\)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB}_\text{INCREASING}(x, \text{VARIABLES}) \land \\
\text{NB}_\text{PROPER}_\text{PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x + 2 \cdot y \leq sv - 1 \]
   - \( sv \geq 7 \): \((sv - 3, 1)\) \((sv - 5, 2)\)
2. \[ y \leq x \]
   - \( sv \geq 7 \): \((1, 1)\) \((2, 2)\)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{INCREASING}(x, \text{VARIABLES}) \land \]
\[ \text{NB}_\text{STEADY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 3: (sv - 2, 1) \ (sv - 3, 2) \)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB}_\text{INCREASING}(x, \text{VARIABLES}) \land \]
\[ \text{NB}_\text{STEADY}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 4: (sv - 2, 1) \ (sv - 3, 2) \)

\[ sv = 12 \]

\[ sv = 11 \]
\( \text{NB\_INCREASING}(x, \text{VARIABLES}) \land \text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 4: (sv - 2, 1) \quad (sv - 3, 2) \)

\( sv = 12 \)

\( sv = 11 \)

\( \text{NB\_INCREASING}(x, \text{VARIABLES}) \land \text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x + y \leq sv \)
   - \( sv \geq 0: (sv - 2, 2) \quad (sv - 3, 3) \)
2. \( y \leq x \)
   - \( sv \geq 4: (1, 1) \quad (2, 2) \)
3. \( sv > 1 \Rightarrow x \leq y + sv - 2 \)
4. \( x > 0 \Rightarrow y \geq 1 \)
   - \( sv \geq 4: (sv - 2, 1) \quad (sv - 3, 1) \)

\( sv = 12 \)

\( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_INCREASING}(x, \text{VARIABLES}) \land \\
\text{NB\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x + y \leq sv - 1\)
   - \(sv \geq 5\): \((sv - 2, 1)\) \((sv - 3, 2)\)
2. \(y \leq x\)
   - \(sv \geq 5\): \((1, 1)\) \((2, 2)\)

\[
\text{NB\_INCREASING}(x, \text{VARIABLES}) \land \\
\text{NB\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x + y \leq sv - 1\)
   - \(sv \geq 5\): \((sv - 2, 1)\) \((sv - 3, 2)\)
2. \(y \leq x\)
   - \(sv \geq 5\): \((1, 1)\) \((2, 2)\)
\(\text{NB}_{-}\text{INCREASING}(x, \text{VARIABLES}) \land \text{NB}_{-}\text{ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\)

1. \(x + y \leq sv - 1\)
2. \(sv \geq 7: (sv - 2, 1) \quad (sv - 3, 2)\)
3. \(y \leq x\)
4. \(sv \geq 7: (1, 1) \quad (2, 2)\)
5. \(y \neq \max(0, \lfloor (sv - 1)/3 \rfloor) - 0,\)
6. \(x < \max(0, \lfloor (sv - 1)/3 \rfloor) + 1,\)
7. \(x > 1 \cdot \max(0, sv - 1) - \max(0, \lfloor (sv - 1)/3 \rfloor) - 1,\)
8. \(1 \neq sv \mod 3\)

\(sv = 12\)

\(sv = 11\)

\(\text{NB}_{-}\text{INCREASING}(x, \text{VARIABLES}) \land \text{SUM}_{-}\text{WIDTH}_{-}\text{DECREASING}_{-}\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\)

1. \(2 \cdot x + y \leq 2 \cdot sv - 2\)
2. \(sv \geq 4: (sv - 2, 2) \quad (sv - 3, 4)\)
3. \(y \neq 1\)

\(sv = 12\)

\(sv = 11\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\( \text{NB\_INCREASING}(x, \text{VARIABLES}) \land \)
\( \text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

\begin{enumerate}
\item \( x > 0 \land y > 0 \Rightarrow x + y \leq sv - 2 \)
\item \( sv \geq 6: (sv-4, 2) \quad (1, sv-3) \)
\item \( y > 0 \Rightarrow x \leq sv - 4 \)
\item \( y \neq 1 \)
\end{enumerate}

\begin{enumerate}
\item \( x > 0 \land y > 0 \Rightarrow x \leq sv - 2 \)
\item \( sv \geq 4: (sv-2, 1) \quad (sv-2, 2) \)
\item \( y > 0 \Rightarrow x \geq 1 \)
\item \( sv \geq 3: (1, sv-2) \quad (1, sv-3) \)
\end{enumerate}
\( \text{NB}_\text{INCREASING}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{INCREASING}_\text{SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x = 0 \Rightarrow y = 0 \)
2. \( y = 0 \Rightarrow x = 0 \)
3. \( sv > 1 \Rightarrow 2 \cdot x \leq y + sv - 2 \)
4. \( sv > 1 \Rightarrow y \leq 2 \cdot x + sv - 2 \)
5. \( x > 0 \land y > 0 \Rightarrow x \leq y - 1 \)
6. \( y > 0 \Rightarrow x \geq 1 \)
7. \( x \neq 1 \lor y < 3 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[(\text{NB\_INCREASING}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|)\]

1. \(2 \cdot x + y \leq 2 \cdot sv - 2\)
2. \(sv \geq 6: (sv - 2, 2) (sv - 3, 4)\)
3. \(x = 0 \Rightarrow y = 0\)
4. \(x > 0 \land y > 0 \Rightarrow x \geq 2\)
5. \(sv \geq 5: (2, sv - 2) (2, sv - 3)\)
6. \(x \neq 1 \lor y < 1\)
7. \(y \neq 1\)

\[sv = 12\]

\[sv = 11\]
\text{NB\_INCREASING}(x,\text{VARIABLES}) \land \text{SUM\_WIDTH\_INFLEXION}(y,\text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. $x > 0 \land y > 0 \Rightarrow x \leq sv - 2$
   - $sv \geq 3$: $(sv - 2, sv - 2)$, $(sv - 2, sv - 3)$

2. $y > 0 \Rightarrow x \geq 1$
   - $sv \geq 3$: $(1, sv - 2)$, $(1, sv - 3)$

\text{NB\_INCREASING}(x,\text{VARIABLES}) \land \text{SUM\_WIDTH\_PEAK}(y,\text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. $x > 0 \land y > 0 \Rightarrow x \leq sv - 2$
   - $sv \geq 3$: $(sv - 2, sv - 2)$, $(sv - 2, sv - 3)$

2. $y > 0 \Rightarrow x \geq 1$
   - $sv \geq 4$: $(1, sv - 2)$, $(1, sv - 3)$
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB} \_ \text{INCREASING}(x, \text{VARIABLES}) \land \text{SUM} \_ \text{WIDTH} \_ \text{PLAIN}(y, \text{VARIABLES}) \text{ with } \text{sv} = |\text{VARIABLES}| \]

1. \( x + y \leq \text{sv} - 1 \)
   - \( \text{sv} \geq 4: (\text{sv} - 2, 1) \quad (\text{sv} - 3, 2) \)
2. \( y > 0 \Rightarrow x \geq 1 \)
   - \( \text{sv} \geq 3: (1, \text{sv} - 2) \quad (1, \text{sv} - 3) \)

\[ \text{NB} \_ \text{INCREASING}(x, \text{VARIABLES}) \land \text{SUM} \_ \text{WIDTH} \_ \text{PLATEAU}(y, \text{VARIABLES}) \text{ with } \text{sv} = |\text{VARIABLES}| \]

1. \( x + y \leq \text{sv} - 1 \)
   - \( \text{sv} \geq 4: (\text{sv} - 2, 1) \quad (\text{sv} - 3, 2) \)
2. \( y > 0 \Rightarrow x \geq 1 \)
   - \( \text{sv} \geq 3: (1, \text{sv} - 2) \quad (1, \text{sv} - 3) \)
\[ \text{NB}_\text{INCREASING}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{PROPER}_\text{PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
2. \( sv \geq 5: (sv - 3, 2) \quad (sv - 4, 3) \)
3. \( y > 0 \Rightarrow x \leq sv - 3 \)
4. \( y > 0 \Rightarrow x \geq 1 \)
5. \( sv \geq 5: (1, sv - 2) \quad (1, sv - 3) \)
6. \( y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{NB\textunderscore INCREASING}(x, \text{VARIABLES}) \land \\
\text{SUM\textunderscore WIDTH\textunderscore PROPER\textunderscore PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. \( x + y \leq sv - 1 \)
2. \( sv \geq 5: \ (sv - 3, 2) \ (sv - 4, 3) \)
3. \( y > 0 \Rightarrow x \leq sv - 3 \)
4. \( y > 0 \Rightarrow x \geq 1 \)
5. \( sv \geq 5: \ (1, sv - 2) \ (1, sv - 3) \)
6. \( y \neq 1 \)

\begin{align*}
\text{NB\textunderscore INCREASING}(x, \text{VARIABLES}) \land \\
\text{SUM\textunderscore WIDTH\textunderscore STEADY\textunderscore SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. \( 2 * x + y \leq 2 * sv - 2 \)
2. \( sv \geq 4: \ (sv - 2, 2) \ (sv - 3, 4) \)
3. \( y \neq 1 \)
\( \text{NB\_INCREASING}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x + y \leq 2 \times sv - 2 \)
2. \( sv \geq 4: (sv - 2, 2) \) \((sv - 3, 4)\)
3. \( y \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)

\( \text{NB\_INCREASING}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( y \leq 2 \times x \)
2. \( sv \geq 4: (1, 2) \) \((2, 4)\)
3. \( x = 0 \Rightarrow y = 0 \)
4. \( y = 0 \Rightarrow x = 0 \)
5. \( sv > 1 \Rightarrow 2 \times x \leq y + sv - 2 \)
6. \( x > 0 \land y > 0 \Rightarrow x \leq y - 1 \)
7. \( sv \geq 4: (sv - 2, sv - 1) \) \((sv - 3, sv - 2)\)

\( sv = 12 \)

\( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_INCREASING}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 2 \)
   - \( sv \geq 3: (sv - 2, sv - 2) \) \( (sv - 2, sv - 3) \)
2. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 5: (1, sv - 2) \) \( (1, sv - 3) \)

\[ \text{NB\_INCREASING}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 2 \)
   - \( sv \geq 4: (sv - 2, 1) \) \( (sv - 2, 2) \)
2. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 3: (1, sv - 2) \) \( (1, sv - 3) \)
\textbf{NB\_INCREASING}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \(2 \cdot x + y \leq 2 \cdot sv - 2\)
   - \(sv \geq 6: (sv - 2, 2) \quad (sv - 3, 4)\)
2. \(y \leq 2 \cdot x\)
   - \(sv \geq 6: (1, 2) \quad (2, 4)\)
3. \(y \neq 1\)

\begin{align*}
\text{sv} = 12 & \quad \text{sv} = 11
\end{align*}

\textbf{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \(2 \cdot x + 2 \cdot y \leq sv - sv \mod 2\)
   - \(sv \mod 2 = 0 \land \text{sv} \geq 4: \lfloor sv/2 \rfloor, 0 \quad \lfloor sv/2 \rfloor - 1, 1\)
   - \(sv \mod 2 = 1 \land \text{sv} \geq 5: \lfloor sv/2 \rfloor, 0 \quad \lfloor sv/2 \rfloor - 1, 1\)
2. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 4: (1, \lfloor (sv - 2)/2 \rfloor) \quad (1, \lfloor (sv - 2)/2 \rfloor - 1)\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \]
\[ \text{NB\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \times x \leq y + 2 \]
   - \( sv \geq 4: \ (1, 0) \quad (2, 2) \]
2. \[ y \leq 2 \times x \]
   - \( sv \geq 6: \ (1, 2) \quad (2, 4) \]

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \]
\[ \text{NB\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x \leq y + 1 \]
   - \( sv \geq 4: \ (1, 0) \quad (2, 1) \]
2. \[ y \leq x \]
   - \( sv \geq 5: \ (1, 1) \quad (2, 2) \]

\[ sv = 12 \]

\[ sv = 11 \]
\begin{align*}
\text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. \( x \leq y + 1 \)  \\
\quad \checkmark \text{sv} \geq 4: (1, 0) \quad (2, 1)

2. \( y \leq x \)  \\
\quad \checkmark \text{sv} \geq 5: (1, 1) \quad (2, 2)

\begin{align*}
\text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. \( x \leq y + 1 \)  \\
\quad \checkmark \text{sv} \geq 4: (1, 0) \quad (2, 1)

2. \( y \leq x \)  \\
\quad \checkmark \text{sv} \geq 5: (1, 1) \quad (2, 2)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB}_\text{PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \ast x + y \leq sv\)
   - \(sv \mod 2 = 0 \land sv \geq 8: ([sv/2], 0) \quad ([sv/2] - 1, 2)\)
   - \(sv \mod 2 = 1 \land sv \geq 11: ([sv/2], 1) \quad ([sv/2] - 1, 3)\)

2. \(y \leq x\)
   - \(sv \geq 7: (1, 1) \quad (2, 2)\)

\[ \text{NB}_\text{INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB}_\text{PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \ast x + y \leq sv\)
   - \(sv \mod 2 = 0 \land sv \geq 8: ([sv/2], 0) \quad ([sv/2] - 1, 2)\)
   - \(sv \mod 2 = 1 \land sv \geq 11: ([sv/2], 1) \quad ([sv/2] - 1, 3)\)

2. \(y \leq x\)
   - \(sv \geq 7: (1, 1) \quad (2, 2)\)
\[ \text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_STEADY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

- \( 2 \cdot x + y \leq sv \)
  - If \( sv \mod 2 = 0 \) and \( sv \geq 4 \):
    - \( sv = 12 \): \( (1, sv - 2) \) and \( (2, sv - 4) \)
  - If \( sv \mod 2 = 1 \) and \( sv \geq 7 \):
    - \( sv = 12 \): \( (1, \lfloor sv/2 \rfloor, 0) \) and \( (2, \lfloor sv/2 \rfloor - 2, 2) \)
    - \( sv = 11 \): \( (1, \lfloor sv/2 \rfloor, 1) \) and \( (2, \lfloor sv/2 \rfloor - 2, 3) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_{\text{INCREASING\_SEQUENCE}}(x, \text{VARIABLES}) \land \text{NB}_{\text{STRICTLY\_DECREASING\_SEQUENCE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 2 \): \((\lfloor sv/2 \rfloor, \lfloor sv/2 \rfloor - 1)\) \((\lfloor sv/2 \rfloor - 1, \lfloor sv/2 \rfloor)\)

2. \( x \leq y + 1 \)
   - \( sv \geq 4 \): \((1, 0)\) \((2, 1)\)

---

\[ \text{NB}_{\text{INCREASING\_SEQUENCE}}(x, \text{VARIABLES}) \land \text{NB}_{\text{STRICTLY\_INCREASING\_SEQUENCE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \geq 4 \): \((1, 1)\) \((2, 2)\)

2. \( sv > 1 \Rightarrow 2 \times y \leq 2 \times x + sv - 2 - (sv - 2) \mod 2 \)

3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 4 \): \((1, \lfloor sv/2 \rfloor)\) \((1, \lfloor sv/2 \rfloor - 1)\)
\[ \text{NB-INCREASING-SEQUENCE}(x, \text{VARIABLES}) \land \text{NB-SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 3 \times x \leq y + sv + 1 \]
   - \( sv \mod 2 = 0 \land sv \geq 8: \left(\left\lfloor \frac{sv}{2} \right\rfloor, \left\lfloor \frac{(sv - 1)}{2} \right\rfloor\right) \]
   - \( sv \mod 2 = 1 \land sv \geq 11: \left(\left\lfloor \frac{sv}{2} \right\rfloor, \left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 2\right) \]

2. \( y \leq x \)
   - \( sv \geq 5: (1, 1) \) (2, 2)

\[ \text{NB-INCREASING-SEQUENCE}(x, \text{VARIABLES}) \land \text{NB-VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y + 1 \)
   - \( sv \geq 4: (1, 0) \) (2, 1)

2. \( y \leq x \)
   - \( sv \geq 5: (1, 1) \) (2, 2)
5. A DATABASE OF PARAMETERISED INVARIANTS

**NB** \_ INCREASING \_ SEQUENCE \((x, \text{VARIABLES})\) \& **NB** \_ ZIGZAG \((y, \text{VARIABLES})\) with \(sv = |\text{VARIABLES}|\)

1. \(2 \cdot x + y \leq sv + 1\)
   - \(sv \mod 2 = 0 \land sv \geq 10\): \((\lfloor sv/2 \rfloor, 1)\) \((\lfloor sv/2 \rfloor - 1, 3)\)
   - \(sv \mod 2 = 1 \land sv \geq 13\): \((\lfloor sv/2 \rfloor, 2)\) \((\lfloor sv/2 \rfloor - 1, 4)\)

2. \(y \leq x\)
   - \(sv \geq 7\): \((1, 1)\) \((2, 2)\)

3. \(y = 0 \Rightarrow 3 \cdot x \leq sv + 1 - (sv + 1) \mod 3\)

\[sv = 12\]

\[sv = 11\]

**NB** \_ INCREASING \_ SEQUENCE \((x, \text{VARIABLES})\) \& **SUM** \_ WIDTH \_ DECREASING \_ SEQUENCE \((y, \text{VARIABLES})\) with \(sv = |\text{VARIABLES}|\)

1. \(2 \cdot x + y \leq 2 \cdot sv - 2\)
   - \(sv \mod 2 = 0 \land sv \geq 2\): \((\lfloor sv/2 \rfloor, sv - 2)\) \((\lfloor sv/2 \rfloor - 1, sv)\)

2. \(2 \cdot x \leq y + 2\)
   - \(sv \geq 4\): \((1, 0)\) \((2, 2)\)

3. \(y \neq 1\)

\[sv = 12\]

\[sv = 11\]
\[ \text{NB}\_\text{INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \times x + y \leq sv \]
   - \[ sv \geq 8: (2, sv - 4) \ (3, sv - 6) \]
2. \[ x > 0 \land y > 0 \Rightarrow x + y \leq sv - 2 \]
   - \[ sv \geq 6: (1, sv - 3) \ (2, sv - 4) \]
3. \[ y > 0 \Rightarrow 2 \times x \leq sv - 2 - (sv - 2) \mod 2 \]
   - \[ sv \mod 2 = 1 \land sv \geq 7: (\lfloor sv/2 \rfloor - 1, 2) \ (\lfloor sv/2 \rfloor - 1, 3) \]
4. \[ y \neq 1 \]

\[ \text{NB}\_\text{INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 3 \times x \leq y + sv + 1 \]
2. \[ x + y \leq sv - 1 \]
   - \[ sv \geq 4: (1, sv - 2) \ (2, sv - 3) \]
3. \[ y > 0 \Rightarrow x \geq 1 \]
   - \[ sv \geq 3: (1, sv - 2) \ (1, sv - 3) \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x \leq y\)
   - \(sv \geq 4\): (1, 2) (2, 4)
2. \(sv > 1 \Rightarrow y \leq 2 \times x + sv - 2\)
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 4\): (1, sv) (1, sv - 1)

\[ \text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x + y \leq sv\)
   - \(sv \geq 6\): (1, sv - 2) (2, sv - 4)
2. \(y > 0 \Rightarrow 2 \times x \leq sv - 2 - (sv - 2) \mod 2\)
   - \(sv \mod 2 = 1 \land sv \geq 5\): ([sv/2] - 1, 2) ([sv/2] - 1, 3)
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 5\): (1, sv - 2) (1, sv - 3)
4. \(y \neq 1\)
\text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( 2 \times x \leq y + 2 \)
   - \( sv \geq 4 \): (1, 0) \quad (2, 2)
2. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 3 \): (1, \(sv - 2\)) \quad (1, \(sv - 3\))

\begin{figure}[h]
\centering
\includegraphics[width=0.4\textwidth]{diagram1}
\includegraphics[width=0.4\textwidth]{diagram2}
\end{figure}

\text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 4 \): (1, \(sv - 2\)) \quad (2, \(sv - 3\))
2. \( x \leq y + 1 \)
   - \( sv \geq 4 \): (1, 0) \quad (2, 1)
3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 3 \): (1, \(sv - 2\)) \quad (1, \(sv - 3\))

\begin{figure}[h]
\centering
\includegraphics[width=0.4\textwidth]{diagram3}
\includegraphics[width=0.4\textwidth]{diagram4}
\end{figure}
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 4: (1, sv - 2), (2, sv - 3) \)
2. \( x \leq y + 1 \)
   - \( sv \geq 4: (1, 0), (2, 1) \)
3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 3: (1, sv - 2), (1, sv - 3) \)

\[
\text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 4: (1, sv - 2), (2, sv - 3) \)
2. \( x \leq y + 1 \)
   - \( sv \geq 4: (1, 0), (2, 1) \)
3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 3: (1, sv - 2), (1, sv - 3) \)
\text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( 4 \times x + y \leq 2 \times sv \)
   - \( sv \mod 2 = 0 \land sv \geq 8: (\lfloor sv/2 \rfloor, 0) \ (\lfloor sv/2 \rfloor - 1, 4) \)
   - \( sv \mod 2 = 1 \land sv \geq 11: (\lfloor sv/2 \rfloor, 2) \ (\lfloor sv/2 \rfloor - 1, 6) \)

2. \( x + y \leq sv - 1 \)
   - \( sv \geq 5: (1, sv - 2) \ (2, sv - 3) \)

3. \( y > 0 \Rightarrow x \geq 1 \)
4. \( sv \geq 5: (1, sv - 2) \ (1, sv - 3) \)
      \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \[4 \times x + y \leq 2 \times sv\]
   - \(sv \mod 2 = 0 \land sv \geq 8\): (\([sv/2], 0\) \(\land\) \([sv/2] - 1, 4\))
   - \(sv \mod 2 = 1 \land sv \geq 11\): (\([sv/2], 2\) \(\land\) \([sv/2] - 1, 6\))

2. \(x + y \leq sv - 1\)
   - \(sv \geq 5\): (1, \(sv - 2\)) \(\land\) (2, \(sv - 3\))

3. \(y > 0 \implies x \geq 1\)
   - \(sv \geq 5\): (1, \(sv - 2\)) \(\land\) (1, \(sv - 3\))

4. \(y \neq 1\)

\[
\text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \[4 \times x + y \leq 2 \times sv\]
   - \(sv \mod 2 = 0 \land sv \geq 4\): (\([sv/2], 0\) \(\land\) \([sv/2] - 1, 4\))
   - \(sv \mod 2 = 1 \land sv \geq 7\): (\([sv/2], 2\) \(\land\) \([sv/2] - 1, 6\))

2. \(y \neq 1\)
\[
\begin{aligned}
\text{NB}_\text{INCREASING}_\text{SEQUENCE}(x, \text{VARIABLES}) \wedge \\
\text{SUM}_\text{WIDTH}_\text{STRITLY}_\text{DECREASING}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{aligned}
\]

1. \(2 * x + y \leq 2 * sv - 2\)
   - \(sv \mod 2 = 0 \land sv \geq 2\): (\(\lfloor sv/2 \rfloor, sv - 2\)) \((\lfloor sv/2 \rfloor - 1, sv)\)
2. \(2 * x \leq y + 2\)
   - \(sv \geq 4\): (1, 0) (2, 2)
3. \(y \neq 1\)

\[
\begin{aligned}
\text{NB}_\text{INCREASING}_\text{SEQUENCE}(x, \text{VARIABLES}) \wedge \\
\text{SUM}_\text{WIDTH}_\text{STRITLY}_\text{INCREASING}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{aligned}
\]

1. \(2 * x \leq y\)
   - \(sv \geq 4\): (1, 2) (2, 4)
2. \(sv > 1 \Rightarrow y \leq 2 * x + sv - 2\)
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 4\): (1, \(sv - 1\)) (1, \(sv - 2\))
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{enumerate}
\item \(3 \cdot x \leq y + sv + 1\)
\item \(x + y \leq sv - 1\)
\item \(sv \geq 4\): \((1, sv - 2), (2, sv - 3)\)
\item \(y > 0 \Rightarrow x \geq 1\)
\item \(sv \geq 3\): \((1, sv - 2), (1, sv - 3)\)
\end{enumerate}

\[
\text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{enumerate}
\item \(x + y \leq sv - 1\)
\item \(sv \geq 4\): \((1, sv - 2), (2, sv - 3)\)
\item \(x \leq y + 1\)
\item \(sv \geq 4\): \((1, 0), (2, 1)\)
\item \(y > 0 \Rightarrow x \geq 1\)
\item \(sv \geq 3\): \((1, sv - 2), (1, sv - 3)\)
\end{enumerate}
\[ \text{NB\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq 2 \times x \)
   - \( sv \geq 6 \): (1, 2) (2, 4)
2. \( y = 0 \Rightarrow 3 \times x \leq sv + 1 - (sv + 1) \mod 3 \)
3. \( x > 0 \land y > 0 \Rightarrow 3 \times x \leq y + sv \)
4. \( y > 0 \Rightarrow 6 \times x \leq y + 2 \times sv + 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 8 \): \((\lfloor sv/2 \rfloor, sv - 2)\) \((\lfloor sv/2 \rfloor - 1, sv - 8)\)
   - \( sv \mod 2 = 1 \land sv \geq 11 \): \((\lfloor sv/2 \rfloor, sv - 5)\) \((\lfloor sv/2 \rfloor - 1, sv - 11)\)
5. \( y \neq 1 \)

\[ \text{NB\_INCREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{NB\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( sv > 1 \Rightarrow 2 \times x + y \leq sv - 2 \)
   - \( sv \geq 4 \): (0, sv - 2) (1, sv - 4)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_INCREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{NB\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \times x + 2 \times y \leq sv - 1 - (sv - 1) \mod 2\)
   \(\begin{array}{ll}
   \text{if } (sv - 1) \mod 2 = 0 \land sv \geq 5: & ((sv - 2)/2, 1) \\
   \text{if } (sv - 1) \mod 2 = 1 \land sv \geq 4: & ((sv - 2)/2, 0)
   \end{array}\)

\[
sv = 12 \\
sv = 11
\]
\text{NB\_INCREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{NB\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. $2 \times x + 2 \times y \leq sv - 1 - (sv - 1) \mod 2$
2. $(sv - 1) \mod 2 = 0 \land sv \geq 5$: $((sv - 2)/2, 1)$
3. $(sv - 1) \mod 2 = 1 \land sv \geq 4$: $((sv - 2)/2, 0)$

$sv = 12$

$sv = 11$

\text{NB\_INCREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{NB\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. $2 \times x + 3 \times y \leq sv - 1$
2. $sv \mod 2 = 0 \land sv \geq 10$: $((sv - 2)/2 - 1, 1)$
3. $x \geq y$
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_{\text{INCREASING\_TERRACE}}(x, \text{VARIABLES}) \land \text{NB}_{\text{PROPER\_PLATEAU}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \cdot x + 3 \cdot y \leq sv - 1\)
2. \(sv \mod 2 = 0 \land sv \geq 10: (\lceil(sv - 2) / 2\rceil - 1, 1) \land (\lceil(sv - 2) / 2\rceil - 4, 3)\)
3. \(sv \geq 10: (\lfloor(sv - 2) / 2\rfloor - 1, 1) \land (\lfloor(sv - 2) / 2\rfloor - 4, 3)\)

\[ \text{NB}_{\text{INCREASING\_TERRACE}}(x, \text{VARIABLES}) \land \text{NB}_{\text{STEADY}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(x \leq y\)
2. \(sv \geq 6: (1, 1) \land (2, 2)\)
3. \(x > 0 \land y > 0 \Rightarrow x + y \leq sv - 2\)
4. \(sv \geq 6: (1, sv - 3) \land (2, sv - 4)\)
\( \text{NB\_INCREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{NB\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \) with \( \text{sv} = |\text{VARIABLES}| \)

\begin{enumerate}
\item \( x \leq y \)
\item \( x > 0 \land y > 0 \Rightarrow x + y \leq \text{sv} - 2 \)
\end{enumerate}

\( \text{sv} \mod 2 = 0 \land \text{sv} \geq 6: ( (\text{sv} - 2)/2, \lfloor \text{sv}/2 \rfloor - 1 ) \land ( (\text{sv} - 2)/2 - 1, \lfloor \text{sv}/2 \rfloor ) \)

\( \text{sv} \mod 2 = 1 \land \text{sv} \geq 5: ( (\text{sv} - 2)/2, 1 ) \land ( (\text{sv} - 2)/2 - 1, 2 ) \)

\( 2 \ast x + y \leq sv - \text{sv} \mod 2 \)

\( \text{sv} \mod 2 = 0 \land \text{sv} \geq 6: ( (\text{sv} - 2)/2 - 1, 2 ) \land ( (\text{sv} - 2)/2 - 2, 3 ) \)

\( \text{sv} \mod 2 = 1 \land \text{sv} \geq 5: ( (\text{sv} - 2)/2, 1 ) \land ( (\text{sv} - 2)/2 - 1, 2 ) \)

\( x > 0 \land y > 0 \Rightarrow 2 \ast x + y \leq \text{sv} - 2 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_INCREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( y = 0 \Rightarrow x = 0 \)
2. \( x > 0 \land y > 0 \Rightarrow x \leq y - 1 \)

\[ sv \geq 6: (1, 2) \quad (2, 3) \]

\[
\text{NB\_INCREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{NB\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( 2 * x + 2 * y \leq sv - 1 - (sv - 1) \mod 2 \)
   - \( (sv - 1) \mod 2 = 0 \land sv \geq 5: ((sv - 2)/2, 1) \quad ((sv - 2)/2 - 1, 2) \)
   - \( (sv - 1) \mod 2 = 1 \land sv \geq 4: ((sv - 2)/2, 0) \quad ((sv - 2)/2 - 1, 1) \)

\[ sv = 12 \quad sv = 11 \]
\[ \text{NB}\_\text{INCREASING}\_\text{TERRACE}(x, \text{VARIABLES}) \land \\
\text{NB}\_\text{VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \times x + 2 \times y \leq sv - 1 - (sv - 1) \mod 2 \]
   - \((sv - 1) \mod 2 = 0 \land sv \geq 5\): \(((sv - 2)/2, 1)\)
   - \((sv - 1) \mod 2 = 1 \land sv \geq 4\): \(((sv - 2)/2, 0)\)

\[ sv = 12 \]

\[ sv = 11 \]

2. \[ sv > 1 \Rightarrow 2 \times x + 2 \times y \leq sv - 2 - (sv - 2) \mod 2 \]
   - \((sv - 2) \mod 2 = 0 \land sv \geq 4\): \(((sv - 2)/2, 0)\)
   - \((sv - 2) \mod 2 = 1 \land sv \geq 5\): \(((sv - 2)/2, 0)\)

\[ sv = 12 \]

\[ sv = 11 \]
A DATABASE OF PARAMETERISED INVARIANTS

\textbf{NB}_{\text{INCREASING\_TERRACE}}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \(2 \cdot x + y \leq sv\)
   \[ sv \geq 6: (0, sv) \land (1, sv - 2) \]
2. \(x = \max(0, \lfloor (sv - 2)/2 \rfloor) \land sv \mod 2 = 1 \land sv > 3 \Rightarrow y \leq 2\)
3. \(x > 0 \land y > 0 \Rightarrow 2 \cdot x \leq sv - 3 - (sv - 3) \mod 2\)
   \[ (sv - 1) \mod 2 = 1 \land sv \geq 4: (\lfloor (sv - 2)/2 \rfloor - 1, 2) \land (\lfloor (sv - 2)/2 \rfloor - 1, 3) \]
4. \(y \neq 1\)
\[ \text{NB\_INCREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( sv > 1 \Rightarrow 2 \cdot x + y \leq sv - 2 \)
2. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq sv - 3 \)
3. \( sv \mod 2 = 0 \land sv \geq 4: \{(sv - 2)/2\}, 0\rangle \quad (0, sv - 2) \)
4. \( sv > 9: \,(1, sv - 5)\rangle \quad (2, sv - 7) \)
5. \( x \neq 2 \lor y \neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 4 \lor sv < 7 \)
6. \( x > (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 2, \quad \begin{align*}
1 &= sv \mod 2, \\
1 &= y \mod 2 \\
y &= (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 2, \quad x < 1, \\
1 &= sv \mod 2, \\
0 &= x \mod 2 \\
x &= \max(0, [(sv - 2)/2]) - 1, \\
y &= (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 2, \\
l &= sv \mod 2
\end{align*} \)
8. \( y \neq 1 \)

\( sv = 12 \)  
\( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
&\text{NB}_{\text{INCREASING TERRACE}}(x, \text{VARIABLES}) \land \\
&\text{SUM}_{\text{WIDTH GORGE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \\
&\begin{cases}
1. x > 0 \land y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2 \\
 & (sv - 1) \mod 2 = 1 \land sv \geq 4: (((sv - 2) / 2) - 1, sv - 2) \\
 & ((sv - 2) / 2) - 1, sv - 3) \\
2. x > 0 \land y > 0 \Rightarrow 2 \times x \leq y + sv - 5 \\
 & x \neq \max(0, \lfloor (sv - 2) / 2 \rfloor) - 1, \\
 & y < 2, \\
 & y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 2, \\
 & 1 = sv \mod 2, \\
 & 1 = y \mod 2 \\
 & x \neq \max(0, \lfloor (sv - 2) / 2 \rfloor), \\
 & y < 1, \\
 & y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 1, \\
 & 0 = sv \mod 2 \\
3. \bigvee \\
 & sv \neq 12 \\
4. \bigvee \\
 & sv \neq 11
\end{cases}
\end{align*}
\]
\textbf{NB\_INCREASING\_TERRACE}(x, VARIABLES) \land \\
\textbf{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, VARIABLES) \text{ with } sv = |VARIABLES| \\

\begin{enumerate}
\item y = 0 \implies x = 0 \\
\item x > 0 \land y > 0 \implies 2 \cdot x \leq y - 2 \\
\item sv \geq 0: (1, 4) \quad (2, 6) \\
\item x < 2 \lor y \neq 5 \\
\item y < 1, \\
\item y > sv \cdot \min(1, \max(0, sv - 1)) - 3, \\
\item 0 = y \mod 2 \\
\item x < 1 \lor y \neq 3 \\
\item x \neq 0 \lor y \neq 1
\end{enumerate}
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}_{\text{INCREASING\_TERRACE}}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH}_{\text{INCREASING\_TERRACE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

- 2 \cdot x \leq y
- sv \geq 4: (0, 0) \ (1, 2)
- y > 0 \Rightarrow x \geq 1
- sv \geq 5: (1, sv - 2) \ (1, sv - 3)
\[\text{NB}_\text{INCREASING}_\text{TERRACE}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{INPLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \[x > 0 \land y > 0 \Rightarrow 2 \cdot x \leq sv - 3 - (sv - 3) \mod 2\]
2. \[(sv - 1) \mod 2 = 0 \land sv \geq 4: \begin{cases} ((sv - 2)/2, sv - 2) & ((sv - 2)/2, 1) \\ ((sv - 2)/2 - 1, sv - 2) & ((sv - 2)/2 - 1, sv - 3) \end{cases}\]
3. \[y < 3, \quad y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 3, \quad 1 = sv \mod 2, \quad 0 = y \mod 2\]
4. \[x \neq \max(0, [(sv - 2)/2]) - 1, \quad y < 2, \quad y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 1, \quad 0 = sv \mod 2\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{NB}_\text{INCREASING\_TERRACE}(x, \text{VARIABLES}) & \land \\
\text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) & \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. \( x > 0 \land y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2 \)
2. \( x > 0 \land y > 0 \Rightarrow 2 \times x \leq y + sv - 5 \)
3. \( \begin{cases} & x \neq \max(0, \lfloor (sv - 2) / 2 \rfloor) - 1, \\ & y < 2, \\ & y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 2, \\ & 1 = sv \mod 2, \\ & 1 = y \mod 2 \\ & x \neq \max(0, \lfloor (sv - 2) / 2 \rfloor), \\ & y < 1, \\ & y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 1, \\ & 0 = sv \mod 2 \end{cases} \)
4. \( \begin{cases} & x \neq \max(0, \lfloor (sv - 2) / 2 \rfloor), \\ & y < 1, \\ & y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 1, \\ & 0 = sv \mod 2 \end{cases} \)

\( sv = 12 \)

\( sv = 11 \)
\[
\text{NB\_INCREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(sv > 1 \Rightarrow 2 \cdot x + y \leq sv - 2\)
2. \(sv \geq 4: (0, sv - 2) \quad (1, sv - 4)\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB \_ INCREASING \_ TERRACE}(x, \text{VARIABLES}) \land \sum \text{WIDTH \_ PROPER \_ PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( sv > 1 \Rightarrow 2 \times x + y \leq sv - 2 \)
2. \( sv \geq 6: (0, sv - 2) \quad (1, sv - 4) \)
3. \( y > 0 \Rightarrow 2 \times x \leq sv - 4 - (sv - 4) \mod 2 \)
4. \( sv \mod 2 = 1 \land sv \geq 5: (\lfloor (sv - 2)/2 \rfloor - 1, 2) \quad (\lfloor (sv - 2)/2 \rfloor - 1, 3) \)
5. \( o^c \geq x + [(y + 1)/2] \)
6. \( y \neq 1 \)
\( \text{NB}_1 \) \_INCREASING \_TERRACE \((x, \text{VARIABLES}) \) \land \\
\text{SUM}_1 \_WIDTH \_STEADY \_SEQUENCE \((y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x \leq y \)
   - \( sv \geq 6 \): (1, 2) (2, 4)
2. \( x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq 2 \times sv - 4 \)
   - \( sv \text{ mod } 2 = 0 \land sv \geq 4 \): \( \lceil (sv - 2) / 2 \rceil, sv - 2 \) \( \lceil (sv - 2) / 2 \rceil - 1, sv \)
3. \( x \neq 0 \lor y \neq 1 \)

\( sv = 12 \)

\( \text{NB}_1 \) \_INCREASING \_TERRACE \((x, \text{VARIABLES}) \) \land \\
\text{SUM}_1 \_WIDTH \_STRICKLY \_DECREASING \_SEQUENCE \((y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x + y \leq sv \)
   - \( sv \geq 8 \): (1, sv - 2) (2, sv - 4)
2. \( x = \max(0, \lceil (sv - 2) / 2 \rceil) \land sv \text{ mod } 2 = 1 \land sv > 3 \Rightarrow y \leq 2 \)
3. \( x > 0 \land y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \text{ mod } 2 \)
   - \( sv - 1 \) \text{ mod } 2 = 1 \land sv \geq 4 \): \( \lceil (sv - 2) / 2 \rceil - 1, 2 \) \( \lceil (sv - 2) / 2 \rceil - 1, 3 \)
4. \( y \neq 1 \)

\( sv = 12 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{NB\_INCREASING\_TERRACE}(x, \text{VARIABLES}) & \land \\
\text{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = \lvert\text{VARIABLES}\rvert
\end{align*}

1. \( y = 0 \Rightarrow x = 0 \)
2. \( x > 0 \land y > 0 \Rightarrow 2 \times x \leq y - 2 \)
   \( sv \geq 6: (1, 4) \quad (2, 6) \)
3. \( x < 2 \lor y \neq 5 \)
4. \( \lor \left\{ \begin{array}{l}
x \neq \max(0, \lfloor (sv - 2)/2 \rfloor) - 1, \\
y < 1,
\end{array} \right.
\quad y > sv \times \min(1, \max(0, sv - 1)) - 3,
\quad 0 = y \mod 2
\)
5. \( x < 1 \lor y \neq 3 \)
6. \( x \neq 0 \lor y \neq 1 \)
\( NB_{\text{INCREASING\_TERRACE}}(x, \text{VARIABLES}) \land 
SUM_{\text{WIDTH\_SUMMIT}}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x > 0 \land y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2 \)
   \( (sv - 1) \mod 2 = 1 \land sv \geq 4: \) \( ((sv - 2)/2) - 1, sv - 2 \)
   \( ((sv - 2)/2) - 1, sv - 3 \)

2. \( x \neq \max(0, [(sv - 2)/2]) - 1, \)
   \( y < 2, \)
   \( y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 2, \)
   \( 1 = sv \mod 2, \)
   \( 1 = y \mod 2 \)

3. \( y < 1, \)
   \( y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 1, \)
   \( 0 = sv \mod 2 \)

\( \text{sv} = 12 \)

\( \text{sv} = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\	extunderscore INCREASING\	extunderscore TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM\	extunderscore WIDTH\	extunderscore VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x > 0 \land y > 0 \Rightarrow 2 \ast x \leq sv - 3 - (sv - 3) \text{ mod } 2\)
2. \((sv - 1) \text{ mod } 2 = 1 \land sv \geq 4: ((sv - 2)/2 - 1, sv - 2) \land ((sv - 2)/2 - 1, sv - 3)\)
3. \(x > 0 \land y > 0 \Rightarrow 2 \ast x \leq y + sv - 5\)
   \[
   \begin{cases}
   x \neq \max(0, [(sv - 2)/2]), y < 2, \\
   y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 2, \\
   1 = sv \text{ mod } 2, \\
   1 = y \text{ mod } 2 \\
   x \neq \max(0, [(sv - 2)/2]), \\
   y < 1, \\
   y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 1, \\
   0 = sv \text{ mod } 2
   \end{cases}
   \]

\(sv = 12\)

\(sv = 11\)
\[
\text{NB INCREASING TERRACE}(x, \text{VARIABLES}) \land \text{SUM WIDTH ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(sv > 1 \Rightarrow 2 \cdot x + y \leq sv - 2\)
   - \(sv \geq 6: \ (0, sv - 2) \ (1, sv - 4)\)
2. \(y > 0 \Rightarrow 2 \cdot x \leq sv - 4 - (sv - 4) \mod 2\)
   - \(sv \mod 2 = 1 \land sv \geq 5: \ (|sv - 2|/2 - 1, 2) \ |(sv - 2)/2 - 1, 3)\)
3. \(y \neq 1\)

\[
\text{NB INFLEXION}(x, \text{VARIABLES}) \land \text{NB PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x \leq 2 \cdot y + 1\)
   - \(sv \geq 5: \ (1, 0) \ (3, 1)\)
2. \(2 \cdot y \leq x + 1\)
   - \(sv \geq 5: \ (1, 1) \ (3, 2)\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_INFLEXION}(x, \text{VARIABLES}) \land \]
\[ \text{NB\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq 2 \times y + 1 \)
   - \( sv \geq 5 \): (1, 0) (3, 1)
2. \( 2 \times y \leq x + 1 \)
   - \( sv \geq 5 \): (1, 1) (3, 2)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_INFLEXION}(x, \text{VARIABLES}) \land \]
\[ \text{NB\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq 2 \times y + 1 \)
   - \( sv \geq 5 \): (1, 0) (3, 1)
2. \( 2 \times y \leq x + 1 \)
   - \( sv \geq 5 \): (1, 1) (3, 2)

\[ sv = 12 \]

\[ sv = 11 \]
$$\text{NB}_\text{INFLEXION}(x, \text{VARIABLES}) \land \text{NB}_\text{PROPER}_\text{PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|$$

1. $2 \times y \leq x + 1$
   - $sv \geq 7$: (1, 1) (3, 2)
2. $sv > 1 \Rightarrow x + y \leq sv - 2$
   - $sv \geq 4$: (sv − 2, 0) (sv − 3, 1)

For $sv = 12$:

For $sv = 11$:

$$\text{NB}_\text{INFLEXION}(x, \text{VARIABLES}) \land \text{NB}_\text{PROPER}_\text{PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|$$

1. $2 \times y \leq x + 1$
   - $sv \geq 7$: (1, 1) (3, 2)
2. $sv > 1 \Rightarrow x + y \leq sv - 2$
   - $sv \geq 4$: (sv − 2, 0) (sv − 3, 1)

For $sv = 12$:

For $sv = 11$: 
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_INFLEXION}(x, \text{VARIABLES}) \land \text{NB\_STEADY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow x + y \leq sv - 2 \)
2. \( sv \geq 4: (sv - 3, 1) \quad (sv - 4, 2) \)

\[ sv = 12 \]

\[ sv = 11 \]
\( \text{NB\_INFLEXION}(x, \text{VARIABLES}) \land \text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x \leq 2 \times y \)
   - \( sv \geq 0: (2, 1) \quad (4, 2) \)

2. \( x > 0 \land y > 0 \Rightarrow 2 \times y \leq 2 \times x + sv - 3 - (sv - 3) \mod 2 \)
   - \( (sv - 1) \mod 2 = 1 \land sv \geq 4: (1, \lfloor sv/2 \rfloor - 1) \quad (2, \lfloor sv/2 \rfloor) \)
   - \( y \neq \lfloor sv/2 \rfloor, \quad x < 1, \quad y > 0 \)
   - \( x > \max(0, sv - 2) - 1, \quad 1 = sv \mod 2, \quad 0 = x \mod 2 \)

\( sv = 12 \) and \( sv = 11 \) conditions are illustrated in diagrams.
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{\texttt{NB\textunderscore INFLEXION}}(x, \text{VARIABLES}) \land \text{\texttt{NB\textunderscore STRICTLY\textunderscore INCREASING\textunderscore SEQUENCE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq 2 \cdot y \)
   - \( sv \geq 6: (2, 1) \quad (4, 2) \)

2. \( x > 0 \land y > 0 \Rightarrow 2 \cdot y \leq 2 \cdot x + sv - 3 \mod 2 \)
   - \( (sv - 1) \mod 2 = 1 \land sv \geq 4: (1, \lfloor sv/2 \rfloor - 1) \quad (2, \lfloor sv/2 \rfloor) \)
   - \[ y \neq \lfloor sv/2 \rfloor, \quad x < 1, \]
   - \( x > \text{max}(0, sv - 2) - 1, \quad 1 = sv \mod 2, \quad 0 = x \mod 2 \)

3. \( \exists \]

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{\texttt{NB\textunderscore SUMMIT}}(x, \text{VARIABLES}) \land \text{\texttt{NB\textunderscore SUMMIT}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2 \cdot y \leq x + 1 \)
   - \( sv \geq 5: (1, 1) \quad (3, 2) \)

2. \( sv > 1 \Rightarrow 3 \cdot x \leq 2 \cdot y + 2 \cdot sv - 3 \)
   - \( sv \mod 2 = 0 \land sv \geq 12: (sv - 3, \lfloor (sv - 1)/2 \rfloor - 2) \quad (sv - 5, \lfloor (sv - 1)/2 \rfloor - 5) \)
   - \( sv \mod 2 = 1 \land sv \geq 9: (sv - 2, \lfloor (sv - 1)/2 \rfloor - 1) \quad (sv - 4, \lfloor (sv - 1)/2 \rfloor - 4) \)
\textbf{NB\_INFLEXION}(x, VARIABLES) \land \\
\textbf{NB\_VALLEY}(y, VARIABLES) with \(sv = |\text{VARIABLES}|\)

\begin{enumerate}
\item \(x \leq 2 \times y + 1\)
\quad \(sv \geq 5: (1, 0)\) \quad (3, 1)
\item \(2 \times y \leq x + 1\)
\quad \(sv \geq 5: (1, 1)\) \quad (3, 2)
\end{enumerate}

\begin{align*}
\text{sv} &= 12 \\
\text{nb\_inflexion} &\leq \text{nb\_valley}
\end{align*}

\begin{align*}
\text{sv} &= 11 \\
\text{nb\_inflexion} &\leq \text{nb\_valley}
\end{align*}

\begin{enumerate}
\item \(x + y \leq sv - 1\)
\quad \(sv \geq 7: (sv - 2, 1)\) \quad (sv - 3, 2)
\item \(2 \times y \leq x\)
\quad \(sv \geq 4: (0, 0)\) \quad (2, 1)
\item \(y = 0 \land sv > 3 \Rightarrow 3 \times x \leq 2 \times sv - 3 - (2 \times sv - 3) \mod 3\)
\end{enumerate}

\begin{align*}
\text{sv} &= 12 \\
\text{nb\_inflexion} &\leq \text{nb\_zigzag}
\end{align*}

\begin{align*}
\text{sv} &= 11 \\
\text{nb\_inflexion} &\leq \text{nb\_zigzag}
\end{align*}
A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_{\text{INflexion}}(x, \text{VARIABLES}) \land \text{SUM}_{\text{WIDTH DECREASING SEQUENCE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \mod 2 = 0 \land sv \geq 4: (sv - 2, sv - 2) \land (sv - 4, sv - 4) \)
   - \( sv \mod 2 = 1 \land sv \geq 5: (sv - 3, sv - 3) \land (sv - 5, sv - 5) \)
2. \( x = \max(0, sv - 2) \land sv \mod 2 = 1 \Rightarrow y \leq sv - 1 \)
3. \( x > 0 \Rightarrow y \geq 2 \)
   - \( sv \geq 4: (1, 2) \land (2, 2) \)
4. \( x > 0 \land y > 0 \Rightarrow y \leq x + sv - 2 \)
   - \( sv \geq 4: (1, sv - 1) \land (2, sv) \)
   \[ \begin{align*}
   & x \neq \max(0, sv - 2), \\
   & y < 3, \\
   & y > sv \ast \min(1, \max(0, sv - 1)) - 1, \\
   & 1 = sv \mod 2, \\
   & 0 = y \mod 2
   \end{align*} \]
5. \( y \neq 1 \)
6. \( y \neq x \lor 0 = x \mod 2 \)
7. \( y \neq x \lor 0 = x \mod 2 \)
   \[ \begin{align*}
   & y \neq sv \ast \min(1, \max(0, sv - 1)), \\
   & x < 1, \\
   & x > \max(0, sv - 2) - 1, \\
   & 0 = x \mod 2
   \end{align*} \]

\[ sv = 12 \]

\[ sv = 11 \]
**NB_INFLEXION**($x, \text{VARIABLES}$) \&
**SUM_WIDTH_DECREASING_TERRACE**($y, \text{VARIABLES}$) with $sv = |\text{VARIABLES}|$

1. $sv > 1 \Rightarrow x + y \leq sv - 2$
2. $sv \geq 4$: $(sv - 2, 0)$ $(sv - 4, 2)$
3. $y > 0 \Rightarrow x \leq sv - 4$
4. $y \neq 1$

**NB_INFLEXION**($x, \text{VARIABLES}$) \&
**SUM_WIDTH_GORGE**($y, \text{VARIABLES}$) with $sv = |\text{VARIABLES}|$

1. $sv > 1 \Rightarrow 3 \cdot x \leq 2 \cdot y + 2 \cdot sv - 3$
2. $sv > 1 \Rightarrow x + 2 \cdot y \leq 2 \cdot sv - 3$
3. $sv \geq 5$: $(1, sv - 2)$ $(3, sv - 3)$
4. $y > 0 \Rightarrow x \geq 1$
5. $sv \geq 3$: $(1, sv - 2)$ $(1, sv - 3)$
\[ \text{NB}_\text{INFLEXION}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \mod 2 = 0 \land sv \geq 4: (sv - 2, sv - 4) \quad (sv - 4, sv - 4) \)
   - \( sv \mod 2 = 1 \land sv \geq 5: (sv - 3, sv - 3) \quad (sv - 5, sv - 5) \)
2. \( x = \max(0, sv - 2) \land sv \mod 2 = 1 \Rightarrow y \leq sv - 1 \)
3. \( x > 0 \Rightarrow y \geq 2 \)
   - \( sv \geq 4: (1, 2) \quad (2, 2) \)
4. \( x > 0 \land y > 0 \Rightarrow y \leq x + sv - 2 \)
   - \( sv \geq 4: (1, sv - 1) \quad (2, sv) \)
     \[ \begin{align*}
     x & \neq \max(0, sv - 2), \\
     y & < 3, \\
     y & > sv \ast \min(1, \max(0, sv - 1)) - 1, \\
     1 & = sv \mod 2, \\
     0 & = y \mod 2
\end{align*} \]
5. \( y \neq 1 \)
6. \( y \neq x \lor 0 = x \mod 2 \)
7. \( y \neq x \lor 0 = x \mod 2 \)
   - \( y \neq sv \ast \min(1, \max(0, sv - 1)), \)
   - \( x < 1, \)
   - \( x > \max(0, sv - 2) - 1, \)
   - \( 0 = x \mod 2 \)

\[ sv = 12 \]

\[ sv = 11 \]
\[ \text{NB\_INFLEXION}(x, \text{VARIABLES}) \land \]
\[ \text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ sv > 1 \Rightarrow x + y \leq sv - 2 \]
2. \[ sv \geq 4: (sv - 2, 0) \quad (sv - 4, 2) \]
3. \[ y > 0 \Rightarrow x \leq sv - 4 \]
4. \[ y \neq 1 \]

\[ \text{NB\_INFLEXION}(x, \text{VARIABLES}) \land \]
\[ \text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x \leq y \]
2. \[ sv \geq 3: (sv - 2, sv - 2) \quad (sv - 3, sv - 3) \]
3. \[ y > 0 \Rightarrow x \geq 1 \]
4. \[ sv \geq 4: (1, sv - 2) \quad (1, sv - 3) \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}_\text{INFLEXION}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x \leq 2y + 1 \)
   - \( sv \geq 5 \): (1, 0) (3, 1)

2. \( sv > 1 \Rightarrow x + 2y \leq 2sv - 3 \)
   - \( sv \geq 5 \): (1, sv - 2) (3, sv - 3)

3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 3 \): (1, sv - 2) (1, sv - 3)

---

The diagrams illustrate the parameterized invariants for different values of \( sv \) (12 and 11). Each quadrant represents a condition on \( x, y, \text{nb}_\text{inflexion}, \text{sum}_\text{width}_\text{peak} \), with \( sv \) as a parameter.
\[
\text{NB\_INFLEXION}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x \leq 2 \times y + 1 \)
   - \( sv \geq 5: (1, 0) \quad (3, 1) \)
2. \( sv > 1 \Rightarrow x + 2 \times y \leq 2 \times sv - 3 \)
   - \( sv \geq 5: (1, sv - 2) \quad (3, sv - 3) \)
3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 3: (1, sv - 2) \quad (1, sv - 3) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB\_INFLEXION}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \( x \leq 2 \times y + 1 \)  
   - \( sv \geq 5 \): (1, 0) (3, 1)
2. \( sv > 1 \Rightarrow x + 2 \times y \leq 2 \times sv - 3 \)  
   - \( sv \geq 5 \): (1, sv - 2) (3, sv - 3)
3. \( y > 0 \Rightarrow x \geq 1 \)  
   - \( sv \geq 3 \): (1, sv - 2) (1, sv - 3)
\[ \text{NB}\_\text{INFLLEXION}(x, \text{VARIABLES}) \land \\
\text{SUM}\_\text{WIDTH}\_\text{PROPER}\_\text{PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( sv > 1 \Rightarrow 2 \cdot x + y \leq 2 \cdot sv - 4 \)
2. \( sv \geq 4: (sv - 2, 0) \quad (sv - 3, 2) \)
3. \( sv > 1 \Rightarrow x + 2 \cdot y \leq 2 \cdot sv - 3 \)
4. \( sv \geq 7: (1, sv - 2) \quad (3, sv - 3) \)
5. \( y > 0 \Rightarrow x \geq 1 \)
6. \( sv \geq 5: (1, sv - 2) \quad (1, sv - 3) \)
7. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\textsc{nb\_inflexion}(x, \textsc{variables}) \land \\
\textsc{sum\_width\_proper\_plateau}(y, \textsc{variables}) \text{ with } sv = |\textsc{variables}|

\begin{enumerate}
\item $sv > 1 \Rightarrow 2 \times x + y \leq 2 \times sv - 4$
\item $sv > 4 : (sv - 2, 0) \quad (sv - 3, 2)$
\item $sv > 1 \Rightarrow x + 2 \times y \leq 2 \times sv - 3$
\item $sv \geq 7 : (1, sv - 2) \quad (3, sv - 3)$
\item $y > 0 \Rightarrow x \geq 1$
\item $sv \geq 5 : (1, sv - 2) \quad (1, sv - 3)$
\item $y \neq 1$
\end{enumerate}

\begin{align*}
\textsc{nb\_inflexion}(x, \textsc{variables}) \land \\
\textsc{sum\_width\_steady\_sequence}(y, \textsc{variables}) \text{ with } sv = |\textsc{variables}|
\end{align*}

\begin{enumerate}
\item $x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq 2 \times sv - 4$
\item $sv \geq 4 : (sv - 3, 2) \quad (sv - 4, 4)$
\item $y \neq 1$
\end{enumerate}
\[\text{NB_INFLEXION}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(x \leq y\)
   - \(sv \mod 2 = 0 \land sv \geq 4: (sv - 2, sv - 2) (sv - 4, sv - 4)\)
   - \(sv \mod 2 = 1 \land sv \geq 5: (sv - 3, sv - 3) (sv - 5, sv - 5)\)
2. \(x = \max(0, sv - 2) \land sv \mod 2 = 1 \Rightarrow y \leq sv - 1\)
3. \(x > 0 \Rightarrow y \geq 2\)
   - \(sv \geq 4: (1, 2) (2, 2)\)
4. \(x > 0 \land y > 0 \Rightarrow y \leq x + sv - 2\)
   - \(sv \geq 4: (1, sv - 1) (2, sv)\)
   - \(y > sv \ast \min(1, \max(0, sv - 1)) - 1, 1 = sv \mod 2, 0 = y \mod 2\)
5. \(y \neq 1\)
6. \(y \neq x \lor 0 = x \mod 2\)
   - \(y \neq sv \ast \min(1, \max(0, sv - 1)), x < 1, x > \max(0, sv - 2) - 1, 0 = x \mod 2\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB}_\text{INFLExION}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \( x \leq y \)
   
   - \( sv \mod 2 = 0 \land sv \geq 4: (sv - 2, sv - 4) \quad (sv - 4, sv - 4) \)
   
   - \( sv \mod 2 = 1 \land sv \geq 5: (sv - 3, sv - 3) \quad (sv - 5, sv - 5) \)

2. \( x = \max(0, sv - 2) \land sv \mod 2 = 1 \Rightarrow y \leq sv - 1 \)

3. \( x > 0 \Rightarrow y \geq 2 \)
   
   - \( sv \geq 4: (1, 2) \quad (2, 2) \)

4. \( x > 0 \land y > 0 \Rightarrow y \leq x + sv - 2 \)
   
   - \( sv \geq 4: (1, sv - 1) \quad (2, sv) \)
     
     \[
     \begin{cases}
     x \neq \max(0, sv - 2), \\
y < 3,
     \end{cases}
     \]

   - \( y > sv \ast \min(1, \max(0, sv - 1)) - 1, \\
     1 = sv \mod 2, \\
     0 = y \mod 2
     \)

5. \( y \neq 1 \)

6. \( y \neq x \lor 0 = x \mod 2 \)

\[
\begin{align*}
\lor \\
( y \neq sv \ast \min(1, \max(0, sv - 1)), \\
x < 1, \\
x \geq \max(0, sv - 2) - 1, \\
0 = x \mod 2
\end{align*}
\]

---

Diagram for \( sv = 12 \)

Diagram for \( sv = 11 \)
\[ \text{NB\_INFLEXION}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( sv > 1 \Rightarrow 3x \leq 2y + 2sv - 3 \)
2. \( sv > 1 \Rightarrow x + 2y \leq 2sv - 3 \)
   - \( sv \geq 5: (1, sv - 2) (3, sv - 3) \)
3. \( y > 0 \Rightarrow x \geq 1 \)
4. \( sv \geq 3: (1, sv - 2) (1, sv - 3) \)

\[ \text{NB\_INFLEXION}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq 2y + 1 \)
   - \( sv \geq 5: (1, 0) (3, 1) \)
2. \( sv > 1 \Rightarrow x + 2y \leq 2sv - 3 \)
   - \( sv \geq 5: (1, sv - 2) (3, sv - 3) \)
3. \( y > 0 \Rightarrow x \geq 1 \)
4. \( sv \geq 3: (1, sv - 2) (1, sv - 3) \)
\textbf{NB\_INFLEXION}(x, VARIABLES) \land \\
\textbf{SUM\_WIDTH\_ZIGZAG}(y, VARIABLES) with sv = |VARIABLES|

1. $y \leq x$
2. $sv \geq 5$: $(sv - 2, sv - 2)$ $(sv - 3, sv - 3)$
3. $y = 0 \land sv > 3 \Rightarrow 3 \cdot x \leq 2 \cdot sv - 3 - (2 \cdot sv - 3) \mod 3$
4. $sv > 1 \Rightarrow 3 \cdot x \leq y + 2 \cdot sv - 4 + (sv \mod 3 = 0)$
5. $x > 0 \land y > 0 \Rightarrow 3 \cdot x \leq y + 2 \cdot sv - 4$
6. $sv \geq 7$: $(sv - 2, sv - 2)$ $(sv - 3, sv - 5)$
7. $y > 0 \Rightarrow x \geq 2$
8. $y \neq 1$

\begin{itemize}
\item sv = 12
\item sv = 11
\end{itemize}
\[ \text{NB\_PEAK}(x, \text{VARIABLES}) \land \text{NB\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y + 1 \)
   - \( sv \geq 5 \): (1, 0) (2, 1)
2. \( y \leq x + 1 \)
   - \( sv \geq 5 \): (0, 1) (1, 2)
3. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \mod 2 = 1 \land \) \( sv \geq 3 \): \((\lfloor sv - 1 \rfloor/2, \lfloor (sv - 1)/2 \rfloor - 1)\) \((\lfloor (sv - 1)/2 \rfloor - 1, \lfloor (sv - 1)/2 \rfloor)\)

\[ \text{NB\_PEAK}(x, \text{VARIABLES}) \land \text{NB\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y \)
   - \( sv \geq 3 \): (0, 0) (1, 1)
2. \( y \leq x \)
   - \( sv \geq 3 \): (0, 0) (1, 1)

\[ sv = 12 \]
\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[\text{NB\_PEAK}(x, \text{VARIABLES}) \land \text{NB\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(2 \times x + y \leq sv - 1\)
   - \(sv \text{ mod } 2 = 0 \land sv \geq 12: \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor, 1 \right) \land \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 1, 3 \right)\)
   - \(sv \text{ mod } 2 = 1 \land sv \geq 15: \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 1, 2 \right) \land \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 2, 4 \right)\)

2. \(y \leq x + 1\)
   - \(sv \geq 7: (0, 1) \land (1, 2)\)

3. \(sv > 1 \Rightarrow x + 2 \times y \leq sv - 2\)

\[\text{NB\_PEAK}(x, \text{VARIABLES}) \land \text{NB\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(2 \times x + y \leq sv - 1\)
   - \(sv \text{ mod } 2 = 0 \land sv \geq 10: \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor, 1 \right) \land \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 1, 3 \right)\)
   - \(sv \text{ mod } 2 = 1 \land sv \geq 13: \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 1, 2 \right) \land \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 2, 4 \right)\)

2. \(y \leq x\)
   - \(sv \geq 4: (0, 0) \land (1, 1)\)
\[ \text{NB}_\text{PEAK}(x, \text{VARIABLES}) \land \text{NB}_\text{STEADY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\[ 2 \times x + y \leq sv - 1 \]

- \( sv \geq 5 \): (1, sv - 3) \( \land \) (2, sv - 5)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB}_\text{PEAK}(x, \text{VARIABLES}) \land \text{NB}_\text{STEADY}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\[ 2 \times x + y \leq sv - 1 \]

- \( sv \mod 2 = 0 \land sv \geq 6 \): \((\lfloor (sv - 1)/2 \rfloor, 1)\) \(\land\) \((\lfloor (sv - 1)/2 \rfloor - 1, 3)\)
- \( sv \mod 2 = 1 \land sv \geq 9 \): \((\lfloor (sv - 1)/2 \rfloor - 1, 2)\) \(\land\) \((\lfloor (sv - 1)/2 \rfloor - 2, 4)\)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_PEAK}(x, \text{VARIABLES}) \land \\
\text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x \leq y \)
   - \( sv \geq 5 \): (1, 1) (2, 2)

\( sv = 12 \)

\( sv = 11 \)

\[
\text{NB\_PEAK}(x, \text{VARIABLES}) \land \\
\text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x \leq y \)
   - \( sv \geq 5 \): (1, 1) (2, 2)

\( sv = 12 \)

\( sv = 11 \)
\[
\text{NB\_PEAK}(\mathbf{x}, \text{VARIABLES}) \land \\
\text{NB\_SUMMIT}(\mathbf{y}, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(3 \times \mathbf{x} \leq \mathbf{y} + sv - 1\)
   - \(sv \mod 2 = 0 \land sv \geq 10: \left(\lfloor (sv - 1)/2 \rfloor, \lceil (sv - 1)/2 \rceil - 1\right) \land \left(\lfloor (sv - 1)/2 \rfloor - 1, \lfloor (sv - 1)/2 \rfloor - 4\right)\)

2. \(\mathbf{y} \leq \mathbf{x}\)
   - \(sv \geq 3: (0, 0) \land (1, 1)\)

\[
\text{NB\_PEAK}(\mathbf{x}, \text{VARIABLES}) \land \\
\text{NB\_VALLEY}(\mathbf{y}, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(\mathbf{x} \leq \mathbf{y} + 1\)
   - \(sv \geq 5: (1, 0) \land (2, 1)\)

2. \(\mathbf{y} \leq \mathbf{x} + 1\)
   - \(sv \geq 5: (0, 1) \land (1, 2)\)

3. \(sv > 1 \Rightarrow \mathbf{x} + \mathbf{y} \leq sv - 2\)
   - \(sv \mod 2 = 1 \land sv \geq 3: \left(\lfloor (sv - 1)/2 \rfloor, \lfloor (sv - 1)/2 \rfloor - 1\right) \land \left(\lfloor (sv - 1)/2 \rfloor - 1, \lfloor (sv - 1)/2 \rfloor\right)\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_{\text{PEAK}}(x, \text{VARIABLES}) \land \]
\[ \text{NB}_{\text{ZIGZAG}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \cdot x + y \leq sv \]
   - \( sv \mod 2 = 0 \land sv \geq 14: \) \( [(sv - 1)/2, 2) \) \( [(sv - 1)/2 - 1, 4) \)
   - \( sv \mod 2 = 1 \land sv \geq 11: \) \( [(sv - 1)/2, 1) \) \( [(sv - 1)/2 - 1, 3) \)

2. \( y \leq x \)
   - \( sv \geq 4: \) \( (0, 0) \) \( (1, 1) \)

3. \( y = 0 \Rightarrow 3 \cdot x \leq sv - sv \mod 3 \)

\[ \text{NB}_{\text{PEAK}}(x, \text{VARIABLES}) \land \]
\[ \text{SUM}_{\text{WIDTH DECREASING SEQUENCE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \cdot x + y \leq 2 \cdot sv - 2 \]

2. \[ 2 \cdot x \leq y \]
   - \( sv \geq 5: \) \( (1, 2) \) \( (2, 4) \)

3. \( x \neq 0 \lor y \neq 1 \)
\[ \text{NB}_\text{PEAK}(x, \text{VARIABLES}) \wedge \text{SUM}_\text{WIDTH}_\text{DECREASING}_\text{TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2 \cdot x + y \leq sv - 1 \)
   - \( sv \geq 7: (1, sv - 3) \quad (2, sv - 5) \)
2. \( y > 0 \Rightarrow 2 \cdot x \leq sv - 3 - (sv - 3) \mod 2 \)
   - \( (sv - 1) \mod 2 = 1 \wedge sv \geq 6: ((sv - 1)/2 - 1, 2) \quad ((sv - 1)/2 - 1, 3) \)
3. \( y \neq 1 \)

\[ \text{NB}_\text{PEAK}(x, \text{VARIABLES}) \wedge \text{SUM}_\text{WIDTH}_\text{GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 3 \cdot x \leq y + sv \)
2. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 3: (0, sv - 2) \quad (1, sv - 3) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{PEAK}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{INCRESSING}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2x + 2y \leq 3sv - 2 - (3sv - 2) \mod 2 \)
   - \( sv \mod 2 = 1 \land sv \geq 3: \left(\left\lfloor (sv - 1)/2\right\rfloor, sv - 1\right) \left(\left\lfloor (sv - 1)/2\right\rfloor - 1, sv\right) \)
2. \( 2x \leq y \)
   - \( sv \geq 5: (1, 2) \quad (2, 4) \)
3. \( x \neq 0 \lor y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB}_\text{PEAK}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{INCRESSING}_\text{TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2x + y \leq sv - 1 \)
   - \( sv \geq 7: (1, sv - 3) \quad (2, sv - 5) \)
2. \( y > 0 \Rightarrow 2x \leq sv - 3 - (sv - 3) \mod 2 \)
   - \( (sv - 1) \mod 2 = 1 \land sv \geq 6: \left(\left\lfloor (sv - 1)/2\right\rfloor - 1, 2\right) \left(\left\lfloor (sv - 1)/2\right\rfloor - 1, 3\right) \)
3. \( y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]
\[
\text{NB\_PEAK}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \[2 \cdot x \leq y + 1\]
   \[sv \geq 5: (1, 1) \quad (2, 3)\]

\[
\text{SV } = 12
\]

\[
\text{SV } = 11
\]

\[
\text{NB\_PEAK}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \[x + y \leq sv - 1\]
   \[sv \geq 5: (1, sv - 2) \quad (2, sv - 3)\]

2. \[x \leq y\]
   \[sv \geq 3: (0, 0) \quad (1, 1)\]

3. \[y > 0 \Rightarrow x \geq 1\]
   \[sv \geq 4: (1, sv - 2) \quad (1, sv - 3)\]

\[
\text{SV } = 12
\]

\[
\text{SV } = 11
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB}_{\text{PEAK}}(x, \text{VARIABLES}) \land \\
\text{SUM}_{\text{WIDTH}\_\text{PLAIN}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \(x \leq y + 1\)
   - \(sv \geq 5: (1, 0) ~ (2, 1)\)
2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 3: (0, sv - 2) ~ (1, sv - 3)\)

\[
\begin{align*}
sv = 12
\end{align*}
\]

\[
\begin{align*}
sv = 11
\end{align*}
\]

\[
\begin{align*}
\text{NB}_{\text{PEAK}}(x, \text{VARIABLES}) \land \\
\text{SUM}_{\text{WIDTH}\_\text{PLATEAU}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \(x + y \leq sv - 1\)
   - \(sv \geq 5: (1, sv - 2) ~ (2, sv - 3)\)
2. \(x \leq y\)
   - \(sv \geq 3: (0, 0) ~ (1, 1)\)
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 4: (1, sv - 2) ~ (1, sv - 3)\)

\[
\begin{align*}
sv = 12
\end{align*}
\]

\[
\begin{align*}
sv = 11
\end{align*}
\]
\[ \text{NB\_PEAK}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 4 \times x + y \leq 2 \times sv - 2 \]
   - \( sv \mod 2 = 0 \land sv \geq 12 \) : \( (\lfloor (sv - 1)/2 \rfloor, 2) \)
   - \( sv \mod 2 = 1 \land sv \geq 15 \) : \( (\lfloor (sv - 1)/2 \rfloor - 1, 4) \)
2. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
3. \( sv \geq 5: (0, sv - 2) \quad (1, sv - 3) \)
4. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[\text{NB}_\text{PEAK}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{PROPER}_\text{PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

\[4 \times x + y \leq 2 \times sv - 2 \]

- \(sv \mod 2 = 0 \land sv \geq 10: \left(\lfloor (sv - 1)/2 \rfloor, 2\right) \quad \left(\lfloor (sv - 1)/2 \rfloor - 1, 6\right)\)
- \(sv \mod 2 = 1 \land sv \geq 13: \left(\lfloor (sv - 1)/2 \rfloor - 1, 4\right) \quad \left(\lfloor (sv - 1)/2 \rfloor - 2, 8\right)\)

\[x + y \leq sv - 1\]

- \(sv \geq 7: (1, sv - 2) \quad (2, sv - 3)\)

\[y > 0 \Rightarrow x \geq 1\]

- \(sv \geq 5: (1, sv - 2) \quad (1, sv - 3)\)

\[y \neq 1\]

\[\text{NV}_\text{PEAK}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{STEADY}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

\[4 \times x + y \leq 2 \times sv - 2 \]

- \(sv \mod 2 = 0 \land sv \geq 6: \left(\lfloor (sv - 1)/2 \rfloor, 2\right) \quad \left(\lfloor (sv - 1)/2 \rfloor - 1, 6\right)\)
- \(sv \mod 2 = 1 \land sv \geq 9: \left(\lfloor (sv - 1)/2 \rfloor - 1, 4\right) \quad \left(\lfloor (sv - 1)/2 \rfloor - 2, 8\right)\)

\[y \neq 1\]
\[\text{NB\_PEAK}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \[2 \times x + y \leq 2 \times sv - 2\]
2. \[2 \times x \leq y\]
3. \[sv \geq 5: (1, 2) \quad (2, 4)\]
4. \[x \neq 0 \lor y \neq 1\]

\[\text{sv} = 12\]

\[\text{sv} = 11\]

\[\text{NB\_PEAK}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \[2 \times x + y \leq 2 \times sv - 2\]
2. \[2 \times x \leq y\]
3. \[sv \geq 5: (1, 2) \quad (2, 4)\]
4. \[x \neq 0 \lor y \neq 1\]

\[\text{sv} = 12\]

\[\text{sv} = 11\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_PEAK}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

- 1. \(3 \cdot x \leq y + sv - 1\)
- 2. \(x + y \leq sv - 1\)
- 3. \(sv \geq 5: (1, sv - 2) \land (2, sv - 3)\)
- 4. \(y > 0 \Rightarrow x \geq 1\)
- 5. \(sv \geq 4: (1, sv - 2) \land (1, sv - 3)\)

\[
\text{NB\_PEAK}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

- 1. \(x \leq y + 1\)
- 2. \(sv \geq 5: (1, 0) \land (2, 1)\)
- 3. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
- 4. \(sv \geq 3: (0, sv - 2) \land (1, sv - 3)\)
\text{NB\_PEAK}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( y \leq 2 \times x + 1 \)
2. \( sv \geq 7: (1, 3) \quad (2, 5) \)
3. \( y = 0 \Rightarrow 3 \times x \leq sv - sv \mod 3 \)
4. \( x > 0 \land y > 0 \Rightarrow 3 \times x \leq y + sv - 2 \)
5. \( y > 0 \Rightarrow 6 \times x \leq y + 2 \times sv - 1 \)
6. \( sv \mod 2 = 0 \land sv \geq 14: ((sv - 1)/2, sv - 5) \quad ((sv - 1)/2 - 1, sv - 11) \)
7. \( sv \mod 2 = 1 \land sv \geq 17: ((sv - 1)/2 - 1, sv - 8) \quad ((sv - 1)/2 - 2, sv - 14) \)
8. \( y > 0 \Rightarrow x \geq 1 \)
9. \( sv \geq 5: (1, 2) \quad (1, 3) \)
10. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{NB\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x \leq y + 1 \)
   - if \( sv \geq 5 \): (1, 0) (2, 1)

2. \( y \leq x + 1 \)
   - if \( sv \geq 5 \): (0, 1) (1, 2)

3. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - if \( sv \mod 2 = 1 \) and \( sv \geq 3 \):
     - \( \lfloor (sv - 1)/2 \rfloor, \lfloor (sv - 1)/2 \rfloor - 1 \) \( \lfloor (sv - 1)/2 \rfloor - 1, \lfloor (sv - 1)/2 \rfloor \)

\[
\text{NB\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{NB\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( 2 * x + y \leq sv - 1 \)
   - if \( sv \mod 2 = 0 \) and \( sv \geq 10 \):
     - \( \lfloor (sv - 1)/2 \rfloor, 1 \) \( \lfloor (sv - 1)/2 \rfloor - 1, 3 \)
   - if \( sv \mod 2 = 1 \) and \( sv \geq 13 \):
     - \( \lfloor (sv - 1)/2 \rfloor - 1, 2 \) \( \lfloor (sv - 1)/2 \rfloor - 2, 4 \)

2. \( y \leq x \)
   - if \( sv \geq 4 \): (0, 0) (1, 1)
\[ \text{NB\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{NB\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x + y \leq sv - 1\)
   - \(sv \mod 2 = 0 \land sv \geq 12: (\lfloor (sv - 1)/2 \rfloor, 1) \quad (\lfloor (sv - 1)/2 \rfloor - 1, 3)\)
   - \(sv \mod 2 = 1 \land sv \geq 15: (\lfloor (sv - 1)/2 \rfloor - 1, 2) \quad (\lfloor (sv - 1)/2 \rfloor - 2, 4)\)
2. \(y \leq x + 1\)
   - \(sv \geq 7: (0, 1) \quad (1, 2)\)
3. \(sv > 1 \Rightarrow x + 2 \times y \leq sv - 2\)

\[ \begin{align*}
\text{nb\_proper\_plateau} & \quad \text{sv = 12} \\
\text{nb\_plain} & \quad \text{sv = 11}
\end{align*} \]

\[ \text{NB\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{NB\_STEADY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x + y \leq sv - 1\)
   - \(sv \geq 5: (1, sv - 3) \quad (2, sv - 5)\)

\[ \begin{align*}
\text{nb\_steady} & \quad \text{sv = 12} \\
\text{nb\_plain} & \quad \text{sv = 11}
\end{align*} \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_PLAIN}(x, \text{VARIABLES}) \land \text{NB\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \cdot x + y \leq sv - 1\)
   - \(sv \mod 2 = 0 \land sv \geq 6: ((sv - 1)/2, 1) \land ((sv - 1)/2 - 1, 3)\)
   - \(sv \mod 2 = 1 \land sv \geq 9: ((sv - 1)/2 - 1, 2) \land ((sv - 1)/2 - 2, 4)\)

2. \(x \leq y\)
   - \(sv \geq 5: (1, 1) \land (2, 2)\)
\( \text{NB\_PLAIN}(x, \text{VAR}) \land \text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VAR}) \) with \( sv = |\text{VAR}| \)

1. \( x \leq y \)
   - \( sv \geq 5: (1, 1), (2, 2) \)

\( sv = 12 \)

\( sv = 11 \)

2. \( sv \text{ mod } 2 = 0 \land sv \geq 12: (\lfloor (sv - 1)/2 \rfloor, \lfloor (sv - 1)/2 \rfloor - 2), (\lfloor (sv - 1)/2 \rfloor - 1, \lfloor (sv - 1)/2 \rfloor - 5) \)
   - \( sv \text{ mod } 2 = 1 \land sv \geq 9: (\lfloor (sv - 1)/2 \rfloor, \lfloor (sv - 1)/2 \rfloor - 1), (\lfloor (sv - 1)/2 \rfloor - 1, \lfloor (sv - 1)/2 \rfloor - 4) \)

3. \( y \leq x + 1 \)
   - \( sv \geq 5: (0, 1), (1, 2) \)

4. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \text{ mod } 2 = 1 \land sv \geq 3: (\lfloor (sv - 1)/2 \rfloor, \lfloor (sv - 1)/2 \rfloor - 1), (\lfloor (sv - 1)/2 \rfloor - 1, \lfloor (sv - 1)/2 \rfloor) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{NB\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{enumerate}
\item \(x \leq y\)
  \begin{itemize}
  \item \(sv \geq 3\): (0, 0) (1, 1)
  \end{itemize}
\item \(y \leq x\)
  \begin{itemize}
  \item \(sv \geq 3\): (0, 0) (1, 1)
  \end{itemize}
\end{enumerate}

\[
\text{NB\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{NB\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{enumerate}
\item \(2 \times x + y \leq sv\)
  \begin{itemize}
  \item \(sv \mod 2 = 0 \land sv \geq 14\): \(((sv - 1)/2), 2\) \(((sv - 1)/2) - 1, 4\)
  \item \(sv \mod 2 = 1 \land sv \geq 11\): \(((sv - 1)/2), 1\) \(((sv - 1)/2) - 1, 3\)
  \end{itemize}
\item \(y \leq x\)
  \begin{itemize}
  \item \(sv \geq 4\): (0, 0) (1, 1)
  \end{itemize}
\item \(y = 0 \Rightarrow 3 \times x \leq sv - sv \mod 3\)
\end{enumerate}
\(\text{NB}_\_\text{PLAIN}(x, \text{VARIABLES}) \land \sum\text{WIDTH}_\_\text{DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\)

1. \(2 \times x + 2 \times y \leq 3 \times sv - 2 - (3 \times sv - 2) \mod 2\)

2. \(sv \mod 2 = 1 \land sv \geq 3: ((sv - 1)/2, sv - 1) \land ((sv - 1)/2 - 1, sv)\)

3. \(2 \times x \leq y\)

\(sv \geq 5:\ (1, 2) \land (2, 4)\)

4. \(x \neq 0 \lor y \neq 1\)

\(sv = 12\)

\(sv = 11\)

\(\text{NB}_\_\text{PLAIN}(x, \text{VARIABLES}) \land \sum\text{WIDTH}_\_\text{DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\)

1. \(2 \times x + y \leq sv - 1\)

2. \(sv \geq 7: (1, sv - 3) \land (2, sv - 5)\)

3. \(y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2\)

\(sv \mod 2 = 1 \land sv \geq 6: ((sv - 1)/2 - 1, 2) \land ((sv - 1)/2 - 1, 3)\)

4. \(y \neq 1\)

\(sv = 12\)

\(sv = 11\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{NB}_\text{PLAIN} & (x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{GORGE} & (y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. $3 \cdot x \leq y + sv - 1$
2. $x + y \leq sv - 1$
   - $sv \geq 5$: (1, sv − 2) (2, sv − 3)
3. $y > 0 \Rightarrow x \geq 1$
   - $sv \geq 4$: (1, sv − 2) (1, sv − 3)

\begin{align*}
\text{NB}_\text{PLAIN} & (x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{INCREASING}_\text{SEQUENCE} & (y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. $2 \cdot x + y \leq 2 \cdot sv - 2$
2. $2 \cdot x \leq y$
   - $sv \geq 5$: (1, 2) (2, 4)
3. $x \neq 0 \lor y \neq 1$

\begin{align*}
\text{NB}_\text{PLAIN} & (x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{INCREASING}_\text{SEQUENCE} & (y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. $2 \cdot x + y \leq 2 \cdot sv - 2$
2. $2 \cdot x \leq y$
   - $sv \geq 5$: (1, 2) (2, 4)
3. $x \neq 0 \lor y \neq 1$
\( \text{NB\_PLAIN}(x, \text{VARIABLES}) \land \)
\( \text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x \leq y \leq sv - 1 \)
   - \( sv \geq 7: \ (1, sv - 3) \ (2, sv - 5) \)
2. \( y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2 \)
   - \( (sv - 1) \mod 2 = 1 \land sv \geq 6: \ (\lfloor (sv - 1)/2 \rfloor - 1, 2) \ (\lfloor (sv - 1)/2 \rfloor - 1, 3) \)
3. \( y \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)

\( \text{NB\_PLAIN}(x, \text{VARIABLES}) \land \)
\( \text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x \leq y + 1 \)
   - \( sv \geq 5: \ (1, 1) \ (2, 3) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\( \text{NB\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x \leq y + 1 \)
   - \( sv \geq 5: \) (1, 0) (2, 1)
2. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 3: \) (0, \( sv - 2 \)) (1, \( sv - 3 \))

\( sv = 12 \)

\( \text{SV} = 11 \)

\( \text{NB\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 5: \) (1, \( sv - 2 \)) (2, \( sv - 3 \))
2. \( x \leq y \)
   - \( sv \geq 3: \) (0, 0) (1, 1)
3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 4: \) (1, \( sv - 2 \)) (1, \( sv - 3 \))

\( sv = 12 \)

\( \text{SV} = 11 \)
\[ \text{NB\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y + 1 \)
   - \( sv \geq 5: (1, 0) \quad (2, 1) \)
2. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 3: (0, sv - 2) \quad (1, sv - 3) \)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 4 \times x + y \leq 2 \times sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 10: (\lfloor (sv - 1)/2 \rfloor, 2) \quad (\lfloor (sv - 1)/2 \rfloor - 1, 6) \)
   - \( sv \mod 2 = 1 \land sv \geq 13: (\lfloor (sv - 1)/2 \rfloor - 1, 4) \quad (\lfloor (sv - 1)/2 \rfloor - 2, 8) \)
2. \( x + y \leq sv - 1 \)
   - \( sv \geq 7: (1, sv - 2) \quad (2, sv - 3) \)
3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 5: (1, sv - 2) \quad (1, sv - 3) \)
4. \( y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{PLAIN}(x, \text{VARIABLES}) \land \] \[
\text{SUM Width Proper Plateau}(y, \text{VARIABLES}) \text{ with } sv = \lvert \text{VARIABLES} \rvert
\]

1. \( 4 \ast x + y \leq 2 \ast sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 12: \) \( \lceil (sv - 1)/2 \rceil, 2 \) \( \lceil (sv - 1)/2 \rceil - 1, 6 \)
   - \( sv \mod 2 = 1 \land sv \geq 15: \) \( \lceil (sv - 1)/2 \rceil - 1, 4 \) \( \lceil (sv - 1)/2 \rceil - 2, 8 \)

2. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 5: \) \( (0, sv - 2) \) \( (1, sv - 3) \)

3. \( y \neq 1 \)

\[ \text{NB}_\text{PLAIN}(x, \text{VARIABLES}) \land \] \[
\text{SUM Width Steady Sequence}(y, \text{VARIABLES}) \text{ with } sv = \lvert \text{VARIABLES} \rvert
\]

1. \( 4 \ast x + y \leq 2 \ast sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 6: \) \( \lceil (sv - 1)/2 \rceil, 2 \) \( \lceil (sv - 1)/2 \rceil - 1, 6 \)
   - \( sv \mod 2 = 1 \land sv \geq 9: \) \( \lceil (sv - 1)/2 \rceil - 1, 4 \) \( \lceil (sv - 1)/2 \rceil - 2, 8 \)

2. \( y \neq 1 \)
\( \text{NB}_{\text{PLAIN}}(x, \text{VARIABLES}) \land \\
\text{SUM WIDTH STRICTLY DECREASING SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \)

1. \( 2 \times x + y \leq 2 \times sv - 2 \)
2. \( 2 \times x \leq y \)
3. \( sv \geq 5: (1, 2) \quad (2, 4) \)
4. \( x \neq 0 \lor y \neq 1 \)

\( sv = 12 \)

\( \text{nb}_{\text{plain}} \)

\( sv = 11 \)

\( \text{nb}_{\text{plain}} \)

\( \text{NB}_{\text{PLAIN}}(x, \text{VARIABLES}) \land \\
\text{SUM WIDTH STRICTLY INCREASING SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \)

1. \( 2 \times x + y \leq 2 \times sv - 2 \)
2. \( 2 \times x \leq y \)
3. \( sv \geq 5: (1, 2) \quad (2, 4) \)
4. \( x \neq 0 \lor y \neq 1 \)

\( sv = 12 \)

\( \text{nb}_{\text{plain}} \)

\( sv = 11 \)

\( \text{nb}_{\text{plain}} \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}_\text{PLAIN}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(3 \times x \leq y + sv\)
2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 3: (0, sv - 2) \quad (1, sv - 3)\)

\[sv = 12\]

\[sv = 11\]

\[
\text{NB}_\text{PLAIN}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x + y \leq sv - 1\)
   - \(sv \geq 5: (1, sv - 2) \quad (2, sv - 3)\)
2. \(x \leq y\)
   - \(sv \geq 3: (0, 0) \quad (1, 1)\)
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 4: (1, sv - 2) \quad (1, sv - 3)\)

\[sv = 12\]

\[sv = 11\]
\( \text{NB} \_\text{PLAIN}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{ZIGZAG}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( y \leq 2 * x + 1 \)
2. \( sv \geq 7: (1,3) \land (2,5) \)
3. \( y = 0 \Rightarrow 3 * x \leq sv - sv \mod 3 \)
4. \( x > 0 \land y > 0 \Rightarrow 3 * x \leq y + sv - 2 \)
5. \( y > 0 \Rightarrow 6 * x \leq y + 2 * sv - 1 \)
6. \( sv \mod 2 = 0 \land sv \geq 14: ((sv - 1)/2, sv - 5) \land ((sv - 1)/2 - 1, sv - 11) \)
7. \( sv \mod 2 = 1 \land sv \geq 17: ((sv - 1)/2 - 1, sv - 8) \land ((sv - 1)/2 - 2, sv - 14) \)
8. \( y > 0 \Rightarrow x \geq 1 \)
9. \( sv \geq 5: (1,2) \land (1,3) \)
10. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{PLATEAU}(x, \text{VARIABLES}) \land \text{NB}_\text{PROPER}_\text{PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x + y \leq sv - 1\)
   - \(sv \mod 2 = 0 \land sv \geq 12: \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor, 1\right) \quad \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 1, 3\right)\)
   - \(sv \mod 2 = 1 \land sv \geq 15: \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 1, 2\right) \quad \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 2, 4\right)\)

2. \(y \leq x + 1\)
   - \(sv \geq 7: (0, 1) \quad (1, 2)\)

3. \(sv > 1 \Rightarrow x + 2 \times y \leq sv - 2\)

\[ \text{NB}_\text{PLATEAU}(x, \text{VARIABLES}) \land \text{NB}_\text{PROPER}_\text{PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x + y \leq sv - 1\)
   - \(sv \mod 2 = 0 \land sv \geq 10: \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor, 1\right) \quad \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 1, 3\right)\)
   - \(sv \mod 2 = 1 \land sv \geq 13: \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 1, 2\right) \quad \left(\left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 2, 4\right)\)

2. \(y \leq x\)
   - \(sv \geq 4: (0, 0) \quad (1, 1)\)
\( \text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \text{NB\_STEADY}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \cdot x + y \leq sv - 1 \)
   - \( sv \geq 5 \): (1, \( sv - 3 \)) (2, \( sv - 5 \))

\[ sv = 12 \]
\[ sv = 11 \]

\( \text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \text{NB\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \cdot x + y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \) \( \land \) \( sv \geq 6 \): (\( \lfloor (sv - 1)/2 \rfloor \), 1) (\( \lfloor (sv - 1)/2 \rfloor - 1 \), 3)
   - \( sv \mod 2 = 1 \) \( \land \) \( sv \geq 9 \): (\( \lfloor (sv - 1)/2 \rfloor - 1 \), 2) (\( \lfloor (sv - 1)/2 \rfloor - 2 \), 4)

\[ sv = 12 \]
\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\[ \begin{array}{l}
\leq y \\
sv \geq 5: (1, 1) \quad (2, 2)
\end{array} \]

\[
\begin{array}{ll}
sv = 12 & sv = 11
\end{array}
\]

\[
\begin{array}{ll}
\text{nb\_plateau} & \text{nb\_strictly\_decreasing\_sequence} \\
\text{nb\_plateau} & \text{nb\_strictly\_decreasing\_sequence}
\end{array}
\]

\[
\begin{array}{ll}
\text{nb\_plateau} & \text{nb\_strictly\_increasing\_sequence} \\
\text{nb\_plateau} & \text{nb\_strictly\_increasing\_sequence}
\end{array}
\]

\[
\begin{array}{l}
\leq y \\
sv \geq 5: (1, 1) \quad (2, 2)
\end{array} \]

\[
\begin{array}{ll}
sv = 12 & sv = 11
\end{array}
\]

\[
\begin{array}{ll}
\text{nb\_plateau} & \text{nb\_strictly\_increasing\_sequence} \\
\text{nb\_plateau} & \text{nb\_strictly\_increasing\_sequence}
\end{array}
\]
\(\text{NB}_{-}\text{PLATEAU}(x, \text{VARIABLES}) \land \text{NB}_{-}\text{SUMMIT}(y, \text{VARIABLES})\) with \(sv = |\text{VARIABLES}|\)

1. \(3 \cdot x \leq y + sv - 1\)
   - \(sv \mod 2 = 0 \land sv \geq 10: (\lfloor (sv - 1)/2 \rfloor, \lfloor (sv - 1)/2 \rfloor - 1)\)
   - \(sv \mod 2 = 1 \land sv \geq 3: (0, 0) \land (1, 1)\)

2. \(y \leq x\)
   - \(sv \geq 3: (0, 0) \land (1, 1)\)

\(\text{NB}_{-}\text{PLATEAU}(x, \text{VARIABLES}) \land \text{NB}_{-}\text{VALLEY}(y, \text{VARIABLES})\) with \(sv = |\text{VARIABLES}|\)

1. \(x \leq y + 1\)
   - \(sv \geq 5: (1, 0) \land (2, 1)\)

2. \(y \leq x + 1\)
   - \(sv \geq 5: (0, 1) \land (1, 2)\)

3. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \mod 2 = 1 \land sv \geq 3: (\lfloor (sv - 1)/2 \rfloor, \lfloor (sv - 1)/2 \rfloor - 1) \land (\lfloor (sv - 1)/2 \rfloor - 1, \lfloor (sv - 1)/2 \rfloor)\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{NB\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \cdot x + y \leq sv\)
   - \(sv \mod 2 = 0 \land sv \geq 14: (\lfloor (sv - 1)/2 \rfloor, 2) \land (\lfloor (sv - 1)/2 \rfloor - 1, 4)\)
   - \(sv \mod 2 = 1 \land sv \geq 11: (\lfloor (sv - 1)/2 \rfloor, 1) \land (\lfloor (sv - 1)/2 \rfloor - 1, 3)\)

2. \(y \leq x\)
   - \(sv \geq 4: (0, 0) \land (1, 1)\)

3. \(y = 0 \Rightarrow 3 \cdot x \leq sv - sv \mod 3\)

\[
\text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \cdot x + y \leq 2 \cdot sv - 2\)

2. \(2 \cdot x \leq y\)
   - \(sv \geq 5: (1, 2) \land (2, 4)\)

3. \(x \neq 0 \lor y \neq 1\)
\[ \text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x + y \leq sv - 1\)
   - \(sv \geq 7: (1, sv - 3) \quad (2, sv - 5)\)
2. \(y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2\)
   - \((sv - 1) \mod 2 = 1 \land sv \geq 6: ((sv - 1)/2 - 1, 2) \quad ((sv - 1)/2 - 1, 3)\)
3. \(y \neq 1\)

4. \(sv = 12\)

\[ \text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(3 \times x \leq y + sv\)
2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 3: (0, sv - 2) \quad (1, sv - 3)\)

4. \(sv = 12\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[\text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2x + 2y \leq 3sv - 2 - (3sv - 2) \mod 2\)
   - \(sv \mod 2 = 1 \land \text{sv} \geq 3: (\lfloor (\text{sv} - 1)/2 \rfloor, \text{sv} - 1) \land ((\text{sv} - 1)/2 - 1, \text{sv})\)
2. \(2x \leq y\)
   - \(\text{sv} \geq 5: (1, 2)\)
3. \(x \neq 0 \lor y \neq 1\)

\[\text{SV} = 12\]

\[\text{SV} = 11\]

\[\text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2x + y \leq sv - 1\)
   - \(\text{sv} \geq 7: (1, \text{sv} - 3)\)
   - \(\text{sv} \geq 5: (2, \text{sv} - 5)\)
2. \(y > 0 \Rightarrow 2x \leq sv - 3 - (sv - 3) \mod 2\)
   - \(\text{sv} \mod 2 = 1 \land \text{sv} \geq 6: (\lfloor (\text{sv} - 1)/2 \rfloor - 1, 2)\)
3. \(y \neq 1\)

\[\text{SV} = 12\]

\[\text{SV} = 11\]
\[
\text{NB}_\text{PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \times x \leq y + 1\)
   - \(sv \geq 5: (1,1), (2,3)\)

\[
sv = 12
\]

\[
sv = 11
\]

\[
\text{NB}_\text{PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x + y \leq sv - 1\)
   - \(sv \geq 5: (1, sv - 2), (2, sv - 3)\)
2. \(x \leq y\)
   - \(sv \geq 3: (0,0), (1,1)\)
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 4: (1, sv - 2), (1, sv - 3)\)

\[
sv = 12
\]

\[
sv = 11
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB} & \quad \text{PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM WIDTH PLAIN}(y, \text{VARIABLES}) & \quad \text{with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \( x \leq y + 1 \)
   - \( sv \geq 5: (1, 0) (2, 1) \)

2. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 3: (0, sv - 2) (1, sv - 3) \)

\[ sv = 12 \]

\[ sv = 11 \]

\[
\begin{align*}
\text{NB} & \quad \text{PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM WIDTH PLATEAU}(y, \text{VARIABLES}) & \quad \text{with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 5: (1, sv - 2) (2, sv - 3) \)

2. \( x \leq y \)
   - \( sv \geq 3: (0, 0) (1, 1) \)

3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 4: (1, sv - 2) (1, sv - 3) \)

\[ sv = 12 \]

\[ sv = 11 \]
\[\text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(4 \times x + y \leq 2 \times sv - 2\)
   - \(sv \mod 2 = 0 \land sv \geq 12\): \(((sv - 1)/2, 2)\) \(((sv - 1)/2 - 1, 6)\)
   - \(sv \mod 2 = 1 \land sv \geq 15\): \(((sv - 1)/2 - 1, 4)\) \(((sv - 1)/2 - 2, 8)\)

2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)

3. \(y \neq 1\)

\[\text{sv} = 12 \quad \text{(1)}\]
\[\text{sv} = 11 \quad \text{(1)}\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \(4 \cdot x + y \leq 2 \cdot sv - 2\)
   - \(sv \mod 2 = 0 \land sv \geq 10: \left(\lfloor (sv - 1)/2 \rfloor, 2\right) \left(\lfloor (sv - 1)/2 \rfloor - 1, 6\right)\)
   - \(sv \mod 2 = 1 \land sv \geq 13: \left(\lfloor (sv - 1)/2 \rfloor - 1, 4\right) \left(\lfloor (sv - 1)/2 \rfloor - 2, 8\right)\)

2. \(x + y \leq sv - 1\)
   - \(sv \geq 7: (1, sv - 2) (2, sv - 3)\)
   - \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 5: (1, sv - 2) (1, sv - 3)\)
   - \(y \neq 1\)

\[
\begin{align*}
\text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \(4 \cdot x + y \leq 2 \cdot sv - 2\)
   - \(sv \mod 2 = 0 \land sv \geq 6: \left(\lfloor (sv - 1)/2 \rfloor, 2\right) \left(\lfloor (sv - 1)/2 \rfloor - 1, 6\right)\)
   - \(sv \mod 2 = 1 \land sv \geq 9: \left(\lfloor (sv - 1)/2 \rfloor - 1, 4\right) \left(\lfloor (sv - 1)/2 \rfloor - 2, 8\right)\)

2. \(y \neq 1\)
\( \text{NB}_\text{PLATEAU}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{Strictly}_\text{Decreasing}_\text{Sequence}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \)

1. \( 2 * x + y \leq 2 * sv - 2 \)
2. \( 2 * x \leq y \)
3. \( sv \geq 5: (1, 2) \quad (2, 4) \)
4. \( x \neq 0 \lor y \neq 1 \)

\( sv = 12 \)

\( \text{NB}_\text{PLATEAU}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{Strictly}_\text{Increasing}_\text{Sequence}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \)

1. \( 2 * x + y \leq 2 * sv - 2 \)
2. \( 2 * x \leq y \)
3. \( sv \geq 5: (1, 2) \quad (2, 4) \)
4. \( x \neq 0 \lor y \neq 1 \)

\( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(3 \cdot x \leq y + sv - 1\)
2. \(x + y \leq sv - 1\)
   - \(sv \geq 5: (1, sv - 2) \quad (2, sv - 3)\)
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 4: (1, sv - 2) \quad (1, sv - 3)\)

\[
\text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x \leq y + 1\)
   - \(sv \geq 5: (1, 0) \quad (2, 1)\)
2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 3: (0, sv - 2) \quad (1, sv - 3)\)
\[ \text{NB\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq 2 \times x + 1 \)
   - \( sv \geq 7: \ (1,3) \quad (2,5) \)
2. \( y = 0 \Rightarrow 3 \times x \leq sv - sv \mod 3 \)
3. \( x > 0 \land y > 0 \Rightarrow 3 \times x \leq y + sv - 2 \)

4. \( y > 0 \Rightarrow 6 \times x \leq y + 2 \times sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 14: \ ([sv-1]/2, sv-5) \quad ([sv-1]/2 - 1, sv-11) \)
   - \( sv \mod 2 = 1 \land sv \geq 17: \ ([sv-1]/2 - 1, sv-8) \quad ([sv-1]/2 - 2, sv-14) \)
5. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 5: \ (1,2) \quad (1,3) \)
6. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_{\text{proper\_plain}}(x, \text{VARIABLES}) \land \text{NB}_{\text{proper\_plateau}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 3x + y \leq sv - 1 \)
   - \( sv \mod 3 = 0 \land sv \geq 24: (\lfloor (sv - 1)/3 \rfloor, 2) \quad (\lfloor (sv - 1)/3 \rfloor - 1, 5) \)
   - \( sv \mod 3 = 1 \land sv \geq 28: (\lfloor (sv - 1)/3 \rfloor - 1, 3) \quad (\lfloor (sv - 1)/3 \rfloor - 2, 6) \)
   - \( sv \mod 3 = 2 \land sv \geq 20: (\lfloor (sv - 1)/3 \rfloor, 1) \quad (\lfloor (sv - 1)/3 \rfloor - 1, 4) \)

2. \( x + 3y \leq sv - 1 \)
   - \( sv \mod 3 = 0 \land sv \geq 24: (2, \lfloor (sv - 1)/3 \rfloor) \quad (5, \lfloor (sv - 1)/3 \rfloor - 1) \)
   - \( sv \mod 3 = 1 \land sv \geq 28: (3, \lfloor (sv - 1)/3 \rfloor - 1) \quad (6, \lfloor (sv - 1)/3 \rfloor - 2) \)
   - \( sv \mod 3 = 2 \land sv \geq 20: (1, \lfloor (sv - 1)/3 \rfloor) \quad (4, \lfloor (sv - 1)/3 \rfloor - 1) \)

3. \( sv > 1 \Rightarrow 2x + 2y \leq sv - 2 - (sv - 2) \mod 2 \)

4. \( o^\omega \geq x + y \)

\[ \begin{align*}
\text{sv} = 12 & \quad & \text{sv} = 11
\end{align*} \]
\[ \text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{NB\_STEADY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 * x + y \leq sv - 1\)
   - \(sv \geq 7: \ (1, sv - 3) \quad (2, sv - 5)\)
2. \(x \leq y\)
   - \(sv \geq 7: \ (1, 1) \quad (2, 2)\)

\[ sv = 12 \]

\[ sv = 11 \]
\[ \text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + 2y \leq sv \)
   - \( sv \mod 2 = 0 \land sv \geq 8: \quad (0, \lfloor sv/2 \rfloor) \quad (2, \lfloor sv/2 \rfloor - 1) \)
   - \( sv \mod 2 = 1 \land sv \geq 11: \quad (1, \lfloor sv/2 \rfloor) \quad (3, \lfloor sv/2 \rfloor - 1) \)
2. \( x \leq y \)
   - \( sv \geq 7: \quad (1, 1) \quad (2, 2) \)

\[ \text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + 2y \leq sv \)
   - \( sv \mod 2 = 0 \land sv \geq 8: \quad (0, \lfloor sv/2 \rfloor) \quad (2, \lfloor sv/2 \rfloor - 1) \)
   - \( sv \mod 2 = 1 \land sv \geq 11: \quad (1, \lfloor sv/2 \rfloor) \quad (3, \lfloor sv/2 \rfloor - 1) \)
2. \( x \leq y \)
   - \( sv \geq 7: \quad (1, 1) \quad (2, 2) \)
\( \text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{NB\_SUMMIT}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 4 \times x \leq y + sv \)
2. \( x + 2 \times y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 12: (1, \lfloor (sv - 1)/2 \rfloor) \)
   - \( (3, \lfloor (sv - 1)/2 \rfloor - 1) \)
   - \( sv \mod 2 = 1 \land sv \geq 15: (2, \lfloor (sv - 1)/2 \rfloor - 1) \)
   - \( (4, \lfloor (sv - 1)/2 \rfloor - 2) \)
3. \( sv > 1 \Rightarrow 2 \times x + y \leq sv - 2 \)

- \( sv = 12 \)
- \( sv = 11 \)

\( \text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{NB\_VALLEY}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x + 2 \times y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 10: (1, \lfloor (sv - 1)/2 \rfloor) \)
   - \( (3, \lfloor (sv - 1)/2 \rfloor - 1) \)
   - \( sv \mod 2 = 1 \land sv \geq 13: (2, \lfloor (sv - 1)/2 \rfloor - 1) \)
   - \( (4, \lfloor (sv - 1)/2 \rfloor - 2) \)
2. \( x \leq y \)
   - \( sv \geq 4: (0, 0) \)
   - \( (1, 1) \)

- \( sv = 12 \)
- \( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
&\text{NB} \_ \text{PROPER} \_ \text{PLAIN}(x, \text{VARIABLES}) \land \\
&\text{NB} \_ \text{ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \[3 \times x + 2 \times y \leq sv - 1\]
   - \(sv \mod 3 = 0 \land sv \geq 18\): \((\lfloor sv - 1 \rfloor / 3, 1)\) \((\lfloor sv - 1 \rfloor / 3 - 2, 4)\)
   - \(sv \mod 3 = 2 \land sv \geq 23\): \((\lfloor sv - 1 \rfloor / 3 - 1, 2)\) \((\lfloor sv - 1 \rfloor / 3 - 3, 5)\)

2. \[2 \times x + 3 \times y \leq sv\]
   - \(sv \mod 3 = 0 \land sv \geq 33\): \((3, \lfloor sv - 1 \rfloor / 3 - 1)\) \((6, \lfloor sv - 1 \rfloor / 3 - 3)\)
   - \(sv \mod 3 = 1 \land sv \geq 28\): \((2, \lfloor sv - 1 \rfloor / 3 - 1)\) \((5, \lfloor sv - 1 \rfloor / 3 - 3)\)
   - \(sv \mod 3 = 2 \land sv \geq 23\): \((1, \lfloor sv - 1 \rfloor / 3)\) \((4, \lfloor sv - 1 \rfloor / 3 - 2)\)
\[
\text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \cdot x \leq y\)
2. \(x + y \leq sv\)
3. \(sv \geq 7: (1, 2) \ (2, 4)\)
4. \(sv \geq 5: (0, sv) \ (1, sv - 1)\)
5. \(sv > 1 \Rightarrow 4 \cdot x + y \leq 2 \cdot sv - 2 - (sv \mod 3 = 0)\)
6. \(x \neq 0 \lor y \neq 1\)

\[
\begin{align*}
sv = 12 & \quad & sv = 11 \\
\end{align*}
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_PROPER\_PLAIN}(x, \text{VAR}) \land \text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VAR}) \text{ with } sv = |\text{VAR}| \]

1. \( 3 \cdot x + y \leq sv - 1 \)
   - \( sv \geq 10: \) (1, \( sv - 4 \)) (2, \( sv - 7 \))
2. \( y > 0 \Rightarrow 3 \cdot x \leq sv - 3 - (sv - 3) \mod 3 \)
   - \( sv \mod 3 = 1 \land sv \geq 7: \) (\( \lfloor (sv - 1)/3 \rfloor - 1, 2 \)) (\( \lfloor (sv - 1)/3 \rfloor - 1, 3 \))
   - \( sv \mod 3 = 2 \land sv \geq 5: \) (\( \lfloor (sv - 1)/3 \rfloor - 1, 2 \)) (\( \lfloor (sv - 1)/3 \rfloor - 1, 3 \))
3. \( sv \mod 3 = 1 \land sv \geq 7: \) (1, \( sv - 5 \)) (2, \( sv - 8 \))
4. \( y \neq 1 \)

\[ \text{NB\_PROPER\_PLAIN}(x, \text{VAR}) \land \text{SUM\_WIDTH\_GORGE}(y, \text{VAR}) \text{ with } sv = |\text{VAR}| \]

1. \( x > 0 \land y > 0 \Rightarrow 3 \cdot x + y \leq sv - 2 \)
   - \( sv \geq 8: \) (1, \( sv - 5 \)) (2, \( sv - 8 \))
\text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \(2 \times x \leq y\)
   - sv \geq 7: (1, 2) (2, 4)

2. \(x + y \leq sv\)
   - sv \geq 5: (0, sv) (1, sv - 1)

3. \(sv > 1 \Rightarrow 4 \times x + y \leq 2 \times sv - 2 - (sv \mod 3 = 0)\)

4. \(x \neq 0 \lor y \neq 1\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(3 \ast x + y \leq sv - 1\)
   - \(sv \geq 9: (1, sv - 4) \ (2, sv - 7)\)
2. \(y > 0 \Rightarrow 3 \ast x \leq sv - 3 - (sv - 3) \mod 3\)
   - \(sv \mod 3 = 1 \land sv \geq 7: \left(\left\lfloor \frac{(sv - 1)}{3} \right\rfloor - 1, 2\right) \ (\left\lfloor \frac{(sv - 1)}{3} \right\rfloor - 1, 3)\)
   - \(sv \mod 3 = 2 \land sv \geq 5: \left(\left\lfloor \frac{(sv - 1)}{3} \right\rfloor - 1, 2\right) \ (\left\lfloor \frac{(sv - 1)}{3} \right\rfloor - 1, 3)\)
3. \(sv \mod 3 = 0 \geq x + \left\lfloor \frac{(y + 1)}{2} \right\rfloor\)
4. \(y \neq 1\)

\[ \text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(3 \ast x \leq y + 1\)
   - \(sv \geq 7: (1, 2) \ (2, 5)\)
\[ \text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y + 1 \)
   - \( sv \geq 7: (1, 0) \quad (2, 1) \)
2. \( sv > 1 \Rightarrow 2 \cdot x + y \leq sv - 2 \)
   - \( sv \geq 4: (0, sv - 2) \quad (1, sv - 4) \)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2 \cdot x \leq y \)
   - \( sv \geq 4: (0, 0) \quad (1, 2) \)
2. \( x + y \leq sv - 1 \)
   - \( sv \geq 7: (1, sv - 2) \quad (2, sv - 3) \)
3. \( x = 0 \Rightarrow 2 \cdot y \leq sv - 1 - (sv - 1) \text{ mod } 2 \)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x \leq y + 1 \)
   - \( sv \geq 7 \): \((1,0)\) \((2,1)\)
2. \( sv > 1 \Rightarrow 2x + y \leq sv - 2 \)
   - \( sv \geq 4 \): \((0,sv - 2)\) \((1,sv - 4)\)

\[
\begin{align*}
sv &= 12 \\
sv &= 11
\end{align*}
\]

\[
\text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( 2x \leq y \)
   - \( sv \geq 4 \): \((0,0)\) \((1,2)\)
2. \( x + y \leq sv - 1 \)
   - \( sv \geq 7 \): \((1,sv - 2)\) \((2,sv - 3)\)
3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 5 \): \((1,sv - 2)\) \((1,sv - 3)\)

\[
\begin{align*}
sv &= 12 \\
sv &= 11
\end{align*}
\]
\[
\text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x = \lfloor (sv - 1)/3 \rfloor - 1 \Rightarrow y \leq 6 + 2 \times (sv - 1) \mod 3 \)
2. \( x = \lfloor (sv - 1)/3 \rfloor \Rightarrow y \leq 2 \times (sv - 1) \mod 3 \)
3. \( sv > 1 \Rightarrow 2 \times x + y \leq sv - 2 \)
   - \( sv \geq 0: \ (0, sv - 2) \quad (1, sv - 4) \)
4. \( x > 0 \land y > 0 \Rightarrow 3 \times x \leq sv - 2 - (sv - 2) \mod 3 \)
   - \( (sv - 2) \mod 3 = 1 \land sv \geq 9: \ ((sv - 1)/3, 2) \quad ((sv - 1)/3, 3) \)
   - \( (sv - 2) \mod 3 = 2 \land sv \geq 7: \ ((sv - 1)/3 - 1, 2) \quad ((sv - 1)/3 - 1, 3) \)
5. \( o = x + \lfloor (y + 1)/2 \rfloor \)
6. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[\text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(4 \cdot x + y \leq 2 \cdot sv - 2\)
2. \(2 \cdot x \leq y\)
3. \(sv \geq 7: (1, 2) \quad (2, 4)\)
4. \(x \neq 0 \lor y \neq 1\)

\[\text{sv} = 12\]

\[\text{sv} = 11\]

\[\text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRONGLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(2 \cdot x \leq y\)
2. \(sv \geq 7: (1, 2) \quad (2, 4)\)
3. \(sv \geq 8: (1, sv - 1) \quad (2, sv - 2)\)
4. \(sv > 1 \Rightarrow 4 \cdot x + y \leq 2 \cdot sv - 2 - (sv \mod 3 = 0)\)
5. \(x \neq 0 \lor y \neq 1\)

\[\text{sv} = 12\]

\[\text{sv} = 11\]
\( \text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x \leq y \)
2. \( x + y \leq sv \)
3. \( sv > 1 \Rightarrow 4 \times x + y \leq 2 \times sv - 2 - (sv \mod 3 = 0) \)
4. \( x \neq 0 \lor y \neq 1 \)

\( \text{NB\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 4 \times x \leq y + sv \)
2. \( sv > 1 \Rightarrow 2 \times x + y \leq sv - 2 \)
3. \( sv \geq 4: (0, sv - 2) \quad (1, sv - 4) \)
A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB} \_\text{PROPER} \_\text{PLAIN}(x, \text{VARIABLES}) \land \\
\text{SUM} _\text{WIDTH} _\text{VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \[2 \times x \leq y\]
   - \(sv \geq 4\): (0, 0) (1, 2)
2. \[x + y \leq sv - 1\]
   - \(sv \geq 7\): (1, \(sv - 2\)) (2, \(sv - 3\))
\[\text{NB PROPER PLAIN}(x, \text{VARIABLES}) \land \text{SUM WIDTH ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(x = \lfloor (sv - 1)/3 \rfloor - 1 \Rightarrow y \leq 6 - 3 \times (sv \mod 3 = 1)\)
2. \(x = \lfloor (sv - 1)/3 \rfloor - 2 \Rightarrow y \leq 9 + 3 \times (sv \mod 3 = 0)\)
3. \(x = \lfloor (sv - 1)/3 \rfloor \Rightarrow y \leq 3 \times (sv \mod 3 = 0)\)
4. \(sv > 1 \Rightarrow 2 \times x + y \leq sv - 2\)
   - \(sv \geq 6: (0, sv - 2) \ (1, sv - 4)\)
5. \(x > 0 \land y > 0 \Rightarrow 3 \times x \leq sv - 3 - (sv - 3) \mod 3\)
   - \(sv \mod 3 = 0 \land sv \geq 9: \ ((sv - 1)/3, 2) \ ((sv - 1)/3, 3)\)
   - \(sv \mod 3 = 1 \land sv \geq 7: \ ((sv - 1)/3 - 1, 2) \ ((sv - 1)/3 - 1, 3)\)
   - \(sv \mod 3 = 2 \land sv \geq 5: \ ((sv - 1)/3 - 1, 2) \ ((sv - 1)/3 - 1, 3)\)
6. \(y \neq 1\)

\[\begin{array}{c|c|c}
sv = 12 & \text{nb_proper_plain} & \text{sum_width_zigzag} \\
0 & 2 & 10 \\
1 & 3 & 9 \\
2 & 4 & 8 \\
3 & 5 & 7 \\
4 & 6 & 6 \\
5 & 7 & 5 \\
6 & 8 & 4 \\
7 & 9 & 3 \\
8 & 10 & 2 \\
\end{array}\]

\[\begin{array}{c|c|c}
sv = 11 & \text{nb_proper_plain} & \text{sum_width_zigzag} \\
0 & 2 & 8 \\
1 & 3 & 7 \\
2 & 4 & 6 \\
3 & 5 & 5 \\
4 & 6 & 4 \\
5 & 7 & 3 \\
6 & 8 & 2 \\
\end{array}\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\(\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \text{NB\_STEADY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\)

1. \(2 \times x + y \leq sv - 1\)
   - \(sv \geq 7: (1, sv - 3) \quad (2, sv - 5)\)
2. \(x \leq y\)
   - \(sv \geq 7: (1, 1) \quad (2, 2)\)

\(\text{sv} = 12\)

\(\text{sv} = 11\)

\(\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \text{NB\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\)

1. \(2 \times x + y \leq sv - 1\)
2. \(x \leq y\)
   - \(sv \geq 7: (1, 1) \quad (2, 2)\)

\(\text{sv} = 12\)

\(\text{sv} = 11\)
\[ \mathbf{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \mathbf{NB\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + 2 \cdot y \leq sv \)
   - \( sv \mod 2 = 0 \land sv \geq 8: \ (0, \lfloor sv/2 \rfloor) \) \( (2, \lfloor sv/2 \rfloor - 1) \)
   - \( sv \mod 2 = 1 \land sv \geq 11: \ (1, \lfloor sv/2 \rfloor) \) \( (3, \lfloor sv/2 \rfloor - 1) \)

2. \( x \leq y \)
   - \( sv \geq 7: \ (1, 1) \) \( (2, 2) \)

\[ \mathbf{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \mathbf{NB\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + 2 \cdot y \leq sv \)
   - \( sv \mod 2 = 0 \land sv \geq 8: \ (0, \lfloor sv/2 \rfloor) \) \( (2, \lfloor sv/2 \rfloor - 1) \)
   - \( sv \mod 2 = 1 \land sv \geq 11: \ (1, \lfloor sv/2 \rfloor) \) \( (3, \lfloor sv/2 \rfloor - 1) \)

2. \( x \leq y \)
   - \( sv \geq 7: \ (1, 1) \) \( (2, 2) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{NB\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(3 \times x + 2 \times y \leq sv - 1\)
2. \(sv \mod 2 = 0 \land sv \geq 10: (1, \lfloor (sv - 1)/2 \rfloor - 1) \quad (3, \lfloor (sv - 1)/2 \rfloor - 4)\)

\(sv = 12\)

\(sv = 11\)

\[
\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{NB\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x + 2 \times y \leq sv - 1\)
2. \(sv \mod 2 = 0 \land sv \geq 12: (1, \lfloor (sv - 1)/2 \rfloor) \quad (3, \lfloor (sv - 1)/2 \rfloor - 1)\)
3. \(sv \mod 2 = 1 \land sv \geq 15: (2, \lfloor (sv - 1)/2 \rfloor - 1) \quad (4, \lfloor (sv - 1)/2 \rfloor - 2)\)
4. \(x \leq y + 1\)
5. \(sv \geq 7: (1, 0) \quad (2, 1)\)
6. \(sv > 1 \Rightarrow 2 \times x + y \leq sv - 2\)

\(sv = 12\)

\(sv = 11\)
\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \text{NB\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

\begin{enumerate}
\item \(3 \times x + 2 \times y \leq sv - 1\)
\begin{itemize}
\item \(sv \mod 3 = 0 \land sv \geq 18: \left(\left\lfloor (sv - 1)/3 \right\rfloor, 1\right) \quad \left(\left\lfloor (sv - 1)/3 \right\rfloor - 2, 4\right)\)
\item \(sv \mod 3 = 2 \land sv \geq 23: \left(\left\lfloor (sv - 1)/3 \right\rfloor - 1, 2\right) \quad \left(\left\lfloor (sv - 1)/3 \right\rfloor - 3, 5\right)\)
\end{itemize}

\item \(2 \times x + 3 \times y \leq sv\)
\begin{itemize}
\item \(sv \mod 3 = 0 \land sv \geq 33: \left(3, \left\lfloor (sv - 1)/3 \right\rfloor - 1\right) \quad \left(6, \left\lfloor (sv - 1)/3 \right\rfloor - 3\right)\)
\item \(sv \mod 3 = 1 \land sv \geq 28: \left(2, \left\lfloor (sv - 1)/3 \right\rfloor - 1\right) \quad \left(5, \left\lfloor (sv - 1)/3 \right\rfloor - 3\right)\)
\item \(sv \mod 3 = 2 \land sv \geq 23: \left(1, \left\lfloor (sv - 1)/3 \right\rfloor\right) \quad \left(4, \left\lfloor (sv - 1)/3 \right\rfloor - 2\right)\)
\end{itemize}
\end{enumerate}

\(sv = 12\)

\(sv = 11\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \(2 \cdot x \leq y\)
2. \(sv \geq 7: (1, 2) (2, 4)\)
3. \(sv \geq 5: (0, sv) (1, sv - 1)\)
4. \(sv > 1 \Rightarrow 4 \cdot x + y \leq 2 \cdot sv - 2 - (sv \mod 3 = 0)\)
5. \(x \neq 0 \lor y \neq 1\)
\textbf{NB\_PROPER\_PLATEAU}(x, VARIABLES) \land \\
\textbf{SUM\_WIDTH\_DECREASING\_TERRACE}(y, VARIABLES) \text{ with } sv = |VARIABLES|

1. \(3 \times x + y \leq sv - 1\)
   \(sv \geq 9: (1, sv - 4) \quad (2, sv - 7)\)

2. \(y > 0 \Rightarrow 3 \times x \leq sv - 3 - (sv - 3) \mod 3\)
   \(sv \mod 3 = 1 \land sv \geq 7: ((sv - 1)/3 - 1, 2) \quad ((sv - 1)/3 - 1, 3)\)
   \(sv \mod 3 = 2 \land sv \geq 5: ((sv - 1)/3 - 1, 2) \quad ((sv - 1)/3 - 1, 3)\)

3. \(o^n \geq x + [(y + 1)/2]\)
4. \(y \neq 1\)

\textbf{NB\_PROPER\_PLATEAU}(x, VARIABLES) \land \\
\textbf{SUM\_WIDTH\_GORGE}(y, VARIABLES) \text{ with } sv = |VARIABLES|

1. \(4 \times x \leq y + sv\)
2. \(sv > 1 \Rightarrow 2 \times x + y \leq sv - 2\)
   \(sv \geq 4: (0, sv - 2) \quad (1, sv - 4)\)

\text{Graphs for } sv = 12 \text{ and } sv = 11.
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) & \text{ with } sv = |\text{VARIABLES}| \\
1 \ & \ 2 \ast x \leq y \\
\ & \ sv \geq 7: \ (1, 2), \ (2, 4) \\
\ & \ sv \geq 5: \ (0, sv), \ (1, sv - 1) \\
\ & \ sv > 1 \ \Rightarrow \ 4 \ast x + y \leq 2 \ast sv - 2 - (sv \mod 3 = 0) \\
\ & \ x \neq 0 \lor y \neq 1
\end{align*}
\]
\(\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \
\text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\)

1. \(3 \cdot x + y \leq sv - 1\)
   - \(sv \geq 10: \ (1, sv - 4) \ (2, sv - 7)\)
2. \(y > 0 \Rightarrow 3 \cdot x \leq sv - 3 - (sv - 3) \mod 3\)
   - \(sv \mod 3 = 1 \land sv \geq 7: \ (((sv - 1)/3) - 1, 2) \ (((sv - 1)/3) - 1, 3)\)
   - \(sv \mod 3 = 2 \land sv \geq 5: \ (((sv - 1)/3) - 1, 2) \ (((sv - 1)/3) - 1, 3)\)
3. \(sv \mod 3 = 1 \land sv \geq 7: \ ((sv - 1)/3 - 1, 2) \ ((sv - 1)/3 - 1, 3)\)
4. \(sv \mod 3 = 2 \land sv \geq 5: \ ((sv - 1)/3 - 1, 2) \ ((sv - 1)/3 - 1, 3)\)
5. \(y \neq 1\)

\(\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \
\text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\)

6. \(3 \cdot x \leq y + 1\)
   - \(sv \geq 7: \ (1, 2) \ (2, 5)\)

\(\text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\)

\(\text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \(2x \leq y\)
   - \(sv \geq 4\): \((0, 0), (1, 2)\)
2. \(x + y \leq sv - 1\)
   - \(sv \geq 7\): \((1, sv - 2), (2, sv - 3)\)

\[
\begin{align*}
\text{SV} = 12
\end{align*}
\]

\[
\begin{align*}
\text{SV} = 11
\end{align*}
\]

\[
\begin{align*}
\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \(x \leq y + 1\)
   - \(sv \geq 7\): \((1, 0), (2, 1)\)
2. \(sv > 1 \Rightarrow 2x + y \leq sv - 2\)
   - \(sv \geq 4\): \((0, sv - 2), (1, sv - 4)\)

\[
\begin{align*}
\text{SV} = 12
\end{align*}
\]

\[
\begin{align*}
\text{SV} = 11
\end{align*}
\]
\[ \text{NB_PROPER_PLATEAU}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x \leq y\)
   - \(sv \geq 4: (0, 0) \quad (1, 2)\)
2. \(x + y \leq sv - 1\)
   - \(sv \geq 7: (1, sv - 2) \quad (2, sv - 3)\)
3. \(x = 0 \Rightarrow 2 \times y \leq sv - 1 - (sv - 1) \mod 2\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x = \lfloor (sv - 1)/3 \rfloor - 1 \Rightarrow y \leq 6 + 2 \ast (sv - 1) \mod 3 \)
2. \( x = \lfloor (sv - 1)/3 \rfloor \Rightarrow y \leq 2 \ast (sv - 1) \mod 3 \)
3. \( sv > 1 \Rightarrow 2 \ast x + y \leq sv - 2 \)
   \( sv \geq 6: \ (0, sv - 2) \ (1, sv - 4) \)
4. \( x > 0 \land y > 0 \Rightarrow 3 \ast x \leq sv - 2 - (sv - 2) \mod 3 \)
   \( (sv - 2) \mod 3 = 1 \land sv \geq 9: \ ((sv - 1)/3, 2) \ ([(sv - 1)/3], 3) \)
   \( (sv - 2) \mod 3 = 2 \land sv \geq 7: \ ((sv - 1)/3 - 1, 2) \ ([(sv - 1)/3], 1, 3) \)
5. \( o \geq x + \lfloor (y + 1)/2 \rfloor \)
6. \( y \neq 1 \)
\[ \text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x \leq y\)
   - \(sv \geq 4: (0,0)\) (1,2)
2. \(x + y \leq sv - 1\)
   - \(sv \geq 7: (1, sv - 2)\) (2, \(sv - 3\))
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 5: (1, sv - 2)\) (1, \(sv - 3\))

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(4 \times x + y \leq 2 \times sv - 2\)
2. \(2 \times x \leq y\)
   - \(sv \geq 7: (1,2)\) (2, 4)
3. \(x \neq 0 \vee y \neq 1\)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{proper_plateau}(x, \text{VARIABLES}) \land \text{SUM}_\text{width_strictly_decreasing_sequence}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2 \ast x \leq y \)
   - \( sv \geq 7: (1, 2) \ (2, 4) \)
2. \( x + y \leq sv \)
   - \( sv \geq 8: (1, sv - 1) \ (2, sv - 2) \)
3. \( sv > 1 \Rightarrow 4 \ast x + y \leq 2 \ast sv - 2 - (sv \mod 3 = 0) \)
4. \( x \neq 0 \lor y \neq 1 \)

\[ \text{NB}_\text{proper_plateau}(x, \text{VARIABLES}) \land \text{SUM}_\text{width_strictly_increasing_sequence}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2 \ast x \leq y \)
   - \( sv \geq 7: (1, 2) \ (2, 4) \)
2. \( x + y \leq sv \)
   - \( sv \geq 8: (1, sv - 1) \ (2, sv - 2) \)
3. \( sv > 1 \Rightarrow 4 \ast x + y \leq 2 \ast sv - 2 - (sv \mod 3 = 0) \)
4. \( x \neq 0 \lor y \neq 1 \)
\[
\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x > 0 \land y > 0 \Rightarrow 3 \times x + y \leq sv - 2\)
2. \(sv \geq 8: (1, sv - 5) \quad (2, sv - 8)\)

\[
\begin{align*}
sv = 12 & \quad \text{sum\_width\_summit} \\
0 & \quad 2 \\
0 & \quad 4 \\
0 & \quad 6 \\
0 & \quad 8 \\
0 & \quad 10
\end{align*}
\]

\[
\begin{align*}
sv = 11 & \quad \text{sum\_width\_summit} \\
0 & \quad 2 \\
0 & \quad 4 \\
0 & \quad 6 \\
0 & \quad 8
\end{align*}
\]

\[
\text{NB\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x \leq y + 1\)
2. \(sv \geq 7: (1, 0) \quad (2, 1)\)
3. \(sv > 1 \Rightarrow 2 \times x + y \leq sv - 2\)
4. \(sv \geq 4: (0, sv - 2) \quad (1, sv - 4)\)

\[
\begin{align*}
sv = 12 & \quad \text{sum\_width\_valley} \\
0 & \quad 2 \\
0 & \quad 4 \\
0 & \quad 6 \\
0 & \quad 8 \\
0 & \quad 10
\end{align*}
\]

\[
\begin{align*}
sv = 11 & \quad \text{sum\_width\_valley} \\
0 & \quad 2 \\
0 & \quad 4 \\
0 & \quad 6 \\
0 & \quad 8
\end{align*}
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB PROPER PLATEAU}(x, \text{VARIABLES}) \land \text{SUM WIDTH ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x = \left\lfloor \frac{(sv - 1)}{3} \right\rfloor - 1 \Rightarrow y \leq 6 - 3 \times (sv \mod 3 = 1) \]
2. \[ x = \left\lfloor \frac{(sv - 1)}{3} \right\rfloor - 2 \Rightarrow y \leq 8 + 3 \times (sv \mod 3 = 0) \]
3. \[ x = \left\lfloor \frac{(sv - 1)}{3} \right\rfloor \Rightarrow y \leq 2 \times (sv \mod 3 = 0) \]
4. \[ sv > 1 \Rightarrow 2 \times x + y \leq sv - 2 \]
5. \[ sv \geq 6: (0, sv - 2) \land (1, sv - 4) \]
6. \[ sv > 0 \land y > 0 \Rightarrow 3 \times x \leq sv - 3 - (sv - 3) \mod 3 \]
7. \[ sv \mod 3 = 0 \land sv \geq 9: (\left\lfloor \frac{(sv - 1)}{3} \right\rfloor, 2) \land (\left\lfloor \frac{(sv - 1)}{3} \right\rfloor, 3) \]
8. \[ sv \mod 3 = 1 \land sv \geq 7: (\left\lfloor \frac{(sv - 1)}{3} \right\rfloor - 1, 2) \land (\left\lfloor \frac{(sv - 1)}{3} \right\rfloor - 1, 3) \]
9. \[ sv \mod 3 = 2 \land sv \geq 5: (\left\lfloor \frac{(sv - 1)}{3} \right\rfloor - 1, 2) \land (\left\lfloor \frac{(sv - 1)}{3} \right\rfloor - 1, 3) \]
10. \[ y \neq 1 \]
\[
\begin{align*}
\text{NB}_\text{STEADY}(x, \text{VARIABLES}) \land \\
\text{NB}_\text{STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \(x + y \leq sv\)
   - \(sv \geq 0\): \((sv - 2, 2)\) \((sv - 3, 3)\)

2. \(y \leq x\)
   - \(sv \geq 4\): \((1, 1)\) \((2, 2)\)

3. \(sv > 1 \Rightarrow x \leq y + sv - 2\)

4. \(x > 0 \Rightarrow y \geq 1\)
   - \(sv \geq 4\): \((sv - 2, 1)\) \((sv - 3, 1)\)

\[
\begin{align*}
\text{NB}_\text{STEADY}(x, \text{VARIABLES}) \land \\
\text{NB}_\text{STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \(x + y \leq sv - 1\)
   - \(sv \geq 4\): \((sv - 2, 1)\) \((sv - 3, 2)\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_STEADY}(x, \text{VARIABLES}) \land \]
\[ \text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\[ 1. \ x + y \leq sv - 1 \]
\[ \square \ sv \geq 4: (sv - 2, 1) \quad (sv - 3, 2) \]

\[ \text{NB\_STEADY}(x, \text{VARIABLES}) \land \]
\[ \text{NB\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\[ 1. \ x + 2y \leq sv - 1 \]
\[ \square \ sv \geq 5: (sv - 3, 1) \quad (sv - 5, 2) \]
\( \text{NB}_\text{STEHAY}(x, \text{VARIABLES}) \land \text{NB}_\text{VALLEY}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x + 2 \cdot y \leq sv - 1 \)
   - \( sv \geq 5: (sv - 3, 1) \quad (sv - 5, 2) \)

\( sv = 12 \)

\( sv = 11 \)

\( \text{NB}_\text{STEHAY}(x, \text{VARIABLES}) \land \text{NB}_\text{ZIGZAG}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x + 3 \cdot y \leq sv - 1 \)
   - \( sv \geq 7: (sv - 4, 1) \quad (sv - 7, 2) \)

\( sv = 12 \)

\( sv = 11 \)

\( \text{NB}_\text{STEHAY}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{DECREASING}_\text{SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x = sv - 2 \Rightarrow y \leq 2 \)
2. \( sv > 1 \Rightarrow 2 \cdot x + y \leq 3 \cdot sv - 4 \)
3. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 2 \)
   
   \[
   \begin{align*}
   x &\neq \max(0, sv - 1) - 1, \\
   y &\neq sv \cdot \min(1, \max(0, sv - 1)) - 4, \\
   sv &< 7 \\
   \end{align*}
   \]
4. \( x \neq \max(0, sv - 1) - 1 \lor y < 3 \lor y > sv \cdot \min(1, \max(0, sv - 1)) - 3 \\
   \) \( y < 1, \quad y > sv \cdot \min(1, \max(0, sv - 1)) - 3, \quad 0 = y \mod 2 \)
5. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{STEADY}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{DECREASING}_\text{TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq 2 \times x \)
2. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \)
3. \( sv \geq 5: (sv - 3, sv - 2) \land (sv - 3, sv - 3) \)
   \( \lor \left( x \neq \max(0, sv - 1) - 1, \right. \)
   \( \left. y < 1, \right) \)
   \( y > (sv - 2) \times \min(1, \max(0, sv - 3)) - 2 \)
4. \( y \neq 1 \)
\[ \text{NB\_STEADY}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\[ x = sv - 3 \Rightarrow y \leq 1 \]
\[ x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \]
\[ (x \neq \max(0, sv - 1) - 1, \]
\[ y < 1, \]
\[ y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 2 \]
\[ x \neq \max(0, sv - 1) - 2, \]
\[ y < 2, \]
\[ y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 1 \]
\[ \text{NB\_STEADY}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x = sv - 2 \Rightarrow y \leq 2 \)
2. \( sv > 1 \Rightarrow 2x + y \leq 3sv - 4 \)
3. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 2 \)
4. \( x \neq \max(0, sv - 1) - 1, \\
y \neq sv \times \min(1, \max(0, sv - 1)) - 4, \\
sv < 7 \)
5. \( x \neq \max(0, sv - 1) - 1 \lor y < 3 \lor y > sv \times \min(1, \max(0, sv - 1)) - 3 \)
6. \( x \neq \max(0, sv - 1) - 1, \\
y < 1, \\
y > sv \times \min(1, \max(0, sv - 1)) - 3, \\
0 = y \mod 2 \)
7. \( y \neq 1 \)
\(NB\_STEADY(x, VARIABLES) \land \sumwidth\_increasing\_terrace(y, VARIABLES)\) with \(sv = |VARIABLES|\)

1. \(y \leq 2 \times x\)
2. \(x > 0 \land y > 0 \Rightarrow x \leq sv - 3\)
3. \(sv \geq 6: (1, 2) (2, 4)\)
4. \(sv \geq 5: (sv - 3, sv - 2) (sv - 3, sv - 3)\)
5. \(\forall x \neq \max(0, sv - 1) - 1,\)
6. \(y < 1,\)
7. \(y > (sv - 2) \times \min(1, \max(0, sv - 3)) - 2\)
8. \(y \neq 1\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{STEADY}(x, \text{VARIABLES}) \land \]
\[ \text{SUM}_\text{WIDTH}_\text{INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\begin{align*}
1. & \quad x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \\
2. & \quad sv \geq 3: (sv - 3, sv - 2) \quad (sv - 3, sv - 3) \\
& \quad x \neq \max(0, sv - 1) - 1, \\
& \quad y < 1, \\
& \quad y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 2
\end{align*}
\[
\begin{align*}
\text{NB\_STEADY}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

\begin{enumerate}
\item \[2 \cdot y \leq x + sv - 1\]
\item \[sv \geq 5: (sv - 3, sv - 2) \quad (sv - 5, sv - 3)\]
\item \[x > 0 \land y > 0 \Rightarrow x \leq sv - 3\]
\item \[sv \geq 3: (sv - 3, sv - 2) \quad (sv - 3, sv - 3)\]
\item \[sv \geq 3: (sv - 3, sv - 2) \quad (sv - 3, sv - 3)\]
\item \[\bigvee \left( \begin{array}{c}
x \neq \max(0, sv - 1) - 1, \\
y < 1, \\
y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 2 \end{array} \right)\]
\end{enumerate}
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB} \text{\_STEADY}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\[ 1 \quad 2 \times y \leq x + sv - 1 \]
\[ \square sv \geq 5: (sv - 3, sv - 2) \quad (sv - 5, sv - 3) \]
\[ 2 \quad x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \]
\[ \square sv \geq 3: (sv - 3, sv - 2) \quad (sv - 3, sv - 3) \]
\[ 3 \quad \bigvee \left( \begin{array}{l}
    x \neq \max(0, sv - 1) - 1, \\
    y < 1, \\
    y > (sv - 2) \times \min(1, \max(0, sv - 2)) - 2
\end{array} \right) \]
\[
\text{NB}_{\text{STEADY}}(x, \text{VARIABLES}) \land \\
\text{SUM}_{\text{WIDTH}}_{\text{PROPER}}_{\text{PLAIN}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \cdot y \leq x + sv - 1\)
2. \(y \leq 2 \cdot x\)
3. \(3 \cdot y \leq 3 \cdot x + sv - 1 - (sv - 1) \mod 3\)
4. \(x > 0 \land y > 0 \Rightarrow x \leq sv - 3\)
5. \(sv \geq 5: (sv - 3, sv - 2) \land (sv - 3, sv - 3)\)
6. \(y \neq 1\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{stead} (x, \text{VARIABLES}) \land \\
\text{SUM}_\text{width, proper plateau} (y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2 \cdot y \leq x + sv - 1 \)
   - \( sv \geq 7 \): \((sv - 3, sv - 2) \quad (sv - 5, sv - 3)\)

2. \( y \leq 2 \cdot x \)
   - \( sv \geq 7 \): \((1, 2) \quad (2, 4)\)

3. \( 3 \cdot y \leq 3 \cdot x + sv - 1 - (sv - 1) \bmod 3 \)

4. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \)
   - \( sv \geq 5 \): \((sv - 3, sv - 2) \quad (sv - 3, sv - 3)\)

5. \( \lor \left( \begin{array}{l}
x \neq \max(0, sv - 1) - 1, \\
y < 1, \\
y > (sv - 2) * \min(1, \max(0, sv - 3)) - 2 \\
\end{array} \right) \)

6. \( y \neq 1 \)
\[ \text{NB\_STEADY}(x, \text{VARIABLES}) \land \] 
\[ \text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \] 

1. \( y \leq 2 \times x \)  
   \[ \text{sv} \geq 4: \ (1, 2) \quad (2, 4) \] 
2. \( x = 0 \Rightarrow y = 0 \)  
3. \( y = 0 \Rightarrow x = 0 \)  
4. \( \text{sv} > 1 \Rightarrow 2 \times x \leq y + \text{sv} - 2 \)  
5. \( x > 0 \land y > 0 \Rightarrow x \leq y - 1 \)  
6. \( \text{sv} \geq 4: \ (\text{sv} - 2, \text{sv} - 1) \quad (\text{sv} - 3, \text{sv} - 2) \)

\[ \text{NB\_STEADY}(x, \text{VARIABLES}) \land \] 
\[ \text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \] 

1. \( 2 \times x + y \leq 2 \times \text{sv} - 2 \)  
   \[ \text{sv} \geq 4: \ (\text{sv} - 2, 2) \quad (\text{sv} - 3, 4) \] 
2. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB\_STEADY}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_STRICKLY\_INCRESHING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

\begin{enumerate}
  \item \(2 \cdot x + y \leq 2 \cdot sv - 2\)
  \item \(sv \geq 4: (sv - 2, 2) \quad (sv - 3, 4)\)
  \item \(y \neq 1\)
\end{enumerate}

\begin{enumerate}
  \item \(x = sv - 3 \Rightarrow y \leq 1\)
  \item \(x > 0 \land y > 0 \Rightarrow x \leq sv - 3\)
  \item \[
  \begin{cases}
    x \neq \max(0, sv - 1) - 1, \\
    y < 1,
  \end{cases}
  \]
  \item \[
  \begin{cases}
    y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 2, \\
    x \neq \max(0, sv - 1) - 2,
  \end{cases}
  \]
  \item \[
  \begin{cases}
    y < 2, \\
    y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 1
  \end{cases}
  \]
\end{enumerate}

\begin{enumerate}
  \item \(sv = 12\)
  \item \(sv = 11\)
\end{enumerate}
\( \text{NB}_\text{STEADY}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{VALLEY}(y, \text{VARIABLES}) \) \text{ with } sv = |\text{VARIABLES}|

1. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \)
2. \( sv \geq 3: (sv - 3, sv - 2) \quad (sv - 3, sv - 3) \)
3. \( \sqrt{\begin{cases} x \neq \max(0, sv - 1) - 1, \\ y < 1, \\ y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 2 \end{cases}} \)

\( sv = 12 \quad sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_STEADY}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow x + y \leq sv - 2 \)
2. \( sv \geq 6: (sv - 4, 2) (1, sv - 3) \)
3. \( y > 0 \Rightarrow x \leq sv - 4 \)
4. \( y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
2. \( sv \mod 2 = 0 \land sv \geq 2: (\lfloor sv/2 \rfloor, \lfloor sv/2 \rfloor - 1) (\lfloor sv/2 \rfloor - 1, \lfloor sv/2 \rfloor) \)

\[ sv = 12 \]

\[ sv = 11 \]
\[
\text{NB\_STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_STRICKLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x + y \leq sv - 1\)
   - \(sv \mod 2 = 0 \land sv \geq 8: \left(\lfloor sv/2 \rfloor, 0\right) \left(\lfloor sv/2 \rfloor - 1, 1\right)\)
   - \(sv \mod 2 = 0 \land sv \geq 11: \left(\lfloor sv/2 \rfloor - 1, 2\right)\)

\[
\text{sv} = 12
\]

\[
\text{sv} = 11
\]

\[
\text{NB\_STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \times x + y \leq sv\)
   - \(sv \mod 2 = 0 \land sv \geq 8: \left(\lfloor sv/2 \rfloor, 0\right) \left(\lfloor sv/2 \rfloor - 1, 1\right)\)
   - \(sv \mod 2 = 1 \land sv \geq 11: \left(\lfloor sv/2 \rfloor - 1, 2\right)\)

2. \(x + 2 \times y \leq sv - 1\)
   - \(sv \mod 2 = 0 \land sv \geq 8: \left(1, \lfloor sv/2 - 1\rfloor\right) \left(3, \lfloor sv/2 - 1\rfloor - 1\right)\)
   - \(sv \mod 2 = 1 \land sv \geq 11: \left(2, \lfloor sv/2 - 1\rfloor - 1\right) \left(4, \lfloor sv/2 - 1\rfloor - 2\right)\)

\[
\text{sv} = 12
\]

\[
\text{sv} = 11
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

**NB_STEADY_SEQUENCE**(*x, VARIABLES*)\(\land\) **NB_VALLEY**(*y, VARIABLES*) with \(sv = |\text{VARIABLES}|\)

1. \(x + 2 \cdot y \leq sv - 1\)
   - \(sv \mod 2 = 0 \land sv \geq 6: (1, \lfloor (sv - 1)/2 \rfloor)\)
   - \(sv \mod 2 = 1 \land sv \geq 9: (2, \lfloor (sv - 1)/2 \rfloor - 1)\)

\(sv = 12\)

\(sv = 11\)

**NB_STEADY_SEQUENCE**(*x, VARIABLES*)\(\land\) **NB_ZIGZAG**(*y, VARIABLES*) with \(sv = |\text{VARIABLES}|\)

1. \(2 \cdot x + 2 \cdot y \leq sv - sv \mod 2\)
   - \(sv \mod 2 = 0 \land sv \geq 6: (\lfloor sv/2 \rfloor, 0)\)
   - \(sv \mod 2 = 1 \land sv \geq 9: (\lfloor sv/2 \rfloor, 0)\)

\(sv = 12\)

\(sv = 11\)

2. \(x + 3 \cdot y \leq sv - 1\)
   - \(sv \mod 3 = 0 \land sv \geq 18: (2, \lfloor (sv - 1)/3 \rfloor)\)
   - \(sv \mod 3 = 1 \land sv \geq 10: (0, \lfloor (sv - 1)/3 \rfloor)\)
   - \(sv \mod 3 = 2 \land sv \geq 14: (1, \lfloor (sv - 1)/3 \rfloor)\)

\(sv = 12\)

\(sv = 11\)
\[
\text{NB\_STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \cdot x + y \leq 2 \cdot sv - 2\)
2. \(sv \mod 2 = 0 \land sv \geq 2: ([sv/2], sv - 2) \land ([sv/2] - 1, sv)\)
   - \(x \neq [sv/2], \quad y < 1, \quad y > sv \cdot \min(1, \max(0, sv - 1)) - 3,
   \)
   - \(1 = sv \mod 2, \quad 0 = y \mod 2\)
3. \(y \neq 1\)

\[sv = 12\]

\[sv = 11\]
A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq 2 \cdot sv - 4 \]
   - \[ sv \mod 2 = 0 \land sv \geq 4: ([sv/2], sv - 4) \land ([sv/2] - 1, sv - 2) \]
2. \[ y > 0 \Rightarrow x \geq 1 \]
   - \[ sv \geq 5: (1, sv - 2) \land (1, sv - 3) \]
   - \[ x \neq [sv/2], \]
   - \[ y < 1, \]
   - \[ y > (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 3, \]
   - \[ 1 = sv \mod 2, \]
   - \[ 0 = y \mod 2 \]
3. \[ \forall \]
4. \[ y \neq 1 \]
\textbf{NB} \_\_ \textbf{STEADY SEQUENCE}(x, \text{VARIABLES}) \land \textbf{SUM WIDTH GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

\begin{enumerate}
\item \(x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq 2 \times sv - 5\)
\item \(sv \mod 2 = 1 \land sv \geq 5: ([sv/2], sv - 4) \quad ([sv/2] - 1, sv - 2)\)
\item \(x > 0 \land y > 0 \Rightarrow 2 \times x \leq sv - 1 - (sv - 1) \mod 2\)
\item \((sv - 1) \mod 2 = 0 \land sv \geq 6: ([sv/2], sv - 4) \quad ([sv/2], 1)\)
\item \((sv - 1) \mod 2 = 1 \land sv \geq 6: ([sv/2] - 1, sv - 3) \quad ([sv/2] - 1, sv - 4)\)
\end{enumerate}

\begin{align*}
sv &= 12 \\
sv &= 11
\end{align*}
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB}_\text{STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \(2 \times x + y \leq 2 \times sv - 2\)
   \(sv \mod 2 = 0 \land sv \geq 2: ([sv/2], sv - 2) \land ([sv/2] - 1, sv)\)
   \(x \neq [sv/2],\)
   \(y < 1,\)

2. \(y > sv \times \min(1, \max(0, sv - 1)) - 3,\)
   \(1 = sv \mod 2,\)
   \(0 = y \mod 2\)

3. \(y \neq 1\)
**NB_STEADY_SEQUENCE**(x, VARIABLES) \&
**SUM_WIDTH_INCREMENTING_TERRACE**(y, VARIABLES) with sv = |VARIABLES|

1. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq 2 \cdot sv - 4 \)
2. \( sv \mod 2 = 0 \land sv \geq 4: (\lfloor sv/2 \rfloor, sv - 4) \land (\lfloor sv/2 \rfloor - 1, sv - 2) \)
3. \( y > 0 \Rightarrow x \geq 1 \)
4. \( sv \geq 5: (1, sv - 2) \land (1, sv - 3) \)
5. \( y \neq 1 \)

\[
\begin{cases}
  x \neq \lfloor sv/2 \rfloor, \\
  y < 1, \\
  y > (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 3, \\
  1 = sv \mod 2, \\
  0 = y \mod 2
\end{cases}
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}_\text{STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x > 0 \land y > 0 \Rightarrow 2 \ast x + y \leq 2 \ast sv - 4 \)
   \( sv \mod 2 = 0 \land sv \geq 8 \): (\( \lfloor sv/2 \rfloor, sv - 4 \)) (\( \lfloor sv/2 \rfloor - 1, sv - 2 \))

2. \( x > 0 \land y > 0 \Rightarrow 2 \ast x \leq y + sv - 2 \)
   \( x = \lfloor sv/2 \rfloor, \) \( y < 1, \)
   \( y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 3, \)
   \( 1 = sv \mod 2, \)
   \( 0 = y \mod 2 \)

3. \( x > 0 \land y > 0 \Rightarrow 2 \ast x \leq y + sv - 2 \)
   \( x = \lfloor sv/2 \rfloor, \) \( y < 1, \)
   \( y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 3, \)
   \( 1 = sv \mod 2, \)
   \( 0 = y \mod 2 \)
**NB_STEADY_SEQUENCE**(x, VARIABLES) ∧ **SUM_WIDTH_PEAK**(y, VARIABLES) with sv = |VARIABLES|

1. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq 2 \cdot sv - 4 \)
2. \( sv \, \text{mod} \, 2 = 0 \land sv \geq 6: \ (\lfloor sv/2 \rfloor, sv - 4) \) \( \land (\lfloor sv/2 \rfloor - 1, sv - 2) \)

- \( x > 0 \land y > 0 \Rightarrow 2 \cdot x \leq y + sv - 2 \)
  - \( x \neq \lfloor sv/2 \rfloor, \)
  - \( y < 1, \)
  - \( y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 3, \)
  - \( 1 = sv \, \text{mod} \, 2, \)
  - \( 0 = y \, \text{mod} \, 2 \)

\( sv = 12 \)

\( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB}_\text{STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 5: \ (1, sv - 2) \quad (2, sv - 3) \)
2. \( x = 0 \Rightarrow 2 \cdot y \leq sv - 1 - (sv - 1) \mod 2 \)
3. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x \leq 2 \cdot y + sv - 3 - (sv - 3) \mod 2 \)
   - \( (sv - 1) \mod 2 = 1 \land sv \geq 6: \ (\lfloor sv/2 \rfloor, 2) \quad (\lfloor sv/2 \rfloor - 1, 1) \)
4. \( 0 = y \mod 2 \quad y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 3, \quad 1 = sv \mod 2 \)

\[ \begin{align*}
\text{sv} = 12 \\
\text{sv} = 11
\end{align*} \]
\[
\text{NB\_STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x + y \leq sv - 1 \)
2. \( sv \geq 5: (1, sv - 2) \land (2, sv - 3) \)
3. \( x = 0 \Rightarrow 2 \cdot y \leq sv - 1 - (sv - 1) \mod 2 \)
4. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x \leq 2 \cdot y + sv - 3 - (sv - 3) \mod 2 \)

\((sv - 1) \mod 2 = 1 \land sv \geq 6: ([sv/2], 2) \land ([sv/2] - 1, 1)\)

\( sv \neq [sv/2], \)
\( y < 1, \)
\( y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 3, \)
\( 1 = sv \mod 2, \)
\( 0 = y \mod 2 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_STEADY\_SEQUENCE}(x, \text{VAR}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VAR}) \text{ with } sv = |\text{VAR}|
\]

1. \( x + y \leq sv - 1 \)
2. \( y > 0 \Rightarrow x \geq 1 \)
3. \( sv \geq 5: (1, sv - 2) \land (2, sv - 3) \)
4. \( y \neq 0 \Rightarrow x \neq 1 \)

\[
\begin{align*}
\begin{cases}
sv \geq 5: (1, sv - 2) \land (2, sv - 3) \\
\text{sv} \geq 5: (1, sv - 2) \land (2, sv - 3) \\
\end{cases}
\end{align*}
\]

\[
\begin{align*}
\begin{cases}
x + y \neq \lfloor sv/2 \rfloor, \\
y < 1, \\
y > (sv - 2) * \min(1, \max(0, sv - 3)) - 3, \\
1 = sv \mod 2, \\
0 = y \mod 2 \\
\end{cases}
\end{align*}
\]

\( sv = 12 \)

\( sv = 11 \)
\[ \text{NB\_STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
2. \( sv \geq 5: (1, sv - 2) \land (2, sv - 3) \)
3. \( y > 0 \Rightarrow x \geq 1 \)
4. \( y > 0 \Rightarrow x \geq 1 \)
5. \( y > (sv - 2) \times \min(1, \max(0, sv - 3)) - 3, \quad 1 = sv \mod 2, \quad 0 = y \mod 2 \)
6. \( y \neq 1 \)
A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}_\text{STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \times x \leq y\)
   - \(sv \geq 4: (1, 2), (2, 4)\)
2. \(sv > 1 \Rightarrow y \leq 2 \times x + sv - 2\)
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 4: (1, sv - 1), (1, sv - 2)\)

Graphs showing the ranges for \(sv = 12\) and \(sv = 11\) respectively.
\( \text{NB\_STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x + y \leq 2 \times sv - 2 \)
2. \( sv \mod 2 = 0 \land sv \geq 2 : ([sv/2], sv - 2) \land ([sv/2] - 1, sv) \)
   \( x \neq [sv/2], \)
   \( y < 1, \)
3. \( y > sv \times \min(1, \max(0, sv - 1)) - 3, \)
   \( 1 = sv \mod 2, \)
   \( 0 = y \mod 2 \)
4. \( y \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)
A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \]
\[ \text{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \cdot x + y \leq 2 \cdot sv - 2 \]
   \[ \begin{align*}
   & \text{sv mod 2} = 0 \land \text{sv} \geq 2: (\lfloor sv/2 \rfloor, sv - 2) \\
   & \quad (\lfloor sv/2 \rfloor - 1, sv) \\
   & \text{\text{sv}} \neq \lfloor sv/2 \rfloor, \\
   & \text{\text{y}} < 1,
   \end{align*} \]

2. \[ \lor \begin{align*}
   & \text{y} > \text{sv} \ast \text{min}(1, \text{max}(0, \text{sv} - 1)) - 3, \\
   & \text{1} = \text{sv mod 2}, \\
   & \text{0} = \text{y mod 2}
   \end{align*} \]

3. \text{y} \neq 1

\[
\begin{array}{c}
\text{sv} = 12 \\
\text{sv} = 11
\end{array}
\]

\[
\begin{array}{c}
\text{nb\_steady\_sequence} \\
\text{sum\_width\_strictly\_increasing\_sequence}
\end{array}
\]

\[
\begin{array}{c}
\text{nb\_steady\_sequence} \\
\text{sum\_width\_strictly\_increasing\_sequence}
\end{array}
\]
\text{NB\_STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } \text{sv} = |\text{VARIABLES}|

1. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq 2 \cdot \text{sv} - 5 \)
2. \( \text{sv mod 2} = 1 \land \text{sv} \geq 5 : (\lfloor \text{sv}/2 \rfloor, \text{sv} - 4) \quad (\lfloor \text{sv}/2 \rfloor - 1, \text{sv} - 2) \)
3. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x \leq \text{sv} - 1 - (\text{sv} - 1) \text{ mod 2} \)

\begin{align*}
\text{sv} = 12 & \quad \text{sv} = 11 \\
\end{align*}

\begin{align*}
& \quad \text{sv} = 10 \quad 10 \quad 8 \quad 6 \quad 4 \quad 2 \quad 0 \\
& \quad 0 \quad 2 \quad 4 \quad 6 \quad 8 \quad 10 \\
& \quad \text{nb\_steady\_sequence} \quad \text{sum\_width\_summit} \\
& \quad \text{sum\_width\_summit} \quad \text{nb\_steady\_sequence}
\end{align*}
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}_{\text{STEADY\_SEQUENCE}}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq 2 \cdot sv - 4\)
2. \(sv \mod 2 = 0 \land sv \geq 6: ([sv/2], sv - 4) \land ([sv/2] - 1, sv - 2)\)
3. \(x > 0 \land y > 0 \Rightarrow 2 \cdot x \leq y + sv - 2\)
4. \(y < 1, \quad y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 3,\)
5. \(l = sv \mod 2, \quad 0 = y \mod 2\)

sv = 12

\[
\begin{array}{c}
\text{nb\_steady\_sequence} \\
\text{sum\_width\_valley}
\end{array}
\]

sv = 11

\[
\begin{array}{c}
\text{nb\_steady\_sequence} \\
\text{sum\_width\_valley}
\end{array}
\]
\[ \text{NB \_STEADY\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \] with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x + y \leq sv \)
   - \( sv \geq 8: (2, sv - 4) \quad (3, sv - 6) \)
2. \( x > 0 \land y > 0 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 6: (1, sv - 3) \quad (2, sv - 4) \)
3. \( y > 0 \Rightarrow 2 \times x \leq sv - 2 - (sv - 2) \mod 2 \)
   - \( sv \mod 2 = 1 \land sv \geq 7: (\lfloor sv/2 \rfloor - 1, 2) \quad (\lfloor sv/2 \rfloor - 1, 3) \)
4. \( y \neq 1 \)

\[ \text{NB \_STRICTLY\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB \_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \] with \( sv = |\text{VARIABLES}| \)

1. \( x + y \leq sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 2: (\lfloor sv/2 \rfloor, \lfloor sv/2 \rfloor - 1) \quad (\lfloor sv/2 \rfloor - 1, \lfloor sv/2 \rfloor) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB}\_\text{STRICTLY\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB}\_\text{SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\[y \leq x\]

\[
\begin{cases} 
\text{sv} \geq 5: & (1,1) & (2,2) \\
\end{cases}
\]

\[
\begin{array}{c}
s_v = 12 \\
s_v = 11
\end{array}
\]

\[
\begin{array}{c}
\text{nb}\_\text{strictly\_decreasing\_sequence} \\
\text{nb}\_\text{summit}
\end{array}
\]

\[
\begin{array}{c}
\text{nb}\_\text{strictly\_decreasing\_sequence} \\
\text{nb}\_\text{valley}
\end{array}
\]

\[
\begin{cases} 
\text{sv} \geq 5: & (1,1) & (2,2) \\
\end{cases}
\]

\[
\begin{array}{c}
s_v = 12 \\
s_v = 11
\end{array}
\]
\text{NB\_STRICKLY\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{NB\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. $x + 2 \ast y \leq sv$
2. $y \leq x$
   - $sv \geq 7$: (1, 1) (2, 2)

\text{sv} = 12
\text{sv} = 11

\text{nb\_strictly\_decreasing\_sequence}

\text{NB\_STRICKLY\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. $2 \ast x \leq y$
   - $sv \geq 4$: (1, 2) (2, 4)
2. $sv > 1 \Rightarrow y \leq 2 \ast x + sv - 2$
3. $y > 0 \Rightarrow x \geq 1$
   - $sv \geq 4$: (1, $sv - 1$) (1, $sv - 2$)

\text{sv} = 12
\text{sv} = 11

\text{nb\_strictly\_decreasing\_sequence}
A DATABASE OF PARAMETERISED INVARIANTS

\[\text{NB}_\text{STRICTLY}_{\text{DECREASING}\_\text{SEQUENCE}}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH}_{\text{DECREASING}\_\text{TERRACE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(x = 0 \Rightarrow y = 0\)
2. \(x > 0 \land y > 0 \Rightarrow x \geq 2\)
   - \(sv \geq 5: (2, sv - 2), (2, sv - 3)\)
3. \(\bigvee\left(\begin{array}{l}
x \neq \lfloor sv/2 \rfloor, \\
y < 1, \\
y > (sv - 2) \times \min(1, \max(0, sv - 3)) - 1, \\
l = sv \text{ mod } 2, \\
o = y \text{ mod } 2\end{array}\right)\)
4. \(x \neq 1 \lor y < 1\)
5. \(y \neq 1\)
\[\text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq 2 \times sv - 3\)
2. \(y > 0 \Rightarrow x \geq 1\)

\[sv \geq 3:\ (1, sv - 2) \ (1, sv - 3)\]

\[sv = 12\]

\[sv = 11\]

\[\text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(2 \times x + y \leq 2 \times sv - 2\)
2. \(sv \mod 2 = 0 \land sv \geq 2:\ (\lfloor sv/2 \rfloor, sv - 2) \ (\lfloor sv/2 \rfloor - 1, sv)\)
   \[
   \begin{align*}
   x & \neq \lfloor sv/2 \rfloor, \\
   y & < 1,
   \end{align*}
   \]
3. \(y > sv \times \min(1, \max(0, sv - 1)) - 3, \ 1 = sv \mod 2, \ 0 = y \mod 2\)
4. \(y \neq 1\)

\[sv = 12\]

\[sv = 11\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(x,y) \land \\
\text{SUM\_WIDTH\_INCREASING\_TERRACE}(y) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \times x + y \leq sv\)
   - \(sv \geq 8\): (2, sv − 4) (3, sv − 6)
2. \(x > 0 \land y > 0 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 6\): (1, sv − 3) (2, sv − 4)
3. \(y > 0 \Rightarrow 2 \times x \leq sv - 2 - (sv - 2) \mod 2\)
   - \(sv \mod 2 = 1 \land sv \geq 7\): (\(\lfloor sv/2 \rfloor - 1, 2\)) (\(\lfloor sv/2 \rfloor - 1, 3\))
4. \(y \neq 1\)

\(sv = 12\)

\(sv = 11\)
\[ \text{NB}_{\text{STRICTLY DECREASING SEQUENCE}}(x, \text{VARIABLES}) \land \\
\text{SUM WIDTH INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x \leq y + sv - 2 \)
2. \( y > 0 \Rightarrow x \geq 1 \)
3. \( sv \geq 3: (1, sv - 2) \) \( (1, sv - 3) \)

\[ \begin{align*}
&x \neq \lfloor sv/2 \rfloor, \\
y < 1, \\
y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 1, \\
1 = sv \mod 2, \\
0 = y \mod 2 
\end{align*} \]

\[ \begin{align*}
\sum &\text{WIDTH PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \\
(1, sv - 2) &\text{ } (1, sv - 3) \\
\begin{align*}
2 \cdot x + y &\leq 2 \cdot sv - 3 \\
y > 0 &\Rightarrow x \geq 1 \\
sv &\geq 3: (1, sv - 2) \) \( (1, sv - 3) \)
\end{align*} \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \wedge \\
\text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{enumerate}
  \item \( x + y \leq sv - 1 \)
  \begin{enumerate}
    \item \( sv \geq 4 \): \((1, sv - 2), (2, sv - 3)\)
    \item \( y > 0 \Rightarrow x \geq 1 \)
    \begin{enumerate}
      \item \( sv \geq 3 \): \((1, sv - 2), (1, sv - 3)\)
    \end{enumerate}
  \end{enumerate}
\end{enumerate}
NB_STRICTLY_DECREASING_SEQUENCE(x, VARIABLES) ∧
SUM_WIDTH_PROPERPLAIN(y, VARIABLES) with sv = |VARIABLES|

1. \[4 \times x + y \leq 2 \times sv\]
   - sv mod 2 = 0 ∧ sv \geq 8: (\lfloor sv/2 \rfloor, 0) (\lfloor sv/2 \rfloor - 1, 4)
   - sv mod 2 = 1 ∧ sv \geq 11: (\lfloor sv/2 \rfloor, 2) (\lfloor sv/2 \rfloor - 1, 6)

2. \[x + y \leq sv - 1\]
   - sv \geq 5: (1, sv - 2) (2, sv - 3)

3. \[y > 0 \implies x \geq 1\]
   - sv \geq 5: (1, sv - 2) (1, sv - 3)

4. y \neq 1

\[\begin{align*}
sv = 12 \\
sv = 11
\end{align*}\]
\[
\text{NB\_\text{STRICTLY\_DECREASING\_SEQUENCE}}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{enumerate}
\item \[4 \times x + y \leq 2 \times sv\]
\item \[sv \mod 2 = 0 \land sv \geq 8: \ (\lfloor sv/2 \rfloor, 0) \quad (\lfloor sv/2 \rfloor - 1, 4)\]
\item \[sv \mod 2 = 1 \land sv \geq 11: \ (\lfloor sv/2 \rfloor, 2) \quad (\lfloor sv/2 \rfloor - 1, 6)\]
\item \[x + y \leq sv - 1\]
\item \[sv \geq 5: \ (1, sv - 2) \quad (2, sv - 3)\]
\item \[y > 0 \Rightarrow x \geq 1\]
\item \[sv \geq 5: \ (1, sv - 2) \quad (1, sv - 3)\]
\item \[y \neq 1\]
\end{enumerate}
\[
\text{NB\_STRICLY\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \[2 \times x + y \leq 2 \times sv - 2\]
2. \(sv \mod 2 = 0 \land sv \geq 2: (\lfloor sv/2 \rfloor, sv - 2) \land (\lceil sv/2 \rceil - 1, sv)\]
3. \(x \neq \lfloor sv/2 \rfloor, y < 1, y > sv \times \min(1, \max(0, sv - 1)) - 3, 1 = sv \mod 2, 0 = y \mod 2\)
4. \(y \neq 1\)

\(sv = 12\)

\(sv = 11\)
\[ \text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x \leq y\)
   \(sv \geq 4: (1, 2) \quad (2, 4)\)
2. \(sv > 1 \Rightarrow y \leq 2 \times x + sv - 2\)
3. \(y > 0 \Rightarrow x \geq 1\)
   \(sv \geq 4: (1, sv - 1) \quad (1, sv - 2)\)
\textbf{NB\_\_STRICLY\_DECREASING\_SEQUENCE}(x, VARIABLES) \land \\
\textbf{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, VARIABLES) with \(sv = |\text{VARIABLES}|\)

1. \(2 \times x + y \leq 2 \times sv - 2\)
   - \(sv \ mod \ 2 = 0 \land sv \geq 2: \left(\lfloor sv/2 \rfloor, \lfloor sv/2 \rfloor - 1, sv\right)\)
   - \(x \neq \lfloor sv/2 \rfloor,\)
   - \(y < 1,\)
   - \(y > sv \times \min(1, \max(0, sv - 1)) - 3,\)
   - \(1 = sv \ mod \ 2,\)
   - \(0 = y \ mod \ 2\)

2. \(y \neq 1\)

\(sv = 12\)

\(\text{nb\_strictly\_decreasing\_sequence}\)

\(\text{sum\_width\_summit}\)

\(sv = 11\)

\(\text{nb\_strictly\_decreasing\_sequence}\)

\textbf{NB\_\_STRICLY\_DECREASING\_SEQUENCE}(x, VARIABLES) \land \\
\textbf{SUM\_WIDTH\_SUMMIT}(y, VARIABLES) with \(sv = |\text{VARIABLES}|\)

1. \(x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq 2 \times sv - 3\)
2. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 3: (1, sv - 2) \land (1, sv - 3)\)

\(sv = 12\)

\(\text{nb\_strictly\_decreasing\_sequence}\)

\(\text{sum\_width\_summit}\)

\(sv = 11\)

\(\text{nb\_strictly\_decreasing\_sequence}\)
A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB \_STRICTLY\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \wedge \]
\[ \text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x > 0 \wedge y > 0 \Rightarrow 2 \cdot x + 2 \cdot y \leq 3 \cdot sv - 5 - (3 \cdot sv - 5) \mod 2 \]
   \[ \wedge (sv - 1) \mod 2 = 1 \wedge sv \geq 4 : ([sv/2], sv - 3) \quad (\lfloor sv/2 \rfloor - 1, sv - 2) \]

2. \[ y > 0 \Rightarrow x \geq 1 \]
   \[ \wedge sv \geq 3 : (1, sv - 2) \quad (1, sv - 3) \]

\[ \text{NB\_STRICTLY\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \wedge \]
\[ \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ y \leq 2 \cdot x \]
   \[ \wedge sv \geq 6 : (1, 2) \quad (2, 4) \]
   \[ x \neq \lfloor sv/2 \rfloor, \]
   \[ y < 1, \]
   \[ y > (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 1, \]
   \[ 1 = sv \mod 2, \]
   \[ 0 = y \mod 2 \]

2. \[ y \neq 1 \]
\[ \text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq x \)
   - \( sv \geq 5: (1, 1), (2, 2) \)

\[ \begin{align*}
sv &= 12 \\
sv &= 11
\end{align*} \]

\[ \text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{NB\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq x \)
   - \( sv \geq 5: (1, 1), (2, 2) \)

\[ \begin{align*}
sv &= 12 \\
sv &= 11
\end{align*} \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[\text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x,\ \text{VARIABLES}) \land \text{NB\_ZIGZAG}(y,\ \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(x + 2y \leq sv\)
2. \(y \leq x\)

\[\begin{array}{ll}
\text{sv} \geq 7: & (1, 1) \quad (2, 2)
\end{array}\]

\[\begin{array}{l}
\text{sv} = 12 \\
\text{sv} = 11
\end{array}\]

\[\text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x,\ \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y,\ \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(2x + y \leq 2sv - 2\)
2. \(sv \mod 2 = 0 \land sv \geq 2: (\lfloor sv/2 \rfloor, sv - 2) \quad (\lfloor sv/2 \rfloor - 1, sv)\)
   \[
   \begin{cases}
   x \neq \lfloor sv/2 \rfloor, \\
y < 1,
   \end{cases}
   \]
3. \(y > sv \times \min(1, \max(0, sv - 1)) - 3, \quad 1 = sv \mod 2, \quad 0 = y \mod 2\)

\[\begin{array}{l}
\text{sv} = 12 \\
\text{sv} = 11
\end{array}\]
NB\_STRICTLY\_INCREASING\_SEQUENCE(x, VARIABLES) ∧
SUM\_WIDTH\_DECREASING\_TERRACE(y, VARIABLES) with sv = |VARIABLES|

1. \(2 \cdot x + y \leq sv\)
   - \(sv \geq 8: (2, sv - 4) (3, sv - 6)\)
2. \(x > 0 \land y > 0 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 6: (1, sv - 3) (2, sv - 4)\)
3. \(y > 0 \Rightarrow 2 \cdot x \leq sv - 2 - (sv - 2) \mod 2\)
   - \(sv \mod 2 = 1 \land sv \geq 7: ([sv/2] - 1, 2) ([sv/2] - 1, 3)\)
4. \(y \neq 1\)

\(\text{nb\_strictly\_increasing\_sequence}\)

\(\text{sum\_width\_decreasing\_terrace}\)

\(\text{sv} = 12\)

\(\text{sv} = 11\)

NB\_STRICTLY\_INCREASING\_SEQUENCE(x, VARIABLES) ∧
SUM\_WIDTH\_GORGE(y, VARIABLES) with sv = |VARIABLES|

1. \(x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq 2 \cdot sv - 3\)
2. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 3: (1, sv - 2) (1, sv - 3)\)

\(\text{nb\_strictly\_increasing\_sequence}\)

\(\text{sum\_width\_gorge}\)

\(\text{sv} = 12\)

\(\text{sv} = 11\)
5. A DATABASE OF PARAMETERISED INVARIENTS

\begin{align*}
\text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } s_v = |\text{VARIABLES}|
\end{align*}

\begin{enumerate}
\item $2 \times x \leq y$
\item $s_v \geq 4$: (1, 2) \hspace{1cm} (2, 4)
\item $s_v > 1 \Rightarrow y \leq 2 \times x + s_v - 2$
\item $y > 0 \Rightarrow x \geq 1$
\item $s_v \geq 4$: (1, $s_v - 1$) \hspace{1cm} (1, $s_v - 2$)
\end{enumerate}
NB\_STRICTLY\_INCREASING\_SEQUENCE(x, VARIABLES) \land 
SUM\_WIDTH\_INCREASING\_TERRACE(y, VARIABLES) with sv = |VARIABLES|

1. \( x = 0 \Rightarrow y = 0 \)
2. \( x > 0 \land y > 0 \Rightarrow x \geq 2 \)
3. \( sv \geq 5: \quad (2, sv - 2) \quad (2, sv - 3) \)
   \[
   \begin{cases}
   x \neq \lfloor sv/2 \rfloor, \\
   y < 1, \\
   y > (sv - 2) \ast \min(1, \max(0, sv - 3)) - 1, \\
   1 = sv \mod 2, \\
   0 = y \mod 2
   \end{cases}
   \]
4. \( x \neq 1 \lor y < 1 \)
5. \( y \neq 1 \)

---

\( sv = 12 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \\
\begin{align*}
&1. \quad x > 0 \land y > 0 \implies 2 \cdot x \leq y + sv - 2 \\
&2. \quad y > 0 \implies x \geq 1 \\
&\quad \begin{cases}
&sv \geq 3: (1, sv - 2) \quad (1, sv - 3) \\
&x \neq \lfloor sv/2 \rfloor, \\
&y < 1, \\
&y > (sv - 2) \cdot \min(1, \max(0, sv - 2) - 1), \\
&1 = sv \mod 2, \\
&0 = y \mod 2
\end{cases}
\end{align*}
\end{align*}

\begin{align*}
\text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \\
\begin{align*}
&1. \quad x > 0 \land y > 0 \implies 2 \cdot x + 2 \cdot y \leq 3 \cdot sv - 5 - (3 \cdot sv - 5) \mod 2 \\
&\quad \begin{cases}
&(sv - 1) \mod 2 = 1 \land sv \geq 4: (\lfloor sv/2 \rfloor, sv - 3) \\
&(\lfloor sv/2 \rfloor - 1, sv - 2)
\end{cases}
\end{align*}
\end{align*}
\( \text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x + y \leq sv - 1 \)
2. \( sv \geq 4: (1, sv - 2) \quad (2, sv - 3) \)
3. \( y > 0 \Rightarrow x \geq 1 \)
4. \( sv \geq 3: (1, sv - 2) \quad (1, sv - 3) \)

\( sv = 12 \)

\( sv = 11 \)
\[\text{NB}_\text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{PROPER}_\text{PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \[4 \times x + y \leq 2 \times sv\]
   - \[sv \mod 2 = 0 \land sv \geq 8: \quad ([sv/2], 0) \quad ([sv/2] - 1, 4)\]
   - \[sv \mod 2 = 1 \land sv \geq 11: \quad ([sv/2], 2) \quad ([sv/2] - 1, 6)\]
2. \[x + y \leq sv - 1\]
   - \[sv \geq 5: \quad (1, sv - 2) \quad (2, sv - 3)\]
3. \[y \geq 0 \Rightarrow x \geq 1\]
   - \[sv \geq 5: \quad (1, sv - 2) \quad (1, sv - 3)\]
4. \[y \neq 1\]
\[ \text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 4 \cdot x + y \leq 2 \cdot sv \)
   - \( sv \mod 2 = 0 \land sv \geq 8: \quad \left\lceil \frac{sv}{2} \right\rceil, 0 \quad \left\lceil \frac{sv}{2} \right\rceil, -1, 4 \)
   - \( sv \mod 2 = 1 \land sv \geq 11: \quad \left\lceil \frac{sv}{2} \right\rceil, 2 \quad \left\lceil \frac{sv}{2} \right\rceil, -1, 6 \)
2. \( x + y \leq sv - 1 \)
   - \( sv \geq 5: \quad (1, sv - 2) \quad (2, sv - 3) \)
3. \( y > 0 \implies x \geq 1 \)
4. \( sv \geq 5: \quad (1, sv - 2) \quad (1, sv - 3) \)
5. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. \( 2 \times x + y \leq 2 \times sv - 2 \)

2. \( sv \mod 2 = 0 \land sv \geq 2: (\lfloor sv/2 \rfloor, sv - 2) \land (\lfloor sv/2 \rfloor - 1, sv) \)

3. \( x \neq \lfloor sv/2 \rfloor, y < 1, y > sv \times \min(1, \max(0, sv - 1)) - 3, 1 = sv \mod 2, 0 = y \mod 2 \)

4. \( y \neq 1 \)

\( sv = 12 \) 

\( sv = 11 \)
\[ \text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \times x + y \leq 2 \times sv - 2 \]
2. \[ sv \mod 2 = 0 \land sv \geq 2 : (\lfloor sv/2 \rfloor, sv - 2) \land (\lfloor sv/2 \rfloor - 1, sv) \]
   \[ x \neq \lfloor sv/2 \rfloor, \]
   \[ y < 1, \]
   \[ y > sv \times \min(1, \max(0, sv - 1)) - 3, \]
   \[ 1 = sv \mod 2, \]
   \[ 0 = y \mod 2 \]
3. \[ y \neq 1 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \times x \leq y\)
   - \(sv \geq 4: (1, 2) (2, 4)\)
2. \(sv > 1 \Rightarrow y \leq 2 \times x + sv - 2\)
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 4: (1, sv - 1) (1, sv - 2)\)

\[
\text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq 2 \times sv - 3\)
2. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 3: (1, sv - 2) (1, sv - 3)\)
\[ \text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \implies 2 \times x + y \leq 2 \times sv - 3 \)
2. \( y > 0 \implies x \geq 1 \)
   - \( sv \geq 3: \) (1, \( sv - 2 \)) (1, \( sv - 3 \))

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq 2 \times x \)
   - \( sv \geq 0: (1, 2) (2, 4) \)
2. \( x \neq \lfloor sv/2 \rfloor, \)
   - \( y < 1, \)
   - \( y > (sv - 2) \times \min(1, \max(0, sv - 3)) - 1, \)
   - \( 1 = sv \mod 2, \)
   - \( 0 = y \mod 2 \)
3. \( y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\( \text{NB\_SUMMIT}(x, \text{VARIABLES}) \land \text{NB\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \)

1. \( x \leq y + 1 \)
   - \( sv \geq 5 \): \((1, 0)\) \((2, 1)\)
2. \( 3 \times y \leq x + sv \)
   - \( sv \mod 2 = 0 \land sv \geq 12 \): \(((sv - 1)/2, (sv - 1)/2)\) \(((sv - 1)/2, (sv - 1)/2 - 1)\)
   - \( sv \mod 2 = 1 \land sv \geq 9 \): \(((sv - 1)/2 - 1, (sv - 1)/2)\) \(((sv - 1)/2 - 4, (sv - 1)/2 - 1)\)
3. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 3 \): \(((sv - 1)/2, (sv - 1)/2 - 1)\) \(((sv - 1)/2 - 1, (sv - 1)/2)\)

\( \text{NB\_SUMMIT}(x, \text{VARIABLES}) \land \text{NB\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \)

1. \( 2 \times x + y \leq sv \)
   - \( sv \mod 2 = 0 \land sv \geq 14 \): \(((sv - 1)/2, 2)\) \(((sv - 1)/2 - 1, 4)\)
   - \( sv \mod 2 = 1 \land sv \geq 11 \): \(((sv - 1)/2 - 1, 1)\) \(((sv - 1)/2 - 1, 3)\)
2. \( y \leq x \)
   - \( sv \geq 4 \): \((0, 0)\) \((1, 1)\)
3. \( y = 0 \Rightarrow 3 \times x \leq sv - sv \mod 3 \)
\( \text{NB\_SUMMIT}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \)

1. \(2 \cdot x + y \leq 2 \cdot sv - 2\)
2. \(2 \cdot x \leq y\)
3. \(\text{sv} \geq 5: (1, 2) \land (2, 4)\)
4. \(x \neq 0 \lor y \neq 1\)

\( sv = 12 \)

\( \text{NB\_SUMMIT}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \)

1. \(2 \cdot x + y \leq sv - 1\)
2. \(sv \geq 7: (1, sv - 3) \land (2, sv - 5)\)
3. \(y > 0 \Rightarrow 2 \cdot x \leq sv - 3 - (sv - 3) \mod 2\)
4. \((sv - 1) \mod 2 = 1 \land sv \geq 6: (\lfloor (sv - 1)/2 \rfloor - 1, 2) \land (\lfloor (sv - 1)/2 \rfloor - 1, 3)\)
5. \(y \neq 1\)

\( sv = 12 \)

\( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_SUMMIT}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{enumerate}
\item \[3 \cdot x \leq y + sv\]
\item \[sv > 1 \Rightarrow x + y \leq sv - 2\]
\item \[sv \geq 3: (0, sv - 2) \quad (1, sv - 3)\]
\end{enumerate}

\[
sv = 12
\]

\[
sv = 11
\]

\[
\text{NB\_SUMMIT}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{enumerate}
\item \[2 \cdot x + y \leq 2 \cdot sv - 2\]
\item \[2 \cdot x \leq y\]
\item \[sv \geq 5: (1, 2) \quad (2, 4)\]
\item \[x \neq 0 \lor y \neq 1\]
\end{enumerate}

\[
sv = 12
\]

\[
sv = 11
\]
\( \text{NB \_SUMMIT}(x, \text{VARIABLES}) \land \text{SUM \_ WIDTH \_ INCREASING \_ TERRACE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x + y \leq sv - 1 \)
   - \( sv \geq 7: \ (1, sv - 3) \quad (2, sv - 5) \)

2. \( y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2 \)
   - \( (sv - 1) \mod 2 = 1 \land sv \geq 6: \ (\lfloor (sv - 1)/2 \rfloor - 1, 2) \quad (\lfloor (sv - 1)/2 \rfloor - 1, 3) \)

3. \( y \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)

\( \text{NB \_SUMMIT}(x, \text{VARIABLES}) \land \text{SUM \_ WIDTH \_ INFLEXION}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2 \times x \leq y + 1 \)
   - \( sv \geq 5: \ (1, 1) \quad (2, 3) \)

\( sv = 12 \)

\( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_SUMMIT}(x, \text{VARIABLES}) \land \]
\[ \text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 5: (1, sv - 2) \) \( (2, sv - 3) \)
2. \( x \leq y \)
   - \( sv \geq 3: (0, 0) \) \( (1, 1) \)
3. \( x \neq 0 \lor y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_SUMMIT}(x, \text{VARIABLES}) \land \]
\[ \text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y + 1 \)
   - \( sv \geq 5: (1, 0) \) \( (2, 1) \)
2. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 3: (0, sv - 2) \) \( (1, sv - 3) \)

\[ sv = 12 \]

\[ sv = 11 \]
\[\text{NB}_\text{SUMMIT}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(x + y \leq sv - 1 - (sv - 1) \mod 2\)
2. \((sv - 1) \mod 2 = 1 \land sv \geq 6\) \((0, sv - 2) \quad (1, sv - 3)\)
3. \(x \leq y\)
4. \(sv \geq 3: \quad (0, 0) \quad (1, 1)\)
5. \(x < \lfloor (sv - 1)/2 \rfloor \Rightarrow x + y \leq sv - 2\)
6. \(sv \geq 6: \quad (0, sv - 2) \quad (1, sv - 3)\)
7. \(y \neq x + 1\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{SUMMIT}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 4 \times x + y \leq 2 \times sv - 2 \]
   - \[ sv \mod 2 = 0 \land \, sv \geq 12: \left( \left \lfloor (sv - 1)/2 \right \rfloor, -2 \right) \left( \left \lfloor (sv - 1)/2 \right \rfloor - 1, 6 \right) \]
   - \[ sv \mod 2 = 1 \land \, sv \geq 15: \left( \left \lfloor (sv - 1)/2 \right \rfloor - 1, 4 \right) \left( \left \lfloor (sv - 1)/2 \right \rfloor - 2, 8 \right) \]
2. \[ sv > 1 \Rightarrow x + y \leq sv - 2 \]
   - \[ sv \geq 5: (0, sv - 2) \quad (1, sv - 3) \]
3. \[ y \neq 1 \]

\[ \text{NB}_\text{SUMMIT}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x > 0 \land \, y > 0 \Rightarrow 2 \times x + y \leq sv - 2 \]
   - \[ sv \geq 8: (1, sv - 4) \quad (2, sv - 6) \]
2. \[ y > 0 \Rightarrow 2 \times x \leq sv - 4 - (sv - 4) \mod 2 \]
   - \[ sv \mod 2 = 1 \land \, sv \geq 5: \left( \left \lfloor (sv - 1)/2 \right \rfloor - 2, 2 \right) \left( \left \lfloor (sv - 1)/2 \right \rfloor - 2, 3 \right) \]
3. \[ y \neq 1 \]

\[ \text{NB}_\text{SUMMIT}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]
\[ \text{NB}_\text{SUMMIT}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 4 \cdot x + y \leq 2 \cdot sv - 2 \)
   - \( sv \ mod \ 2 = 0 \land sv \geq 8: \ (\lfloor (sv - 1)/2 \rfloor, 2) \) (\( \lfloor (sv - 1)/2 \rfloor - 1, 6 \))
   - \( sv \ mod \ 2 = 1 \land sv \geq 11: \ (\lfloor (sv - 1)/2 \rfloor - 1, 4) \) (\( \lfloor (sv - 1)/2 \rfloor - 2, 8 \))
2. \( x + y \leq sv \)
   - \( sv \geq 8: \ (1, sv - 1) \) (\( 2, sv - 2 \))
3. \( y \neq 1 \)

\[ \text{NB}_\text{SUMMIT}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2 \cdot x + y \leq 2 \cdot sv - 2 \)
2. \( 2 \cdot x \leq y \)
   - \( sv \geq 5: \ (1, 2) \) (\( 2, 4 \))
3. \( x \neq 0 \lor y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB} \_\text{SUMMIT}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x + 2 \times y \leq 3 \times sv - 2 - (3 \times sv - 2) \mod 2\)
   - \(sv \mod 2 = 1 \land sv \geq 3: (\lfloor (sv - 1)/2 \rfloor, sv - 1) \land ((sv - 1)/2 - 1, sv)\)
2. \(2 \times x \leq y\)
   - \(sv \geq 5: (1, 2) \land (2, 4)\)
3. \(x \neq 0 \lor y \neq 1\)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB} \_\text{SUMMIT}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(x + y \leq sv - 1\)
   - \(sv \geq 5: (1, sv - 2) \land (2, sv - 3)\)
2. \(x \leq y\)
   - \(sv \geq 3: (0, 0) \land (1, 1)\)
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 4: (1, sv - 2) \land (1, sv - 3)\)

\[ sv = 12 \]

\[ sv = 11 \]
\[ \text{NB} \_\text{SUMMIT}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y + 1 \)
   - \( sv \geq 5: \ (1, 0) \quad (2, 1) \)
2. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 3: \ (0, sv - 2) \quad (1, sv - 3) \)

\( sv = 12 \)

\( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB} & \_{\text{SUMMIT}}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \( y \leq 3 \times x \)
   - \( sv \geq 5 \): (0, 0) (1, 3)
2. \( 3 \times y \leq 4 \times x + sv \)
3. \( y = 0 \Rightarrow 3 \times x \leq sv - sv \mod 3 \)
4. \( x > 0 \land y > 0 \Rightarrow 3 \times x \leq y + sv - 2 \)
5. \( y > 0 \Rightarrow 6 \times x \leq y + 2 \times sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 14 \): \((\lfloor (sv - 1)/2 \rfloor, sv - 5)\) \((\lfloor (sv - 1)/2 \rfloor - 1, sv - 11)\)
   - \( sv \mod 2 = 1 \land sv \geq 17 \): \((\lfloor (sv - 1)/2 \rfloor - 1, sv - 8)\) \((\lfloor (sv - 1)/2 \rfloor - 2, sv - 14)\)
6. \( y \neq 1 \)
\[ \text{NB\_VALLEY}(x, \text{VARIABLES}) \land \text{NB\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \cdot x + y \leq sv\)
2. \(sv \mod 2 = 0 \land sv \geq 14: \left( \left\lfloor \frac{(sv - 1)}{2} \right\rfloor, 1 \right) \land \left( \left\lfloor \frac{(sv - 1)}{2} \right\rfloor + 1, 2 \right)\)
3. \(sv \mod 2 = 1 \land sv \geq 11: \left( \left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 1, 2 \right) \land \left( \left\lfloor \frac{(sv - 1)}{2} \right\rfloor, 1 \right)\)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \cdot x + 2 \cdot y \leq 3 \cdot sv - 2 - (3 \cdot sv - 2) \mod 2\)
2. \(sv \mod 2 = 1 \land sv \geq 3: \left( \left\lfloor \frac{(sv - 1)}{2} \right\rfloor, sv - 1 \right) \land \left( \left\lfloor \frac{(sv - 1)}{2} \right\rfloor - 1, sv \right)\)
3. \(sv \geq 5: (1, 3) \land (2, 4)\)
4. \(x \neq 0 \lor y \neq 1\)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{VALLEY}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{DECREASING}_\text{TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 \times x + y \leq sv - 1\)
   - \(sv \geq 7: (1, sv - 3) \land (2, sv - 5)\)
2. \(y > 0 \Rightarrow 2 \times x \leq sv - 3 - (sv - 3) \mod 2\)
   - \((sv - 1) \mod 2 = 1 \land sv \geq 6: (\lfloor (sv - 1)/2 \rfloor - 1, 2) \land (\lfloor (sv - 1)/2 \rfloor - 1, 3)\)
3. \(y \neq 1\)

\[ \text{NB}_\text{VALLEY}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(3 \times x \leq y + sv - 1\)
2. \(x + y \leq sv - 1\)
   - \(sv \geq 5: (1, sv - 2) \land (2, sv - 3)\)
3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 4: (1, sv - 2) \land (1, sv - 3)\)
\[ \text{NB VALLEY}(x, \text{VARIABLES}) \land \text{SUM WIDTH INCREASING SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \cdot x + y \leq 2 \cdot sv - 2 \]
2. \[ 2 \cdot x \leq y \]
3. \[ sv \geq 5: (1, 2) \quad (2, 4) \]
4. \[ x \neq 0 \lor y \neq 1 \]

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB VALLEY}(x, \text{VARIABLES}) \land \text{SUM WIDTH INCREASING TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \cdot x + y \leq sv - 1 \]
2. \[ sv \geq 7: (1, sv - 3) \quad (2, sv - 5) \]
3. \[ y > 0 \Rightarrow 2 \cdot x \leq sv - 3 - (sv - 3) \mod 2 \]
4. \[ (sv - 1) \mod 2 = 1 \land sv \geq 6: \cdot ((sv - 1)/2 - 1, 2) \quad ((sv - 1)/2 - 1, 3) \]
5. \[ y \neq 1 \]

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_VALLEY}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\[ 1. \quad 2 \times x \leq y + 1 \]
\[ \square \quad sv \geq 5: \ (1, 1) \quad (2, 3) \]

\[ sv = 12 \]

\[ \square \]

\[ \square \]

\[ \text{NB\_VALLEY}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\[ 1. \quad x \leq y + 1 \]
\[ \square \quad sv \geq 5: \ (1, 0) \quad (2, 1) \]
\[ \square \quad sv > 1 \Rightarrow x + y \leq sv - 2 \]
\[ \square \quad sv \geq 3: \ (0, sv - 2) \quad (1, sv - 3) \]

\[ sv = 12 \]

\[ \square \]

\[ \square \]

\[ \square \]

\[ \square \]

\[ \text{sv} = 11 \]
\[ \text{NB \_ VALLEY}(x, \text{VARIABLES}) \land \text{SUM \_ WIDTH \_ PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv - 1 \)
   - \( sv \geq 5 \): (1, \( sv - 2 \)) (2, \( sv - 3 \))

2. \( x \leq y \)
   - \( sv \geq 3 \): (0, 0) (1, 1)

3. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 4 \): (1, \( sv - 2 \)) (1, \( sv - 3 \))

\[ \text{NB \_ VALLEY}(x, \text{VARIABLES}) \land \text{SUM \_ WIDTH \_ PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x \leq y + 1 \)
   - \( sv \geq 5 \): (1, 0) (2, 1)

2. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 3 \): (0, \( sv - 2 \)) (1, \( sv - 3 \))
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{NB\_VALLEY}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with sv } = |\text{VARIABLES}|
\end{align*}
\]

1. \(4 \times x + y \leq 2 \times \text{sv} - 2\)
   - \(\text{sv mod 2} = 0 \land \text{sv} \geq 10: ((\text{sv} - 1)/2, 2) ((\text{sv} - 1)/2 - 1, 6)\)
   - \(\text{sv mod 2} = 1 \land \text{sv} \geq 13: ((\text{sv} - 1)/2 - 1, 4) ((\text{sv} - 1)/2 - 2, 8)\)

2. \(x + y \leq \text{sv} - 1\)
   - \(\text{sv} \geq 7: (1, \text{sv} - 2) (2, \text{sv} - 3)\)

3. \(y > 0 \Rightarrow x \geq 1\)
   - \(\text{sv} \geq 5: (1, \text{sv} - 2) (1, \text{sv} - 3)\)

4. \(y \neq 1\)
\[ \text{NB\_VALLEY}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(4 \times x + y \leq 2 \times sv - 2\)
   - \(sv \mod 2 = 0 \land sv \geq 12: (\lfloor (sv - 1)/2 \rfloor, 2) (\lfloor (sv - 1)/2 \rfloor - 1, 6)\)
   - \(sv \mod 2 = 1 \land sv \geq 15: (\lfloor (sv - 1)/2 \rfloor - 1, 4) (\lfloor (sv - 1)/2 \rfloor - 2, 8)\)

2. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 5: (0, sv - 2) (1, sv - 3)\)

3. \(y \neq 1\)

\[ \text{NB\_VALLEY}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(4 \times x + y \leq 2 \times sv - 2\)
   - \(sv \mod 2 = 0 \land sv \geq 6: (\lfloor (sv - 1)/2 \rfloor, 2) (\lfloor (sv - 1)/2 \rfloor - 1, 6)\)
   - \(sv \mod 2 = 1 \land sv \geq 9: (\lfloor (sv - 1)/2 \rfloor - 1, 4) (\lfloor (sv - 1)/2 \rfloor - 2, 8)\)

2. \(y \neq 1\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\( \text{NB}_\text{VALLEY}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{STRICTLY}_\text{DECREASING}_\text{SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2x + y \leq 2sv - 2 \)
2. \( 2x \leq y \)
3. \( sv \geq 5: (1, 2) (2, 4) \)
4. \( x \neq 0 \lor y \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)

\( \text{NB}_\text{VALLEY}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( 2x + y \leq 2sv - 2 \)
2. \( 2x \leq y \)
3. \( sv \geq 5: (1, 2) (2, 4) \)
4. \( x \neq 0 \lor y \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)
\[ \text{NB\_VALLEY}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 3x \leq y + sv \]
2. \[ sv > 1 \Rightarrow x + y \leq sv - 2 \]
3. \[ sv \geq 3: \ (0, sv - 2) \ \ (1, sv - 3) \]

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_VALLEY}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ x + y \leq sv - 1 \]
2. \[ sv \geq 5: \ (1, sv - 2) \ \ (2, sv - 3) \]
3. \[ x \leq y \]
4. \[ sv \geq 3: \ (0, 0) \ \ (1, 1) \]
5. \[ y > 0 \Rightarrow x \geq 1 \]
6. \[ sv \geq 4: \ (1, sv - 2) \ \ (1, sv - 3) \]

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB\_VALLEY}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq 2 \cdot x + 1 \)
   - \( sv \geq 7 \): (1, 3) (2, 5)
2. \( y = 0 \Rightarrow 3 \cdot x \leq sv - sv \mod 3 \)
3. \( x > 0 \land y > 0 \Rightarrow 3 \cdot x \leq y + sv - 2 \)
4. \( y > 0 \Rightarrow 6 \cdot x \leq y + 2 \cdot sv - 1 \)
   - \( sv \mod 2 = 0 \land sv \geq 14 \): \( \lfloor (sv - 1)/2 \rfloor, sv - 5 \) \( (sv - 1)/2 - 1, sv - 11 \)
   - \( so \mod 2 = 1 \land sv \geq 17 \): \( \lfloor (sv - 1)/2 \rfloor - 1, sv - 8 \) \( (sv - 1)/2 - 2, sv - 14 \)
5. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 5 \): (1, 2) (1, 3)
6. \( y \neq 1 \)
\[ \text{NB}_\text{ZIGZAG}(x, \text{VARIABLES}) \land \]
\[ \text{SUM_WIDTH_DECREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ 2 \times x \leq y \]
2. \[ sv \geq 7: (1, 2) \quad (2, 4) \]
3. \[ \lor \left( \begin{array}{l}
  x \neq \max(0, \left\lfloor (sv - 1)/3 \right\rfloor) - 0,
  y < 2 \times \max(0, \left\lfloor (sv - 1)/3 \right\rfloor) + 1,
  y > sv \times \min(1, \max(0, sv - 1)) - 1,
  1 \neq sv \mod 3
\end{array} \right) \]
4. \[ y \neq 2 \times x + 1 \]
\[ \text{NB}_ZIGZAG(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_DECREASING_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x = \left\lfloor (sv - 1)/3 \right\rfloor - 1 \land sv > 3 \Rightarrow y \leq 6 + 2 \ast (sv - 1) \mod 3 \)
2. \( x = \left\lfloor (sv - 1)/3 \right\rfloor \land sv > 3 \Rightarrow y \leq 2 \ast (sv - 1) \mod 3 \)
3. \( sv > 1 \Rightarrow 2 \ast x + y \leq sv - 2 \)
   - \( sv \geq 6: (0, sv - 2) \land (1, sv - 4) \)
4. \( x > 0 \land y > 0 \Rightarrow 3 \ast x \leq sv - 2 - (sv - 2) \mod 3 \)
   - \( (sv - 2) \mod 3 = 1 \land sv \geq 9: \left(\left\lfloor (sv - 1)/3 \right\rfloor, 2 \right) \land \left(\left\lfloor (sv - 1)/3 \right\rfloor, 3 \right) \)
   - \( (sv - 2) \mod 3 = 2 \land sv \geq 7: \left(\left\lfloor (sv - 1)/3 \right\rfloor - 1, 2 \right) \land \left(\left\lfloor (sv - 1)/3 \right\rfloor - 1, 3 \right) \)
5. \( y \neq 1 \)
\[ \text{NB\_ZIGZAG}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 5 \cdot x \leq y + sv \)
   - \( sv \geq 4: (0, 0) \) (1, 1)

2. \( x \leq y \)
   - \( sv \geq 4: (0, sv - 2) \) (1, sv - 3)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{NB\_ZIGZAG}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2 \cdot x \leq y \)
   - \( sv \geq 7: (1, 2) \) (2, 4)
   - \( y < 2 \cdot \max(0, \left\lfloor (sv - 1)/3 \right\rfloor) + 1 \)
   - \( y > sv \cdot \min(1, \max(0, sv - 1)) - 1 \)
   - \( 1 \neq sv \mod 3 \)

2. \( y \neq 2 \cdot x + 1 \)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_ZIGZAG}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x = \lceil (sv - 1)/3 \rceil - 1 \land sv > 3 \Rightarrow y \leq 6 + 2 \ast (sv - 1) \mod 3\)
2. \(x = \lceil (sv - 1)/3 \rceil \land sv > 3 \Rightarrow y \leq 2 \ast (sv - 1) \mod 3\)
3. \(sv > 1 \Rightarrow 2 \ast x + y \leq sv - 2\)
   - \(sv \geq 6: (0, sv - 2) \land (1, sv - 4)\)
4. \(x > 0 \land y > 0 \Rightarrow 3 \ast x \leq sv - 2 - (sv - 2) \mod 3\)
   - \((sv - 2) \mod 3 = 1 \land sv \geq 9: ((sv - 1)/3, 2) \land ((sv - 1)/3, 3)\)
   - \((sv - 2) \mod 3 = 2 \land sv \geq 7: ((sv - 1)/3 - 1, 2) \land ((sv - 1)/3 - 1, 3)\)
5. \(y \neq 1\)

\[
\text{NB\_ZIGZAG}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INFLUXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(3 \ast x \leq y + 1\)
   - \(sv \geq 7: (1, 2) \land (2, 5)\)
2. \(y > 0 \Rightarrow 3 \ast x \leq y + 1\)
   - \(sv \geq 7: (1, 2) \land (2, 5)\)
\[
\text{NB\_ZIGZAG}(x, \text{VAR\_IBL\_ES}) \land \\
\text{SUM\_WIDTH\_PEAK}(y, \text{VAR\_IBL\_ES}) \text{ with } sv = |\text{VAR\_IBL\_ES}|
\]

1. \(5 \times x \leq y + sv\)
2. \(2 \times x \leq y + 2\)
   - \(sv \geq 11\): (2, 2) (3, 4)
3. \(x \leq y\)
   - \(sv \geq 4\): (0, 0) (1, 1)
4. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   - \(sv \geq 4\): (0, \(sv - 2\)) (1, \(sv - 3\))
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{NB}_\text{ZIGZAG}(x, \text{VARIABLES}) \land \]
\[ \text{SUM}_\text{WIDTH}_{\text{PLAIN}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( 2 * x + y \leq sv \)
   - \( sv \geq 11 \): (2, \( sv - 4 \)) (3, \( sv - 6 \))

2. \( x \leq y \)
   - \( sv \geq 4 \): (0, 0) (1, 1)

3. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 4 \): (0, \( sv - 2 \)) (1, \( sv - 3 \))
\[ \text{NB}_\text{ZIGZAG}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{PROPER}_\text{PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[3 \times x + y \leq sv\]
   \[sv \geq 11: (2, sv - 6) \quad (3, sv - 9)\]
2. \[x > 0 \land y > 0 \Rightarrow x + y \leq sv - 3\]
3. \[y > 0 \Rightarrow 3 \times x \leq sv - 2 - (sv - 2) \mod 3\]
   \[[sv - 2] \mod 3 = 1 \land sv \geq 9: \left\lfloor (sv - 1)/3 \right\rfloor, 2 \quad \left\lfloor (sv - 1)/3 \right\rfloor, 3\]
   \[[sv - 2] \mod 3 = 2 \land sv \geq 7: \left\lfloor (sv - 1)/3 \right\rfloor - 1, 2 \quad \left\lfloor (sv - 1)/3 \right\rfloor - 1, 3\]
4. \[y \neq 1\]
\[
\text{NB}_\text{zigzag}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{width}_\text{proper}_\text{plateau}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(3 \times x + y \leq sv\)
   - \(sv \geq 11: (2, sv - 6) \quad (3, sv - 9)\)
2. \(x > 0 \land y > 0 \Rightarrow x + y \leq sv - 3\)
3. \(y > 0 \Rightarrow 3 \times x \leq sv - 2 - (sv - 2) \mod 3\)
   - 
   - 
4. \(y \not= 1\)
$3579$

$\text{nb\_zigzag}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|$

\begin{enumerate}
  \item $2 \cdot x + y \leq sv$
  \item $sv \geq 10$: $(1, sv - 2) \quad (2, sv - 4)$
  \item $x = \lfloor (sv - 1)/3 \rfloor - 1 \land sv > 3 \Rightarrow y \leq 6 + 2 \cdot (sv - 1) \mod 3$
  \item $x = \lfloor (sv - 1)/3 \rfloor \land sv > 3 \Rightarrow y \leq 2 \cdot (sv - 1) \mod 3$
  \item $x > 0 \land y > 0 \Rightarrow 3 \cdot x \leq sv - 2 - (sv - 2) \mod 3$
  \item $(sv - 2) \mod 3 = 1 \land sv \geq 6$: $(\lfloor (sv - 1)/3 \rfloor, 2) \quad (\lfloor (sv - 1)/3 \rfloor, 3)$
  \item $(sv - 2) \mod 3 = 2 \land sv \geq 4$: $(\lfloor (sv - 1)/3 \rfloor - 1, 2) \quad (\lfloor (sv - 1)/3 \rfloor - 1, 3)$
  \item $y \neq 1$
\end{enumerate}
5. A DATABASE OF PARAMETERISED INVARIANTS

\text{NB\_ZIGZAG}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

\begin{align*}
\text{1. } & 2 \ast x \leq y \\
\text{2. } & sv \geq 7: (1, 2) \quad (2, 4) \\
\text{3. } & sv \neq 7 & (sv - 1) / 3) - 0,
\begin{aligned}
& x \neq \text{max}(0, [sv - 1] / 3) - 0, \\
& y < 2 \ast \text{max}(0, [sv - 1] / 3) + 1,
\quad y > sv \ast \text{min}(1, \text{max}(0, sv - 1)) - 1,
\quad 1 \neq sv \mod 3
\end{aligned}
\text{4. } & y \neq 2 \ast x + 1
\end{align*}
\( \text{NB}_ZIGZAG(x, \text{VARIABLES}) \land \text{SUM}_WIDE\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \)

1. \(2 \times x \leq y\)
2. \(sv \geq 7: (1, 2) \land (2, 4)\)
3. \(sv = 7 \land (x \neq \max(0, ((sv - 1)/3)) - 0, y < 2 \times \max(0, ((sv - 1)/3)) + 1, y > sv \times \min(1, \max(0, sv - 1)) - 1, 1 \neq sv \mod 3)\)
4. \(y \neq 2 \times x + 1\)

\(sv = 12\)

\(sv = 11\)

\(\text{NB}_ZIGZAG(x, \text{VARIABLES}) \land \text{SUM}_WIDE\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \)

1. \(5 \times x \leq y + sv\)
2. \(x \leq y\)
3. \(sv \geq 4: (0, 0) \land (1, 1)\)
4. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
5. \(sv \geq 4: (0, sv - 2) \land (1, sv - 3)\)

\(sv = 12\)

\(sv = 11\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{NB\_ZIGZAG}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \\
\begin{align*}
1. & \quad 5 \cdot x \leq y + sv \\
2. & \quad 2 \cdot x \leq y + 2 \\
& \quad sv \geq 11: \quad (2, 2) \quad (3, 4) \\
3. & \quad x \leq y \\
& \quad sv \geq 4: \quad (0, 0) \quad (1, 1) \\
4. & \quad sv \geq 1 \Rightarrow x + y \leq sv - 2 \\
& \quad sv \geq 4: \quad (0, sv - 2) \quad (1, sv - 3)
\end{align*}
\]
\[
\text{NB\_ZIGZAG}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \times x \leq y\)
   - \(sv \geq 4: \) (0, 0) (1, 2)

2. \(x + y \leq sv - 1\)
   - \(sv \geq 7: \) (1, \(sv - 2\)) (2, \(sv - 3\))

3. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 5: \) (1, \(sv - 2\)) (1, \(sv - 3\))
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \wedge \text{SUM\_WIDTH\_DECREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow y \leq x - 2 \)
2. \( y > 0 \Rightarrow x \geq 4 \)
3. \( x \neq 5 \lor y < 4 \)
4. \( x \neq sv \times \min(1, \max(0, sv - 1)) - 4, \frac{y}{(sv - 2) \times \min(1, \max(0, sv - 3)) - 3} \)
5. \( x \neq sv \times \min(1, \max(0, sv - 1)) - 5, \frac{y}{(sv - 2) \times \min(1, \max(0, sv - 3)) - 4} \)
6. \( x \neq 1 \lor y \neq 0 \)
7. \( x \neq 4 \lor 0 = y \mod 2 \)
8. \( x \neq 3 \lor y < 1 \)
9. \( y \neq 1 \)
$$\text{SUM_WIDTH_DECREASING_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|$$

1. $x > 0 \land y > 0 \Rightarrow x + y \leq 2 * sv - 3$
2. $sv \geq 3: (sv, sv - 3) \quad (sv - 1, sv - 2)$
3. $y > 0 \Rightarrow x \geq 2$
4. $sv \geq 5: (2, sv - 2) \quad (2, sv - 3)$
5. $x \neq 3 \lor y \neq 1$
6. $x \neq 1$

$sv = 12$

$sv = 11$
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM_WIDTH_DECREASING_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_INCREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq 2 \cdot sv - 2 \)
2. \( sv \mod 2 = 0 \land sv \geq 2: (sv, sv - 2) \text{ if } (sv - 2, sv) \)
3. \( x = sv \land sv \mod 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 3 \)
4. \( x = sv - 2 \land sv \mod 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 1 \)
5. \( x \neq sv \cdot \min(1, \max(0, sv - 1)) - 1, y \neq sv \cdot \min(1, \max(0, sv - 1)) - 1, 1 = sv \mod 2 \)
6. \( y < 1, y > sv \cdot \min(1, \max(0, sv - 1)) - 3, 0 = y \mod 2 \)
7. \( x < 1, x > sv \cdot \min(1, \max(0, sv - 1)) - 3, 0 = x \mod 2 \)
8. \( y \neq 1 \land x \neq 1 \)
\[
\text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x + y \leq sv\)
   - \(sv \geq 0: (sv, 0) \quad (sv - 2, 2)\)
2. \(x > 0 \land y > 0 \Rightarrow x + 2 \ast y \leq 2 \ast sv - 4\)
   - \(sv \geq 6: (2, sv - 3) \quad (4, sv - 4)\)
3. \(y > 0 \Rightarrow x \leq sv - 2\)
4. \(x \neq 1\)
5. \(y \neq 1\)

\(sv = 12\)

\(sv = 11\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM_WIDTH_DECREASING_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{enumerate}
\item $x > 0 \land y > 0 \Rightarrow x \leq y + sv - 2$
\item $sv \geq 6$: $(sv, 2)$ $(sv - 1, 1)$
\item $y > 0 \Rightarrow x \geq 2$
\item $sv \geq 3$: $(2, sv - 2)$ $(2, sv - 3)$
\item $x \neq sv \ast \min(1, \max(0, sv - 1)) \lor y \neq 1$
\item $x \neq 1$
\end{enumerate}
\textit{SUM\_WIDTH\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land
\textit{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 \times sv - 3 \)
   \( sv \geq 0: \ (sv, \ sv - 3) \quad (sv - 1, \ sv - 2) \)

2. \( y > 0 \Rightarrow x \geq 2 \)
   \( sv \geq 3: \ (2, \ sv - 2) \quad (2, \ sv - 3) \)

3. \( x \neq 3 \lor y \neq 1 \)

4. \( x \neq 1 \)

\begin{figure}
\hspace{1cm}
\textit{SUM\_WIDTH\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land
\textit{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + 2 \times y \leq 2 \times sv - 2 \)
   \( sv \geq 4: \ (2, \ sv - 2) \quad (4, \ sv - 3) \)

2. \( y > 0 \Rightarrow x \geq 2 \)
   \( sv \geq 3: \ (2, \ sv - 2) \quad (2, \ sv - 3) \)

3. \( x \neq 1 \)

\end{figure}
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM_WIDTH_DECREASING_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x + 2 \times y \leq 2 \times sv - 2\)
2. \(sv \geq 4: (2, sv - 2), (4, sv - 3)\)
3. \(y > 0 \Rightarrow x \geq 2\)
4. \(sv \geq 3: (2, sv - 2), (2, sv - 3)\)
5. \(x \neq 1\)

\[sv = 12\]

\[sv = 11\]
SUM_WIDTH_DECREASING_SEQUENCE(x, VARIABLES) \land SUM_WIDTH_PROPERPLAIN(y, VARIABLES) with sv = |VARIABLES|

1. $3 \cdot x + 3 \cdot y \leq 4 \cdot sv - 2 - (4 \cdot sv - 2) \mod 3$
2. $2 \cdot x + y \leq 2 \cdot sv$
   - $sv \geq 5$: $(sv, 0)$  $(sv - 1, 2)$
3. $x + 2 \cdot y \leq 2 \cdot sv - 2$
   - $sv \geq 5$: $(2, sv - 2)$  $(4, sv - 3)$
4. $y > 0 \Rightarrow x \geq 2$
   - $sv \geq 5$: $(2, sv - 2)$  $(2, sv - 3)$
5. $x \neq 1$
6. $y \neq 1$
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{SUM_WIDTH_DECREASING_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_PROPER_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. \(2 \cdot x + y \leq 2 \cdot sv\)
   - \(sv \geq 5\): \((sv, 0)\) \((sv - 1, 2)\)

2. \(x + 2 \cdot y \leq 2 \cdot sv - 2\)
   - \(sv \geq 5\): \((2, sv - 2)\) \((4, sv - 3)\)

3. \(y > 0 \Rightarrow x \geq 2\)
   - \(sv \geq 5\): \((2, sv - 2)\) \((2, sv - 3)\)

4. \(x \neq 1\)

5. \(y \neq 1\)
\[ \text{SUM_WIDTH_DECREASING_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_STEADY_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq 2 \times sv - 2 \)
2. \( sv \geq 5: (sv, sv - 2) \quad (sv - 1, sv - 1) \)
3. \( x \neq 3 \lor y \neq sv \times \min(1, \max(0, sv - 1)) \)
4. \( y \neq 1 \)
5. \( x \neq 1 \)

sv = 12

sv = 11
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \]
\[ \text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq x \)
2. \( sv \geq 4 \quad (sv - 1, sv - 1) \quad (sv - 2, sv - 2) \)
3. \( sv > 1 \Rightarrow x \leq y + sv - 2 \)
4. \( x > 0 \Rightarrow y \geq 2 \)
5. \( y \neq 1 \)
6. \( x < 3 \lor y \neq 2 \)
7. \( x < 4 \lor y \neq 3 \)

sv = 12

\[
\begin{array}{c}
\text{sum\_width\_decreasing\_sequence} \\
\text{sum\_width\_strictly\_decreasing\_sequence}
\end{array}
\]

sv = 11

\[
\begin{array}{c}
\text{sum\_width\_decreasing\_sequence} \\
\text{sum\_width\_strictly\_decreasing\_sequence}
\end{array}
\]
\[
\text{SUM_WIDTH_DECREASING_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\[
x + y \leq 2 \cdot sv - 2
\]

\[
sv \text{ mod } 2 = 0 \land sv \geq 2: (sv, sv - 2) (sv - 2, sv)
\]

\[
x = sv \land sv \text{ mod } 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 3
\]

\[
x = sv - 2 \land sv \text{ mod } 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 1
\]

\[
y = sv \land sv \text{ mod } 2 = 1 \land sv > 1 \Rightarrow x \leq sv - 3
\]

\[
\begin{align*}
&x \neq sv \cdot \min(1, \max(0, sv - 1)) - 1, \\
y \neq sv \cdot \min(1, \max(0, sv - 1)) - 1, \\
&1 = sv \text{ mod } 2 \\
x \neq sv \cdot \min(1, \max(0, sv - 1)), \\
y < 1, \\
y > sv \cdot \min(1, \max(0, sv - 1)) - 3, \\
0 = y \text{ mod } 2 \\
y \neq sv \cdot \min(1, \max(0, sv - 1)), \\
x < 1, \\
x > sv \cdot \min(1, \max(0, sv - 1)) - 3, \\
0 = x \text{ mod } 2
\end{align*}
\]

\[
y \neq 1
\]

\[
x \neq 1
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 * sv - 3 \)
   - \( sv \geq 6: (sv, sv - 3) \quad (sv - 1, sv - 2) \)
2. \( y > 0 \Rightarrow x \geq 2 \)
   - \( sv \geq 3: (2, sv - 2) \quad (2, sv - 3) \)
3. \( x \neq 3 \lor y \neq 1 \)
4. \( x \neq 1 \)

\[
\begin{array}{c}
\text{sum\_width\_decreasing\_sequence} \\
\hline
\text{sum\_width\_summit} \\
\end{array}
\]

\( sv = 12 \)

\( sv = 11 \)
\text{SUM_WIDTH_DECREASING_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|$

1. $x > 0 \land y > 0 \Rightarrow x + y \leq 2 \times sv - 3$
2. $sv \geq 3: (sv, sv - 3) \quad (sv - 1, sv - 2)$
3. $y > 0 \Rightarrow x \geq 2$
4. $sv \geq 4: (2, sv - 2) \quad (2, sv - 3)$
5. $x \neq 3 \lor y \neq 1$
6. $x \neq 1$

$s v = 12$

$s v = 11$
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM_WIDTH_DECREASING_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq x \)
   - \( sv \mod 2 = 0 \land sv \geq 3: (sv - 2, sv - 2) \) (0, 0)
   - \( sv \mod 2 = 1 \land sv \geq 6: (sv - 3, sv - 3) \) (2, 2)
2. \( x = sv \land sv \mod 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 3 \)
3. \( y > 0 \Rightarrow x \geq 2 \)
   - \( y \neq (sv - 2) \ast \min(1, \max(0, sv - 3)), x < 3, \)
   - \( x > sv \ast \min(1, \max(0, sv - 1)) - 1, 1 = sv \mod 2, \)
   - \( 0 = x \mod 2 \)
4. \( x \neq 5 \lor y < 4 \)
5. \( x \neq 1 \)
6. \( x \neq 3 \lor y < 1 \)
7. \( y \neq x \lor 0 = x \mod 2 \)
8. \( y \neq x \lor 0 = x \mod 2 \)
   - \( x \neq sv \ast \min(1, \max(0, sv - 1)), y < 1, \)
   - \( y > (sv - 2) \ast \min(1, \max(0, sv - 3)) - 1, 0 = y \mod 2 \)
9. \( y \neq 1 \)

\[ \begin{align*}
\text{sum_width_decreasing_sequence} & \quad \text{sum_width_zigzag} \\
0 & \quad 0 \\
2 & \quad 2 \\
4 & \quad 4 \\
6 & \quad 6 \\
8 & \quad 8 \\
10 & \quad 10
\end{align*} \]

\[ \begin{align*}
\text{sum_width_decreasing_sequence} & \quad \text{sum_width_zigzag} \\
0 & \quad 0 \\
2 & \quad 2 \\
4 & \quad 4 \\
6 & \quad 6 \\
8 & \quad 8 \\
10 & \quad 10
\end{align*} \]
\[
\text{SUM_WIDTH_DECENDING_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_GORGE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x > 0 \land y > 0 \Rightarrow x \leq sv - 3\)
2. \(y \neq (sv - 2) \times \min(1, \max(0, sv - 3)) - 2\)
3. \(sv < 5\)
4. \(x \neq (sv - 2) \times \min(1, \max(0, sv - 3)) - 2\)
5. \(y \neq 2\)
6. \(sv < 5\)

\[
\begin{array}{c}
\text{sv } = 12 \\
\text{sv } = 11
\end{array}
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

SUM_WIDTH_DECREASING_TERRACE(x, VARIABLES) ∧
SUM_WIDTH_INCREASING_SEQUENCE(y, VARIABLES) with sv = |VARIABLES|

1. \( x + y \leq sv \)
   - \( sv \geq 7: (sv - 4, 4) \quad (sv - 5, 5) \)

2. \( x > 0 \Rightarrow y \leq sv - 2 \)

3. \( x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq 2 \times sv - 4 \)
   - \( sv \geq 6: (sv - 3, 2) \quad (sv - 4, 4) \)

4. \( y \neq 1 \)

5. \( x \neq 1 \)

\[ x \times y \leq sv \]
\[ sv \geq 7: (sv - 4, 4) \quad (sv - 5, 5) \]
\[ x > 0 \Rightarrow y \leq sv - 2 \]
\[ x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq 2 \times sv - 4 \]
\[ sv \geq 6: (sv - 3, 2) \quad (sv - 4, 4) \]
\[ y \neq 1 \]
\[ x \neq 1 \]
\[\text{SUM_WIDTH_DECREASING_TERRACE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_INCREASING_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(sv > 1 \implies x + y \leq sv - 2\)
2. \(x > 0 \land y > 0 \implies x + y \leq sv - 3\)
3. \(sv \geq 4: (sv - 2, 0) \land (0, sv - 2)\)
4. \(sv \geq 8: (sv - 5, 2) \land (2, sv - 5)\)
5. \(sv = 12\)
6. \(sv = 11\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM WIDTH DECREASING TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM WIDTH INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \)
2. \( sv \geq 5: (sv - 3, sv - 2) \land (sv - 3, 1) \)
3. \( x \neq (sv - 2) * \min(1, \max(0, sv - 3)) - 2 \lor y \neq 3 \lor sv < 7 \)
4. \( y < 2, \\
   y > (sv - 2) * \min(1, \max(0, sv - 2)) - 1 \)
5. \( x \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)
\[
\text{SUM\_WIDTH\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x > 0 \land y > 0 \Rightarrow 2x \leq y + 2sv - 9 \)
2. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \)
3. \( x \neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 2, \quad y \neq (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 2, \quad sv < 5 \)
4. \( x \neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 2 \lor y \neq 2 \lor sv < 5 \)
5. \( x \neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 1, \quad y < 1, \quad y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 1 \)
6. \( x \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM\_WIDTH\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
2. \(x > 0 \Rightarrow y \leq sv - 4\)
3. \(x \neq 1\)

\[
sv > 1 \Rightarrow x + y \leq sv - 2
\]

\[
sv \geq 5: (sv - 2, 0) (sv - 3, 1)
\]

\[
x > 0 \Rightarrow y \leq sv - 4
\]

\[
x \neq 1
\]
\( \text{SUM_WIDTH_DECREASING_TERRACE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_PROPER.PLAIN}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
2. \( sv \geq 7: (sv - 2, 0) \) \( (sv - 4, 2) \)
3. \( x > 0 \Rightarrow y \leq sv - 4 \)
4. \( y > 0 \Rightarrow x \leq sv - 4 \)
5. \( o^m \geq \left\lfloor (x + 1)/2 \right\rfloor + \left\lfloor (y + 1)/2 \right\rfloor \)
6. \( y \neq 1 \)
7. \( x \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM WIDTH DECREASING TERRACE}(x, \text{VARIABLES}) \land \text{SUM WIDTH PROPER PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
2. \( sv \geq 6: (sv - 2, 0) \quad (sv - 4, 2) \)
3. \( x > 0 \Rightarrow y \leq sv - 4 \)
4. \( y > 0 \Rightarrow x \leq sv - 4 \)
5. \( z \geq f = \left\lfloor \frac{x + 1}{2} \right\rfloor + \left\lfloor \frac{y + 1}{2} \right\rfloor \)
6. \( y \neq 1 \)
7. \( z \neq 1 \)
\[
\text{SUM WIDTH DECREASING TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM WIDTH STEADY SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x \leq y \)
2. \( x > 0 \Rightarrow y \geq 2 \)
3. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 \times sv - 4 \)
4. \( y \neq 4 \lor 0 = x \mod 2 \)
5. \( y \neq 1 \)
6. \( y \neq x + 1 \)
7. \( x \neq 1 \)

\[
\begin{array}{c}
\text{sv} = 12 \\
\text{sv} = 11
\end{array}
\]
5. A DATABASE OF PARAMETERISED INVARIENTS

\[
\text{SUM}_\text{WIDTH}_\text{DECREASING}_\text{TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{STRICTLY}_\text{DECREASING}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x = \max(0, sv - 2) \land sv \mod 2 = 1 \land sv > 3 \Rightarrow y \leq sv - 1 \)
2. \( y = 0 \Rightarrow x = 0 \)
3. \( x > 0 \land y > 0 \Rightarrow y \geq 4 \)
   \( sv \geq 5: (sv - 2, 4) (sv - 3, 4) \)
4. \( x \neq (sv - 2) \times \min(1, \max(0, sv - 3)), \)
   \( y < 3, \)
   \( y > sv \times \min(1, \max(0, sv - 1)) - 1, \)
   \( 0 = y \mod 2 \)
5. \( x < 1 \lor y \neq 2 \)
   \( y \neq sv \times \min(1, \max(0, sv - 1)), \)
6. \( x < 1 \)
7. \( x > (sv - 2) \times \min(1, \max(0, sv - 3)) - 1, \)
   \( 0 = x \mod 2 \)
8. \( x < 1 \lor y \neq 3 \)
9. \( x \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)
\[ \text{SUM}_\text{WIDTH\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv \)
2. \( sv \geq 7: (sv - 4, 4) \quad (sv - 5, 5) \)
3. \( x > 0 \Rightarrow y \leq sv - 2 \)
4. \( x > 0 \land y > 0 \Rightarrow 2 \times x + y \leq 2 \times sv - 4 \)
5. \( sv \geq 6: (sv - 3, 2) \quad (sv - 4, 4) \)
6. \( y \neq 1 \)
7. \( x \neq 1 \)

\[ \text{sum_width_decreasing_terrace} \]

\[ \text{sum_width_strictly_increasing_sequence} \]

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM}_\text{WIDTH}_\text{DECREASING}_\text{TERRACE}(x, \text{VARIABLES}) \land \]
\[ \text{SUM}_\text{WIDTH}_\text{SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\begin{align*}
\mathbf{1} \quad & x > 0 \land y > 0 \implies 2x \leq y + 2sv - 9 \\
\mathbf{2} \quad & x > 0 \land y > 0 \implies x \leq sv - 3 \\
\mathbf{3} \quad & \bigvee \left( \begin{array}{l}
x \neq (sv - 2) \ast \min(1, \max(0, sv - 3)) - 2, \\
y \neq (sv - 2) \ast \min(1, \max(0, sv - 2)) - 2, \\
sv < 5 \end{array} \right) \\
\mathbf{4} \quad & x \neq (sv - 2) \ast \min(1, \max(0, sv - 3)) - 2 \lor y \neq 2 \lor sv < 5 \\
\mathbf{5} \quad & \bigvee \left( \begin{array}{l}
x \neq (sv - 2) \ast \min(1, \max(0, sv - 3)) - 1, \\
y < 1, \\
y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 1 \end{array} \right) \\
\mathbf{6} \quad & x \neq 1
\end{align*}

\begin{align*}
& sv = 12 \\
& sv = 11
\end{align*}
$\text{SUM\_WIDTH\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES})$ with $sv = |\text{VARIABLES}|

1. $x > 0 \land y > 0 \Rightarrow x \leq sv - 3$
2. $x > 0 \land y > 0 \Rightarrow x \leq y + sv - 5$
3. $x \neq (sv - 2) \ast \min(1, \max(0, sv - 3)) - 2, y \neq (sv - 2) \ast \min(1, \max(0, sv - 2)) - 2, sv < 5$
4. $x \neq (sv - 2) \ast \min(1, \max(0, sv - 3)) - 2 \lor y \neq 2 \lor sv < 5$
5. $y < 1, y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 1$
6. $x \neq 1$
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM\_WIDTH\_DECREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. $sv > 1 \Rightarrow x + y \leq sv - 2$
2. $sv \geq 6: (sv - 2, 0) \ (sv - 4, 2)$
3. $x > 0 \Rightarrow y \leq sv - 4$
4. $y > 0 \Rightarrow x \leq sv - 4$
5. $y \neq 1$
6. $x \neq 1$
\[
\text{SUM\_WIDTH\_GORGE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x > 0 \Rightarrow y \geq 2\)
   - \(sv \geq 3: (sv - 2, 2) \quad (sv - 3, 2)\)
2. \(x > 0 \land y > 0 \Rightarrow x + y \leq 2sv - 3\)
   - \(sv \geq 6: (sv - 2, sv - 1) \quad (sv - 3, sv)\)
3. \(x \neq 1 \lor y \neq 3\)
4. \(y \neq 1\)

\(sv = 12\) \hspace{1cm} \(sv = 11\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM\_WIDTH\_GORGE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INCREASING\_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x > 0 \land y > 0 \Rightarrow y \leq sv - 3 \)
2. \( x > 0 \land y > 0 \Rightarrow 2 \cdot y \leq x + 2 \cdot sv - 9 \)
3. \( \bigvee \left( \begin{array}{c} 
    x \neq (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 2, \\
    y \neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 2, \\
    sv < 5 
\end{array} \right) \)
4. \( x \neq 2 \lor y \neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 2 \lor sv < 5 \)
5. \( \bigvee \left( \begin{array}{c} 
    y \neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 1, \\
    x < 1, \\
    x > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 1 
\end{array} \right) \)
6. \( y \neq 1 \)
\[ \text{SUM\_WIDTH\_GORGE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_INFLEXION}(y, \text{VARIABLES}) \text{ with } \text{sv} = |\text{VARIABLES}| \]

1. \( x > 0 \Rightarrow y \geq 1 \)
2. \( \text{sv} \geq 5 : (\text{sv} - 2, 1) \quad (\text{sv} - 3, 1) \)

\( \text{sv} = 12 \)

\[ \begin{array}{|c|c|}
\hline
\text{sum\_width\_gorge} & \text{sum\_width\_inflexion} \\
\hline
0 & 0 \\
1 & 1 \\
2 & 2 \\
3 & 3 \\
4 & 4 \\
5 & 5 \\
6 & 6 \\
7 & 7 \\
8 & 8 \\
9 & 9 \\
10 & 10 \\
\hline
\end{array} \]

\( \text{sv} = 11 \)

\[ \begin{array}{|c|c|}
\hline
\text{sum\_width\_gorge} & \text{sum\_width\_inflexion} \\
\hline
0 & 0 \\
1 & 1 \\
2 & 2 \\
3 & 3 \\
4 & 4 \\
5 & 5 \\
6 & 6 \\
7 & 7 \\
8 & 8 \\
9 & 9 \\
\hline
\end{array} \]

\[ \text{SUM\_WIDTH\_GORGE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PEAK}(y, \text{VARIABLES}) \text{ with } \text{sv} = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow 2x + y \leq 3x - sv - 9 \)
2. \( x > 0 \land y > 0 \Rightarrow x + 2y \leq 3x - sv - 9 \)
3. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \)
4. \( \text{sv} \geq 4 : (sv - 3, sv - 3) \quad (sv - 3, sv - 4) \)
5. \( x > 0 \land y > 0 \Rightarrow y \leq sv - 3 \)
6. \( \text{sv} \geq 4 : (sv - 3, sv - 3) \quad (sv - 4, sv - 3) \)

\( \text{sv} = 12 \)

\[ \begin{array}{|c|c|}
\hline
\text{sum\_width\_gorge} & \text{sum\_width\_peak} \\
\hline
0 & 0 \\
1 & 1 \\
2 & 2 \\
3 & 3 \\
4 & 4 \\
5 & 5 \\
6 & 6 \\
7 & 7 \\
8 & 8 \\
9 & 9 \\
10 & 10 \\
\hline
\end{array} \]

\( \text{sv} = 11 \)

\[ \begin{array}{|c|c|}
\hline
\text{sum\_width\_gorge} & \text{sum\_width\_peak} \\
\hline
0 & 0 \\
1 & 1 \\
2 & 2 \\
3 & 3 \\
4 & 4 \\
5 & 5 \\
6 & 6 \\
7 & 7 \\
8 & 8 \\
9 & 9 \\
\hline
\end{array} \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM}_\text{WIDTH}_\text{GORGE}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x + y \leq sv - 1\)
   \(sv \geq 5:\ (sv - 2, 1) \quad (sv - 3, 2)\)
2. \(x > 0 \Rightarrow y \geq 1\)
   \(sv \geq 4:\ (sv - 2, 1) \quad (sv - 3, 1)\)
3. \(x \neq 0 \lor y \neq 1\)
4. \(x \neq 1 \lor y \neq 2\)

\[
\begin{align*}
\text{sv} = 12 & \quad \text{sv} = 11 \\
\text{sum_width}_\text{gorge} & \quad \text{sum_width}_\text{gorge} \\
\text{sum_width}_\text{plain} & \quad \text{sum_width}_\text{plain}
\end{align*}
\]

\[
\text{SUM}_\text{WIDTH}_\text{GORGE}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
   \(sv \geq 3:\ (sv - 2, 0) \quad (sv - 3, 1)\)

\[
\begin{align*}
\text{sv} = 12 & \quad \text{sv} = 11 \\
\text{sum_width}_\text{gorge} & \quad \text{sum_width}_\text{gorge} \\
\text{sum_width}_\text{plateau} & \quad \text{sum_width}_\text{plateau}
\end{align*}
\]
\text{SUM_WIDTH_GORGE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_PROPER.PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
2. \(sv \geq 3: (sv - 2, 0) \land (0, sv - 2)\)
3. \(x > 0 \land y > 0 \Rightarrow x + y \leq sv - 3\)
4. \(sv \geq 7: (sv - 5, 2) \land (1, sv - 4)\)
5. \(x \neq 4 \lor y \neq (sv - 2) * \min(1, \max(0, sv - 3)) - 4 \lor sv < 7\)
6. \(x \neq (sv - 2) * \min(1, \max(0, sv - 2)) - 4 \lor y < 4 \lor sv < 7\)
7. \(y \neq (sv - 2) * \min(1, \max(0, sv - 3)) - 3, x < 3, x > (sv - 2) * \min(1, \max(0, sv - 2)) - 1, 0 = x \mod 2 \lor y \neq (sv - 2) * \min(1, \max(0, sv - 3)) - 1, x < 1, x > (sv - 2) * \min(1, \max(0, sv - 2)) - 2, 0 = x \mod 2\)
8. \(y \neq (sv - 2) * \min(1, \max(0, sv - 3)) - 2, x < 2, x > (sv - 2) * \min(1, \max(0, sv - 2)) - 1, 1 = x \mod 2 \lor x \neq (sv - 2) * \min(1, \max(0, sv - 2)) - 3, y < 1, y > (sv - 2) * \min(1, \max(0, sv - 3)) - 1, 0 = y \mod 2 \lor x \neq (sv - 2) * \min(1, \max(0, sv - 2)) - 2, y < 1, y > (sv - 2) * \min(1, \max(0, sv - 3)) - 1\)
9. \(y \neq 1\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM_WIDTH_GORGE}(x, \text{VARIABLES}) \land \]
\[ \text{SUM_WIDTH_PROPER_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
2. \( sv \geq 4: (sv - 2, 0) \ (sv - 4, 2) \)
3. \( y > 0 \Rightarrow x \leq sv - 4 \)
4. \( y \neq 1 \)

\[ sv = 12 \]

\[ sy = 11 \]

\[ \text{SUM_WIDTH_GORGE}(x, \text{VARIABLES}) \land \]
\[ \text{SUM_WIDTH_STEADY_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow x + 2 * y \leq 3 * sv - 6 \)
2. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 * sv - 5 \)
3. \( x > 0 \land y > 0 \Rightarrow y \leq sv - 1 \)
4. \( x \neq 2 \lor y \neq sv \times \min(1, \max(0, sv - 1)) - 1 \)
5. \( y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]
\[ \text{SUM_WIDTH_GORGE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \implies y \geq 2 \)
2. \( sv \geq 5 \implies (sv - 2, 2) \land (sv - 3, 2) \)
3. \( x > 0 \land y > 0 \implies x + y \leq 2 \times sv - 3 \)
4. \( sv \geq 3 \implies (sv - 2, sv - 1) \land (sv - 3, sv) \)
5. \( x \neq 1 \lor y \neq 3 \)
6. \( y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM}_\text{WIDTH}_\text{GORGE}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \Rightarrow y \geq 2 \)
   - \( sv \geq 3: (sv - 2, 2) \quad (sv - 3, 2) \)
2. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 \times sv - 3 \)
   - \( sv \mod 2 = 0 \land sv \geq 4: (sv - 2, sv - 1) \quad (sv - 3, sv) \)
   - \( sv \mod 2 = 1 \land sv \geq 5: (sv - 2, sv - 1) \quad (sv - 3, sv) \)
3. \( x \neq 1 \lor y \neq 3 \)
4. \( y \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)
\[ \text{SUM_WIDTH_GORGE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow 2x + y \leq 3sv - 9 \)
2. \( x > 0 \land y > 0 \Rightarrow x + 2y \leq 3sv - 9 \)
3. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \)
   - \( sv \geq 4: (sv - 3, sv - 3) \quad (sv - 3, sv - 4) \)
4. \( x > 0 \land y > 0 \Rightarrow y \leq sv - 3 \)
   - \( sv \geq 4: (sv - 3, sv - 3) \quad (sv - 4, sv - 3) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{SUM}_\text{WIDTH}_\text{GORGE}(x, \text{VARIABLES}) & \land \\
\text{SUM}_\text{WIDTH}_\text{VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \(x \leq y\)
2. \(sv \geq 3: (sv - 2, sv - 2) \land (sv - 3, sv - 3)\)
   \[
   \begin{cases}
   x \neq (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 3, \\
   0 = y \mod 2,
   \\
   0 = sv \mod 2,
   \\
   sv < 6
   \end{cases}
   \]
3. \(sv < 6\)
   \[
   \begin{cases}
   x \neq (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 3, \\
   1 = y \mod 2,
   \\
   1 = sv \mod 2
   \end{cases}
   \]
4. \(y \neq x + 1\)
5. \(sv \geq 3\)

- For \(sv = 12\):
- Graph showing points satisfying the conditions.
- For \(sv = 11\):
- Graph showing points satisfying the conditions.
\[ \text{SUM_WIDTH_GORGE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq 3 \times x \)
2. \( sv \geq 5: (0, 0) \) (1, 3)
3. \( y = \max(0, sv - 2) \land sv > 1 \Rightarrow 2 \times x \geq sv - 2 - sv \mod 2 \)
4. \( sv > 1 \Rightarrow 2 \times x + y \leq 2 \times sv - 3 \)
5. \( sv \geq 7: (sv - 3, 3) (sv - 4, 5) \)
6. \( y > 0 \Rightarrow x \leq sv - 3 \)
7. \( sv \geq 5: (sv - 3, 2) (sv - 3, 3) \)
8. \( y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM_WIDTH_INCREASING_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_INCREASING_TERRACE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x > 0 \land y > 0 \Rightarrow y \leq x - 2\)
2. \(y > 0 \Rightarrow x \geq 4\)
3. \(x \neq 5 \lor y < 4\)
4. \[\begin{align*}
x &\neq sv \cdot \min(1, \max(0, sv - 1)) - 4, \\
y &\neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 3
\end{align*}\]
5. \[\begin{align*}
x &\neq sv \cdot \min(1, \max(0, sv - 1)) - 5, \\
y &\neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 4
\end{align*}\]
6. \(x \neq 1 \lor y \neq 0\)
7. \(x \neq 4 \lor 0 = y \mod 2\)
8. \(x \neq 3 \lor y < 1\)
9. \(y \neq 1\)
\textsc{sum\_width\_increasing\_sequence}(x, \text{VARIABLES}) \land \\
\textsc{sum\_width\_inflexion}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

\begin{enumerate}
\item $x > 0 \land y > 0 \Rightarrow x \leq y + sv - 2$
\item $sv \geq 0: (sv, 2) \ (sv - 1, 1)$
\item $y > 0 \Rightarrow x \geq 2$
\item $sv \geq 3: (2, sv - 2) \ (2, sv - 3)$
\item $x \neq sv \ast \min(1, \max(0, sv - 1)) \lor y \neq 1$
\item $x \neq 1$
\end{enumerate}
5. A DATABASE OF PARAMETERISED INVARIANTS

**SUM_WIDTH_INCREASING_SEQUENCE**$(x, \text{VARIABLES}) \land \;
\text{SUM_WIDTH_PEAK}(y, \text{VARIABLES})$ with $sv = |\text{VARIABLES}|

1. $x > 0 \land y > 0 \Rightarrow x + y \leq 2 * sv - 3$

   $\square$ $sv \geq 3$: $(sv, sv - 3)$ $(sv - 1, sv - 2)$

2. $y > 0 \Rightarrow x \geq 2$

   $\square$ $sv \geq 4$: $(2, sv - 2)$ $(4, sv - 3)$

3. $x \neq 3 \lor y \neq 1$

4. $x \neq 1$

---

**SUM_WIDTH_INCREASING_SEQUENCE**$(x, \text{VARIABLES}) \land \;
\text{SUM_WIDTH_PLAIN}(y, \text{VARIABLES})$ with $sv = |\text{VARIABLES}|

1. $x + 2 * y \leq 2 * sv - 2$

   $\square$ $sv \geq 4$: $(2, sv - 2)$ $(4, sv - 3)$

2. $y > 0 \Rightarrow x \geq 2$

   $\square$ $sv \geq 3$: $(2, sv - 2)$ $(2, sv - 3)$

3. $x \neq 1$
\[ \text{SUM_WIDTH_INCREASING_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_PLATEAU}(y, \text{VARIABLES}) \text{ with } s_v = |\text{VARIABLES}| \]

1. \( x + 2 \cdot y \leq 2 \cdot s_v - 2 \)
2. \( s_v \geq 4: (2, s_v - 2) \quad (4, s_v - 3) \)
3. \( y > 0 \Rightarrow x \geq 2 \)
4. \( s_v \geq 3: (2, s_v - 2) \quad (2, s_v - 3) \)
5. \( x \neq 1 \)

\[ x = 2 \]
\[ y = 4 \]
\[ s_v = 12 \]
\[ s_v = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM_WIDTH_INCREASING_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_PROPERPLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(2 * x + y \leq 2 * sv\)
   - \(sv \geq 5: (sv - 1, 2, 2)\)
2. \(x + 2 * y \leq 2 * sv - 2\)
   - \(sv \geq 5: (2, sv - 2, 4, sv - 3)\)
3. \(y > 0 \Rightarrow x \geq 2\)
   - \(sv \geq 5: (2, sv - 2, 2, sv - 3)\)
4. \(x \neq 1\)
5. \(y \neq 1\)
\[
\text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(3 \cdot x + 3 \cdot y \leq 4 \cdot sv - 2 - (4 \cdot sv - 2) \mod 3\)
2. \(2 \cdot x + y \leq 2 \cdot sv\)
   - \(sv \geq 5: (sv, 0) \quad (sv - 1, 2)\)
3. \(x + 2 \cdot y \leq 2 \cdot sv - 2\)
   - \(sv \geq 5: (2, sv - 2) \quad (4, sv - 3)\)
4. \(y > 0 \Rightarrow x \geq 2\)
   - \(sv \geq 5: (2, sv - 2) \quad (2, sv - 3)\)
5. \(x \neq 1\)
6. \(y \neq 1\)
A DATABASE OF PARAMETERISED INVARIANTS

\[ \begin{align*}
\text{SUM_WIDTH_INCREASING_SEQUENCE} & (x, \text{VARiABLES}) \land \\
\text{SUM_WIDTH_STEADYSEQUENCE} & (y, \text{VARiABLES}) \text{ with } sv = |\text{VARiABLES}|
\end{align*} \]

1. \( x + y \leq 2 + sv - 2 \)
2. \( sv \geq 5: (sv, sv - 2), (sv - 1, sv - 1) \)
3. \( x \neq 3 \lor y \neq sv \land \min(1, \max(0, sv - 1)) \)
4. \( y \neq 1 \)
5. \( x \neq 1 \)
\[
\begin{align*}
\text{SUM_WIDTH_INCREASING_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

- \( x + y \leq 2 \star sv - 2 \)
- \( sv \mod 2 = 0 \land sv \geq 2 : (sv, sv - 2) \quad (sv - 2, sv) \)
- \( x = sv \land sv \mod 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 3 \)
- \( y = sv \land sv \mod 2 = 1 \land sv > 1 \Rightarrow x \leq sv - 3 \)
- \( y \neq sv \star \min(1, \max(0, sv - 1)) - 1, 1 = sv \mod 2 \)
- \( x \neq sv \star \min(1, \max(0, sv - 1)), y < 1, y > sv \star \min(1, \max(0, sv - 1)) - 3, 0 = y \mod 2 \)
- \( y \neq sv \star \min(1, \max(0, sv - 1)), x < 1, x > sv \star \min(1, \max(0, sv - 1)) - 3, 0 = x \mod 2 \)
- \( y \neq 1 \)
- \( x \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\sum_{\text{width increasing sequence}}(x, \text{variables}) \land \\
\sum_{\text{width strictly increasing sequence}}(y, \text{variables}) \text{ with } sv = |\text{variables}|
\]

1. \( y \leq x \)
2. \( sv \geq 4: (sv - 1, sv - 1) \text{ (}sv - 2, sv - 2) \)
3. \( sv > 1 \Rightarrow x \leq y + sv - 2 \)
4. \( x > 0 \Rightarrow y \geq 2 \)
5. \( y \neq 1 \)
6. \( x < 3 \lor y \neq 2 \)
7. \( x < 4 \lor y \neq 3 \)

\begin{align*}
sv &= 12 \\
sv &= 11
\end{align*}
\[ \text{SUM WIDTH INCREASING SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM WIDTH SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 \cdot sv - 3 \)
2. \( sv \geq 3: \quad (sv, sv - 3) \quad (sv - 1, sv - 2) \)
3. \( y > 0 \Rightarrow x \geq 2 \)
4. \( sv \geq 5: \quad (2, sv - 2) \quad (2, sv - 3) \)
5. \( x \neq 3 \lor y \neq 1 \)
6. \( x \neq 1 \)

### Diagrams

- **sv = 12**
- **sv = 11**
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM\_WIDTH\_INCREASING\_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 * sv - 3 \)
   - \( sv \geq 6: (sv, sv - 3) \land (sv - 1, sv - 2) \)
2. \( y > 0 \Rightarrow x \geq 2 \)
   - \( sv \geq 3: (2, sv - 2) \land (2, sv - 3) \)
3. \( x \neq 3 \lor y \neq 1 \)
4. \( x \neq 1 \)

---

![Diagram](image-url)
\[ \text{SUM_WIDTH_INCREASING_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq x \)
   - \( sv \mod 2 = 0 \land sv \geq 3: (sv - 2, sv - 2) \quad (0, 0) \)
   - \( sv \mod 2 = 1 \land sv \geq 6: (sv - 3, sv - 3) \quad (2, 2) \)

2. \( x = sv \land sv \mod 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 3 \)

3. \( y > 0 \Rightarrow x \geq 2 \)
   - \( y \neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) \),
   - \( x < 3, \)
   - \( x > sv \cdot \min(1, \max(0, sv - 1)) - 1, \)
   - \( 1 = sv \mod 2, \)
   - \( 0 = x \mod 2 \)

4. \( x \neq 5 \lor y < 4 \)

5. \( x \neq 1 \)

6. \( x \neq 3 \lor y < 1 \)

7. \( y \neq x \lor 0 = x \mod 2 \)

8. \( x \neq sv \cdot \min(1, \max(0, sv - 1)), \)
   - \( y < 1, \)
   - \( y > (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 1, \)
   - \( 0 = y \mod 2 \)

9. \( y \neq 1 \)

10. \( y \neq 4 \lor x < 5 \)

11. \( y \neq 2 \lor x < 3 \)

12. \( y \neq x \lor 0 = x \mod 2 \)

13. \( x \neq sv \cdot \min(1, \max(0, sv - 1)), \)
   - \( y < 1, \)
   - \( y > (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 1, \)
   - \( 0 = y \mod 2 \)

14. \( y \neq 1 \)

15. \( y \neq 5 \lor x < 4 \)

16. \( y \neq 3 \lor x < 2 \)

17. \( y \neq x \lor 0 = x \mod 2 \)

18. \( x \neq sv \cdot \min(1, \max(0, sv - 1)), \)
   - \( y < 1, \)
   - \( y > (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 1, \)
   - \( 0 = y \mod 2 \)

19. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{SUM_WIDTH_INCREASING_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_INFLEXION}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \\
\quad 1. x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \\
\quad 2. sv \geq 5: (sv - 3, sv - 2) \land (sv - 3, 1) \\
\quad 3. x \neq (sv - 2) \ast \min(1, \max(0, sv - 3)) - 2 \lor y \neq 3 \lor sv < 7 \\
\quad 4. x \neq 1
\end{align*}
\]
\[\begin{align*}
\text{SUM_WIDTH_INCREASING_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_PEAK}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}\]

1. \(x > 0 \land y > 0 \implies x \leq sv - 3\)
2. \(x > 0 \land y > 0 \implies x \leq y + sv - 5\)
3. \(x \neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 2, y \neq (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 2, sv < 5\)
4. \(x \neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 2 \lor y \neq 2 \lor sv < 5\)
5. \(y < 1, y > (sv - 2) \cdot \min(1, \max(0, sv - 2)) - 1\)
6. \(x \neq 1\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM_WIDTH_INCREASING_TERRACE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
2. \(sv \geq 5: (sv - 2, 0) \ (sv - 3, 1)\)
3. \(x > 0 \Rightarrow y \leq sv - 4\)
4. \(x \neq 1\)
\( \text{SUM\_WIDTH\_INCREASING\_TERRACE}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
2. \( x > 0 \Rightarrow y \leq sv - 4 \)
3. \( y > 0 \Rightarrow x \leq sv - 4 \)
4. \( o^\circ \geq \lfloor (x+1)/2 \rfloor + \lfloor (y+1)/2 \rfloor \)
5. \( y \neq 1 \)
6. \( x \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM WIDTH INCREASING TERRACE}(x, \text{VARIABLES}) \land \text{SUM WIDTH PROPER PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
2. \( sv \geq 7: (sv - 2, 0) \text{ and } (sv - 4, 2) \)
3. \( x > 0 \Rightarrow y \leq sv - 4 \)
4. \( y > 0 \Rightarrow x \leq sv - 4 \)
5. \( y \neq 1 \)
6. \( x \neq 1 \)
\textsc{sum\_width\_increasing\_terrace}(x, \text{variables}) \land \textsc{sum\_width\_steady\_sequence}(y, \text{variables}) \text{ with } sv = |\text{variables}|

1. \(x \leq y\)
2. \(sv \geq 5\): \((sv - 2, sv - 2)\), \((sv - 3, sv - 3)\)
3. \(x > 0 \Rightarrow y \geq 2\)
4. \(x > 0 \land y > 0 \Rightarrow x + y \leq 2 \ast sv - 4\)
5. \(sv \geq 5\): \((sv - 2, sv - 2)\), \((sv - 3, sv - 1)\)
6. \(y \neq 4 \lor 0 = x \mod 2\)
7. \(y \neq 1\)
8. \(y \neq x + 1\)
9. \(x \neq 1\)
SUM_WIDTH_INCREASING_TERRACE(x, VARIABLES) ∧ SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE(y, VARIABLES) with sv = |VARIABLES|

1. \( x + y \leq sv \)
   - \( sv \geq 7 \): \((sv - 4, 4) \quad (sv - 5, 5)\)

2. \( x > 0 \Rightarrow y \leq sv - 2 \)

3. \( x > 0 \land y > 0 \Rightarrow 2x + y \leq 2 \times sv - 4 \)
   - \( sv \geq 6 \): \((sv - 3, 2) \quad (sv - 4, 4)\)

4. \( y \neq 1 \)

5. \( x \neq 1 \)
\[
\begin{align*}
\text{SUM_WIDTH_INCREASING_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE}(y, \text{VARIABLES}) & \text{ with } sv = |\text{VARIABLES}| \\
\text{1. } x &= \max(0, sv - 2) \land sv \mod 2 = 1 \land sv > 3 \Rightarrow y \leq sv - 1 \\
\text{2. } y &= 0 \Rightarrow x = 0 \\
\text{3. } x > 0 \land y > 0 \Rightarrow y \geq 4 \\
\text{4. } \forall sv \geq 5: & \ (sv - 2, 4) \ (sv - 3, 4) \\
& \left( \begin{array}{l}
{x \neq (sv - 2) \ast \min(1, \max(0, sv - 3)),} \\
{y < 3,} \\
{y > sv \ast \min(1, \max(0, sv - 1)) - 1,} \\
{0 = y \mod 2}
\end{array} \right) \\
\text{5. } y & \neq 1 \\
\text{6. } x & < 1 \lor y \neq 2 \\
\text{7. } \forall y \neq sv \ast \min(1, \max(0, sv - 1)), & \left( \begin{array}{l}
x < 1, \\
x > (sv - 2) \ast \min(1, \max(0, sv - 3)) - 1, \\
0 = x \mod 2
\end{array} \right) \\
\text{8. } x & < 1 \lor y \neq 3 \\
\text{9. } x & \neq 1
\end{align*}
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM\_WIDTH\_INCREASING\_TERRACE}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x > 0 \land y > 0 \Rightarrow x \leq sv - 3\)
2. \(x \neq (sv - 2) \ast \min(1, \max(0, sv - 3)) - 2,\)
   \(y \neq (sv - 2) \ast \min(1, \max(0, sv - 2)) - 2,\)
   \(sv < 5\)
3. \(x \neq (sv - 2) \ast \min(1, \max(0, sv - 3)) - 2 \lor y \neq 2 \lor sv < 5\)
4. \(x \neq (sv - 2) \ast \min(1, \max(0, sv - 3)) - 1,\)
   \(y < 1,\)
   \(y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 1\)
5. \(x \neq 1\)

\[
\begin{array}{c}
\text{sv = 12} \\
\text{sv = 11}
\end{array}
\]
\[ \text{SUM WIDTH INCREASING TERRACE}(x, \text{VAR}) \land \text{SUM WIDTH VALLEY}(y, \text{VAR}) \text{ with } sv = |\text{VAR}| \]

1. \( x > 0 \land y > 0 \Rightarrow 2x \leq y + 2sv - 9 \)
2. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \)
3. \( x \neq (sv - 2) \min(1, \max(0, sv - 3)) - 2, \)  
   \( y \neq (sv - 2) \min(1, \max(0, sv - 2)) - 2, \)  
   \( sv < 5 \)
4. \( x \neq (sv - 2) \min(1, \max(0, sv - 3)) - 2 \lor y \neq 2 \lor sv < 5 \)
5. \( y < 1, \)  
   \( y > (sv - 2) \min(1, \max(0, sv - 2)) - 1 \)
6. \( x \neq 1 \)}
5. A DATABASE OF PARAMETERISED INVARIENTS

**SUM_WIDTH_INCREASING_TERRACE**\((x, \text{VARIABLES}) \land \text{SUM_WIDTH_ZIGZAG}(y, \text{VARIABLES})\) with \(sv = |\text{VARIABLES}|\)

1. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
2. \(sv \geq 6: (sv - 2, 0) \quad (sv - 4, 2)\)
3. \(x > 0 \Rightarrow y \leq sv - 4\)
4. \(y > 0 \Rightarrow x \leq sv - 4\)
5. \(y \neq 1\)
6. \(x \neq 1\)

**SUM_WIDTH_INFLEXION**\((x, \text{VARIABLES}) \land \text{SUM_WIDTH_PEAK}(y, \text{VARIABLES})\) with \(sv = |\text{VARIABLES}|\)

1. \(y > 0 \Rightarrow x \geq 1\)
2. \(sv \geq 5: (1, sv - 2) \quad (1, sv - 3)\)
\begin{itemize}
\item $y \leq x$
\item $sv \geq 3$: $(sv - 2, sv - 2)$, $(sv - 3, sv - 3)$
\end{itemize}

$sv = 12$

$sv = 11$
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM\_WIDTH\_INFLEXION}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( y \leq x \)
2. \( sv \geq 5: (sv - 2, sv - 2) \quad (sv - 3, sv - 3) \)
3. \( y > 0 \Rightarrow x \geq 2 \)
4. \( y \neq 1 \)
$\text{SUM}_\text{WIDTH}_\text{INFLEXION}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{STEADY}_\text{SEQUENCE}(y, \text{VARIABLES})$ with $sv = |\text{VARIABLES}|$

1. $x > 0 \land y > 0 \Rightarrow x + y \leq 2 \cdot sv - 4$
2. $sv \geq 5: (sv - 2, sv - 2) \land (sv - 3, sv - 1)$
3. $x > 0 \land y > 0 \Rightarrow y \leq x + sv - 2$
4. $sv \geq 6: (1, sv - 1) \land (2, sv)$
5. $x \neq 1 \lor y \neq sv \cdot \min(1, \max(0, sv - 1))$
6. $y \neq 1$

---

$sv = 12$

$sv = 11$
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM WIDTH}_\text{INFLEXION}(x, \text{VARIABLES}) \land \\
\text{SUM WIDTH}_\text{STRICTLY DECREASING SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x > 0 \Rightarrow y \geq 2 \)
   \( \square sv \geq 3: (sv - 2, 2) \quad (sv - 3, 2) \)

2. \( x > 0 \land y > 0 \Rightarrow y \leq x + sv - 2 \)
   \( \square sv \mod 2 = 0 \land sv \geq 4: (1, sv - 1) \quad (2, sv) \)
   \( \square sv \mod 2 = 1 \land sv \geq 5: (1, sv - 1) \quad (2, sv) \)

3. \( x \neq 1 \lor y \neq sv \ast \min(1, \max(0, sv - 1)) \)

4. \( y \neq 1 \)
\[ \text{SUM_WIDTH_INFLEXION}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \Rightarrow y \geq 2 \)
   - \( sv \geq 3: (sv - 2, 2) \) (\( sv - 3, 2 \))
2. \( x > 0 \land y > 0 \Rightarrow y \leq x + sv - 2 \)
   - \( sv \mod 2 = 0 \land sv \geq 4: (1, sv - 1) \) (\( 2, sv \))
   - \( sv \mod 2 = 1 \land sv \geq 5: (1, sv - 1) \) (\( 2, sv \))
3. \( x \neq 1 \lor y \neq sv \ast \min(1, \max(0, sv - 1)) \)
4. \( y \neq 1 \)

\[ SV = 12 \]

\[ SV = 11 \]

\[ SV = 12 \]

\[ SV = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM}_\text{WIDTH}_{\text{INFLEXION}}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_{\text{VALLEY}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( y > 0 \Rightarrow x \geq 1 \)
2. \( sv \geq 5: (1, sv - 2) \quad (1, sv - 3) \)

\[ sv = 12 \]

\[ sv = 11 \]

\[
\text{SUM}_\text{WIDTH}_{\text{INFLEXION}}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_{\text{ZIGZAG}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( y \leq x \)
2. \( y > 0 \Rightarrow x \geq 2 \)
3. \( y \not= 1 \)

\[ sv = 12 \]

\[ sv = 11 \]
\textbf{SUM\_WIDTH\_PEAK}(x, \text{VARIABLES}) \land \\
\textbf{SUM\_WIDTH\_PLAIN}(y, \text{VARIABLES}) \text{ with } \text{sv} = |\text{VARIABLES}|

\begin{enumerate}
  \item \(\text{sv} > 1 \Rightarrow x + y \leq \text{sv} - 2\)
  \begin{enumerate}
    \item \(\text{sv} \geq 3: (\text{sv} - 2, 0) \quad (\text{sv} - 3, 1)\)
  \end{enumerate}
\end{enumerate}

\textbf{SUM\_WIDTH\_PEAK}(x, \text{VARIABLES}) \land \\
\textbf{SUM\_WIDTH\_PLATEAU}(y, \text{VARIABLES}) \text{ with } \text{sv} = |\text{VARIABLES}|

\begin{enumerate}
  \item \(y \leq x\)
    \begin{enumerate}
      \item \(\text{sv} \geq 3: (\text{sv} - 2, \text{sv} - 2) \quad (\text{sv} - 3, \text{sv} - 3)\)
    \end{enumerate}
  \item \(x > 0 \Rightarrow y \geq 1\)
    \begin{enumerate}
      \item \(\text{sv} \geq 4: (\text{sv} - 2, 1) \quad (\text{sv} - 3, 1)\)
    \end{enumerate}
\end{enumerate}
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{SUM\_WIDTH\_PEAK}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLAIN}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. sv > 1 \Rightarrow x + y \leq sv - 2
2. sv \geq 4: (sv - 2, 0) \ (sv - 4, 2)
3. y > 0 \Rightarrow x \leq sv - 4
4. y \neq 1

\[
\begin{align*}
\text{SUM\_WIDTH\_PEAK}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. y \leq x
2. sv \geq 5: (sv - 2, sv - 2) \ (sv - 3, sv - 3)
3. y > 0 \Rightarrow x \geq 2
4. y \neq 1
\[ \text{SUM\_WIDTH\_PEAK}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STEADY\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 \times sv - 4 \)
2. \( sv \geq 5: (sv - 2, sv - 2) \text{ or } (sv - 3, sv - 1) \)
3. \( x > 0 \land y > 0 \Rightarrow y \leq x + sv - 2 \)
4. \( sv \geq 6: (1, sv - 1) \text{ or } (2, sv) \)
5. \( x \neq 1 \lor y \neq sv \times \min(1, \max(0, sv - 1)) \)
6. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM_WIDTH_PEAK}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \Rightarrow y \geq 2 \)
   - \( sv \geq 3: (sv - 2, 2) \) \((sv - 3, 2)\)
2. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 \ast sv - 3 \)
   - \( sv \mod 2 = 0 \land sv \geq 4: (sv - 2, sv - 1) \) \((sv - 3, sv)\)
   - \( sv \mod 2 = 1 \land sv \geq 5: (sv - 2, sv - 1) \) \((sv - 3, sv)\)
3. \( x \neq 1 \lor y \neq 3 \)
4. \( y \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)
\[
\text{SUM\_WIDTH\_PEAK}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

\begin{enumerate}
\item \(x > 0 \Rightarrow y \geq 2\)
\item \(sv \geq 4: (sv - 2, 2) \quad (sv - 3, 2)\)
\item \(x > 0 \land y > 0 \Rightarrow x + y \leq 2 \ast sv - 3\)
\item \(sv \geq 3: (sv - 2, sv - 1) \quad (sv - 3, sv)\)
\item \(x \neq 1 \lor y \neq 3\)
\item \(y \neq 1\)
\end{enumerate}
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\begin{align*}
\text{SUM}_\text{WIDTH}_\text{PEAK}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}
\]

1. \( y \leq x \)
   \[ sv \geq 3: \ (sv - 2, sv - 2) \ (sv - 3, sv - 3) \]
   \[ y \neq (sv - 2) \ast \min(1, \max(0, sv - 2)) - 3, \]
   \[ 0 = x \mod 2, \]
   \[ 0 = sv \mod 2, \]
   \[ sv < 6 \]
   \[ y \neq (sv - 2) \ast \min(1, \max(0, sv - 2)) - 3, \]
   \[ 1 = x \mod 2, \]
   \[ 1 = sv \mod 2 \]
2. \( \lor \)
   \[ x \neq (sv - 2) \ast \min(1, \max(0, sv - 2)), \]
   \[ y < 1, \]
   \[ y > (sv - 2) \ast \min(1, \max(0, sv - 2)) - 1 \]
3. \( y \neq x - 1 \)

\( sv = 12 \)

\( sv = 11 \)
\[ \text{SUM_WIDTH_PEAK}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq 3 \cdot sv - 9 \)
2. \( x > 0 \land y > 0 \Rightarrow x + 2 \cdot y \leq 3 \cdot sv - 9 \)
3. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \)
4. \( x > 0 \land y > 0 \Rightarrow y \leq sv - 3 \)

- \( sv \geq 4: (sv - 3, sv - 3) \) \( (sv - 3, sv - 4) \)
- \( sv \geq 4: (sv - 3, sv - 3) \) \( (sv - 4, sv - 3) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM\_WIDTH\_PEAK}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq 2 \cdot x + 1 \)
   - \( sv \geq 7 \): (1, 3) (2, 5)
2. \( sv > 1 \Rightarrow 2 \cdot x + y \leq 2 \cdot sv - 3 \)
   - \( sv \geq 7 \): (sv - 3, 3) (sv - 4, 5)
3. \( y > 0 \Rightarrow x \leq sv - 3 \)
   - \( sv \geq 5 \): (sv - 3, 2) (sv - 3, 3)
4. \( y > 0 \Rightarrow x \geq 1 \)
   - \( sv \geq 5 \): (1, 2) (1, 3)
5. \( y \neq 1 \)

\[ \text{SUM\_WIDTH\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   - \( sv \geq 3 \): (sv - 2, 0) (sv - 3, 1)

\( sv = 12 \)

\[ \text{sum\_width\_peak} \]

\[ \text{sum\_width\_plain} \]

\[ \text{sum\_width\_plateau} \]

\( sv = 11 \)
\[ \sum_{\text{WIDTH}}(x, \text{VARIABLES}) \land \sum_{\text{WIDTH, PROPER}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq x \)
2. \( sv \geq 5: (sv - 2, sv - 2) \quad (sv - 3, sv - 3) \)
3. \( y = 0 \Rightarrow 2x \leq sv - 1 - (sv - 1) \text{ mod } 2 \)
4. \( x > 0 \land y > 0 \Rightarrow 2x \leq y + sv - 2 \)
5. \( sv \geq 6: (sv - 2, sv - 2) \quad (sv - 3, sv - 4) \)
6. \( y > 0 \Rightarrow x \geq 2 \)
7. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM}_\text{WIDTH}_\text{PLAIN}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{PROPER}_\text{PLATEAU}(y, \text{VARIABLES}) \] with \( sv = |\text{VARIABLES}| \)

1. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
2. \( sv \geq 4: (sv - 2, 0) \ (sv - 4, 2) \)
3. \( y > 0 \Rightarrow x \leq sv - 4 \)
4. \( y \neq 1 \)
\[ \sum_{\text{WIDTH}}(x, \text{VARIABLES}) \land \sum_{\text{WIDTH \_ STEADY \_ SEQUENCE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y = 0 \Rightarrow 2 \times x \leq sv - 1 - (sv - 1) \mod 2 \)
2. \( x > 0 \land y > 0 \Rightarrow 2 \times x \leq y + sv - 2 \)
3. \( sv \geq 6: (sv - 2, sv - 2) \quad (sv - 3, sv - 4) \)
4. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 \times sv - 4 \)
5. \( x \neq 1 \lor y \neq sv \times \min(1, \max(0, sv - 1)) \)
6. \( y \neq sv \times \min(1, \max(0, sv - 2)) - 1, \quad \begin{cases} y \neq sv \times \min(1, \max(0, sv - 1)) - 2, \\ sv < 5 \end{cases} \)

\( y \neq 1 \)
A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM}_\text{WIDTH}_{\text{PLAIN}}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_{\text{STRICTLY DECREASING SEQUENCE}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[2 \times x + y \leq 2 \times sv - 2\]
   \[sv \geq 4: (sv - 2, 2) \quad (sv - 3, 4)\]
2. \[x > 0 \Rightarrow y \geq 2\]
   \[sv \geq 3: (sv - 2, 2) \quad (sv - 3, 2)\]
3. \[y \neq 1\]
\( \text{SUM_WIDTH}_\text{PLAIN}(x, \text{VARIABLES}) \land \text{SUM_WIDTH}_\text{SUMMIT}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( sv > 1 \implies x + y \leq sv - 2 \)
2. \( sv \geq 3: (sv - 2, 0) \quad (sv - 3, 1) \)

\( sv = 12 \)

\( sv = 11 \)

---

\( \text{SUM_WIDTH}_\text{PLAIN}(x, \text{VARIABLES}) \land \text{SUM_WIDTH}_\text{VALLEY}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x \leq y \)
2. \( sv \geq 3: (sv - 2, sv - 2) \quad (sv - 3, sv - 3) \)
3. \( y > 0 \implies x \geq 1 \)
4. \( sv \geq 4: (1, sv - 2) \quad (1, sv - 3) \)

\( sv = 12 \)

\( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

SUM_WIDTHPLAIN(x, VARIABLES) \(\land\) SUM_WIDTHZIGZAG(y, VARIABLES) with \(sv = |VARIABLES|\)

1. \(y \leq 2 \times x + 1\)
   - \(sv \geq 7\): (1, 3) (2, 5)
2. \(y < \max(0, sv - 2) \land sv > 1 \Rightarrow 2 \times x + y \leq 2 \times sv - 4\)
   - \(sv \geq 6\): (sv - 2, 0) (sv - 3, 2)
3. \(sv > 1 \Rightarrow 2 \times x + y \leq 2 \times sv - 3 - (sv - 1) \mod 2\)
   - \(sv - 1) \mod 2 = 1 \land sv \geq 6\): (sv - 2, 0) (sv - 3, 2)
4. \(y > 0 \Rightarrow x \leq sv - 3\)
5. \(y > 0 \Rightarrow x \geq 1\)
   - \(sv \geq 5\): (1, 2) (1, 3)
6. \(\bigvee \begin{cases} \begin{array}{l} x \neq (sv - 2) \times \min(1, \max(0, sv - 2)) - 2, \\ y \neq (sv - 2) \times \min(1, \max(0, sv - 3)) - 1, \\ 0 = sv \mod 2, \\ sv < 6 \end{array} \end{cases} \)
7. \(y \neq 1\)
\[ \text{SUM WIDTH PLATEAU}(x, \text{VARIABLES}) \land \text{SUM WIDTH PROPER PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
2. \( sv \geq 4: (sv - 2, 0) \ (sv - 4, 2) \)
3. \( y > 0 \Rightarrow x \leq sv - 4 \)
4. \( y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{SUM WIDTH PLATEAU}(x, \text{VARIABLES}) \land \text{SUM WIDTH PROPER PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y \leq x \)
2. \( sv \geq 5: (sv - 2, sv - 2) \ (sv - 3, sv - 3) \)
3. \( y = 0 \Rightarrow 2 \times x \leq sv - 1 - (sv - 1) \mod 2 \)
4. \( x > 0 \land y > 0 \Rightarrow 2 \times x \leq y + sv - 2 \)
5. \( sv \geq 6: (sv - 2, sv - 2) \ (sv - 3, sv - 4) \)
6. \( y > 0 \Rightarrow x \geq 2 \)
7. \( y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIENTS

\[ \text{SUM}_\text{WIDTH}_\text{PLATEAU}(x, \text{VARIABLES}) \land \text{SUM}_\text{WIDTH}_\text{STEADY}_\text{SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( y = 0 \Rightarrow 2 \cdot x \leq sv - 1 - (sv - 1) \mod 2 \)
2. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x \leq y + sv - 2 \)
3. \( sv \geq 6: (sv - 2, sv - 2) \land (sv - 3, sv - 4) \)
4. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 \cdot sv - 4 \)
5. \( sv \geq 5: (sv - 2, sv - 2) \lor (sv - 3, sv - 1) \)
6. \( sv \geq 6: (1, sv - 1) \lor (2, sv) \)
7. \( x \neq 1 \lor y \neq sv \cdot \min(1, \max(0, sv - 1)) \)
8. \( y \neq \begin{cases} sv \cdot \min(1, \max(0, sv - 2)) - 1, \\ sv < 5 \end{cases} \)
9. \( y \neq 1 \)

\[ \sum \text{width}_{\text{plateau}} \]

\[ \sum \text{width}_{\text{steady sequence}} \]

\[ sv = 12 \]

\[ sv = 11 \]
\[ \text{SUM_WIDTH_PLATEAU}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[2 \times x + y \leq 2 \times sv - 2\]
   - \( sv \geq 4 \): \((sv - 2, 2)~(sv - 3, 4)\)
   - \( x > 0 \Rightarrow y \geq 2 \)
   - \( sv \geq 3 \): \((sv - 2, 2)~(sv - 3, 2)\)
   - \( y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]

\[ \text{SUM_WIDTH_PLATEAU}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[2 \times x + y \leq 2 \times sv - 2\]
   - \( sv \geq 4 \): \((sv - 2, 2)~(sv - 3, 4)\)
   - \( x > 0 \Rightarrow y \geq 2 \)
   - \( sv \geq 3 \): \((sv - 2, 2)~(sv - 3, 2)\)
   - \( y \neq 1 \)

\[ sv = 12 \]

\[ sv = 11 \]
5. A DATABASE OF PARAMETERISED INVARIANTS

\text{SUM\_WIDTH\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( x + y \leq sv - 1 \)
   \( sv \geq 5: (1, sv - 2) \quad (2, sv - 3) \)
2. \( y > 0 \Rightarrow x \geq 1 \)
   \( sv \geq 4: (1, sv - 2) \quad (1, sv - 3) \)
3. \( x \neq 1 \lor y \neq 0 \)
4. \( x \neq 2 \lor y \neq 1 \)

\text{SUM\_WIDTH\_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

1. \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
   \( sv \geq 3: (sv - 2, 0) \quad (sv - 3, 1) \)

\( sv = 12 \)          \( sv = 11 \)
\text{SUM WIDTH PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM WIDTH ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|

\begin{itemize}
\item \( y \leq 2 \times x + 1 \)
\item \( sv \geq 7: (1, 3) \quad (2, 5) \)
\item \( y < \max(0, sv - 2) \land sv > 1 \Rightarrow 2 \times x + y \leq 2 \times sv - 4 \)
\item \( sv \geq 6: (sv - 2, 0) \quad (sv - 3, 2) \)
\item \( sv > 1 \Rightarrow 2 \times x + y \leq 2 \times sv - 3 - (sv - 1) \mod 2 \)
\item \( (sv - 1) \mod 2 = 1 \land sv \geq 6: (sv - 2, 0) \quad (sv - 3, 2) \)
\item \( y > 0 \Rightarrow x \leq sv - 3 \)
\item \( y > 0 \Rightarrow x \geq 1 \)
\item \( sv \geq 5: (1, 2) \quad (1, 3) \)
\item \( \bigvee \left( \begin{array}{l}
 x \neq (sv - 2) \times \min(1, \max(0, sv - 2)) - 2, \\
 y \neq (sv - 2) \times \min(1, \max(0, sv - 3)) - 1, \\
 0 = sv \mod 2, \\
 sv < 6
\end{array} \right) \)
\item \( y \neq 1 \)
\end{itemize}
5. A DATABASE OF PARAMETERISED INVARIANTS

\[\text{SUM\_WIDTH\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_PROPER\_PLATEAU}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
2. \(sv \geq 6 : (sv - 2, 0) \quad (sv - 4, 2)\)
3. \(x > 0 \Rightarrow y \leq sv - 4\)
4. \(y > 0 \Rightarrow x \leq sv - 4\)
5. \(o^= \geq [(x + 1)/2] + [(y + 1)/2]\)
6. \(y \neq 1\)
7. \(x \neq 1\)
\textsc{sum\_width\_proper\_plain}(x, \text{VARiABLEs}) \land \textsc{sum\_width\_steady\_sequence}(y, \text{VARiABLEs}) \text{ with } sv = |\text{VARiABLEs}|

1. $x \leq y$
2. $sv \geq 5: (sv - 2, sv - 2) \land (sv - 3, sv - 3)$
3. $x > 0 \Rightarrow y \geq 2$
4. $x > 0 \land y > 0 \Rightarrow x + y \leq 2 * sv - 4$
5. $sv \geq 5: (sv - 2, sv - 2) \land (sv - 3, sv - 1)$
6. $y \neq 4 \lor 0 = x \mod 2$
7. $y \neq 1$
8. $y \neq x + 1$
9. $x \neq 1$

\[sv = 12\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{SUM_WIDTH_PROPER_PLAIN}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. $3x + 3y \leq 4sv - 2 - (4sv - 2) \mod 3$
2. $2x + y \leq 2sv - 2$
3. $sv \geq 5$: $(sv - 2, 2)$ $(sv - 3, 4)$
4. $x + 2y \leq 2sv$
5. $sv \geq 8$: $(2sv - 1)$ $(sv - 2)$
6. $x > 0 \Rightarrow y \geq 2$
7. $sv \geq 5$: $(sv - 2, 2)$ $(sv - 3, 2)$
8. $y \neq 1$
9. $x \neq 1$

\begin{figure}[h]
\centering
\begin{tabular}{ll}
\text{sv} = 12 & \text{sv} = 11 \\
\end{tabular}
\end{figure}
\[\text{SUM_WIDTH_PROPER.PLAIN}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \[2 \times x + y \leq 2 \times sv - 2\]
2. \[sv \geq 5: \ (sv - 2, 2) \quad (sv - 3, 4)\]
3. \[x + 2 \times y \leq 2 \times sv\]
4. \[sv \geq 8: \ (2, sv - 1) \quad (4, sv - 2)\]
5. \[x > 0 \Rightarrow y \geq 2\]
6. \[sv \geq 5: \ (sv - 2, 2) \quad (sv - 3, 2)\]
7. \[y \neq 1\]
8. \[x \neq 1\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{SUM}_\text{\_}\text{WIDTH}_\text{\_}\text{PROPER\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{\_}\text{WIDTH}_\text{\_}\text{SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
2. \(sv \geq 5: (sv - 2, 0) \ (sv - 3, 1)\)
3. \(x > 0 \Rightarrow y \leq sv - 4\)
4. \(x \neq 1\)

\begin{align*}
\text{SUM}_\text{\_}\text{WIDTH}_\text{\_}\text{PROPER\_PLAIN}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{\_}\text{WIDTH}_\text{\_}\text{VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. \(x \leq y\)
2. \(sv \geq 5: (sv - 2, sv - 2) \ (sv - 3, sv - 3)\)
3. \(x > 0 \Rightarrow y \geq 2\)
4. \(x \neq 1\)
\[\text{SUM\_WIDTH\_PROPER\_PLAIN}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|\]

1. \(sv > 1 \Rightarrow x + y \leq sv - 2\)
2. \(sv \geq 0: (sv - 2, 0) \quad (sv - 4, 2)\)
3. \(x > 0 \Rightarrow y \leq sv - 4\)
4. \(y > 0 \Rightarrow x \leq sv - 4\)
5. \(y \neq 1\)
6. \(x \neq 1\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{SUM_WIDTH_PROPER_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_STEADY_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. \( x \leq y \)
2. \( sv \geq 5: (sv - 2, sv - 2) (sv - 3, sv - 3) \)
3. \( x > 0 \Rightarrow y \geq 2 \)
4. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 * sv - 4 \)
5. \( y \neq 4 \lor 0 = x \mod 2 \)
6. \( y \neq 1 \)
7. \( y \neq x + 1 \)
8. \( x \neq 1 \)
\[
\text{SUM_WIDTH_PROPER_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(2 \times x + y \leq 2 \times sv - 2\)
   - sv \geq 5: \((sv - 2, 2)\) \((sv - 3, 4)\)
2. \(x + 2 \times y \leq 2 \times sv\)
   - sv \geq 8: \((2, sv - 1)\) \((4, sv - 2)\)
3. \(x > 0 \Rightarrow y \geq 2\)
   - sv \geq 5: \((sv - 2, 2)\) \((sv - 3, 2)\)
4. \(y \neq 1\)
5. \(x \neq 1\)
5. A DATABASE OF PARAMETERISED INVARIANTS

\begin{align*}
\text{SUM_WIDTH_PROPER_PLATEAU}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}

1. \( 3 \cdot x + 3 \cdot y \leq 4 \cdot sv - 2 - (4 \cdot sv - 2) \mod 3 \)
2. \( 2 \cdot x + y \leq 2 \cdot sv - 2 \)
   - \( sv \geq 5: (sv - 2, 2), (sv - 3, 4) \)
3. \( x + 2 \cdot y \leq 2 \cdot sv \)
   - \( sv \geq 8: (2, sv - 1), (4, sv - 2) \)
4. \( x > 0 \Rightarrow y \geq 2 \)
   - \( sv \geq 5: (sv - 2, 2), (sv - 3, 2) \)
5. \( y \neq 1 \)
6. \( x \neq 1 \)

\begin{align*}
sv = 12
\end{align*}

\begin{align*}
sv = 11
\end{align*}
SUM_WIDTH_PROPER_PLATEAU(x, VARIABLES) \land \\
SUM_WIDTH_SUMMIT(y, VARIABLES) with sv = \|VARIABLES\|

1. sv > 1 \Rightarrow x + y \leq sv - 2
2. sv \geq 3: (sv - 2, 0) \land (0, sv - 2)
3. x > 0 \land y > 0 \Rightarrow x + y \leq sv - 3
4. sv \geq 7: (sv - 4, 1) \land (sv - 5, 2)
5. x \neq (sv - 2) * min(1, max(0, sv - 3)) - 4 \lor y \neq 4 \lor sv < 7
6. x < 4 \lor y \neq (sv - 2) * min(1, max(0, sv - 2)) - 4 \lor sv < 7
7. \begin{cases} \begin{align*}
x &\neq (sv - 2) * min(1, max(0, sv - 3)) - 3, \\
y &< 3, \\
y &> (sv - 2) * min(1, max(0, sv - 2)) - 1, \\
0 & = y \mod 2 \\
x &\neq (sv - 2) * min(1, max(0, sv - 3)) - 1, \\
y &< 1, \\
y &> (sv - 2) * min(1, max(0, sv - 2)) - 2, \\
0 & = y \mod 2 \\
x &\neq (sv - 2) * min(1, max(0, sv - 3)) - 2, \\
y &< 2, \\
y &> (sv - 2) * min(1, max(0, sv - 2)) - 2, \\
1 & = y \mod 2 \\
y &\neq (sv - 2) * min(1, max(0, sv - 2)) - 3, \\
x &< 1, \\
x &> (sv - 2) * min(1, max(0, sv - 3)) - 1, \\
0 & = x \mod 2 \\
y &\neq (sv - 2) * min(1, max(0, sv - 2)) - 2, \\
x &< 1, \\
x &> (sv - 2) * min(1, max(0, sv - 3)) - 1 \\
\end{align*} \end{cases}
8. 10 \begin{cases} x = 1 \end{cases}

sv = 12

sv = 11
5. A DATABASE OF PARAMETERISED INVARIANTS

\[ \text{SUM\_WIDTH\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\begin{enumerate}
\item \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
\item \( sv \geq 5: (sv - 2, 0) \quad (sv - 3, 1) \)
\item \( x > 0 \Rightarrow y \leq sv - 4 \)
\item \( x \neq 1 \)
\end{enumerate}

\[ \text{SUM\_WIDTH\_PROPER\_PLATEAU}(x, \text{VARIABLES}) \land \text{SUM\_WIDTH\_ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

\begin{enumerate}
\item \( sv > 1 \Rightarrow x + y \leq sv - 2 \)
\item \( sv \geq 6: (sv - 2, 0) \quad (sv - 4, 2) \)
\item \( x > 0 \Rightarrow y \leq sv - 4 \)
\item \( y > 0 \Rightarrow x \leq sv - 4 \)
\item \( y \neq 1 \)
\item \( x \neq 1 \)
\end{enumerate}
\( \text{SUM_WIDTH_STEADY_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_STRICTLY_DECENDING_SEQUENCE}(y, \text{VARIABLES}) \) with \( sv = |\text{VARIABLES}| \)

1. \( x + y \leq 2 \cdot sv - 2 \)
2. \( sv \mod 2 = 0 \land sv \geq 2: (sv, sv - 2) \quad (sv - 2, sv) \)
3. \( x = sv \land sv \mod 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 3 \)
4. \( x = sv - 2 \land sv \mod 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 1 \)
5. \( x \neq sv \cdot \min(1, \max(0, sv - 1)) - 1, \quad 1 = sv \mod 2 \)
6. \( y \neq sv \cdot \min(1, \max(0, sv - 1)) - 1, \quad y < 1, \quad y > sv \cdot \min(1, \max(0, sv - 1)) - 3, \quad 0 = y \mod 2 \)
7. \( x < 1, \quad x > sv \cdot \min(1, \max(0, sv - 1)) - 3, \quad 0 = x \mod 2 \)
8. \( y \neq 1 \)
9. \( x \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)
A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM}_{\text{WIDTH}} \cdot \text{STEADY \_ SEQUENCE}\left(x, \text{VARIABLES}\right) \land \\
\text{SUM}_{\text{WIDTH}} \cdot \text{STRICTLY \_ INCREASING \_ SEQUENCE}\left(y, \text{VARIABLES}\right) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x + y \leq 2 \cdot sv - 2\)
   - \(sv \mod 2 = 0 \land sv \geq 2: (sv, sv - 2) \land (sv - 2, sv)\)

2. \(x = sv \land sv \mod 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 3\)

3. \(x = sv - 2 \land sv \mod 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 1\)
   - \(x \neq sv \ast \min(1, \max(0, sv - 1)) - 1,\)
   - \(y \neq sv \ast \min(1, \max(0, sv - 1)) - 1, \land y < 1,\)
   - \(y > sv \ast \min(1, \max(0, sv - 1)) - 3, \land 0 = y \mod 2,\)
   - \(x < 1,\)
   - \(x > sv \ast \min(1, \max(0, sv - 1)) - 3, \land 0 = x \mod 2\)

4. \(y \neq 1\)

5. \(sv = 12\)

6. \(sv = 11\)
\[
\begin{align*}
&\text{SUM_WIDTH_STEADY_SEQUENCE}(x, \text{VARIABLES}) \land \\
&\text{SUM_WIDTH_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \\
&1 \quad x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq 3 \cdot sv - 6 \\
&2 \quad x > 0 \land y > 0 \Rightarrow x + y \leq 2 \cdot sv - 5 \\
&\quad \square sv \geq 7: (sv - 1, sv - 4) \quad (sv - 2, sv - 3) \\
&3 \quad x > 0 \land y > 0 \Rightarrow x \leq sv - 1 \\
&\quad \square sv \geq 8: (sv - 1, sv - 4) \quad (sv - 1, sv - 5) \\
&4 \quad x \neq sv \cdot \min(1, \max(0, sv - 1)) - 1 \lor y \neq 2 \\
&5 \quad x \neq 1
\end{align*}
\]
5. A DATABASE OF PARAMETERISED INVARIANTS

\[\begin{align*}
\text{SUM}_\text{WIDTH}_\text{STEADY}_\text{SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\end{align*}\]

1. \(x > 0 \land y > 0 \Rightarrow x + y \leq 2 * sv - 4\)
   
   \(sv \geq 6: (sv, sv - 4) \quad (sv - 1, sv - 3)\)

2. \(x > 0 \land y > 0 \Rightarrow x \leq y + sv - 2\)
   
   \(sv \geq 6: (sv, 2) \quad (sv - 1, 1)\)

3. \(x \neq sv \ast \min(1, \max(0, sv - 1)) \lor y \neq 1\)

4. \(x \neq 1\)

sv = 12

sv = 11
\[ \sum_{\text{width steady sequence}}(x, \text{VARIABLES}) \land \sum_{\text{width zigzag}}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x + y \leq sv \)
2. \( sv \geq 7: \ (sv - 2, 2) \ (sv - 3, 3) \)
3. \( x > 0 \land y > 0 \Rightarrow x + 2 \times y \leq 2 \times sv - 4 \)
4. \( sv \geq 6: \ (2, sv - 3) \ (4, sv - 4) \)
5. \( y > 0 \Rightarrow x \leq sv - 2 \)
6. \( x \neq 1 \)
7. \( y \neq 1 \)

\( sv = 12 \)

\( sv = 11 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( x + y \leq 2 \ast sv - 2 \)
   - sv mod 2 = 0 \land sv \geq 2: (sv, sv - 2) \land (sv - 2, sv)

2. \( x = sv \land sv \mod 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 3 \)

3. \( x = sv - 2 \land sv \mod 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 1 \)
   - \( x \neq sv \ast \min(1, \max(0, sv - 1)) - 1, \)
   - \( y \neq sv \ast \min(1, \max(0, sv - 1)) - 1, \)
   - \( 1 = sv \mod 2, \)
   - \( y < 1, \)
   - \( y > sv \ast \min(1, \max(0, sv - 1)) - 3, \)
   - \( 0 = y \mod 2, \)
   - \( y \neq sv \ast \min(1, \max(0, sv - 1)), \)
   - \( x < 1, \)
   - \( x > sv \ast \min(1, \max(0, sv - 1)) - 3, \)
   - \( 0 = x \mod 2, \)

4. \( y \neq 1 \)

5. \( x \neq 1 \)
\[ \text{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_SUMMIT}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \(x > 0 \land y > 0 \Rightarrow x + y \leq 2 \ast sv - 3\)
2. \(sv \mod 2 = 0 \land sv \geq 4: (sv, sv - 3) \land (sv - 1, sv - 2)\)
3. \(sv \mod 2 = 1 \land sv \geq 5: (sv, sv - 3) \land (sv - 1, sv - 2)\)
4. \(y > 0 \Rightarrow x \geq 2\)
5. \(sv \geq 3: (2, sv - 2) \land (2, sv - 3)\)
6. \(x \neq 3 \lor y \neq 1\)
7. \(x \neq 1\)

---

**sv = 12**

**sv = 11**
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}(x, VARIABLES) \land \\
\text{SUM\_WIDTH\_VALLEY}(y, VARIABLES) \text{ with } sv = |VARIABLES|
\]

1. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2 \cdot sv - 3 \)
2. \( sv \geq 3: \ (sv, sv - 3), \ (sv - 1, sv - 2) \)
3. \( y > 0 \Rightarrow x \geq 2 \)
4. \( sv \geq 4: \ (2, sv - 2), \ (2, sv - 3) \)
5. \( x \neq 3 \lor y \neq 1 \)
6. \( x \neq 1 \)

\[ \text{sv} = 12 \]

\[ \text{sv} = 11 \]
\texttt{SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE}(x, VARIABLES) \land \texttt{SUM_WIDTH_ZIGZAG}(y, VARIABLES) \text{ with } sv = |VARIABLES|

1. \( y \leq x \)
2. \( \text{sv mod 2} = 0 \land \text{sv} \geq 3: (\text{sv} - 2, \text{sv} - 2) \) (0, 0)
3. \( \text{sv mod 2} = 1 \land \text{sv} \geq 6: (\text{sv} - 3, \text{sv} - 3) \) (2, 2)
4. \( x = \text{sv} \land \text{sv mod 2} = 1 \land \text{sv} > 1 \Rightarrow y \leq \text{sv} - 3 \)
5. \( y > 0 \Rightarrow x \geq 2 \)
   \( y \neq (\text{sv} - 2) \ast \min(1, \max(0, \text{sv} - 3)), \)
   \( x < 3, \)
   \( x > \text{sv} \ast \min(1, \max(0, \text{sv} - 1)) - 1, \)
   \( 1 = \text{sv mod 2}, \)
   \( 0 = x \mod 2 \)
6. \( x \neq 5 \lor y < 4 \)
7. \( x \neq 1 \)
8. \( x \neq 3 \lor y < 1 \)
9. \( y \neq x \lor 0 = x \mod 2 \)
   \( x \neq \text{sv} \ast \min(1, \max(0, \text{sv} - 1)), \)
   \( y < 1, \)
   \( y > (\text{sv} - 2) \ast \min(1, \max(0, \text{sv} - 3)) - 1, \)
   \( 0 = y \mod 2 \)
10. \( y \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM_WIDTH.Strictly.Increasing.Sequence}(x, \text{VARIABLES}) \land \\
\text{SUM_WIDTH.Summit}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \(x > 0 \land y > 0 \Rightarrow x + y \leq 2 \times sv - 3\)
   - \(sv \geq 3\): \((sv, sv - 3)\) \((sv - 1, sv - 2)\)

2. \(y > 0 \Rightarrow x \geq 2\)
   - \(sv \geq 5\): \((2, sv - 2)\) \((2, sv - 3)\)

3. \(x \neq 3 \lor y \neq 1\)
4. \(x \neq 1\)
\[ \text{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(x, \text{VARs}) \land \text{SUM\_WIDTH\_VALLEY}(y, \text{VARs}) \text{ with } sv = |\text{VARs}| \]

1. \( x > 0 \land y > 0 \Rightarrow x + y \leq 2sv - 3 \)
2. \( sv \mod 2 = 0 \land sv \geq 4 : (sv, sv - 3) \text{ or } (sv - 1, sv - 2) \)
3. \( y > 0 \Rightarrow x \geq 2 \)
4. \( sv \geq 3 : (2, sv - 2) \text{ or } (2, sv - 3) \)
5. \( x \neq 3 \lor y \neq 1 \)
6. \( x \neq 1 \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\textbf{SUM\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}(x, VARIABLES) \land
\textbf{SUM\_WIDTH\_ZIGZAG}(y, VARIABLES) with sv = |VARIABLES|

1. \( y \leq x \)
   - sv mod 2 = 0 \land sv \geq 3: (sv - 2, sv - 2) (0, 0)
   - sv mod 2 = 1 \land sv \geq 6: (sv - 3, sv - 3) (2, 2)

2. \( x = sv \land sv \mod 2 = 1 \land sv > 1 \Rightarrow y \leq sv - 3 \)

3. \( y > 0 \Rightarrow x \geq 2 \)
   - \( y \neq (sv - 2) \cdot \min(1, \max(0, sv - 3)) \)
   - \( x < 3 \)
   - \( x > sv \cdot \min(1, \max(0, sv - 1)) - 1 \)
   - \( 1 = sv \mod 2 \)
   - \( 0 = x \mod 2 \)

4. \( x \neq 5 \lor y < 4 \)

5. \( x \neq 1 \)

6. \( x \neq 3 \lor y < 1 \)

7. \( y \neq x \lor 0 = x \mod 2 \)

8. \( x \neq sv \cdot \min(1, \max(0, sv - 1)) \)
   - \( y < 1 \)
   - \( y > (sv - 2) \cdot \min(1, \max(0, sv - 3)) - 1 \)
   - \( 0 = y \mod 2 \)

9. \( y \neq 1 \)

Diagram:

- For \( sv = 12 \)
- For \( sv = 11 \)
\[ \text{SUM_WIDTH_SUMMIT}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_VALLEY}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \( x > 0 \land y > 0 \Rightarrow 2 \cdot x + y \leq 3 \cdot sv - 9 \)
2. \( x > 0 \land y > 0 \Rightarrow x + 2 \cdot y \leq 3 \cdot sv - 9 \)
3. \( x > 0 \land y > 0 \Rightarrow x \leq sv - 3 \)
   - \( sv \geq 4: (sv - 3, sv - 3) \quad (sv - 3, sv - 4) \)
4. \( x > 0 \land y > 0 \Rightarrow y \leq sv - 3 \)
   - \( sv \geq 4: (sv - 3, sv - 3) \quad (sv - 4, sv - 3) \)
5. A DATABASE OF PARAMETERISED INVARIANTS

\[
\text{SUM}_\text{WIDTH}_\text{SUMMIT}(x, \text{VARIABLES}) \land \\
\text{SUM}_\text{WIDTH}_\text{ZIGZAG}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}|
\]

1. \( y \leq 3 \times x \)
2. \( y = \max(0, sv - 2) \land sv > 1 \Rightarrow 2 \times x \geq sv - 2 - sv \mod 2 \)
3. \( sv > 1 \Rightarrow 2 \times x + y \leq 2 \times sv - 3 \)
   - \( sv \geq 7 \): \((sv - 3, 3)\) \((sv - 4, 5)\)
4. \( y > 0 \Rightarrow x \leq sv - 3 \)
   - \( sv \geq 5 \): \((sv - 3, 2)\) \((sv - 3, 3)\)
5. \( y \neq 1 \)
\[ \text{SUM_WIDTH_Valley}(x, \text{VARIABLES}) \land \text{SUM_WIDTH_Zigzag}(y, \text{VARIABLES}) \text{ with } sv = |\text{VARIABLES}| \]

1. \[ y \leq 2 \cdot x + 1 \]
2. \[ sv \geq 7: (1, 3) \ (2, 5) \]
3. \[ sv > 1 \Rightarrow 2 \cdot x + y \leq 2 \cdot sv - 3 \]
4. \[ sv \geq 7: (sv - 3, 3) \ (sv - 4, 5) \]
5. \[ y > 0 \Rightarrow x \leq sv - 3 \]
6. \[ sv \geq 5: (sv - 3, 2) \ (sv - 3, 3) \]
7. \[ y > 0 \Rightarrow x \geq 1 \]
8. \[ sv \geq 5: (1, 2) \ (1, 3) \]
9. \[ y \neq 1 \]
5. A DATABASE OF PARAMETERISED INVARIANTS
Appendix A

Electronic Constraint Catalogue

The file attached to each constraint is available from a link located on the leftmost upper corner of the first page of each constraint provided you are using Adobe Acrobat Reader.

- The file `eval.pl`, containing all utilities, is available from this link.
- The file `src.pl`, that allows one to download all time-series constraints in SIC-Stus, is available from the link.
- The MiniZinc implementation of the time-series constraints is available from this link. It corresponds to a reformulation of the corresponding automata constraints.
- Besides the Prolog checkers available in the corresponding attached Prolog files, we also have Java and C synthesised code for the checkers from the following links:
  - `names.h, names.c, checker.h, checker.c`.
  - `TimeSeriesNames.java, TimeSeriesChecker.java`.
- Finally, the machine-readable version `invariants.pl` of the database of parameterised invariants described in Chapter 5 is available from the following link.

3699
Bibliography


Index

Page numbers in bold face (as in 160) point to a definition of a constraint, keyword, restriction or system. Page numbers in bold-italic face (as in 160) notify an abbreviation of a constraint name. Finally, page numbers in serif face (as in 160) indicate an occurrence of constraint name, keyword, system or author name.

A

ALL_EQUAL_HEIGHT_DECREASING_TERRACE, 158
ALL_EQUAL_HEIGHT_INCREASING_TERRACE, 162
ALL_EQUAL_HEIGHT_PLAIN, 166
ALL_EQUAL_HEIGHT_PLATEAU, 170
ALL_EQUAL_HEIGHT_PROPER_PLAIN, 174
ALL_EQUAL_HEIGHT_PROPER_PLATEAU, 178
ALL_EQUAL_HEIGHT_STEADY, 182
ALL_EQUAL_HEIGHT_STEADY_SEQUENCE, 186
ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE, 190
ALL_EQUAL_MAX_DECREASING, 194
ALL_EQUAL_MAX_DECREASING_SEQUENCE, 198
ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE, 202
ALL_EQUAL_MAX_INCREASING, 206
ALL_EQUAL_MAX_INCREASING_SEQUENCE, 210
ALL_EQUAL_MAX_INFLEXION, 214
ALL_EQUAL_MAX_PEAK, 218
ALL_EQUAL_MAX STRICTLY DECREASING_SEQUENCE, 222
ALL_EQUAL_MAX STRICTLY INCREASING_SEQUENCE, 226
ALL_EQUAL_MAX_SUMMIT, 230
ALL_EQUAL_MAX_ZIGZAG, 234
ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE, 238
ALL_EQUAL_MIN_DECREASING, 242
ALL_EQUAL_MIN_DECREASING_SEQUENCE, 246
ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE, 250
ALL_EQUAL_MIN_GORGE, 254
ALL_EQUAL_MIN_INCREASING, 258
ALL_EQUAL_MIN_INCREASING_SEQUENCE, 262
ALL_EQUAL_MIN_INFLEXION, 266
ALL_EQUAL_MIN STRICTLY DECREASING_SEQUENCE, 270
ALL_EQUAL_MIN STRICTLY INCREASING_SEQUENCE, 274

3703
INDEX

ALL_EQUAL_MIN_VALLEY, 278
ALL_EQUAL_MIN_ZIGZAG, 282
ALL_EQUAL_RANGE_DECREASING, 286
ALL_EQUAL_RANGE_DECREASING_SEQUENCE, 290
ALL_EQUAL_RANGE_INCREASING, 294
ALL_EQUAL_RANGE_INCREASING_SEQUENCE, 298
ALL_EQUAL_RANGE.Strictly_DECREASING_SEQUENCE, 302
ALL_EQUAL_RANGE.Strictly_INCREASING_SEQUENCE, 306
ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE, 310
ALL_EQUAL_SURF_DECREASING, 314
ALL_EQUAL_SURF_DECREASING_SEQUENCE, 318
ALL_EQUAL_SURF_DECREASING_TERRACE, 322
ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE, 326
ALL_EQUAL_SURF_GORGE, 330
ALL_EQUAL_SURF_INCREASING, 334
ALL_EQUAL_SURF_INCREASING_SEQUENCE, 338
ALL_EQUAL_SURF_INCREASING_TERRACE, 342
ALL_EQUAL_SURF_INFLEXION, 346
ALL_EQUAL_SURF_PEAK, 350
ALL_EQUAL_SURF_PLAIN, 354
ALL_EQUAL_SURF_PLATEAU, 358
ALL_EQUAL_SURF_PROPER_PLAIN, 362
ALL_EQUAL_SURF_PROPER_PLATEAU, 366
ALL_EQUAL_SURF_STEADY, 370
ALL_EQUAL_SURF_STEADY_SEQUENCE, 374
ALL_EQUAL_SURF_STRICTLY_DECREASING_SEQUENCE, 378
ALL_EQUAL_SURF_STRICTLY_INCREASING_SEQUENCE, 382
ALL_EQUAL_SURF_SUMMIT, 386
ALL_EQUAL_SURF_VALLEY, 390
ALL_EQUAL_SURF_ZIGZAG, 394
ALL_EQUAL_WIDTH_DECREASING_SEQUENCE, 398
ALL_EQUAL_WIDTH_DECREASING_TERRACE, 402
ALL_EQUAL_WIDTH_GORGE, 406
ALL_EQUAL_WIDTH_INCREASING_SEQUENCE, 410
ALL_EQUAL_WIDTH_INCREASING_TERRACE, 414
ALL_EQUAL_WIDTH_INFLEXION, 418
ALL_EQUAL_WIDTH_PEAK, 422
ALL_EQUAL_WIDTH_PLAIN, 426
ALL_EQUAL_WIDTH_PLATEAU, 430
ALL_EQUAL_WIDTH_PROPER_PLAIN, 434
ALL_EQUAL_WIDTH_PROPER_PLATEAU, 438
ALL_EQUAL_WIDTH_STEADY_SEQUENCE, 442
ALL_EQUAL_WIDTH.Strictly_DECREASING_SEQUENCE, 446
ALL_EQUAL_WIDTH.Strictly_INCREASING_SEQUENCE, 450
ALL_EQUAL_WIDTH_SUMMIT, 454
ALL_EQUAL_WIDTH_VALLEY, 458
ALL_EQUAL_WIDTH_ZIGZAG, 462

AMONG, 7, 1408, 1416, 1420, 1424, 1428, 1436, 1440, 1444, 1448, 1452, 1456, 1460, 1464, 1468, 1472, 1476, 1480, 1484, 1488, 1492, 1782, 1790, 1794, 1798, 1802, 1810, 1814, 1818, 1822, 1826, 1830, 1834, 1838, 1842, 1846, 1850, 1854, 1858, 1862,
DECREASING_HEIGHT_DECreasIng_TERRACE, 466
DECREASING_HEIGHT_INCREASING_TERRACE, 470
DECREASING_HEIGHT.PLAIN, 474
DECREASING_HEIGHT_PLATEAU, 478
DECREASING_HEIGHT_PROPER.PLAIN, 482
DECREASING_HEIGHT_PROPER_PLATEAU, 486
DECREASING_HEIGHT_STeadY, 490
DECREASING_HEIGHT_STeady_SEQUENCE, 494
DECREASING_MAX_BUMP_ON_DECreasIng_SEQUENCE, 498
DECREASING_MAX_DECreasIng, 502
DECREASING_MAX_DECreasIng_SEQUENCE, 506
DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE, 510
DECREASING_MAX_INCREASING, 514
DECREASING_MAX_INCREASING_SEQUENCE, 518
DECREASING_MAX_INflexION, 522
DECREASING_MAX_Peak, 526
DECREASING_MAX_strictLY_DECreasIng_SEQUENCE, 530
DECREASING_MAX_strictLY_InCREASING_SEQUENCE, 534
DECREASING_MAX_sumMIT, 538
DECREASING_MAX_ZIGZAG, 542
DECREASING_MIN_BUMP_ON_DECreasIng_SEQUENCE, 546
DECREASING_MIN_DECreasIng, 550
DECREASING_MIN_DECreasIng_SEQUENCE, 554
DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE, 558
DECREASING_MIN_GORge, 562
DECREASING_MIN_INCREASING, 566
DECREASING_MIN_INCREASING_SEQUENCE, 570
DECREASING_MIN_INflexION, 574
DECREASING_MIN_strictLY_DECreasIng_SEQUENCE, 578
DECREASING_MIN_strictLY_InCREASING_SEQUENCE, 582
DECREASING_MIN_valLEY, 586
DECREASING_MIN_ZIGZAG, 590
DECREASING_RANGE_DECreasIng, 594
DECREASING_RANGE_DECreasIng_SEQUENCE, 598
DECREASING_RANGE_INCREASING, 602
DECREASING_RANGE_INCREASING_SEQUENCE, 606
DECREASING_RANGE_strictLY_DECreasIng_SEQUENCE, 610
DECREASING_RANGE_strictLY_InCREASING_SEQUENCE, 614
DECREASING_SURF_BUMP_ON_DECreasIng_SEQUENCE, 618
DECREASING_SURF_DECREASING, 622
DECREASING_SURF_DECREASING_SEQUENCE, 626
DECREASING_SURF_DECREASING_TERRACE, 630
DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE, 634
DECREASING_SURF_GORGE, 638
DECREASING_SURF_INCREASING, 642
DECREASING_SURF_INCREASING_SEQUENCE, 646
DECREASING_SURF_INCREASING_TERRACE, 650
DECREASING_SURF_INFLEXION, 654
DECREASING_SURF_PEAK, 658
DECREASING_SURF.PLAIN, 662
DECREASING_SURF_PLATEAU, 666
DECREASING_SURF_PROPER.PLAIN, 670
DECREASING_SURF_PROPER_PLATEAU, 674
DECREASING_SURF_STEADY, 678
DECREASING_SURF_STEADY_SEQUENCE, 682
DECREASING_SURF.Strictly_DECREASING_SEQUENCE, 686
DECREASING_SURF.Strictly_INCREASING_SEQUENCE, 690
DECREASING_SURF_SUMMIT, 694
DECREASING_SURF_Valley, 698
DECREASING_SURF_ZIGZAG, 702
DECREASING_WIDTH_DECREASING_SEQUENCE, 706
DECREASING_WIDTH_DECREASING_TERRACE, 710
DECREASING_WIDTH_GORGE, 714
DECREASING_WIDTH_INCREASING_SEQUENCE, 718
DECREASING_WIDTH_INCREASING_TERRACE, 722
DECREASING_WIDTHINFLEXION, 726
DECREASING_WIDTH.Peak, 730
DECREASING_WIDTH.PLAIN, 734
DECREASING_WIDTH_PLATEAU, 738
DECREASING_WIDTH.PROPER.PLAIN, 742
DECREASING_WIDTH.PROPER_PLATEAU, 746
DECREASING_WIDTH.STEADY_SEQUENCE, 750
DECREASING_WIDTH.Strictly_DECREASING_SEQUENCE, 754
DECREASING_WIDTH.Strictly_INCREASING_SEQUENCE, 758
DECREASING_WIDTH_SUMMIT, 762
DECREASING_WIDTH_Valley, 766
DECREASING_WIDTH.ZIGZAG, 770
INDEX

DISTINCT, 4

F

first, 5

H

HEIGHT_DECREASING_TERRACE, 774
HEIGHT_INCREASING_TERRACE, 778
HEIGHT_PLAIN, 782
HEIGHT_PLATEAU, 786
HEIGHT_PROPER_PLAIN, 790
HEIGHT_PROPER_PLATEAU, 794
HEIGHT_STEADY, 798
HEIGHT_STEADY_SEQUENCE, 802

I

in_attr, 3
in_list, 3
INCREASING_HEIGHT_DECREASING_TERRACE, 806
INCREASING_HEIGHT_INCREASING_TERRACE, 810
INCREASING_HEIGHT_PLAIN, 814
INCREASING_HEIGHT_PLATEAU, 818
INCREASING_HEIGHT_PROPER_PLAIN, 822
INCREASING_HEIGHT_PROPER_PLATEAU, 826
INCREASING_HEIGHT_STEADY, 830
INCREASING_HEIGHT_STEADY_SEQUENCE, 834
INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE, 838
INCREASING_MAX_DECREASING, 842
INCREASING_MAX_DECREASING_SEQUENCE, 846
INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE, 850
INCREASING_MAX_INCREASING, 854
INCREASING_MAX_INCREASING_SEQUENCE, 858
INCREASING_MAX_INFLEXION, 862
INCREASING_MAX_MAX_INCREASING, 866
INCREASING_MAX_MAX_INCREASINGSEQUENCE, 870
INCREASING_MAX_MAX_INCREASINGSEQUENCE, 874
INCREASING_MAX_MAX_INCREASINGSEQUENCE, 878
INCREASING_MAX_MAX_INCREASINGSEQUENCE, 882
INCREASING_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE, 886
INCREASING_MIN_DECREASING, 890
INDEX

INCREASING_WIDTH_STEADY_SEQUENCE, 1090
INCREASING_WIDTH STRICTLY_DECREASING_SEQUENCE, 1094
INCREASING_WIDTH STRICTLY_INCREASING_SEQUENCE, 1098
INCREASING_WIDTH_SUMMIT, 1102
INCREASING_WIDTH VALLEY, 1106
INCREASING_WIDTH_ZIGZAG, 1110
INDEX_BUMP_ON_DECREASING_SEQUENCE, 1114
INDEX_DECREASING, 1118
INDEX_DECREASING_SEQUENCE, 1122
INDEX_DECREASING_TERRACE, 1126
INDEX_DIP_ON_INCREASING_SEQUENCE, 1130
INDEX_GORGE, 1134
INDEX_INCREASING, 1138
INDEX_INCREASING_SEQUENCE, 1142
INDEX_INCREASING_TERRACE, 1146
INDEX_INFLEXION, 1150
INDEX_PEAK, 1154
INDEX_PLAIN, 1158
INDEX_PLATEAU, 1162
INDEX_PROPER_PLAIN, 1166
INDEX_PROPER_PLATEAU, 1170
INDEX_STEADY, 1174
INDEX_STEADY_SEQUENCE, 1178
INDEX_STRICTLY_DECREASING_SEQUENCE, 1182
INDEX_STRICTLY_INCREASING_SEQUENCE, 1186
INDEX_SUMMIT, 1190
INDEX_VALLEY, 1194
INDEX_ZIGZAG, 1198

L

last, 5

M

MAX_BUMP_ON_DECREASING_SEQUENCE, 1202
MAX_DECREASING, 1206
MAX_DECREASING_SEQUENCE, 1210
MAX_DIP_ON_INCREASING_SEQUENCE, 1214
MAX_HEIGHT_Decreasing_TERRACE, 1218, 2092
MAX_HEIGHT_INCREASING_TERRACE, 1222, 2096
MAX_HEIGHT_PLAIN, 1226, 2100
MAX_HEIGHT_PLATEAU, 1230, 2104
MAX_HEIGHT_PROPER_PLAIN, 1234, 2108
MAX_HEIGHT_PROPER_PLATEAU, 1238, 2112
INDEX

MAX_HEIGHT_STEADY, 1242, 2116
MAX_HEIGHT_STEADY_SEQUENCE, 1246, 2120
MAX_INCREASING, 1250
MAX_INCREASING_SEQUENCE, 1254
MAX_INFLEXION, 1258
MAX_MAX_BUMP_ON_DECREASING_SEQUENCE, 1262, 2124, 3206
MAX_MAX_DECREASING, 1266, 2128, 3206
MAX_MAX_DECREASING_SEQUENCE, 1270, 2132, 3206
MAX_MAX_DIP_ON_INCREASING_SEQUENCE, 1274, 2136, 3206
MAX_MAX_INCREASING, 1278, 2140, 3206
MAX_MAX_INCREASING_SEQUENCE, 1282, 2144, 3206
MAX_MAX_INFLEXION, 1286, 2148, 3206
MAX_MAX_PEAK, ii, 1290, 2152, 3206
MAX_MAX.Strictly_DECREASING_SEQUENCE, 1294, 2156, 3206
MAX_MAX.Strictly_INCREMENTING_SEQUENCE, 1298, 2160, 3206
MAX_MAX_SUMMIT, 1302, 2164, 3206
MAX_MAX_ZIGZAG, 1306, 2168, 3206
MAX_MIN_BUMP_ON_DECREASING_SEQUENCE, 1314, 2172, 3206
MAX_MIN_DECREASING, 1318, 2176, 3207
MAX_MIN_DECREASING_SEQUENCE, 1322, 2180, 3207
MAX_MIN_DIP_ON_INCREASING_SEQUENCE, 1326, 2184, 3207
MAX_MIN_GORGE, 1330, 2188, 3207
MAX_MIN_INCREMENT, 1336, 2192, 3207
MAX_MIN_INCREMENT_SEQUENCE, 1340, 2196, 3207
MAX_MIN_INFLEXION, 1344, 2200, 3207
MAX_MIN.Strictly_DECREASING_SEQUENCE, 1348, 2204, 3207
MAX_MIN.Strictly_INCREMENTING_SEQUENCE, 1352, 2208, 3207
MAX_MIN[valley, 1356, 2212, 3207
MAX_MIN.ZIGZAG, 1360, 2216, 3207
MAX_PEAK, 15, 1368
MAX_RANGE_DECREASING, ii, 1372, 3207
MAX_RANGE_DECREASING_SEQUENCE, 1376, 3208
MAX_RANGE_INCREMENT, ii, 1380, 3208
MAX_RANGE_INCREMENT_SEQUENCE, 1384, 3208
MAX_RANGE.Strictly_DECREASING_SEQUENCE, 1388, 3208
MAX_RANGE.Strictly_INCREMENTING_SEQUENCE, 1392, 3208
MAX.Strictly_DECREASING_SEQUENCE, 1396
MAX.Strictly_INCREMENTING_SEQUENCE, 1400
MAX_SUMMIT, 1404
MAX_SURF_BUMP_ON_DECREASING_SEQUENCE, 1408, 2220, 3208
MAX_SURF_DECREASING, 1412, 2224, 3208
MAX_SURF_DECREASING_SEQUENCE, 1416, 2228, 3208
MAX_SURF_DECREASING_TERRACE, 1420, 2232, 3208
MAX_SURF_DIP_ON_INCREASING_SEQUENCE, 1424, 2236, 3208
MAX_SURF_GORGE, 1428, 2240, 3208
MAX_SURF_INCREMENT, 1432, 2244, 3208
MAX_SURF_INCREMENTING_SEQUENCE, 1436, 2248, 3209
MAX_SURF_INCREMENTING_TERRACE, 1440, 2252, 3209
MAX_SURF_INFLEXION, 1444, 2256, 3209
MAX_SURF_PEAK, 1448, 2260, 3209
MIN_MAX_INCREASING, 1660, 2424, 3212
MIN_MAX_INCREASING_SEQUENCE, 1664, 2428, 3212
MIN_MAX_INFLEXION, 1668, 2432, 3212
MIN_MAX_PEAK, iii, 1672, 2436, 3212
MIN_MAX_STRICTLY_DECREASING_SEQUENCE, 1676, 2440, 3212
MIN_MAX_STRICTLY_INCREASING_SEQUENCE, 1680, 2444, 3212
MIN_MAX_SUMMIT, 1684, 2448, 3212
MIN_MAX_ZIGZAG, 1688, 2452, 3212
MIN_MIN_BUMP_ON_DECREASING_SEQUENCE, 1696, 2456, 3212
MIN_MIN_DECREASING, 1700, 2460, 3213
MIN_MIN_DECREASING_SEQUENCE, 1704, 2464, 3213
MIN_MIN_DIP_ON_INCREASING_SEQUENCE, 1708, 2468, 3213
MIN_MIN_GORGE, 1712, 2472, 3213
MIN_MIN_INCREASING, 1718, 2476, 3213
MIN_MIN_INCREASING_SEQUENCE, 1722, 2480, 3213
MIN_MIN_INFLEXION, 1726, 2484, 3213
MIN_MIN_STRICTLY_DECREASING_SEQUENCE, 1730, 2488, 3213
MIN_MIN_STRICTLY_INCREASING_SEQUENCE, 1734, 2492, 3213
MIN_MIN_VALLEY, 1738, 2496, 3213
MIN_MIN_ZIGZAG, 1742, 2500, 3213
MIN_RANGE_DECREASING, 1750, 3213
MIN_RANGE_DECREASING_SEQUENCE, 1754, 3213
MIN_RANGE_INCREASING, 1758, 3214
MIN_RANGE_INCREASING_SEQUENCE, 1762, 3214
MIN_RANGE_STRICTLY_DECREASING_SEQUENCE, 1766, 3214
MIN_RANGE_STRICTLY_INCREASING_SEQUENCE, 1770, 3214
MIN_STRICTLY_DECREASING_SEQUENCE, 1774
MIN_STRICTLY_INCREASING_SEQUENCE, 1778
MIN_SURF_BUMP_ON_DECREASING_SEQUENCE, 1782, 2504, 3214
MIN_SURF_DECREASING, 1786, 2508, 3214
MIN_SURF_DECREASING_SEQUENCE, 1790, 2512, 3214
MIN_SURF_DECRESING_TERRACE, 1794, 2516, 3214
MIN_SURF_DIP_ON_INCREASING_SEQUENCE, 1798, 2520, 3214
MIN_SURF_GORGE, 1802, 2524, 3214
MIN_SURF_INCREASING, 1806, 2528, 3214
MIN_SURF_INCREASING_SEQUENCE, 1810, 2532, 3214
MIN_SURF_INCREASING_TERRACE, 1814, 2536, 3214
MIN_SURF_INFLEXION, 1818, 2540, 3215
MIN_SURF_PEAK, 1822, 2544, 3215
MIN_SURF_PLAIN, 1826, 2548, 3215
MIN_SURF_PLATEAU, 1830, 2552, 3215
MIN_SURF_PROPER_PLAIN, 1834, 2556, 3215
MIN_SURF_PROPER_PLATEAU, 1838, 2560, 3215
MIN_SURF_STEADY, 1842, 2564, 3215
MIN_SURF_STEADY_SEQUENCE, 1846, 2568, 3215
MIN_SURF_STRICTLY_DECREASING_SEQUENCE, 1850, 2572, 3215
MIN_SURF_STRICTLY_INCREASING_SEQUENCE, 1854, 2576, 3215
MIN_SURF_SUMMIT, 1858, 2580, 3215
MIN_SURF_VALLEY, 1862, 2584, 3215
MIN_SURF_ZIGZAG, 1866, 2588, 3215
INDEX

MIN_WIDTH, 1874
MIN_WIDTH_DECREASING_SEQUENCE, 1878, 2592, 3216
MIN_WIDTH_DECREASING_TERRACE, 1882, 2596, 3216
MIN_WIDTH_GORGE, 1886, 2600, 3216
MIN_WIDTH_INCREASING_SEQUENCE, 1890, 2604, 3216
MIN_WIDTH_INCREASING_TERRACE, 1894, 2608, 3216
MIN_WIDTH_INFLEXION, 1898, 2612, 3216
MIN_WIDTH_PEAK, 1902, 2616, 3216
MIN_WIDTH_PLAIN, ii, 1906, 2620, 3216
MIN_WIDTH_PLATEAU, iii, 1910, 2624, 3216
MIN_WIDTH_PROPER_PLAIN, 1914, 2628, 3216
MIN_WIDTH_PROPER_PLATEAU, 1918, 2632, 3216
MIN_WIDTH_STEADY_SEQUENCE, 1922, 2636, 3216
MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE, 1926, 2640, 3216
MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE, 1930, 2644, 3217
MIN_WIDTH_SUMMIT, 1934, 2648, 3217
MIN_WIDTH_VALLEY, 1938, 2652, 3217
MIN_WIDTH_ZIGZAG, 1942, 2656, 3217
MIN_ZIGZAG, 1950

N

NB_BUMP_ON_DECREASING_SEQUENCE, ii, 1954, 3223–3243
NB_DECREASING, 1960, 3217, 3223, 3244–3263
NB_DECREASING_SEQUENCE, 1966, 3217, 3223, 3244, 3264–3282
NB_DECREASING_TERRACE, 1972, 3217, 3224, 3244, 3264, 3283–3307
NB_DIP_ON_INCREASING_SEQUENCE, iii, 1978, 3203, 3204, 3225, 3245, 3264, 3283, 3307–3324
NB_GORGE, iii, 1984, 3217, 3226, 3245, 3265, 3283, 3307, 3325–3341
NB_INCREASING, 1992, 3217, 3226, 3246, 3265, 3284, 3308, 3325, 3342–3359
NB_INCREASING_SEQUENCE, 1998, 3218, 3227, 3246, 3266, 3284, 3308, 3325, 3342, 3359–3375
NB_INCREASING_TERRACE, 2004, 3218, 3228, 3247, 3266, 3285, 3309, 3326, 3342, 3359, 3375–3395
NB_INFLEXION, 2010, 3218, 3229, 3247, 3267, 3286, 3309, 3326, 3343, 3360, 3375, 3395–3414
NB_PEAK, iii, 14, 2016, 3218, 3229, 3248, 3267, 3287, 3310, 3327, 3343, 3360, 3376, 3395, 3415–3429
NB_PLAIN, 2022, 3218, 3230, 3248, 3268, 3287, 3310, 3327, 3344, 3361, 3376, 3396, 3415, 3430–3443
NB_PLATEAU, 2028, 3218, 3230, 3249, 3268, 3288, 3311, 3328, 3344, 3361, 3377, 3396, 3415, 3430, 3444–3457
NB_PROPER_PLAIN, 2034, 3218, 3231, 3249, 3269, 3288, 3311, 3328, 3345, 3362, 3377, 3397, 3416, 3430, 3444, 3458–3473
NB_PROPER_PLATEAU, 2040, 3219, 3231, 3250, 3269, 3289, 3312, 3329, 3345, 3362, 3378, 3397, 3416, 3431, 3444, 3458, 3474–3488
NB_STEADY, 2046, 3219, 3232, 3250, 3270, 3289, 3312, 3329, 3346, 3363, 3378, 3398, 3417, 3431, 3445, 3459, 3474, 3489–3504
NB_STEADY_SEQUENCE, 2052, 3203, 3219, 3232, 3251, 3270, 3290, 3313, 3330, 3346, 3363, 3379, 3398, 3417, 3432, 3445, 3459, 3474, 3489, 3504–3523
NB_STRICTLY_DECREASING_SEQUENCE, 2058, 3219, 3233, 3251, 3271, 3290, 3313, 3330, 3347, 3364, 3379, 3399, 3418, 3432, 3446, 3460, 3475, 3489, 3504, 3523–3536
NB_STRICTLY_INCREASING_SEQUENCE, 2064, 3219, 3233, 3252, 3271, 3291, 3314, 3331, 3347, 3364, 3380, 3400, 3418, 3433, 3446, 3460, 3475, 3490, 3505, 3523, 3537–3549
NB_SUMMIT, iii, 2070, 3219, 3234, 3252, 3272, 3291, 3314, 3331, 3348, 3365, 3380, 3400, 3419, 3433, 3447, 3461, 3476, 3490, 3505, 3524, 3537, 3550–3560
NB_VALLEY, iii, 2078, 3219, 3234, 3253, 3272, 3292, 3315, 3332, 3348, 3365, 3381, 3401, 3419, 3434, 3447, 3461, 3476, 3491, 3506, 3524, 3537, 3550–3560
NB_ZIGZAG, iii, 2084, 3220, 3235, 3253, 3273, 3292, 3315, 3332, 3349, 3366, 3381, 3401, 3420, 3434, 3448, 3462, 3477, 3491, 3506, 3525, 3538, 3550, 3561, 3571–3583
non_increasing_size, 4
nval, 6

P

POS_MAX_HEIGHT_DECREASING_TERRACE, 2092
POS_MAX_HEIGHT_InCREASING_TERRACE, 2096
POS_MAX_HEIGHT_PLAIN, 2100
POS_MAX_HEIGHT_PLATEAU, 2104
POS_MAX_HEIGHT_PROPER_PLAIN, 2108
POS_MAX_HEIGHT_PROPER_PLATEAU, 2112
POS_MAX_HEIGHT_STEADY, 2116
POS_MAX_HEIGHT_STEADY_SEQUENCE, 2120
POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE, 2124
POS_MAX_MAX_DECENTSING, 2128
POS_MAX_MAX_DECLINING_SEQUENCE, 2132
POS_MAX_MAX_DIP_ON_INCREASING_SEQUENCE, 2136
POS_MAX_MAX_INCREASING, 2140
POS_MAX_MAX_INCREASING_SEQUENCE, 2144
POS_MAX_MAX_INFLEXION, 2148
POS_MAX_MAX_PEAK, 2152
POS_MAX_MAX_STRICTLY_DECREASING_SEQUENCE, 2156
POS_MAX_MAX_STRICTLY_INCREASING_SEQUENCE, 2160
POS_MAX_MAX_SUMMIT, 2164
POS_MAX_MAX_ZIGZAG, 2168
POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE, 2172
POS_MAX_MIN_DECESING, 2176
POS_MAX_MIN_DECREASING_SEQUENCE, 2180
POS_MAX_MIN_DIP_ON_INCREASING_SEQUENCE, 2184
POS_MAX_MIN_GORGE, 2188
POS_MAX_MIN_INCREASING, 2192
POS_MAX_MIN_INCREASING_SEQUENCE, 2196
<table>
<thead>
<tr>
<th>INDEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>POS_MAX_MIN_INFLEXION, 2200</td>
</tr>
<tr>
<td>POS_MAX_MIN STRICTLY DECREASING_SEQUENCE, 2204</td>
</tr>
<tr>
<td>POS_MAX_MIN STRICTLY INCREASING_SEQUENCE, 2208</td>
</tr>
<tr>
<td>POS_MAX_MIN VALLEY, 2212</td>
</tr>
<tr>
<td>POS_MAX_MIN ZIGZAG, 2216</td>
</tr>
<tr>
<td>POS_MAX_SURF_BUMP ON DECREASING_SEQUENCE, 2220</td>
</tr>
<tr>
<td>POS_MAX_SURF DECREASING, 2224</td>
</tr>
<tr>
<td>POS_MAX_SURF DECREASING_SEQUENCE, 2228</td>
</tr>
<tr>
<td>POS_MAX_SURF DECREASING_TERRACE, 2232</td>
</tr>
<tr>
<td>POS_MAX_SURF_DIP ON INCREASING_SEQUENCE, 2236</td>
</tr>
<tr>
<td>POS_MAX_SURF GORGE, 2240</td>
</tr>
<tr>
<td>POS_MAX_SURF_INCREASING, 2244</td>
</tr>
<tr>
<td>POS_MAX_SURF_INCREASING_SEQUENCE, 2248</td>
</tr>
<tr>
<td>POS_MAX_SURF_INCREASING_TERRACE, 2252</td>
</tr>
<tr>
<td>POS_MAX_SURF_INFLEXION, 2256</td>
</tr>
<tr>
<td>POS_MAX_SURF_PEAK, 2260</td>
</tr>
<tr>
<td>POS_MAX_SURF_PLAIN, 2264</td>
</tr>
<tr>
<td>POS_MAX_SURF_PLATEAU, 2268</td>
</tr>
<tr>
<td>POS_MAX_SURF_PROPER_PLAIN, 2272</td>
</tr>
<tr>
<td>POS_MAX_SURF_PROPER_PLATEAU, 2276</td>
</tr>
<tr>
<td>POS_MAX_SURF_STEADY, 2280</td>
</tr>
<tr>
<td>POS_MAX_SURF_STEADY_SEQUENCE, 2284</td>
</tr>
<tr>
<td>POS_MAX_SURF STRICTLY DECREASING_SEQUENCE, 2288</td>
</tr>
<tr>
<td>POS_MAX_SURF STRICTLY INCREASING_SEQUENCE, 2292</td>
</tr>
<tr>
<td>POS_MAX_SURF SUMMIT, 2296</td>
</tr>
<tr>
<td>POS_MAX_SURF VALLEY, 2300</td>
</tr>
<tr>
<td>POS_MAX_SURF ZIGZAG, 2304</td>
</tr>
<tr>
<td>POS_MAX_WIDTH DECREASING_SEQUENCE, 2308</td>
</tr>
<tr>
<td>POS_MAX_WIDTH DECREASING_TERRACE, 2312</td>
</tr>
<tr>
<td>POS_MAX_WIDTH GORGE, 2316</td>
</tr>
<tr>
<td>POS_MAX_WIDTH INCREASING_SEQUENCE, 2320</td>
</tr>
<tr>
<td>POS_MAX_WIDTH INCREASING_TERRACE, 2324</td>
</tr>
<tr>
<td>POS_MAX_WIDTH INFLEXION, 2328</td>
</tr>
<tr>
<td>POS_MAX_WIDTH PEAK, 2332</td>
</tr>
<tr>
<td>POS_MAX_WIDTH PLAIN, 2336</td>
</tr>
<tr>
<td>POS_MAX_WIDTH PLATEAU, 2340</td>
</tr>
<tr>
<td>POS_MAX_WIDTH_PROPER_PLAIN, 2344</td>
</tr>
<tr>
<td>POS_MAX_WIDTH_PROPER_PLATEAU, 2348</td>
</tr>
<tr>
<td>POS_MAX_WIDTH_STEADY_SEQUENCE, 2352</td>
</tr>
<tr>
<td>POS_MAX_WIDTH STRICTLY DECREASING_SEQUENCE, 2356</td>
</tr>
<tr>
<td>POS_MAX_WIDTH STRICTLY INCREASING_SEQUENCE, 2360</td>
</tr>
<tr>
<td>POS_MAX_WIDTH_SUMMIT, 2364</td>
</tr>
<tr>
<td>POS_MAX_WIDTH_VALLEY, 2368</td>
</tr>
<tr>
<td>POS_MAX_WIDTH_ZIGZAG, 2372</td>
</tr>
<tr>
<td>POS_MIN_HEIGHT DECREASING_TERRACE, 2376</td>
</tr>
<tr>
<td>POS_MIN_HEIGHT INCREASING_TERRACE, 2380</td>
</tr>
<tr>
<td>POS_MIN_HEIGHT PLAIN, 2384</td>
</tr>
<tr>
<td>POS_MIN_HEIGHT PLATEAU, 2388</td>
</tr>
<tr>
<td>POS_MIN_HEIGHT_PROPER_PLAIN, 2392</td>
</tr>
<tr>
<td>POS_MIN_HEIGHT_PROPER_PLATEAU, 2396</td>
</tr>
</tbody>
</table>
POS_MIN_HEIGHT_STEADY, 2400
POS_MIN_HEIGHT_STEADY_SEQUENCE, 2404
POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE, 2408
POS_MIN_MAX_DECREASING, 2412
POS_MIN_MAX_DECREASING_SEQUENCE, 2416
POS_MIN_MAX_DIP_ON_INCREASING_SEQUENCE, 2420
POS_MIN_MAX_INCREASING, 2424
POS_MIN_MAX_INCREASING_SEQUENCE, 2428
POS_MIN_MAX_INFLEXION, 2432
POS_MIN_MAX_PEAK, 16, 17, 2436
POS_MIN_MAX_strictly_decreasing_sequence, 2440
POS_MIN_MAX_strictly_increasing_sequence, 2444
POS_MIN_MAX_summit, 2448
POS_MIN_MAX_zigzag, 2452
POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE, 2456
POS_MIN_MIN_DECREASING, 2460
POS_MIN_MIN_DECREASING_SEQUENCE, 2464
POS_MIN_MIN_DIP_ON_INCREASING_SEQUENCE, 2468
POS_MIN_MIN_GORGE, 2472
POS_MIN_MIN_INCREASING, 2476
POS_MIN_MIN_INCREASING_SEQUENCE, 2480
POS_MIN_MIN_INFLEXION, 2484
POS_MIN_MIN_strictly_decreasing_sequence, 2488
POS_MIN_MIN_strictly_increasing_sequence, 2492
POS_MIN_MIN_VALLEY, 2496
POS_MIN_MIN_zigzag, 2500
POS_MIN_SURF_BUMP_ON_DECREASING_SEQUENCE, 2504
POS_MIN_SURF_DECREASING, 2508
POS_MIN_SURF_DECREASING_SEQUENCE, 2512
POS_MIN_SURF_DECREASING_TERRACE, 2516
POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE, 2520
POS_MIN_SURF_GORGE, 2524
POS_MIN_SURF_INCREASING, 2528
POS_MIN_SURF_INCREASING_SEQUENCE, 2532
POS_MIN_SURF_INCREASING_TERRACE, 2536
POS_MIN_SURF_INFLEXION, 2540
POS_MIN_SURF_PEAK, 2544
POS_MIN_SURF_PLAIN, 2548
POS_MIN_SURF_PLATEAU, 2552
POS_MIN_SURF_PROPER_PLAIN, 2556
POS_MIN_SURF_PROPER_PLATEAU, 2560
POS_MIN_SURF_STEADY, 2564
POS_MIN_SURF_STEADY_SEQUENCE, 2568
POS_MIN_SURF_strictly_decreasing_sequence, 2572
POS_MIN_SURF_strictly_increasing_sequence, 2576
POS_MIN_SURF_summit, 2580
POS_MIN_SURF_VALLEY, 2584
POS_MIN_SURF_zigzag, 2588
POS_MIN_WIDTH_DECREASING_SEQUENCE, 2592
POS_MIN_WIDTH_DECREASING_TERRACE, 2596
POS_MIN_WIDTH_GORGE, 2600
POS_MIN_WIDTH_INCREASING_SEQUENCE, 2604
POS_MIN_WIDTH_INCREASING_TERRACE, 2608
POS_MIN_WIDTH_INFLEXION, 2612
POS_MIN_WIDTH_PEAK, 2616
POS_MIN_WIDTH_PLAIN, 2620
POS_MIN_WIDTH_PLATEAU, 2624
POS_MIN_WIDTH_PROPER_PLAIN, 2628
POS_MIN_WIDTH_PROPER_PLATEAU, 2632
POS_MIN_WIDTH_STEADY_SEQUENCE, 2636
POS_MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE, 2640
POS_MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE, 2644
POS_MIN_WIDTH_SUMMIT, 2648
POS_MIN_WIDTH_VALLEY, 2652
POS_MIN_WIDTH_ZIGZAG, 2656

prod, 6

R
range, 6
require_at_least, 5
required, 4

S
same_size, 5
signature

AUTOMATON
ALL_EQUAL_HEIGHT_DECREASING_TERRACE, 158
ALL_EQUAL_HEIGHT_INCREASING_TERRACE, 162
ALL_EQUAL_HEIGHT_PLAIN, 166
ALL_EQUAL_HEIGHT_PLATEAU, 170
ALL_EQUAL_HEIGHT_PROPER_PLAIN, 174
ALL_EQUAL_HEIGHT_PROPER_PLATEAU, 178
ALL_EQUAL_HEIGHT_STEADY, 182
ALL_EQUAL_HEIGHT_STEADY_SEQUENCE, 186
ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE, 190
ALL_EQUAL_MAX_DECREASING, 194
ALL_EQUAL_MAX_DECREASING_SEQUENCE, 198
ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE, 202
ALL_EQUAL_MAX_INCREASING, 206
ALL_EQUAL_MAX_INCREASING_SEQUENCE, 210
ALL_EQUAL_MAX_INFLEXION, 214
ALL_EQUAL_MAX_PEAK, 218
ALL_EQUAL_MAX STRICTLY DECREASING_SEQUENCE, 222
INDEX

ALL_EQUAL_MAX_STRONGLY_INCREASING_SEQUENCE, 226
ALL_EQUAL_MAX_SUMMIT, 230
ALL_EQUAL_MAX_ZIGZAG, 234
ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE, 238
ALL_EQUAL_MIN_DECREASING, 242
ALL_EQUAL_MIN_DECREASING_SEQUENCE, 246
ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE, 250
ALL_EQUAL_MIN_GORGE, 254
ALL_EQUAL_MIN_INCREASING, 258
ALL_EQUAL_MIN_INCREASING_SEQUENCE, 262
ALL_EQUAL_MIN_INFLexion, 266
ALL_EQUAL_MIN_STRONGLY_DECREASING_SEQUENCE, 270
ALL_EQUAL_MIN_STRONGLY_INCREASING_SEQUENCE, 274
ALL_EQUAL_MIN_VALLEY, 278
ALL_EQUAL_MIN_ZIGZAG, 282
ALL_EQUAL_RANGE_DECReasing, 286
ALL_EQUAL_RANGE_DECReasing_SEQUENCE, 290
ALL_EQUAL_RANGE_INCREASING, 294
ALL_EQUAL_RANGE_INCREASING_SEQUENCE, 298
ALL_EQUAL_RANGE_STRONGLY_DECReasing_SEQUENCE, 302
ALL_EQUAL_RANGE_STRONGLY_INCREASING_SEQUENCE, 306
ALL_EQUAL_SURF_BUMP_ON_DECReasing_SEQUENCE, 310
ALL_EQUAL_SURF_DECReasing, 314
ALL_EQUAL_SURF_DECReasing_SEQUENCE, 318
ALL_EQUAL_SURF_DECReasing_TERRACE, 322
ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE, 326
ALL_EQUAL_SURF_GORGE, 330
ALL_EQUAL_SURF_INCREASING, 334
ALL_EQUAL_SURF_INCREASING_SEQUENCE, 338
ALL_EQUAL_SURF_INCREASING_TERRACE, 342
ALL_EQUAL_SURF_INFLexion, 346
ALL_EQUAL_SURF_PEAK, 350
ALL_EQUAL_SURF_PLAIN, 354
ALL_EQUAL_SURF_PLATEAU, 358
ALL_EQUAL_SURF_PROPER_PLAIN, 362
ALL_EQUAL_SURF_PROPER_PLATEAU, 366
ALL_EQUAL_SURF_STEADY, 370
ALL_EQUAL_SURF_STEADY_SEQUENCE, 374
ALL_EQUAL_SURF_STRONGLY_DECReasing_SEQUENCE, 378
ALL_EQUAL_SURF_STRONGLY_INCREASING_SEQUENCE, 382
ALL_EQUAL_SURF_SUMMIT, 386
ALL_EQUAL_SURF_VALLEY, 390
ALL_EQUAL_SURF_ZIGZAG, 394
ALL_EQUAL_WIDTH_DECReasing_SEQUENCE, 398
ALL_EQUAL_WIDTH_DECReasing_TERRACE, 402
ALL_EQUAL_WIDTH_GORGE, 406
ALL_EQUAL_WIDTH_INCREASING_SEQUENCE, 410
ALL_EQUAL_WIDTH_INCREASING_TERRACE, 414
ALL_EQUAL_WIDTH_INFLexion, 418
ALL_EQUAL_WIDTH_PEAK, 422
INDEX

ALL_EQUAL_WIDTHPLAIN, 426
ALL_EQUAL_WIDTH PLATEAU, 430
ALL_EQUAL_WIDTH PROPERPLAIN, 434
ALL_EQUAL_WIDTH PROPERPLATEAU, 438
ALL_EQUAL_WIDTH STEADYSEQUENCE, 442
ALL_EQUAL_WIDTH STRICTLYDECREASINGSEQUENCE, 446
ALL_EQUAL_WIDTH STRICTLYINCREASINGSEQUENCE, 450
ALL_EQUAL_WIDTH SUMMIT, 454
ALL_EQUAL_WIDTH VALLEY, 458
ALL_EQUAL_WIDTH ZIGZAG, 462
DECREASING_HEIGHTDECREASING TERRACE, 466
DECREASING_HEIGHTINCREASING TERRACE, 470
DECREASING_HEIGHTPLAIN, 474
DECREASING_HEIGHT PLATEAU, 478
DECREASING_HEIGHT PROPERPLAIN, 482
DECREASING_HEIGHT PROPERPLATEAU, 486
DECREASING_HEIGHT STEADY, 490
DECREASING_HEIGHT STEADYSEQUENCE, 494
DECREASING_MAX BUMP_ON DECREASINGSEQUENCE, 498
DECREASING_MAX DECREASING, 502
DECREASING_MAX DECREASINGSEQUENCE, 506
DECREASING_MAX_DIP_ON_INCREASINGSEQUENCE, 510
DECREASING_MAX_INCREASING, 514
DECREASING_MAX_INCREASINGSEQUENCE, 518
DECREASING_MAX_INFLEXION, 522
DECREASING_MAX_PEEK, 526
DECREASING_MAX.StrictLY DECREASINGSEQUENCE, 530
DECREASING_MAX.StrictLY INCREASINGSEQUENCE, 534
DECREASING_MAX SUMMIT, 538
DECREASING_MAX ZIGZAG, 542
DECREASING_MIN BUMP_ON DECREASINGSEQUENCE, 546
DECREASING_MIN DECREASING, 550
DECREASING_MIN DECREASINGSEQUENCE, 554
DECREASING_MIN_DIP_ON_INCREASINGSEQUENCE, 558
DECREASING_MIN GORGE, 562
DECREASING_MIN INCREASING, 566
DECREASING_MIN INCREASINGSEQUENCE, 570
DECREASING_MIN_INFLEXION, 574
DECREASING_MIN.StrictLY DECREASINGSEQUENCE, 578
DECREASING_MIN.StrictLY INCREASINGSEQUENCE, 582
DECREASING_MIN VALLEY, 586
DECREASING_MIN ZIGZAG, 590
DECREASING_RANGE DECREASING, 594
DECREASING_RANGE DECREASINGSEQUENCE, 598
DECREASING_RANGE INCREASING, 602
DECREASING_RANGE INCREASINGSEQUENCE, 606
DECREASING_RANGE.StrictLY DECREASINGSEQUENCE, 610
DECREASING_RANGE.StrictLY INCREASINGSEQUENCE, 614
DECREASING_SURF_BUMP_ON DECREASINGSEQUENCE, 618
DECREASING_SURF DECREASING, 622
DECREASING_SURF_DECREASING_SEQUENCE, 626
DECREASING_SURF_DECREASING_TERRACE, 630
DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE, 634
DECREASING_SURF_GORGE, 638
DECREASING_SURF_INCREASING, 642
DECREASING_SURF_INCREASING_SEQUENCE, 646
DECREASING_SURF_INCREASING_TERRACE, 650
DECREASING_SURF_INFLEXION, 654
DECREASING_SURF_PEAK, 658
DECREASING_SURF.PLAIN, 662
DECREASING_SURF_PLATEAU, 666
DECREASING_SURF_PROPER.PLAIN, 670
DECREASING_SURF_PROPER_PLATEAU, 674
DECREASING_SURF_STEADY, 678
DECREASING_SURF_STEADY_SEQUENCE, 682
DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE, 686
DECREASING_SURF_STRICTLY_INCREASING_SEQUENCE, 690
DECREASING_SURF_SUMMIT, 694
DECREASING_SURF_VALLEY, 698
DECREASING_SURF_ZIGZAG, 702
DECREASING_WIDTH_DECREASING_SEQUENCE, 706
DECREASING_WIDTH_DECREASING_TERRACE, 710
DECREASING_WIDTH_GORGE, 714
DECREASING_WIDTH_INCREASING_SEQUENCE, 718
DECREASING_WIDTH_INCREASING_TERRACE, 722
DECREASING_WIDTH_INFLEXION, 726
DECREASING_WIDTH_PEAK, 730
DECREASING_WIDTH.PLAIN, 734
DECREASING_WIDTH_PLATEAU, 738
DECREASING_WIDTH_PROPER.PLAIN, 742
DECREASING_WIDTH_PROPER_PLATEAU, 746
DECREASING_WIDTH_STEADY_SEQUENCE, 750
DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE, 754
DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE, 758
DECREASING_WIDTH_SUMMIT, 762
DECREASING_WIDTH_VALLEY, 766
DECREASING_WIDTH_ZIGZAG, 770
HEIGHT_DECREASING_TERRACE, 774
HEIGHT_INCREASING_TERRACE, 778
HEIGHT.PLAIN, 782
HEIGHT_PLATEAU, 786
HEIGHT_PROPER.PLAIN, 790
HEIGHT_PROPER_PLATEAU, 794
HEIGHT_STEADY, 798
HEIGHT_STEADY_SEQUENCE, 802
INCREASING_HEIGHT_DECREASING_TERRACE, 806
INCREASING_HEIGHT_INCREASING_TERRACE, 810
INCREASING_HEIGHT.PLAIN, 814
INCREASING_HEIGHT_PLATEAU, 818
INCREASING_HEIGHT_PROPER.PLAIN, 822
INDEX

INCREASING_HEIGHT_PROPER_PLATEAU, 826
INCREASING_HEIGHT_STEADY, 830
INCREASING_HEIGHT_STEADY_SEQUENCE, 834
INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE, 838
INCREASING_MAX_DECENDING, 842
INCREASING_MAX_DECENDING_SEQUENCE, 846
INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE, 850
INCREASING_MAX_INCREASING, 854
INCREASING_MAX_INCREASING_SEQUENCE, 858
INCREASING_MAX_INFLEXION, 862
INCREASING_MAX_PEAK, 866
INCREASING_MAX.Strictly_DECENDING_SEQUENCE, 870
INCREASING_MAX.Strictly_INCREASING_SEQUENCE, 874
INCREASING_MAX_SUMMIT, 878
INCREASING_MAX_ZIGZAG, 882
INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE, 886
INCREASING_MIN_DECENDING, 890
INCREASING_MIN_DecENDING_SEQUENCE, 894
INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE, 898
INCREASING_MIN_GORGE, 902
INCREASING_MIN_INCREASING, 906
INCREASING_MIN_INCREASING_SEQUENCE, 910
INCREASING_MIN_INFLEXION, 914
INCREASING_MIN.Strictly_DECENDING_SEQUENCE, 918
INCREASING_MIN.Strictly_INCREASING_SEQUENCE, 922
INCREASING_MIN_VALLEY, 926
INCREASING_MIN_ZIGZAG, 930
INCREASING_RANGE_DECENDING, 934
INCREASING_RANGE_DECENDING_SEQUENCE, 938
INCREASING_RANGE_INCREASING, 942
INCREASING_RANGE_INCREASING_SEQUENCE, 946
INCREASING_RANGE.Strictly_DECENDING_SEQUENCE, 950
INCREASING_RANGE.Strictly_INCREASING_SEQUENCE, 954
INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE, 958
INCREASING_SURF_DECENDING, 962
INCREASING_SURF_DECENDING_SEQUENCE, 966
INCREASING_SURF_DECENDING_TERRACE, 970
INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE, 974
INCREASING_SURF_GORGE, 978
INCREASING_SURF_INCREASING, 982
INCREASING_SURF_INCREASING_SEQUENCE, 986
INCREASING_SURF_INCREASING_TERRACE, 990
INCREASING_SURF_INFLEXION, 994
INCREASING_SURF_PEAK, 998
INCREASING_SURF_PLAIN, 1002
INCREASING_SURF_PLATEAU, 1006
INCREASING_SURF_PROPER_PLAIN, 1010
INCREASING_SURF_PROPER_PLATEAU, 1014
INCREASING_SURF_STEADY, 1018
INCREASING_SURF_STEADY_SEQUENCE, 1022
<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCREASING_SURF_STRICTLY_DECREASING_SEQUENCE</td>
<td>1026</td>
</tr>
<tr>
<td>INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE</td>
<td>1030</td>
</tr>
<tr>
<td>INCREASING_SURF_SUMMIT</td>
<td>1034</td>
</tr>
<tr>
<td>INCREASING_SURF_VALLEY</td>
<td>1038</td>
</tr>
<tr>
<td>INCREASING_SURF_ZIGZAG</td>
<td>1042</td>
</tr>
<tr>
<td>INCREASING_WIDTH_DECREASING_SEQUENCE</td>
<td>1046</td>
</tr>
<tr>
<td>INCREASING_WIDTH_DECREASING_TERRACE</td>
<td>1050</td>
</tr>
<tr>
<td>INCREASING_WIDTH_GORGE</td>
<td>1054</td>
</tr>
<tr>
<td>INCREASING_WIDTH_INCREASING_SEQUENCE</td>
<td>1058</td>
</tr>
<tr>
<td>INCREASING_WIDTH_INCREASING_TERRACE</td>
<td>1062</td>
</tr>
<tr>
<td>INCREASING_WIDTH_INFLEXION</td>
<td>1066</td>
</tr>
<tr>
<td>INCREASING_WIDTH_PEEK</td>
<td>1070</td>
</tr>
<tr>
<td>INCREASING_WIDTH_PLAIN</td>
<td>1074</td>
</tr>
<tr>
<td>INCREASING_WIDTH_PLATEAU</td>
<td>1078</td>
</tr>
<tr>
<td>INCREASING_WIDTH_PROPER_PLAIN</td>
<td>1082</td>
</tr>
<tr>
<td>INCREASING_WIDTH_PROPER_PLATEAU</td>
<td>1086</td>
</tr>
<tr>
<td>INCREASING_WIDTH_STEADY_SEQUENCE</td>
<td>1090</td>
</tr>
<tr>
<td>INCREASING_WIDTH.Strictly_Decreasing.Sequence</td>
<td>1094</td>
</tr>
<tr>
<td>INCREASING_WIDTH.Strictly_Increasing.Sequence</td>
<td>1098</td>
</tr>
<tr>
<td>INCREASING_WIDTH_SUMMIT</td>
<td>1102</td>
</tr>
<tr>
<td>INCREASING_WIDTH_VALLEY</td>
<td>1106</td>
</tr>
<tr>
<td>INCREASING_WIDTH_ZIGZAG</td>
<td>1110</td>
</tr>
<tr>
<td>INDEX_BUMP_ON_DECREASING_SEQUENCE</td>
<td>1114</td>
</tr>
<tr>
<td>INDEX_DECREASING</td>
<td>1118</td>
</tr>
<tr>
<td>INDEX_DECREASING_SEQUENCE</td>
<td>1122</td>
</tr>
<tr>
<td>INDEX_Decreasing_TERRACE</td>
<td>1126</td>
</tr>
<tr>
<td>INDEX_DIP_ON_INCREASING_SEQUENCE</td>
<td>1130</td>
</tr>
<tr>
<td>INDEX_GORGE</td>
<td>1134</td>
</tr>
<tr>
<td>INDEX_INCREASING</td>
<td>1138</td>
</tr>
<tr>
<td>INDEX_INCREASING_SEQUENCE</td>
<td>1142</td>
</tr>
<tr>
<td>INDEX_INCREASING_TERRACE</td>
<td>1146</td>
</tr>
<tr>
<td>INDEX_INFLEXION</td>
<td>1150</td>
</tr>
<tr>
<td>INDEX_PEAK</td>
<td>1154</td>
</tr>
<tr>
<td>INDEX_PLAIN</td>
<td>1158</td>
</tr>
<tr>
<td>INDEX_PLATEAU</td>
<td>1162</td>
</tr>
<tr>
<td>INDEX_PROPER_PLAIN</td>
<td>1166</td>
</tr>
<tr>
<td>INDEX_PROPER_PLATEAU</td>
<td>1170</td>
</tr>
<tr>
<td>INDEX_STEADY</td>
<td>1174</td>
</tr>
<tr>
<td>INDEX_STEADY_SEQUENCE</td>
<td>1178</td>
</tr>
<tr>
<td>INDEX STRICTLY DECREASING_SEQUENCE</td>
<td>1182</td>
</tr>
<tr>
<td>INDEX STRICTLY INCREASING_SEQUENCE</td>
<td>1186</td>
</tr>
<tr>
<td>INDEX_SUMMIT</td>
<td>1190</td>
</tr>
<tr>
<td>INDEX_VALLEY</td>
<td>1194</td>
</tr>
<tr>
<td>INDEX_ZIGZAG</td>
<td>1198</td>
</tr>
<tr>
<td>MAX_BUMP_ON_DECREASING_SEQUENCE</td>
<td>1202</td>
</tr>
<tr>
<td>MAX_DECREASING</td>
<td>1206</td>
</tr>
<tr>
<td>MAX_DECREASING_SEQUENCE</td>
<td>1210</td>
</tr>
<tr>
<td>MAX_DIP_ON_INCREASING_SEQUENCE</td>
<td>1214</td>
</tr>
<tr>
<td>MAX_HEIGHT_DECREASING_TERRACE</td>
<td>1218</td>
</tr>
<tr>
<td>MAX_HEIGHT_INCREASING_TERRACE</td>
<td>1222</td>
</tr>
</tbody>
</table>
INDEX

MAX_HEIGHTPLAIN, 1226
MAX_HEIGHTPLATEAU, 1230
MAX_HEIGHTPROPERPLAIN, 1234
MAX_HEIGHTPROPERPLATEAU, 1238
MAX_HEIGHTSTEADY, 1242
MAX_HEIGHTSTEADYSEQUENCE, 1246
MAX_INCREASING, 1250
MAX_INCREASINGSEQUENCE, 1254
MAX_INFLEXION, 1258
MAX_MAXBUMponDECREASINGSEQUENCE, 1262
MAX_MAXDECREASING, 1266
MAX_MAXDECREASINGSEQUENCE, 1270
MAX_MAX_DIPonINCREASINGSEQUENCE, 1274
MAX_MAX_INCREASING, 1278
MAX_MAX_INCREASINGSEQUENCE, 1282
MAX_MAX_PEAK, 1290
MAX_MAXSTRICELYDECREASINGSEQUENCE, 1294
MAX_MAXSTRICLYINCREASINGSEQUENCE, 1298
MAX_MAX_SUMMIT, 1302
MAX_MAX_ZIGZAG, 1306
MAX_MIN_BUMPonDECREASINGSEQUENCE, 1314
MAX_MIN_DECREASING, 1318
MAX_MIN_DECREASINGSEQUENCE, 1322
MAX_MIN_DIPonINCREASINGSEQUENCE, 1326
MAX_MIN_GORGE, 1330
MAX_MIN_INCREASING, 1336
MAX_MIN_INCREASINGSEQUENCE, 1340
MAX_MIN_INFLEXION, 1344
MAX_MIN_STRICELYDECREASINGSEQUENCE, 1348
MAX_MINSTRICELYINCREASINGSEQUENCE, 1352
MAX_MIN VALLEY, 1356
MAX_MIN_ZIGZAG, 1360
MAX_PEAK, 1368
MAX_RANGE_DECREASING, 1372
MAX_RANGE_DECREASINGSEQUENCE, 1376
MAX_RANGE_INCREASING, 1380
MAX_RANGE_INCREASINGSEQUENCE, 1384
MAX_RANGE_STRICELYDECREASINGSEQUENCE, 1388
MAX_RANGE_STRICLYINCREASINGSEQUENCE, 1392
MAX_STRICTLYDECREASINGSEQUENCE, 1396
MAX_STRICTLYINCREASINGSEQUENCE, 1400
MAX_SUMMIT, 1404
MAX_SURF_BUMPonDECREASINGSEQUENCE, 1408
MAX_SURF_DECREASING, 1412
MAX_SURF_DECREASINGSEQUENCE, 1416
MAX_SURF_DECREASING_TERRACE, 1420
MAX_SURF_DIPonINCREASINGSEQUENCE, 1424
MAX_SURF_GORGE, 1428
MAX_SURF_INCREASING, 1432
INDEX

MAX_SURF_INCREASING_SEQUENCE, 1436
MAX_SURF_INCREASING_TERRACE, 1440
MAX_SURF_INFLEXION, 1444
MAX_SURF_PEAK, 1448
MAX_SURF_PLAIN, 1452
MAX_SURF_PLATEAU, 1456
MAX_SURF_PROPER_PLAIN, 1460
MAX_SURF_PROPER_PLATEAU, 1464
MAX_SURF_STEADY, 1468
MAX_SURF_STEADY_SEQUENCE, 1472
MAX_SURF_STRICTLY_DECREASING_SEQUENCE, 1476
MAX_SURF_STRICTLY_INCREASING_SEQUENCE, 1480
MAX_SURF_SUMMIT, 1484
MAX_SURF_VALLEY, 1488
MAX_SURF_ZIGZAG, 1492
MAX_WIDTH_DECREASING_SEQUENCE, 1500
MAX_WIDTH_DECREASING_TERRACE, 1504
MAX_WIDTH_GORGE, 1508
MAX_WIDTH_INCREASING_SEQUENCE, 1514
MAX_WIDTH_INCREASING_TERRACE, 1518
MAX_WIDTH_INFLEXION, 1522
MAX_WIDTH_PEAK, 1526
MAX_WIDTH_PLAIN, 1530
MAX_WIDTH_PLATEAU, 1534
MAX_WIDTH_PROPER_PLAIN, 1538
MAX_WIDTH_PROPER_PLATEAU, 1542
MAX_WIDTH_STEADY_SEQUENCE, 1546
MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE, 1550
MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE, 1554
MAX_WIDTH_SUMMIT, 1558
MAX_WIDTH_VALLEY, 1564
MAX_WIDTH_ZIGZAG, 1568
MAX_ZIGZAG, 1576
MIN_BUMP_ON_DECREASING_SEQUENCE, 1580
MIN_DECREASING, 1584
MIN_DECREASING_SEQUENCE, 1588
MIN_DIP_ON_INCREASING_SEQUENCE, 1592
MIN_GORGE, 1596
MIN_HEIGHT_DECREASING_TERRACE, 1600
MIN_HEIGHT_INCREASING_TERRACE, 1604
MIN_HEIGHT_PLAIN, 1608
MIN_HEIGHT_PLATEAU, 1612
MIN_HEIGHT_PROPER_PLAIN, 1616
MIN_HEIGHT_PROPER_PLATEAU, 1620
MIN_HEIGHT_STEADY, 1624
MIN_HEIGHT_STEADY_SEQUENCE, 1628
MIN_INCREASING, 1632
MIN_INCREASING_SEQUENCE, 1636
MIN_INFLEXION, 1640
MIN_MAX_BUMP_ON_DECREASING_SEQUENCE, 1644
MIN_MAX_DECREASING, 1648
MIN_MAX_DECREEASING_SEQUENCE, 1652
MIN_MAX_DIP_ON_INCREASING_SEQUENCE, 1656
MIN_MAX_INCREASING, 1660
MIN_MAX_INCREASING_SEQUENCE, 1664
MIN_MAX_INFLEXION, 1668
MIN_MAX_PEAK, 1672
MIN_MAX_STRICTLY_DECREASING_SEQUENCE, 1676
MIN_MAX_STRICTLY_INCREASING_SEQUENCE, 1680
MIN_MAX_SUMMIT, 1684
MIN_MAX_ZIGZAG, 1688
MIN_MIN_BUMP_ON_DECREASING_SEQUENCE, 1696
MIN_MIN_DECREEASING, 1700
MIN_MIN_DECREEASING_SEQUENCE, 1704
MIN_MIN_DIP_ON_INCREASING_SEQUENCE, 1708
MIN_MIN_GORGE, 1712
MIN_MIN_INCREASING, 1716
MIN_MIN_INCREASING_SEQUENCE, 1722
MIN_MIN_INFLEXION, 1726
MIN_MIN_STRICTLY_DECREASING_SEQUENCE, 1730
MIN_MIN_STRICTLY_INCREASING_SEQUENCE, 1734
MIN_MIN_VALLEY, 1738
MIN_MIN_ZIGZAG, 1742
MIN_RANGE_DECREEASING, 1750
MIN_RANGE_DECREEASING_SEQUENCE, 1754
MIN_RANGE_INCREASING, 1758
MIN_RANGE_INCREASING_SEQUENCE, 1762
MIN_RANGE_STRICTLY_DECREASING_SEQUENCE, 1766
MIN_RANGE_STRICTLY_INCREASING_SEQUENCE, 1770
MIN_STRICTLY_DECREEASING_SEQUENCE, 1774
MIN_STRICTLY_INCREASING_SEQUENCE, 1778
MIN_SURF_BUMP_ON_DECREEASING_SEQUENCE, 1782
MIN_SURF_DECREEASING, 1786
MIN_SURF_DECREEASING_SEQUENCE, 1790
MIN_SURF_DECREEASING_TERRACE, 1794
MIN_SURF_DIP_ON_INCREASING_SEQUENCE, 1798
MIN_SURF_GORGE, 1802
MIN_SURF_INCREASING, 1806
MIN_SURF_INCREASING_SEQUENCE, 1810
MIN_SURF_INCREASING_TERRACE, 1814
MIN_SURF_INFLEXION, 1818
MIN_SURF_PEAK, 1822
MIN_SURFPLAIN, 1826
MIN_SURFPLATEAU, 1830
MIN_SURFPROPERPLAIN, 1834
MIN_SURFPROPERPLATEAU, 1838
MIN_SURFSTEADY, 1842
MIN_SURF_STEADY_SEQUENCE, 1846
MIN_SURF_STRICTLY_DECREEASING_SEQUENCE, 1850
MIN_SURF_STRICTLY_INCREASING_SEQUENCE, 1854
INDEX

MIN_SURF_SUMMIT, 1858
MIN_SURF_VALLEY, 1862
MIN_SURF_ZIGZAG, 1866
MIN_VALLEY, 1874
MIN_WIDTH_DECREEING_SEQUENCE, 1878
MIN_WIDTH_DECREEING_TERRACE, 1882
MIN_WIDTH_GORGE, 1886
MIN_WIDTH_INCREASESEQUENCE, 1890
MIN_WIDTH_INCREASESEQUENCE, 1894
MIN_WIDTH_INFLATION, 1898
MIN_WIDTH_PEAK, 1902
MIN_WIDTH_PLAIN, 1906
MIN_WIDTH_PLATEAU, 1910
MIN_WIDTH_PROPER_PLAIN, 1914
MIN_WIDTH_PROPER_PLATEAU, 1918
MIN_WIDTH_STEADY_SEQUENCE, 1922
MIN_WIDTH_STRICTLY_DECREEINGSEQUENCE, 1926
MIN_WIDTH_STRICTLY_INCREASESEQUENCE, 1930
MIN_WIDTH_SUMMIT, 1934
MIN_WIDTH_VALLEY, 1938
MIN_WIDTH_ZIGZAG, 1942
MIN_ZIGZAG, 1950
NB_BUMP_ON_DECREEINGSEQUENCE, 1954
NB_DECREEING, 1960
NB_DECREEINGSEQUENCE, 1966
NB_DECREEING_TERRACE, 1972
NB_DIP_ON_INCREASESEQUENCE, 1978
NB_GORGE, 1984
NB_INCREASEING, 1992
NB_INCREASESEQUENCE, 1998
NB_INCREASESEQUENCE, 2004
NB_INFLATION, 2010
NB_PEAK, 2016
NB_PLAIN, 2022
NB_PLATEAU, 2028
NB_PROPER_PLAIN, 2034
NB_PROPER_PLATEAU, 2040
NB_STEADY, 2046
NB_STEADY_SEQUENCE, 2052
NB_STRICTLY_DECREEINGSEQUENCE, 2058
NB_STRICTLY_INCREASESEQUENCE, 2064
NB_SUMMIT, 2070
NB_VALLEY, 2078
NB_ZIGZAG, 2084
POS_MAX_HEIGHT_DECREEING_TERRACE, 2092
POS_MAX_HEIGHT_INCREASESEQUENCE, 2096
POS_MAX_HEIGHT_PLAIN, 2100
POS_MAX_HEIGHT_PLATEAU, 2104
POS_MAX_HEIGHT_PROPER_PLAIN, 2108
POS_MAX_HEIGHT_PROPER_PLATEAU, 2112
INDEX

POS_MAX_WIDTH_GORGE, 2316
POS_MAX_WIDTH_INCREASING_SEQUENCE, 2320
POS_MAX_WIDTH_INCREASING_TERRACE, 2324
POS_MAX_WIDTH_INFLEXION, 2328
POS_MAX_WIDTH_PEAK, 2332
POS_MAX_WIDTH_PLAIN, 2336
POS_MAX_WIDTH_PLATEAU, 2340
POS_MAX_WIDTH_PROPER_PLAIN, 2344
POS_MAX_WIDTH_PROPER_PLATEAU, 2348
POS_MAX_WIDTH_STEADY_SEQUENCE, 2352
POS_MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE, 2356
POS_MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE, 2360
POS_MAX_WIDTH_SUMMIT, 2364
POS_MAX_WIDTH_VALLEY, 2368
POS_MAX_WIDTH_ZIGZAG, 2372
POS_MIN_HEIGHT_DECREASING_TERRACE, 2376
POS_MIN_HEIGHT_INCREASING_TERRACE, 2380
POS_MIN_HEIGHT_PLAIN, 2384
POS_MIN_HEIGHT_PLATEAU, 2388
POS_MIN_HEIGHT_PROPER_PLAIN, 2392
POS_MIN_HEIGHT_PROPER_PLATEAU, 2396
POS_MIN_HEIGHT_STEADY, 2400
POS_MIN_HEIGHT_STEADY_SEQUENCE, 2404
POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE, 2408
POS_MIN_MAX_DECREASING, 2412
POS_MIN_MAX_DECREASING_SEQUENCE, 2416
POS_MIN_MAX_DIP_ON_INCREASING_SEQUENCE, 2420
POS_MIN_MAX_INCREASING, 2424
POS_MIN_MAX_INCREASING_SEQUENCE, 2428
POS_MIN_MAX_INFLEXION, 2432
POS_MIN_MAX_PEAK, 2436
POS_MIN_MAX_STRICTLY_DECREASING_SEQUENCE, 2440
POS_MIN_MAX_STRICTLY_INCREASING_SEQUENCE, 2444
POS_MIN_MAX_SUMMIT, 2448
POS_MIN_MAX_ZIGZAG, 2452
POS_MIN_MIN_BUMP_ON_DECREASING_SEQUENCE, 2456
POS_MIN_MIN_DECREASING, 2460
POS_MIN_MIN_DECREASING_SEQUENCE, 2464
POS_MIN_MIN_DIP_ON_INCREASING_SEQUENCE, 2468
POS_MIN_MIN_GORGE, 2472
POS_MIN_MIN_INCREASING, 2476
POS_MIN_MIN_INCREASING_SEQUENCE, 2480
POS_MIN_MIN_INFLEXION, 2484
POS_MIN_MIN_STRICTLY_DECREASING_SEQUENCE, 2488
POS_MIN_MIN_STRICTLY_INCREASING_SEQUENCE, 2492
POS_MIN_MIN_VALLEY, 2496
POS_MIN_MIN_ZIGZAG, 2500
POS_MIN_SURF_BUMP_ON_DECREASING_SEQUENCE, 2504
POS_MIN_SURF_DECREASING, 2508
POS_MIN_SURF_DECREASING_SEQUENCE, 2512
INDEX

POS_MIN_SURF_DECREASING_TERRACE, 2516
POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE, 2520
POS_MIN_SURF_GORGE, 2524
POS_MIN_SURF_INCREASING, 2528
POS_MIN_SURF_INCREASING_SEQUENCE, 2532
POS_MIN_SURF_INCREASING_TERRACE, 2536
POS_MIN_SURF_INFLEXION, 2540
POS_MIN_SURF_PEAK, 2544
POS_MIN_SURF_PLAIN, 2548
POS_MIN_SURF_PLATEAU, 2552
POS_MIN_SURF_PROPER_PLAIN, 2556
POS_MIN_SURF_PROPER_PLATEAU, 2560
POS_MIN_SURF_STEADY, 2564
POS_MIN_SURF_STEADY_SEQUENCE, 2568
POS_MIN_SURF.Strictly_Decreasing_Sequence, 2572
POS_MIN_SURF.Strictly_Increasing_Sequence, 2576
POS_MIN_SURF_SUMMIT, 2580
POS_MIN_SURF_VALLEY, 2584
POS_MIN_SURF_ZIGZAG, 2588
POS_MIN_WIDTH_DECREASING_SEQUENCE, 2592
POS_MIN_WIDTH_DECREASING_TERRACE, 2596
POS_MIN_WIDTH_GORGE, 2600
POS_MIN_WIDTH_INCREASING_SEQUENCE, 2604
POS_MIN_WIDTH_INCREASING_TERRACE, 2608
POS_MIN_WIDTH_INFLEXION, 2612
POS_MIN_WIDTH_PEAK, 2616
POS_MIN_WIDTH_PLAIN, 2620
POS_MIN_WIDTH_PLATEAU, 2624
POS_MIN_WIDTH_PROPER_PLAIN, 2628
POS_MIN_WIDTH_PROPER_PLATEAU, 2632
POS_MIN_WIDTH_STEADY_SEQUENCE, 2636
POS_MIN_WIDTH.Strictly_Decreasing_Sequence, 2640
POS_MIN_WIDTH.Strictly_Increasing_Sequence, 2644
POS_MIN_WIDTH_SUMMIT, 2648
POS_MIN_WIDTH_VALLEY, 2652
POS_MIN_WIDTH_ZIGZAG, 2656
SUM_HEIGHT_DECREASING_TERRACE, 2660
SUM_HEIGHT_INCREASING_TERRACE, 2666
SUM_HEIGHT_PLAIN, 2672
SUM_HEIGHT_PLATEAU, 2676
SUM_HEIGHT_PROPER_PLAIN, 2680
SUM_HEIGHT_PROPER_PLATEAU, 2684
SUM_HEIGHT_STEADY, 2688
SUM_HEIGHT_STEADY_SEQUENCE, 2692
SUM_MAX_BUMP_ON_DECREASING_SEQUENCE, 2696
SUM_MAX_DECREASING, 2700
SUM_MAX_DECREASING_SEQUENCE, 2706
SUM_MAX_DIP_ON_INCREASING_SEQUENCE, 2710
SUM_MAX_INCREASING, 2714
SUM_MAX_INCREASING_SEQUENCE, 2720
SUM_MAX_INFLEXION, 2724
SUM_MAX_PEAK, 2728
SUM_MAX.Strictly Decreasing_sequence, 2732
SUM_MAX.Strictly Increasing_sequence, 2736
SUM_MAX_Summit, 2740
SUM_MAX_Zigzag, 2746
SUM_MIN_Bump_on_Decreasing_sequence, 2754
SUM_MIN_Decreasing, 2758
SUM_MIN_Decreasing_sequence, 2764
SUM_MIN_Dip_on_Increasing_sequence, 2768
SUM_MIN_Gorge, 2772
SUM_MIN_Increasing, 2778
SUM_MIN_Increasing_sequence, 2784
SUM_MIN_Inflexion, 2788
SUM_MIN.Strictly Decreasing_sequence, 2792
SUM_MIN.Strictly Increasing_sequence, 2796
SUM_MIN_Valley, 2800
SUM_MIN_Zigzag, 2804
SUM_Range_Decreasing, 2812
SUM_Range_Decreasing_sequence, 2816
SUM_Range_Increasing, 2820
SUM_Range_Increasing_sequence, 2824
SUM_Range.Strictly Decreasing_sequence, 2828
SUM_Range.Strictly Increasing_sequence, 2832
SUM_Surf_Bump_on_Decreasing_sequence, 2836
SUM_Surf_Decreasing, 2840
SUM_Surf_Decreasing_sequence, 2846
SUM_Surf_Decreasing_Terrace, 2850
SUM_Surf_Dip_on_Increasing_sequence, 2854
SUM_Surf_Gorge, 2858
SUM_Surf_Increasing, 2864
SUM_Surf_Increasing_sequence, 2870
SUM_Surf_Increasing_Terrace, 2874
SUM_Surf_Inflexion, 2878
SUM_Surf_Peak, 2882
SUM_Surf_Plain, 2886
SUM_Surf_Plateau, 2890
SUM_Surf_Proper_Plain, 2894
SUM_Surf_Proper_Plateau, 2898
SUM_Surf_Steady, 2902
SUM_Surf_Steady_sequence, 2906
SUM_Surf.Strictly Decreasing_sequence, 2910
SUM_Surf.Strictly Increasing_sequence, 2914
SUM_Surf_Summit, 2918
SUM_Surf_Valley, 2924
SUM_Surf_Zigzag, 2928
SUM_Width_Decreasing_sequence, 2936
SUM_Width_Decreasing_Terrace, 2942
SUM_Width_Gorge, 2948
SUM_Width_Increasing_sequence, 2956
INDEX

3731

SUM_WIDTH_INCREASING_TERRACE, 2962
SUM_WIDTH_INFLEXION, 2968
SUM_WIDTH_PEAK, 2974
SUM_WIDTH_PLAIN, 2980
SUM_WIDTH_PLATEAU, 2986
SUM_WIDTH_PROPER_PLAIN, 2992
SUM_WIDTH_PROPER_PLATEAU, 2998
SUM_WIDTH_STEADY_SEQUENCE, 3004
SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE, 3010
SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE, 3016
SUM_WIDTH_SUMMIT, 3022
SUM_WIDTH_VALLEY, 3030
SUM_WIDTH_ZIGZAG, 3036
SURF_BUMP_ON_DECREASING_SEQUENCE, 3046
SURF_DECREASING, 3050
SURF_DECREASING_SEQUENCE, 3054
SURF_DECREASING_TERRACE, 3058
SURF_DIP_ON_INCREASING_SEQUENCE, 3062
SURF_GORGE, 3066
SURF_INCREASING, 3070
SURF_INCREASING_SEQUENCE, 3074
SURF_INCREASING_TERRACE, 3078
SURF_INFLEXION, 3082
SURF_PEAK, 3086
SURF_PLAIN, 3090
SURF_PLATEAU, 3094
SURF_PROPER_PLAIN, 3098
SURF_PROPER_PLATEAU, 3102
SURF_STEADY, 3106
SURF_STEADY_SEQUENCE, 3110
SURF_STRICTLY_DECREASING_SEQUENCE, 3114
SURF_STRICTLY_INCREASING_SEQUENCE, 3118
SURF_SUMMIT, 3122
SURF_VALLEY, 3126
SURF_ZIGZAG, 3130
WIDTH_DECREASING_SEQUENCE, 3134
WIDTH_DECREASING_TERRACE, 3138
WIDTH_GORGE, 3142
WIDTH_INCREASING_SEQUENCE, 3146
WIDTH_INCREASING_TERRACE, 3150
WIDTH_INFLEXION, 3154
WIDTH_PEAK, 3158
WIDTH_PLAIN, 3162
WIDTH_PLATEAU, 3166
WIDTH_PROPER_PLAIN, 3170
WIDTH_PROPER_PLATEAU, 3174
WIDTH_STEADY_SEQUENCE, 3178
WIDTH_STRICTLY_DECREASING_SEQUENCE, 3182
WIDTH_STRICTLY_INCREASING_SEQUENCE, 3186
WIDTH_SUMMIT, 3190
INDEX

WIDTH_VALLEY, 3194
WIDTH_ZIGZAG, 3198

sum, 5
SUM_HEIGHT_DECREASING_TERRACE, 2660
SUM_HEIGHT_INCREASING_TERRACE, 2666
SUM_HEIGHT_PLAIN, 2672
SUM_HEIGHT_PLATEAU, 2676
SUM_HEIGHT_PROPER_PLAIN, 2680
SUM_HEIGHT_PROPER_PLATEAU, 2684
SUM_HEIGHT_STEADY, 2688
SUM_HEIGHT_STEADY_SEQUENCE, 2692
SUM_MAX_BUMP_ON_DECREASING_SEQUENCE, 2696
SUM_MAX_DECREASING, 2700
SUM_MAX_DECREASING_SEQUENCE, 2706
SUM_MAX_DIP_ON_INCREASING_SEQUENCE, 2710
SUM_MAX_INCREASING, 2714
SUM_MAX_INCREASING_SEQUENCE, 2720
SUM_MAX_INFLATION, 2724
SUM_MAX_PEAK, 2728
SUM_MAX_STRICLY_DECREASING_SEQUENCE, 2732
SUM_MAX_STRICLY_INCREASING_SEQUENCE, 2736
SUM_MAX_SUMMIT, 2740
SUM_MAX_ZIGZAG, 2746
SUM_MIN_BUMP_ON_DECREASING_SEQUENCE, 2754
SUM_MIN_DECREASING, 2758
SUM_MIN_DECREASING_SEQUENCE, 2764
SUM_MIN_DIP_ON_INCREASING_SEQUENCE, 2768
SUM_MIN_GORGE, 2772
SUM_MIN_INCREASING, 2778
SUM_MIN_INCREASING_SEQUENCE, 2784
SUM_MIN_INFLATION, 2788
SUM_MIN_STRICLY_DECREASING_SEQUENCE, 2792
SUM_MIN_STRICLY_INCREASING_SEQUENCE, 2796
SUM_MIN_VALLEY, 2800
SUM_MIN_ZIGZAG, 2804
SUM_RANGE_DECREASING, 2812, 3220
SUM_RANGE_DECREASING_SEQUENCE, 2816
SUM_RANGE_INCREASING, 2820
SUM_RANGE_INCREASING_SEQUENCE, 2824
SUM_RANGE_STRICLY_DECREASING_SEQUENCE, 2828
SUM_RANGE_STRICLY_INCREASING_SEQUENCE, 2832
SUM_SURF_BUMP_ON_DECREASING_SEQUENCE, 2836
SUM_SURF_DECREASING, 2840
SUM_SURF_DECREASING_SEQUENCE, 2846
SUM_SURF_DECREASING_TERRACE, 2850
SUM_SURF_DIP_ON_INCREASING_SEQUENCE, 2854
SUM_SURF_GORGE, 2858
SUM_SURF_INCREASING, 2864
SUM_SURF_INCREASING_SEQUENCE, 2870
SUM_SURF_INCREASING_TERRACE, 2874
INDEX

SUM_SURF_INFLEXION, 2878
SUM_SURF_PEAK, 2882
SUM_SURF_PLAIN, 2886
SUM_SURF_PLATEAU, 2890
SUM_SURF_PROPER_PLAIN, 2894
SUM_SURF_PROPER_PLATEAU, 2898
SUM_SURF_STEADY, 2902
SUM_SURF_STEADY_SEQUENCE, 2906
SUM_SURF_STRICTLY_DECREASING_SEQUENCE, 2910
SUM_SURF_STRICTLY_INCREASING_SEQUENCE, 2914
SUM_SURF_SUMMIT, 2918
SUM_SURF_VALLEY, 2924
SUM_SURF_ZIGZAG, 2928
SUM_WIDTH_DECREASING_SEQUENCE, 2936, 3203, 3220, 3235, 3254, 3273, 3293, 3316, 3333, 3349, 3366, 3382, 3402, 3420, 3435, 3448, 3463, 3478, 3491, 3507, 3525, 3538, 3551, 3561, 3571, 3584–3598
SUM_WIDTH_DECREASING_TERRACE, 2942, 3220, 3236, 3255, 3274, 3294, 3316, 3333, 3350, 3367, 3383, 3403, 3421, 3435, 3449, 3464, 3479, 3492, 3508, 3526, 3539, 3551, 3562, 3572, 3584, 3599–3612
SUM_WIDTH_GORGE, 2948, 3220, 3236, 3256, 3274, 3295, 3317, 3334, 3350, 3367, 3384, 3403, 3421, 3436, 3449, 3464, 3479, 3493, 3509, 3527, 3539, 3552, 3562, 3573, 3585, 3599, 3613–3623
SUM_WIDTH_INCREASING_SEQUENCE, 2956, 3220, 3237, 3256, 3275, 3296, 3317, 3334, 3351, 3368, 3385, 3404, 3422, 3436, 3450, 3465, 3480, 3494, 3510, 3527, 3540, 3552, 3563, 3573, 3586, 3600, 3613, 3624–3635
SUM_WIDTH_INCREASING_TERRACE, 2962, 3220, 3237, 3257, 3275, 3297, 3318, 3335, 3352, 3368, 3386, 3405, 3422, 3437, 3450, 3466, 3481, 3495, 3511, 3528, 3541, 3553, 3563, 3574, 3587, 3601, 3614, 3624, 3636–3646
SUM_WIDTH_INFLEXION, 2968, 3221, 3238, 3257, 3276, 3298, 3318, 3335, 3353, 3369, 3387, 3405, 3423, 3437, 3451, 3466, 3481, 3496, 3512, 3529, 3542, 3553, 3564, 3574, 3588, 3602, 3615, 3625, 3636, 3646–3652
SUM_WIDTH_PEAK, 2974, 3221, 3238, 3258, 3276, 3299, 3319, 3336, 3353, 3369, 3388, 3406, 3423, 3438, 3451, 3467, 3482, 3496, 3513, 3529, 3542, 3554, 3564, 3575, 3589, 3603, 3615, 3626, 3637, 3646, 3653–3660
SUM_WIDTH_PLAIN, 2980, 3221, 3239, 3258, 3277, 3300, 3319, 3336, 3354, 3370, 3389, 3407, 3424, 3438, 3452, 3467, 3482, 3497, 3514, 3530, 3543, 3554, 3565, 3576, 3589, 3604, 3616, 3626, 3638, 3647, 3653, 3660–3666
SUM_WIDTH_PLATEAU, 2986, 3204, 3221, 3239, 3259, 3277, 3300, 3320, 3337, 3354, 3370, 3389, 3408, 3424, 3439, 3452, 3468, 3483, 3498, 3515, 3530, 3543, 3555, 3565, 3576, 3590, 3604, 3616, 3627, 3638, 3647, 3653, 3660, 3667–3671
SUM_WIDTH_PROPER_PLAIN, 2992, 3221, 3240, 3259, 3278, 3301, 3320, 3337, 3355, 3371, 3390, 3409, 3425, 3439, 3453, 3468, 3484, 3499, 3516, 3531, 3544, 3556, 3566, 3577, 3591, 3605, 3617, 3628, 3639, 3648, 3654, 3661, 3667, 3672–3677
SUM_WIDTH_PROPER_PLATEAU, 2998, 3221, 3240, 3260, 3279, 3301, 3321, 3338, 3356, 3372, 3390, 3410, 3426, 3440, 3454, 3469, 3485, 3500, 3517, 3532, 3545, 3556, 3567, 3578, 3592, 3606, 3618, 3629, 3640, 3648, 3654, 3662, 3667, 3672, 3678–3682
SUM_WIDTH_STEADY_SEQUENCE, 3004, 3222, 3241, 3260, 3279, 3302, 3321, 3338, 3356, 3372, 3391, 3410, 3426, 3440, 3454, 3470, 3485, 3501, 3518, 3533, 3546, 3557, 3567, 3579, 3593, 3607, 3618, 3630, 3641, 3649, 3655, 3663, 3668, 3673, 3678,
INDEX

3683–3687
SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE, \textbf{3010}, 3203, 3204, 3222, 3241, 3261, 3280, 3303, 3322, 3339, 3357, 3373, 3391, 3411, 3427, 3441, 3455, 3470, 3486, 3501, 3519, 3534, 3547, 3557, 3568, 3580, 3594, 3608, 3619, 3631, 3642, 3650, 3656, 3664, 3669, 3674, 3679, 3683, 3688–3691

SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE, \textbf{3016}, 3222, 3242, 3262, 3280, 3304, 3323, 3339, 3357, 3373, 3392, 3412, 3427, 3441, 3455, 3471, 3486, 3502, 3520, 3535, 3548, 3558, 3568, 3581, 3595, 3609, 3620, 3632, 3643, 3651, 3657, 3664, 3669, 3675, 3680, 3684, 3688, 3692–3694

SUM_WIDTH_SUMMIT, \textbf{3022}, 3203, 3222, 3242, 3262, 3281, 3305, 3323, 3340, 3358, 3374, 3393, 3413, 3428, 3442, 3456, 3471, 3487, 3502, 3521, 3535, 3548, 3558, 3569, 3581, 3596, 3610, 3621, 3633, 3644, 3651, 3658, 3665, 3670, 3676, 3681, 3685, 3689, 3692, 3695, 3696

SUM_WIDTH_VALLEY, \textbf{3030}, 3222, 3243, 3263, 3281, 3306, 3324, 3340, 3358, 3374, 3394, 3413, 3428, 3442, 3456, 3472, 3487, 3503, 3522, 3536, 3549, 3559, 3569, 3582, 3597, 3611, 3622, 3634, 3645, 3652, 3659, 3665, 3670, 3676, 3682, 3686, 3690, 3693, 3695, 3697

SUM_WIDTH_ZIGZAG, \textbf{3036}, 3203, 3204, 3222, 3243, 3263, 3282, 3307, 3324, 3341, 3359, 3375, 3395, 3414, 3429, 3443, 3457, 3473, 3488, 3504, 3523, 3536, 3549, 3560, 3570, 3583, 3598, 3612, 3623, 3635, 3646, 3652, 3660, 3666, 3671, 3677, 3682, 3687, 3691, 3694, 3696, 3697

SURF_BUMP_ON_DECREASING_SEQUENCE, \textbf{3046}
SURF_DECREASING, \textbf{3050}
SURF_DECREASING_SEQUENCE, \textbf{3054}
SURF_DECREASING_TERRACE, \textbf{3058}
SURF_DIP_ON_INCREASING_SEQUENCE, \textbf{3062}
SURF_GORGE, \textbf{3066}
SURF_INCREASING, \textbf{3070}
SURF_INCREASING_SEQUENCE, \textbf{3074}
SURF_INCREASING_TERRACE, \textbf{3078}
SURF_INFLEXION, \textbf{3082}
SURF_PEAK, \textbf{3086}
SURF.PLAIN, \textbf{3090}
SURF_PLATEAU, \textbf{3094}
SURF_PROPER.PLAIN, \textbf{3098}
SURF_PROPER_PLATEAU, \textbf{3102}
SURF_STEADY, \textbf{3106}
SURF_STEADY_SEQUENCE, \textbf{3110}
SURF_STRICTLY_DECREASING_SEQUENCE, \textbf{3114}
SURF_STRICTLY_INCREASING_SEQUENCE, \textbf{3118}
SURF_SUMMIT, \textbf{3122}
SURF_VALLEY, \textbf{3126}
SURF_ZIGZAG, \textbf{3130}

W

WIDTH_DECREASING_SEQUENCE, \textbf{3134}
INDEX

WIDTH_DECREASING_TERRACE, 3138
WIDTH_GORGE, 3142
WIDTH_INCREASING_SEQUENCE, 3146
WIDTH_INCREASING_TERRACE, 3150
WIDTH_INFLEXION, 3154
WIDTH_PEAK, 3158
WIDTHPLAIN, 3162
WIDTH_PLATEAU, 3166
WIDTH_PROPER_PLAIN, 3170
WIDTH_PROPER_PLATEAU, 3174
WIDTH_STEADY_SEQUENCE, 3178
WIDTH_STRICTLY_DECREASING_SEQUENCE, 3182
WIDTH_STRICTLY_INCREASING_SEQUENCE, 3186
WIDTH_SUMMIT, 3190
WIDTH_VALLEY, 3194
WIDTH_ZIGZAG, 3198