



**HAL**  
open science

# An $O(n^2)$ time algorithm for the minimal permutation completion problem

Christophe Crespelle, Anthony Perez, Ioan Todinca

## ► To cite this version:

Christophe Crespelle, Anthony Perez, Ioan Todinca. An  $O(n^2)$  time algorithm for the minimal permutation completion problem. *Discrete Applied Mathematics*, 2019, 254, pp.80-95. 10.1016/j.dam.2018.06.036 . hal-01969498

**HAL Id: hal-01969498**

**<https://inria.hal.science/hal-01969498>**

Submitted on 21 Oct 2021

**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.



Distributed under a Creative Commons Attribution - NonCommercial 4.0 International License

# An $O(n^2)$ time Algorithm for the Minimal Permutation Completion Problem<sup>\*</sup>

Christophe Crespelle<sup>1</sup>, Anthony Perez<sup>2</sup>, and Ioan Todinca<sup>2</sup>

<sup>1</sup> Université Claude Bernard Lyon 1 and CNRS, DANTE/INRIA, LIP UMR CNRS 5668,  
ENS de Lyon, Université de Lyon, [christophe.crespelle@inria.fr](mailto:christophe.crespelle@inria.fr)  
<sup>2</sup> LIFO, Université d'Orleans, BP 6759, F-45067 Orleans Cedex 2,  
[ioan.todinca](mailto:ioan.todinca@univ-orleans.fr), [anthony.perez](mailto:anthony.perez@univ-orleans.fr)

**Abstract.** In the MINIMAL PERMUTATION COMPLETION problem, one is given an arbitrary graph  $G = (V, E)$  and the aim is to find a permutation super-graph  $H = (V, F)$  defined on the same vertex set and such that  $F \supseteq E$  is inclusion-minimal among all possibilities. The graph  $H$  is then called a *minimal permutation completion* of  $G$ . We provide an  $O(n^2)$  incremental algorithm computing such a minimal permutation completion. To the best of our knowledge, this result leads to the first polynomial algorithm for this problem.

## 1 Introduction

In graph modification problems, we are given an arbitrary input graph and the goal is to transform it into a graph satisfying some property  $\Pi$  using a small number of modifications. Typically, modifications consist in adding and/or removing edges and/or vertices. Here we restrict to the case where we are only allowed to add edges to the input graph  $G = (V, E)$ , transforming it into a super-graph  $H = (V, F)$  that belongs to some target class of graphs. Probably the most famous problem of this kind is MINIMUM FILL-IN, where the goal is to add as few edges as possible in order to obtain a *chordal* graph  $H$ . A graph is chordal if it has no induced cycles with four or more vertices. The problem being NP-hard [18], it triggered the attention to a simpler one, where we are only required to compute an inclusion-minimal chordal supergraph  $H$  of  $G$ . Such a graph is called a *minimal triangulation* of  $G$ , and the MINIMAL TRIANGULATION problem has been known to be polynomial since 1976 [15, 17]. The problem can be solved in  $O(nm)$  time [17], and when the graph is dense the current best algorithm is the one of Heggernes et al. [8]. A detailed survey on this problem is provided in [6]. Minimal completions into other graph classes have been intensively studied. There are polynomial algorithms computing minimal completions into *interval graphs* [5, 14], *proper interval graphs* [16], *split graphs* [9], *cographs* [10] and *comparability graphs* [7]. Note that the *minimum* versions of these problems are NP-complete (see e.g. the thesis of Mancini [11] for further discussion and references).

In this paper we consider the MINIMAL PERMUTATION COMPLETION problem. A graph is a permutation graph if we can assign to each vertex a segment having an endpoint on a “top” line and the other on a parallel “bottom” line, such that two vertices are adjacent if and only if the two corresponding segments intersect. Such a representation is called a *permutation model* of the graph. We give an  $O(n^2)$  time algorithm computing a minimal permutation completion of an arbitrary graph. To the best of our knowledge, this is the first polynomial algorithm for the problem. Let us point out that computing a permutation completion with a minimum number of edges is NP-hard [2]. Our result is based on a vertex-incremental approach, also used for other types of completions. More specifically, we take the vertices of the input graph one by one in an arbitrary order, and at each step we add the new vertex  $x_i$  to the previously computed minimal permutation completion  $H_{i-1}$ . The new minimal permutation completion  $H_i$  is obtained by adding only edges between  $x_i$  and the rest of the graph. Very informally, if we are given a permutation model of  $H_{i-1}$ , we need to insert segment  $x_i$  in a minimal way. In Section 3 we consider the case when  $H_{i-1}$  has a unique permutation model and we provide an  $O(n)$  time computation of such an insertion position. Somehow surprisingly, even this task is non-trivial (the similar algorithm is very simple in the case of minimal interval completions [5]). Then we need to take into account the fact that  $H_{i-1}$  may have many different models (Section 4). Fortunately they are all encoded in its *modular decomposition*. Eventually, we compute the modular decomposition of the new completion  $H_i$ . This is done thanks to the algorithm of Crespelle and Paul [3], which incrementally maintains the modular decomposition of permutation graphs.

<sup>\*</sup> A preliminary extended abstract of this paper appeared as [4] in the Proceedings of WG 2015.

## 2 Preliminaries

Every graph  $G = (V, E)$  considered here will be finite, undirected, loopless and without multiple edges. We denote  $V(G)$  the set of vertices of  $G$  and  $n = |V(G)|$ . The edge between vertices  $x$  and  $y$  will arbitrarily be denoted either  $xy$  or  $yx$ , and the neighbourhood of a vertex  $x \in V$  is denoted  $N(x)$ . For a subset  $S \subseteq V$  of vertices, we denote by  $N_G(S) = \bigcup_{x \in S} N(x) \setminus S$  its neighbourhood, and by  $G[S]$  the subgraph of  $G$  induced by  $S$ , i.e.  $G[S] = (S, E_S)$  with  $E_S = \{xy \in E : x \in S \text{ and } y \in S\}$ .

**Permutation model.** A graph  $G = (V, E)$  is a *permutation graph* if and only if it admits a *permutation model*  $(\pi_1, \pi_2)$ , i.e., two one-to-one mappings  $\pi_1, \pi_2 : V \rightarrow \{1, \dots, n\}$  such that two vertices  $x$  and  $y$  are adjacent in  $G$  if and only if  $(\pi_1(x) - \pi_1(y))(\pi_2(x) - \pi_2(y)) < 0$ . An equivalent geometric definition of a permutation model for  $G$  is to associate to each vertex  $x$  a segment having one endpoint on a *top line* and the other on a parallel *bottom line*, all endpoints being pairwise distinct. In such a model two vertices  $x$  and  $y$  are adjacent if and only if their corresponding segments intersect. The correspondence between these two definitions is that the order in which the endpoints appear on the top (resp. bottom) line of the permutation model is the order defined by mapping  $\pi_1$  (resp.  $\pi_2$ ). We equally use these two visions in the rest of the article, depending on which one is more convenient for our purpose. Note that a permutation model can be encoded by storing mappings  $\pi_1$  and  $\pi_2$  as arrays, which uses  $O(n)$  space under the usual assumption that integers smaller than  $n$  are stored in constant space. Then, the condition for  $x$  and  $y$  to be adjacent can be tested in  $O(1)$  time by two comparisons of integers.

**Permutation completion.** Let  $G = (V, E)$  be an arbitrary graph. A *permutation completion* of  $G$  is a permutation graph  $H = (V, F)$ , on the same vertex set, such that  $E \subseteq F$ . If, moreover, set  $F$  is inclusion-minimal under these constraints, we say that  $H$  is a *minimal permutation completion* of  $G$ . In this paper we only deal with permutation completions. Hence, we will sometimes omit the term “permutation” and simply refer to them as (minimal) completions. Note that every graph  $G$  has a permutation completion: one can simply add all the missing edges to  $G$  and observe that the complete graph is a permutation graph. Permutation graphs are also *hereditary*, i.e. an induced subgraph of a permutation graph is also a permutation graph.

Our approach for computing a minimal completion of an arbitrary graph  $G$  is incremental. More precisely, we take the vertices of  $G$  one by one in an arbitrary order  $(x_1, \dots, x_n)$ , and at step  $i$  we compute a minimal completion  $H_i$  of  $G_i = G[\{x_1, \dots, x_i\}]$  from a minimal completion  $H_{i-1}$  of  $G_{i-1}$ . Notice that  $H_0$  is an empty graph and that  $H_n$  will turn out to be a permutation completion of  $G_n = G$ . This is done by adding edges incident to  $x_i$  only. This is possible thanks to the following observation that is general to all hereditary graph classes that are also stable by addition of a universal vertex, including permutation graphs.

**Lemma 1 (see e.g. [14]).** *Let  $G$  be an arbitrary graph and let  $H$  be a minimal permutation completion of  $G$ . Consider a new graph  $G' = G + x$  obtained by adding to  $G$  a new vertex  $x$  adjacent to an arbitrary set  $N(x)$  of vertices of  $G$ . There is a minimal permutation completion  $H'$  of  $G'$  such that  $H' - x = H$ .*

**Filling vertices.** Following notations of Lemma 1, for any subset  $W \subseteq V(G)$  of vertices, we say that we *fill*  $W$  in  $H'$  if we make all the vertices of  $W \setminus N(x)$  adjacent to  $x$  in the completion  $H'$  of  $G + x$ .

**Modular decomposition.** We now give some known definitions and results on modular decomposition (see [12, 13] for surveys). A *module* of a graph  $G = (V, E)$  is a subset of vertices  $M \subseteq V$  such that for any vertex  $y \notin M$ , the set  $M$  is either contained in the neighbourhood of  $y$  or  $M$  does not intersect the neighbourhood of  $y$ . Note that  $V$  and the singletons  $\{y\}, y \in V$  are modules, namely the *trivial modules*. A graph is *prime* if it contains only trivial modules. Let  $\mathcal{P} = \{M_1, \dots, M_p\}$  be a partition of  $V$  into modules of  $G$ . We denote by  $G/\mathcal{P}$  the *quotient graph* obtained by shrinking each module  $M_i$  into a unique vertex  $a_i$ . Formally,  $G/\mathcal{P}$  has vertex set  $\{a_1, \dots, a_p\}$ , and two vertices  $a_i$  and  $a_j$  are adjacent in  $G/\mathcal{P}$  if and only if there are two vertices  $b_i \in M_i$  and  $b_j \in M_j$  adjacent in  $G$ . The *modular decomposition tree*  $T$  (or simply *modular decomposition*) of a graph  $G = (V, E)$  is recursively defined as follows. If

90  $G$  has a unique vertex  $\{y\}$ , then  $T$  has a unique vertex (a leaf) labelled  $y$ . Let  $\mathcal{P} = \{M_1, \dots, M_p\}$  be a partition of  $V$  90  
 91 into modules, defined as follows: 91

- 92 1. If  $G$  is not connected, then each  $M_i$  is the vertex set of a connected component of  $G$ . 92
- 93 2. If  $G$  is connected but its complement  $\bar{G}$  is not connected, then each  $M_i$  is the vertex set of a connected component 93  
 94 of  $\bar{G}$ . 94
- 95 3. Otherwise,  $\mathcal{P}$  is the unique partition of  $V$  into maximal modules strictly contained in  $V$ . 95

96 The root  $r$  of  $T$  is labelled *parallel* in the first case, *series* in the second case and *prime* in the third case. A node 96  
 97  $u$  of the modular decomposition tree has  $k$  children  $v_1, \dots, v_k$ , each  $v_i$  being the root of the modular decomposition 97  
 98 tree of  $G[M_i]$ . We also associate to node  $u$  the quotient graph  $P_u = G[\cup_{1 \leq i \leq k} M_i] / \mathcal{P}$ ; for simplicity, we consider that 98  
 99 the vertex set of  $P_u$  is exactly  $\{v_1, \dots, v_k\}$ . Note that when  $u$  is a series node, graph  $P_u$  is a complete graph, when  $u$  99  
 100 is a parallel node graph  $P_u$  is an independent set and when  $u$  is a prime node, graph  $P_u$  is prime. Note also that  $P_u$  is 100  
 101 always a subgraph of  $G$ . Consequently, when  $G$  is a permutation graph, so is  $P_u$ . Hence in the modular decomposition 101  
 102 of  $G$  we also store, for each prime node  $u$ , a permutation model for the quotient graph  $P_u$ . It is well known that a 102  
 103 prime permutation graph admits a unique permutation model up to symmetries [12] (exchanging the two orders of the 103  
 104 model or reversing both orders). 104

105 **The new problem.** From now on, we consider the following problem, with slightly modified notations. Let  $G$  be a 105  
 106 permutation graph, and let  $G + x$  be the graph obtained by inserting to  $G$  a new vertex  $x$  adjacent to some set  $N(x)$  106  
 107 of vertices of  $G$ . Our goal is to compute a minimal permutation completion  $H$  of  $G + x$  such that  $H - x = G$ . 107  
 108 We will solve this problem in  $O(n)$  time, where  $n$  is the number of vertices of  $G$ . As we shall see, graph  $G$  will be 108  
 109 given together with its *modular decomposition*, which encodes all its possible permutation models. Our algorithm will 109  
 110 compute the set  $N'(x) \supseteq N(x)$  of neighbours of  $x$  in graph  $H$ , and will update this data structure for the completion 110  
 111  $H$ , thanks to the algorithm of [3]. 111

### 112 3 Minimal completion respecting a permutation model 112

113 **Definition 1 (Minimal completion respecting a permutation model).** Let  $(\pi_1, \pi_2)$  be a permutation model of a 113  
 114 permutation graph  $G$  and  $x$  a vertex to insert in  $G$ . A permutation completion  $H$  of  $G + x$  respects  $(\pi_1, \pi_2)$  if there 114  
 115 exists a permutation model of  $H$  such that removing  $x$  from it results in  $(\pi_1, \pi_2)$ . Moreover, if  $H$  is inclusion minimal 115  
 116 among such completions, then  $H$  is called a minimal permutation completion respecting  $(\pi_1, \pi_2)$ . 116

117 Notice that such a completion always exists: starting from  $(\pi_1, \pi_2)$  and making  $x$  adjacent to every vertex of  $G$  117  
 118 yields a permutation completion  $H$  respecting  $(\pi_1, \pi_2)$ . The aim of this section is to provide an  $O(n)$  time algorithm 118  
 119 computing a minimal permutation completion respecting a given permutation model of a graph  $G$ . This does not pro- 119  
 120 vide in general a minimal completion of  $G$  since a permutation graph may admit many different permutation models. 120  
 121 Nevertheless, such a completion is minimal when graph  $G$  is prime (in which case it admits a unique permutation 121  
 122 model up to symmetries — see Section 2 for definitions). This is the only case where we will use this approach in our 122  
 123 global algorithm. However, the algorithm we design in this section is not specific to prime graphs and we present it in 123  
 124 the general setting. In the remaining of this section, we use  $(\pi_1, \pi_2)$  to denote the given permutation model of graph 124  
 125  $G$ . 125

#### 126 3.1 Combinatorial characterisation 126

127 Given a permutation graph  $G$ , a vertex  $x$  to insert in  $G$  and a permutation completion  $H$  of  $G + x$ , we want to obtain 127  
 128 a permutation model of  $H$  respecting  $(\pi_1, \pi_2)$ . To that aim, we extend the definition of mappings  $\pi_1$  and  $\pi_2$ , initially 128  
 129 defined from  $V(G)$  to  $\{1, \dots, n\}$ , to vertex  $x$  as well, in the following way. 129

130 **Definition 2 (Insertion position).** An insertion position of  $x$  in  $(\pi_1, \pi_2)$  is a couple  $(\pi_1(x), \pi_2(x)) = (i + 0.5, j + 0.5)$  130  
 131 for some integers  $i, j \in \{0, \dots, n\}$ . 131

132  
133  
134  
135  
136  
137

Observe that a necessary and sufficient condition for insertion position  $(\pi_1(x), \pi_2(x))$  to give a permutation completion of  $G + x$  is that  $\{z \in N(x) \mid \pi_1(z) < \pi_1(x)\} = \{t \in N(x) \mid \pi_2(t) > \pi_2(x)\}$ . We now characterise all such insertion positions of  $x$  by a family of *slots*  $(I_j, J_j)$ , i.e. couples of intervals such that a position  $Pos = (p_1, p_2)$  can be assigned to  $x$  in order to obtain a completion of  $G + x$  if and only if  $(p_1, p_2)$  is contained in one of the slots  $(I_j, J_j)$ . Moreover, the intervals  $I_j$  (resp.  $J_j$ ) will be pairwise disjoint. To this purpose, we adopt the following notations, see their illustration on Figure 1.

132  
133  
134  
135  
136  
137

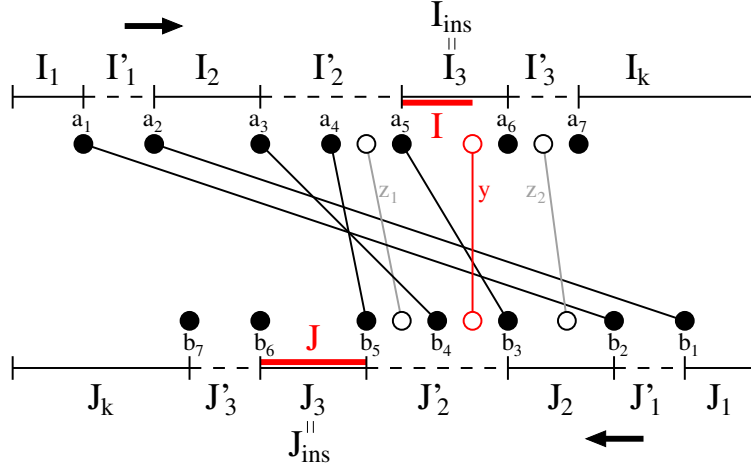


Fig. 1. Illustration of the valid insertion slots  $(I_i, J_i)$ .

138  
139  
140  
141  
142  
143  
144  
145

Let  $l = |N(x)|$ ,  $a_1, a_2, \dots, a_l$  denote the neighbours of  $x$  in **increasing** order in  $\pi_1$ , i.e.  $\pi_1(a_1) < \pi_1(a_2) < \dots < \pi_1(a_l)$ , and  $b_1, b_2, \dots, b_l$  denote the neighbours of  $x$  in **decreasing** order in  $\pi_2$ , i.e.  $\pi_2(b_l) < \dots < \pi_2(b_2) < \pi_2(b_1)$ . For  $1 \leq i \leq l-1$ , we also denote  $A_i = \{a_1, \dots, a_i\}$  and  $B_i = \{b_1, \dots, b_i\}$ . We define  $Valid = \{i \in \{1, \dots, l-1\} \mid A_i = B_i\}$ . Let  $k = |Valid| + 2$  and  $v_2, v_3, \dots, v_{k-1}$  denote the elements of  $Valid$  in increasing order. Then, for any  $j \in \{2, \dots, k-1\}$ , let  $I_j = [\pi_1(a_{v_j}), \pi_1(a_{v_j+1})]$  and  $J_j = [\pi_2(b_{v_j+1}), \pi_2(b_{v_j})]$  (see. Figure 1). We also denote  $I_1 = [0, \pi_1(a_1)]$ ,  $J_1 = [\pi_2(b_1), n+1]$ ,  $I_k = [\pi_1(a_l), n+1]$  and  $J_k = [0, \pi_2(b_l)]$ . Note that the intervals  $I_j$ ,  $1 \leq j \leq k$ , are pairwise disjoint and numbered from left to right in  $\pi_1$ , while the intervals  $J_j$  are pairwise disjoint and numbered from right to left in  $\pi_2$ .

146

147  
148  
149  
150  
151  
152  
153  
154

In order to define a completion of  $G + x$ , an insertion position must make  $x$  adjacent to at least all the vertices of  $N(x)$ . The insertion positions  $(\pi_1(x), \pi_2(x))$  that define a completion  $H$  of  $G + x$  are thus exactly the insertion positions such that  $\pi_1(x) \in I_i$  and  $\pi_2(x) \in J_i$  for some  $i \in \{1, \dots, k\}$ . Our next goal is to choose among all slots  $(I_i, J_i)$  an insertion slot  $(I_{ins}, J_{ins})$  that will provide an optimal insertion position (Definition 4). Then we prove through a sequence of lemmas that this slot contains indeed a position providing a minimal permutation completion respecting our permutation model (Lemma 2). Given an interval  $I$ , let  $l(I)$  (resp.  $r(I)$ ) denote the left endpoint (resp. right endpoint) of  $I$ . For any  $i \in \{1, \dots, k-1\}$ , we define  $I'_i = [r(I_i), l(I_{i+1})]$  in  $\pi_1$  (see Figure 1), and  $I'_k = \emptyset$ . Symmetrically, we define  $J'_i = [r(J_{i+1}), l(J_i)]$  in  $\pi_2$ , and  $J'_k = \emptyset$ . Finally, we denote  $\hat{I}_i = I_i \cup I'_i$  and  $\hat{J}_i = J_i \cup J'_i$ .

155  
156  
157

**Definition 3 (Forced, forwarding).** A vertex  $z$  is forced if  $z \in N(x)$  or there exists  $i \in \{1, \dots, k\}$  such that  $\pi_1(z) \in I'_i$  and  $\pi_2(z) \in J'_i$ . We say that  $i \in \{1, \dots, k\}$  is forwarding if all vertices  $y$  such that  $\pi_1(y) \in \bigcup_{1 \leq j \leq i} \hat{I}_j$  and  $\pi_2(y) \in \bigcup_{1 \leq j \leq i} \hat{J}_j$  are forced.

158  
159  
160

Note that forced vertices are adjacent to  $x$  in any completion of  $G + x$  respecting  $(\pi_1, \pi_2)$ . It may happen that all  $i \in \{1, \dots, k\}$  are forwarding; in this case,  $x$  is adjacent to all the vertices of  $G$  in any completion respecting  $(\pi_1, \pi_2)$ . Consequently, the unique minimal such completion is easily obtained by inserting  $x$  in any arbitrary slot  $(I_i, J_i)$ , with

161  $i \in \{i, \dots, k\}$ . In the following, we do not consider this case anymore and we assume that there exists a slot  $i$  which  
 162 is not forwarding. 162

163 **Definition 4 (Insertion slot and vertex  $y$ ).** The insertion slot  $(I_{ins}, J_{ins})$  is defined by  $ins = \min\{i \in \{1, \dots, k\} \mid$   
 164  $i \text{ is not forwarding}\}$ . Moreover, we denote by  $y$  a non-forced vertex that makes  $ins$  not forwarding. 164

165 In the remaining of this section,  $y$  will be used to denote a non-forced vertex that makes  $ins$  not forwarding. By  
 166 the previous observations, the insertion slot  $ins$  and the vertex  $y$  are well defined. Observe that, by minimality of  $ins$ ,  
 167 we either have  $\pi_1(y) \in \hat{I}_{ins}$  and  $\pi_2(y) \in \bigcup_{1 \leq j \leq ins} \hat{J}_j$ , or  $\pi_2(y) \in \hat{J}_{ins}$  and  $\pi_1(y) \in \bigcup_{1 \leq j \leq ins} \hat{I}_j$ .  
 168 We make some straightforward yet crucial observations about the slot  $(I_{ins}, J_{ins})$ , that will allow us to prove that one  
 169 can find a minimal completion by inserting  $x$  in this slot. 169

170 **Observation 1** There is a completion obtained by inserting  $x$  in the slot  $(I_{ins}, J_{ins})$  such that  $x$  is not adjacent to  $y$ .  
 171 This holds exactly for insertion positions in a sub-slot  $(I, J) \subseteq (I_{ins}, J_{ins})$  defined as follows. Denote  $I_y^- = [0, \pi_1(y)]$ ,  
 172  $I_y^+ = [\pi_1(y), n + 1]$ ,  $J_y^- = [0, \pi_2(y)]$  and  $J_y^+ = [\pi_2(y), n + 1]$ . We distinguish two cases: 172

- 173 1. if  $\pi_1(y) \in I_{ins}$  or  $\pi_2(y) \in J_{ins}$  but not both, then there exists a unique  $\alpha \in \{-, +\}$  s.t.  $I_{ins} \cap I_y^\alpha \neq \emptyset$  and  
 174  $J_{ins} \cap J_y^\alpha \neq \emptyset$ , and we set  $(I, J) = (I_{ins} \cap I_y^\alpha, J_{ins} \cap J_y^\alpha)$ ;
- 175 2. otherwise,  $(\pi_1(y), \pi_2(y)) \in (I_{ins}, J_{ins})$  and then there are two suitable slots  $(I, J) = (I_{ins} \cap I_y^-, J_{ins} \cap J_y^-)$   
 176 and  $(I, J) = (I_{ins} \cap I_y^+, J_{ins} \cap J_y^+)$ . 176

177 **Lemma 2.** Let  $C$  be a completion obtained by inserting  $x$  in the slot  $(I_{ins}, J_{ins})$  such that  $x$  is non adjacent to  $y$  in the  
 178 completion, and  $C$  is minimal among such completions. Then  $C$  is a minimal completion of  $G + x$  respecting  $(\pi_1, \pi_2)$ . 178

179 **Proof.** Observe first that such a completion  $C$  exists by Observation 1. Remind that all the completions respecting  
 180  $(\pi_1, \pi_2)$  are obtained by inserting  $x$  in some slot  $(I_i, J_i)$ , with  $1 \leq i \leq k$ . Moreover, if  $i > ins$  it is straightforward  
 181 to see that the completions obtained by inserting  $x$  in  $(I_i, J_i)$  make  $x$  adjacent to  $y$  and are therefore not subgraphs of  
 182 completion  $C$ . 182

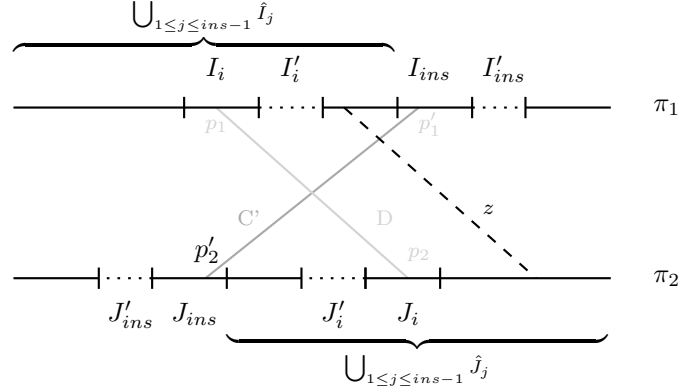
183 Now, let  $D$  be a completion obtained by inserting  $x$  at position  $(p_1, p_2)$  in  $(I_i, J_i)$  with  $i < ins$  and let us prove  
 184 that  $D$  is not a proper subgraph of  $C$ . Consider the completion  $C'$  obtained by inserting  $x$  at position  $(p'_1, p'_2) =$   
 185  $(l(I_{ins}) + 0.5, r(J_{ins}) - 0.5)$  in  $(I_{ins}, J_{ins})$ . We first prove that  $C'$  is contained in  $D$ . Assume for a contradiction  
 186 that there is a vertex  $z$  adjacent to  $x$  in  $C'$  but not in  $D$ . Then  $\pi_1(z)$  is between  $p_1$  and  $p'_1$  or  $\pi_2(z)$  is between  $p'_2$   
 187 and  $p_2$ . We can assume w.l.o.g. that we are in the first case (the other case is symmetrical). Since  $z$  is not adjacent to  
 188  $x$  in  $D$ , we have  $\pi_2(z) > p_2 > r(J'_i)$  (recall that  $\pi_1(z) > p_1 > r(I'_{i-1})$ ). It follows that  $\pi_1(z)$  (resp.  $\pi_2(z)$ ) does  
 189 not belong to  $\bigcup_{1 \leq j \leq i-1} I'_j$  (resp.  $\bigcup_{i \leq j \leq k} J'_j$ ). Since  $l(I_{ins}) \in N(x)$  by construction, it follows that  $\pi_1(z) < l(I_{ins})$ .  
 190 Moreover, since  $i < ins$  we have  $\pi_2(z) > r(J_{ins})$ . Hence, segment  $z$  has an endpoint in  $\bigcup_{j \leq ins-1} \hat{I}_j$  and the other  
 191 one in  $\bigcup_{j \leq ins-1} \hat{J}_j$  (see Figure 2). This implies that vertex  $z$ , which is not adjacent to  $x$  in  $D$ , is not forced. It follows  
 192 from Definition 3 that vertex  $z$  makes  $ins - 1$  non-forwarding, leading to a contradiction. 192

193 We can now finish the proof that  $D$  is not a proper subgraph of  $C$ . If  $x$  is adjacent to  $y$  in  $C'$ , then  $x$  is adjacent to  
 194  $y$  in  $D$  and clearly  $D$  is not a subgraph of  $C$ . If  $x$  is not adjacent to  $y$  in  $C'$ , then  $C'$  is not a strict subgraph of  $C$  (by  
 195 minimality of  $C$ ), thus  $D$ , which contains  $C'$ , is not a strict subgraph of  $C$ .  $\square$  195

196 Our next goal is to compute an optimal insertion position contained in a given slot  $(I, J)$ . We need the following  
 197 definition. 197

198 **Definition 5 ((Right-maximal) left-stable insertion position).** Let  $(I, J)$  be two intervals with endpoints in  $\{0, \dots, n +$   
 199  $1\}$ . An insertion position  $Pos = (p_1, p_2)$  for  $x$  in  $(I, J)$  is left-stable if all vertices  $z$  such that  $l(I) < \pi_1(z) < p_1$   
 200 (resp.  $l(J) < \pi_2(z) < p_2$ ) satisfy  $\pi_2(z) < p_2$  (resp.  $\pi_1(z) < p_1$ ).  
 201 If an insertion position  $Pos = (p_1, p_2)$  is left-stable and there is no left-stable insertion position  $Pos' = (p'_1, p'_2)$   
 202 strictly on its right in  $(I, J)$ , i.e. different from  $Pos$  and having  $p'_1 \geq p_1$  and  $p'_2 \geq p_2$ , we say that  $Pos$  is right-  
 203 maximal. 203

204 We now show that in any couple of intervals  $(I, J)$ , there exists a *unique* right-maximal left-stable position, called  
 205 the *rightmost left-stable insertion position*. 205



**Fig. 2.** Representation of the completions  $D$  and  $C'$  and of the respective position of the segments of  $x$  and  $z$  in such completions.

**Lemma 3 (Rightmost left-stable insertion position).** Given a couple of intervals  $(I, J)$ , there exists a unique right maximal left-stable insertion position  $Pos$  in  $(I, J)$ , called the rightmost left-stable insertion position.

**Proof.** First, note that insertion position  $(p_1, p_2) = (l(I) + 0.5, l(J) + 0.5)$  is left-stable. Now consider two left-stable insertion positions  $Pos = (p_1, p_2)$  and  $Pos' = (p'_1, p'_2)$  of  $(I, J)$  such that the segments defined by  $Pos$  and  $Pos'$  are crossing. Assume w.l.o.g. that  $p'_1 < p_1$  in  $\pi_1$  and  $p_2 < p'_2$  in  $\pi_2$ . We show that the insertion position  $Pos'' = (p_1, p'_2)$ , which is strictly on the right of both  $Pos$  and  $Pos'$  is left-stable too. Indeed, since  $Pos$  is left-stable, any vertex  $z$  such that  $\pi_1(z) < p_1$  satisfies  $\pi_2(z) < p_2$  and then  $\pi_2(z) < p'_2$ . Similarly, since  $Pos'$  is left-stable, any vertex  $z$  such that  $\pi_2(z) < p'_2$  satisfies  $\pi_1(z) < p'_1$  and then  $\pi_1(z) < p_1$ . This achieves to show that  $Pos''$  is left-stable. Consequently, two right-maximal left-stable insertion positions cannot cross each other and it follows that there is one unique such insertion position in  $(I, J)$ .  $\square$

**Lemma 4.** Let  $Pos = (p_1, p_2)$  be the rightmost left-stable insertion position in  $(I, J) \subseteq (I_{ins}, J_{ins})$  (Observation 1). The completion obtained by inserting  $x$  at position  $Pos$  is minimal among completions obtained by insertion of  $x$  in  $(I, J)$ .

**Proof.** Let us start by two basic properties of  $Pos$ . Let  $z$  be the vertex immediately on the right of  $p_1$  in  $\pi_1$ . If  $z \neq r(I_{ins})$  then  $\pi_2(z) > p_2$ , since otherwise incrementing  $p_1$  to place it just after  $z$  in  $\pi_1$  would give a left-stable insertion position strictly on the right of  $Pos$ , contradicting the right-maximality of  $Pos$ . Similarly, the vertex  $z$  immediately on the right of  $p_2$  in  $\pi_2$  satisfies either  $z = r(J_{ins})$  or  $\pi_1(z) > p_1$ . Since  $Pos$  is left-stable, we also have that the vertex  $z$  immediately on the left of  $p_1$  in  $\pi_1$  satisfies either  $z = l(I_{ins})$  or  $\pi_2(z) < p_2$ . Similarly, the vertex  $z$  immediately on the left of  $p_2$  in  $\pi_2$  satisfies either  $z = l(J_{ins})$  or  $\pi_1(z) < p_1$ .

Let us show that the completion obtained from  $Pos$  is minimal among the completions obtained from insertion positions in  $(I, J)$ . First, let  $Pos'$  be an insertion position in  $(I, J)$  such that segment  $Pos'$  crosses the one of  $Pos$ . If  $p'_1 < p_1$ , then, since  $Pos$  is left-stable and since  $Pos'$  crosses  $Pos$ , the vertex  $z$  which is immediately on the left of  $p_1$  in  $\pi_1$  is not adjacent to  $x$  w.r.t.  $Pos$ , while it is w.r.t.  $Pos'$ . Thus the completion defined by  $Pos'$  is not included in the one defined by  $Pos$ . The same arguments hold if  $p'_2 < p_2$ .

Let now  $Pos'$  be an insertion position strictly on the left of  $Pos$ . Assume there exists one vertex  $z$  adjacent to  $x$  w.r.t.  $Pos$  which is not adjacent to  $x$  w.r.t.  $Pos'$ . Then,  $z$  is between  $Pos$  and  $Pos'$  in one of  $\pi_1$  or  $\pi_2$ , and on the right of  $Pos$  in the other order. This violates the left-stability of  $Pos$  — a contradiction. Thus, the completion obtained by inserting  $x$  in  $Pos'$  is not properly included in the one where  $x$  is inserted in  $Pos$ . Let now  $Pos'$  be an insertion position strictly on the right of  $Pos$ . Assume that the completion obtained by inserting  $x$  at position  $Pos'$  is included in the one where  $x$  is inserted at position  $Pos$ . Then, all vertices between  $Pos$  and  $Pos'$  in  $\pi_1$  or in  $\pi_2$  must be on the left of  $Pos'$  in the other order, since otherwise their segments would cross  $Pos'$  and not  $Pos$ . Thus,  $Pos'$  is left-stable, which contradicts the right-maximality of  $Pos$ .

To conclude, by inserting  $x$  at position  $Pos$  we obtain a permutation completion of  $G + x$  that is minimal among completions obtained by insertion of  $x$  in the slot  $(I, J)$ .  $\square$

Assume there are two possible couples of subintervals of  $(I_{ins}, J_{ins})$  where  $x$  can be inserted without becoming adjacent to vertex  $y$  (see the second case of Observation 1). Lemma 5 below allows to choose one of these two possibilities such that the resulting completion is minimal.

**Lemma 5.** *Let  $(I_l, J_l)$  and  $(I_r, J_r)$  be two pairs of intervals such that  $I_l$  is on the left of  $I_r$  and  $J_l$  is on the left of  $J_r$ . Let  $Pos_l$  (resp.  $Pos_r$ ) be the rightmost left-stable insertion position in  $(I_l, J_l)$  (resp.  $(I_r, J_r)$ ). Let  $C_l$  (resp.  $C_r$ ) be the completion obtained by inserting  $x$  in position  $Pos_l$  (resp.  $Pos_r$ ).*

*If  $C_r$  is a subgraph of  $C_l$ , then  $C_r$  is a minimal completion among all completions obtained by inserting  $x$  in one of the slots  $(I_l, J_l)$  or  $(I_r, J_r)$ . Otherwise,  $C_l$  is minimal among all such completions.*

**Proof.** By Lemma 4,  $C_l$  (resp.  $C_r$ ) is minimal among all completions obtained by inserting  $x$  in the slot  $(I_l, J_l)$  (resp.  $(I_r, J_r)$ ). Therefore, if  $C_r$  is a subgraph of  $C_l$ , then  $C_r$  is minimal among all completions obtained by inserting  $x$  in one of the slots  $(I_l, J_l)$  or  $(I_r, J_r)$ . By symmetry, if  $C_l$  is a subgraph of  $C_r$ , then  $C_l$  is minimal among all such completions.

It remains to consider the case when  $C_l$  and  $C_r$  are incomparable w.r.t. the subgraph relation. In this case, one needs to show that  $C_l$  is minimal among the desired completions. By contradiction, assume there is a completion  $C$  strictly contained in  $C_l$ , obtained by inserting  $x$  at some position  $Pos$  contained in the slot  $(I_l, J_l)$  or  $(I_r, J_r)$ . Recall that  $Pos$  cannot be contained in  $(I_l, J_l)$  since  $Pos_l$  is rightmost left-stable for the couple of intervals  $(I_l, J_l)$ . Hence  $Pos$  is in the slot  $(I_r, J_r)$ . We will show that in this case  $C_r$  is a subgraph of  $C$ , contradicting the fact that  $C_r$  and  $C_l$  are incomparable.

First, we show that  $Pos = (p_1, p_2)$  is left-stable. Indeed, all vertices  $z$  such that  $l(I_r) < \pi_1(z) < p_1$  satisfy  $\pi_2(z) < p_2$ , since otherwise they would be adjacent to  $x$  w.r.t.  $Pos$  (in  $C$ ) and not w.r.t.  $Pos_l$  (in  $C_l$ ). For the same reason, all vertices  $z$  such that  $l(J_r) < \pi_2(z) < p_2$  satisfy  $\pi_1(z) < p_1$ . This shows that  $Pos$  is left-stable. Since  $Pos_r$  is rightmost left-stable either  $Pos = Pos_r$  or  $Pos_r$  is strictly on the right of  $Pos$ . In the first case,  $C$  and  $C_r$  are the same completions and thus the contradiction immediately follows. We now consider the second case.

Suppose for contradiction that there exists a vertex  $z$  adjacent to  $x$  in  $C_r$  but not in  $C$ , and let  $Pos_r = (p'_1, p'_2)$ . We have either  $p_1 < \pi_1(z) < p'_1$  and  $p'_2 < \pi_2(z)$ , or  $p_2 < \pi_2(z) < p'_2$  and  $p'_1 < \pi_1(z)$ . In both cases, this contradicts the left-stability of  $Pos_r$ . Thus, all vertices adjacent to  $x$  in  $C_r$  are also adjacent to  $x$  in  $C$ . Hence  $C_r$  is a subgraph of  $C$ , which is a subgraph of  $C_l$ —a contradiction.  $\square$

### 3.2 The algorithm

The aim of this section is to provide an algorithm that given a permutation graph  $G$  with model  $(\pi_1, \pi_2)$  and a vertex  $x$  to insert in  $G$  computes a minimal permutation completion  $H$  of  $G + x$  respecting  $(\pi_1, \pi_2)$  in  $O(n)$  time. Such an algorithm will moreover return a permutation model of  $H$ . Our algorithm is in two steps.

**First step.** The algorithm first determines the insertion slot  $(I_{ins}, J_{ins})$ . To that purpose we first compute the slots  $(I_i, J_i)$  defined above in  $O(n)$  time [3]. Then, by increasing  $i$ , we scan the vertices having one endpoint in the slot  $(I_i, J_i)$  until we find a non-forced vertex  $y$ , which can be tested in  $O(1)$  time. We then have  $ins = i$  and a suitable vertex  $y$  as in Definition 4. This takes  $O(n)$  time.

**Second step.** The algorithm then inserts  $x$  in  $(I_{ins}, J_{ins})$ . By Observation 1, either there is a unique slot  $(I, J)$  that contains the optimal insertion position, or there are two possible slots  $(I_l, J_l)$  and  $(I_r, J_r)$  such that the second one is on the right. In the first case, we find (Algorithm 1) the rightmost left-stable insertion position in  $(I, J)$ , which is minimal among the completions respecting the permutation model  $(\pi_1, \pi_2)$  (Lemma 4). In the second case, we compute the rightmost left-stable insertion positions  $Pos_l$  and  $Pos_r$  in each of the slots  $(I_l, J_l)$  and  $(I_r, J_r)$ . If the completion  $C_r$  obtained from  $Pos_r$  is included in the completion  $C_l$  obtained from  $Pos_l$ , then the algorithm returns the completion



285  $C = C_r$ ; otherwise the algorithm returns the completion  $C = C_l$ . Lemma 5 ensures that the completion  $C$  chosen in 285  
286 this way is minimal among the completions respecting  $(\pi_1, \pi_2)$ . Note that the inclusion of  $C_r$  into  $C_l$  can be tested 286  
287 in  $O(n)$  time: while scanning one order of the permutation model, one simply needs to check for each encountered 287  
288 vertex  $z$  that if segment  $z$  crosses the segment defined by  $Pos_r$ , then segment  $z$  also crosses the segment defined by 288  
289  $Pos_l$ . 289

290  
291 To conclude this section, we show how to compute the rightmost left-stable insertion position in a slot  $(I, J)$  in 291  
292  $O(n)$  time. This is done by Algorithm 1, whose correctness and time complexity is formally stated and proved by 292  
293 Lemma 6 below. 293

---

**Algorithm 1:** Rightmost(I,J): Computing the rightmost left-stable insertion position  $Pos$  in a pair  $(I, J)$  of intervals of  $(\pi_1, \pi_2)$ .

---

**Input:** The permutation model  $(\pi_1, \pi_2)$  of  $G$  and a pair  $(I, J)$  of its intervals

**Output:** The rightmost left-stable insertion position  $Pos$  in  $(I, J)$

```

1   $p_1 \leftarrow l(I) + 0.5;$ 
2   $p_2 \leftarrow l(J) + 0.5;$ 
3  Denote  $Pos = (p_1, p_2)$  and let  $f(p_1)$  (resp.  $f(p_2)$ ) be the vertex immediately on the right of  $p_1$  in  $\pi_1$  (resp. of  $p_2$  in  $\pi_2$ );
4   $old_1 \leftarrow p_1 - 1;$ 
5   $old_2 \leftarrow p_2 - 1;$ 
6  while  $(p_1, p_2) \neq (old_1, old_2)$  do
7      while  $f(p_1) \neq r(I)$  and  $\pi_2(f(p_1)) < p_2$  do
8           $p_1 \leftarrow p_1 + 1$ 
9      while  $f(p_2) \neq r(J)$  and  $\pi_1(f(p_2)) < p_1$  do
10          $p_2 \leftarrow p_2 + 1$ 
11      $(old_1, old_2) \leftarrow (p_1, p_2);$ 
12      $(q_1, q_2) \leftarrow (p_1 + 1, p_2 + 1);$ 
13     while  $(p_1, p_2) \in (I, J)$  and  $(p_1, p_2) \neq (q_1, q_2)$  do
14          $q'_2 \leftarrow \max\{\pi_2(z) \mid p_1 < \pi_1(z) < q_1\} + 0.5;$ 
15          $q'_2 \leftarrow \max\{q'_2, q_2\};$ 
16          $q'_1 \leftarrow \max\{\pi_1(z) \mid p_2 < \pi_2(z) < q_2\} + 0.5;$ 
17          $q'_1 \leftarrow \max\{q'_1, q_1\};$ 
18          $(p_1, p_2) \leftarrow (q_1, q_2);$ 
19          $(q_1, q_2) \leftarrow (q'_1, q'_2);$ 
20     if  $(p_1, p_2) \notin (I, J)$  then
21          $(p_1, p_2) \leftarrow (old_1, old_2)$ 
22 Return  $Pos = (p_1, p_2);$ 

```

---

294 **Lemma 6.** Given a permutation model  $(\pi_1, \pi_2)$  of a permutation graph  $G$  and a pair  $(I, J)$  of intervals of  $(\pi_1, \pi_2)$ , 294  
295 Algorithm 1 correctly outputs the rightmost left-stable insertion position  $Pos$  in  $(I, J)$  and runs in  $O(n)$  time. 295

296 **Proof.** For the running time, first note that  $p_1$  and  $p_2$  can only increase during the algorithm, except at Line 21 where 296  
297 they are reset to their previous values. But in this case, the algorithm immediately stops as the next test of the condition 297  
298 of the main loop fails (Line 6). The two loops of Lines 7 and 9 only shift  $p_1$  and  $p_2$  to the right, in constant time. So 298  
299 their total execution time along the algorithm is  $O(n)$ . The inner loop of Line 13 sweeps through  $\pi_1$  and  $\pi_2$  from left to 299  
300 right. Because  $p_1$  and  $p_2$  only increase, the intervals of  $(\pi_1, \pi_2)$  parsed during each execution of this loop are pairwise 300  
301 disjoint. Then, the total running time devoted to the execution of this loop during the whole algorithm is  $O(n)$  and so 301  
302 is the total complexity of Algorithm 1. 302  
303 303

We now prove the correctness. An invariant of our main loop is that position  $Pos = (p_1, p_2)$  is always left-stable. This is clear at the initialization (Line 1) and the left-stability is preserved during the two loops of Lines 7 and 9. Suppose that the value of  $Pos$  has been modified by the loop of Line 13. Assume for a contradiction that, at the end of this loop, there is a vertex  $z$  such that  $l(I) < \pi_1(z) < p_1$  but  $\pi_2(z) > p_2$  (the arguments for the symmetrical case  $l(J) < \pi_2(z) < p_2$  and  $\pi_1(z) > p_1$  are similar). The way  $(p_1, p_2)$  and  $(q_1, q_2)$  evolve in the loop of Line 13 ensures that, at the execution of Line 15 in some iteration of the loop, we have encountered vertex  $z$ . Moreover, at that moment  $q'_2$  has been set to a value of at least  $\pi_2(z)$ . Therefore, at the end of the loop of Line 13, we have  $p_2 \geq \pi_2(z)$  — a contradiction. Then, at the end of Algorithm 1  $Pos$  is left-stable.

We still have to show that the returned position  $Pos$  is right-maximal among the left-stable insertion positions (Definition 5). By contradiction, assume there is another left-stable insertion position  $Pos^* = (p_1^*, p_2^*) \in (I, J)$  strictly on the right of  $Pos$ . Among all such positions, we consider a leftmost one. The only reason for the algorithm to stop is that  $Pos$  is reset at Line 21 to the value it had at Line 11, that is after the execution of the two loops of Lines 7 and 9. If  $p_1^* = p_1$  then  $p_2^* > p_2$  and since  $Pos^*$  is left-stable, the vertex  $z$  which is immediately on the right of  $p_2$  in  $\pi_2$  satisfies  $\pi_1(z) < p_1$ . Consequently, the loop of Line 9 should have incremented  $p_2$  once more before it stops — a contradiction. On the other hand, if  $p_2^* = p_2$  then  $p_1^* > p_1$  and since  $Pos^*$  is left-stable, the vertex  $z$  which is immediately on the right of  $p_1$  in  $\pi_1$  satisfies  $\pi_2(z) < p_2$ . Moreover  $\pi_2(z)$  cannot be in the interval between the value of  $p_2$  before the loop of Line 9 starts and the value of  $p_2$  when it ends. Indeed, all vertices  $z'$  having one endpoint in this interval satisfy  $\pi_1(z') < p_1$ , while  $\pi_1(z) > p_1$ . It follows that  $\pi_2(z) < p_2$  before the loop of Line 9 starts, i.e. just after the loop of Line 7 stops. Since  $z = f(p_1)$ , the loop of Line 7 should have incremented  $p_1$  once more before it stops — a contradiction.

Thus, in the last iteration of the main loop, at Line 12 we have  $p_1^* \neq p_1$  and  $p_2^* \neq p_2$ ? This implies that position  $(p_1^*, p_2^*)$  is also on the right of  $(p_1 + 1, p_2 + 1)$ . As we noted before, the fact that our algorithm stopped and returned  $Pos$  at the end of this iteration of the main loop implies that  $Pos$  was reset at Line 21 with the value it had at Line 12. This means that in the execution of the loop of Line 13, the values of  $p_1$  or  $p_2$  went beyond the right bounds of  $I$  or  $J$  (Line 20). Notice that the execution of the loop of Line 13 started with  $(q_1, q_2) = (p_1 + 1, p_2 + 1)$ . Since position  $(p_1^*, p_2^*)$  is left-stable, for any vertex  $z$  such that  $p_1 < \pi_1(z) < p_1^*$  (resp. such that  $p_2 < \pi_2(z) < p_2^*$ ) we have that  $\pi_2(z) \leq p_2^*$  (resp.  $\pi_1(z) \leq p_1^*$ ). Therefore, the values of  $q'_1$  and  $q'_2$ , and consequently of  $p_1$  and  $p_2$ , are never set to values greater than  $p_1^*$  and  $p_2^*$  respectively (Lines 15 and 17). This contradicts the fact that the values of  $p_1, p_2$  went beyond the right bounds of  $I$  or  $J$ . Thus, there is no other left-stable insertion position on the right of the position  $Pos$  returned by the algorithm. This shows that position  $Pos$  is right-maximal among the left-stable insertion positions in  $(I, J)$  and concludes the correctness proof of Algorithm 1.  $\square$

We are now ready to state the main result of this section. Observe that the correctness of Theorem 1 directly follows from the correctness of Algorithm 1 and hence Lemma 6.

**Theorem 1.** *Given a permutation model  $(\pi_1, \pi_2)$  of a permutation graph  $G$  and a vertex  $x$  to insert in  $G$  together with its neighbourhood  $N(x)$ , a minimal permutation completion  $H$  of  $G + x$  respecting  $(\pi_1, \pi_2)$  can be computed in  $\mathcal{O}(n)$  time. The output is a permutation model of  $H$ .*

## 4 General minimal completion of $G + x$

For our general algorithm computing a minimal permutation completion of  $G + x$ , we consider the modular decomposition of  $G$  (see Section 2). For each node  $u$  of the modular decomposition tree  $T$ , we denote by  $V[u]$  the subset of vertices of  $G$  appearing in the subtree rooted in  $u$ , and by  $G[u]$  the subgraph of  $G$  induced by these vertices. We denote by  $G[u] + x$  the graph obtained from  $G[u]$  by adding  $x$  with neighbourhood  $N(x) \cap V[u]$ . Our algorithm operates in a bottom-up fashion on the modular decomposition tree  $T$  of  $G$ . For each node  $u$  with children  $v_1, \dots, v_k$ , we determine a minimal completion of  $G[u] + x$ , thanks to the minimal completions we already computed for  $G[v_i] + x$ . The algorithm ends with the computation of a minimal completion of  $G[r] + x = G + x$ , where  $r$  is the root of  $T$ . Before describing the algorithm, we first prove the combinatorial properties that will guarantee its validity. We start with some notations and definitions.

352 A subset  $W \subseteq V(G)$  of vertices is *hit* in  $G + x$  (resp. in a completion  $H$  of  $G + x$ ) if it intersects the neighbourhood 352  
353  $N(x)$  of  $x$  in  $G + x$  (resp. the neighbourhood  $N_H(x)$  of  $x$  in  $H$ ). If  $W$  is contained in  $N(x)$  (resp.  $N_H(x)$ ),  $W$  is *fully* 353  
354 *hit* in  $G + x$  (resp.  $H$ ). If  $W$  is hit but not fully hit, it is *partially hit*. A node  $u$  of the modular decomposition tree  $T$  of 354  
355  $G$  is hit, fully hit or partially hit according to the status of its associated set of vertices  $V[u]$ . When we omit to precise 355  
356 it, the graph referred to in these notions is  $G + x$ . Observe that the set of hit nodes of  $T$  can be computed in  $\mathcal{O}(n)$  356  
357 time by a bottom-up parsing of  $T$ . For each node  $u$  of  $T$ ,  $P_u = (V_u, E_u)$  denotes the corresponding quotient graph 357  
358 (see Section 2). If  $u$  is a *prime* node,  $P_u$  is prime, and stored together with its unique (up to symmetries) permutation 358  
359 model. If  $u$  is a *series* or *parallel* node, then graph  $P_u$  is a complete graph, respectively an independent set. 359  
360 In this section, it is more convenient to work with the geometric version of a permutation model. Each vertex is rep- 360  
361 resented by a segment having an extremity on the *top line* of the model and another one on the *bottom* line. By [1], 361  
362 any permutation model of  $G[u]$  is obtained from a model of  $P_u$  by *expanding each segment corresponding to a ver-* 362  
363 *tex  $v_i$  of  $P_u$  into a model of  $G[v_i]$*  as follows. segment  $v_i$  is enlarged into a parallelogram by enlarging its top (resp. 363  
364 bottom) endpoint into a top (resp. bottom) edge, lying on the top (resp. bottom) line of the permutation model. These 364  
365 expansions are such that the top edges (resp. bottom edges) of the parallelograms are pairwise disjoint. Moreover, they 365  
366 appear on the top (resp. bottom) line in the same order as the endpoints of the segments. Therefore, for any pair of 366  
367 vertices  $v_i$  and  $v_j$  of  $P_u$ , the corresponding parallelograms intersect if and only if  $v_i v_j$  is an edge of  $P_u$ . Now, for each 367  
368  $v_i$  (recall that  $v_i$  is a child of  $u$  in the modular decomposition tree), we insert a model of  $G[v_i]$  into the parallelogram 368  
369 of  $v_i$ . 369

#### 370 4.1 Combinatorial arguments 370

371 At each step of our algorithm, in order to compute a minimal completion of  $G[u] + x$ , we use the completions of 371  
372  $G[v_i] + x$  previously computed for each child  $v_i$  of  $u$ . We also take into account the new considered node  $u$  by using a 372  
373 completion of  $P_u + x$ . Note however that there is no canonical definition of what the graph  $P_u + x$  is. The following 373  
374 definition fills this gap in a way that suits our purpose. 374

375 **Definition 6 (Contracted graph  $P_u + x$ ).** *The contracted graph  $P_u + x$  is the graph obtained from  $P_u$  by adding the* 375  
376 *vertex  $x$  with neighbourhood  $N_{P_u+x}(x) = \{v_i \in V_u \mid v_i \text{ is hit}\}$ .* 376

377 In the remaining of this section, we let  $H_u$  be a minimal permutation completion of  $P_u + x$ . In order to obtain a 377  
378 permutation completion of  $G[u] + x$ , we could consider each node  $v_i$  adjacent to  $x$  in  $H_u$  and fill the set  $V[v_i]$  (i.e., 378  
379 make all vertices of  $V[v_i]$  adjacent to  $x$ ). It is not hard to see that this construction yields a permutation completion 379  
380 of  $G[u] + x$ . Indeed, take a permutation model  $H_u^{mod}$  of  $H_u$ , and expand for each node  $v_j$  of  $P_u$  segment  $v_j$  into a 380  
381 parallelogram (segment  $x$  remains unchanged). By inserting a model of  $G[v_j]$  into the parallelogram of  $v_j$ , we obtain 381  
382 a model for the completion of  $G[u] + x$ . Notice that such a completion is not necessarily minimal: in the construction 382  
383 above we can sometimes “shift” the top and/or bottom endpoint of segment  $x$  inside a parallelogram and save some 383  
384 edges in the completion. This can be done by avoiding to have segment  $x$  crossing some of the segments inside this 384  
385 parallelogram. As we shall see, there are at most two hit children  $v_i$  of  $u$  for which we do not fill  $V[v_i]$ . Depending on 385  
386 the cases, for such a child  $v_i$  we can either use any minimal permutation completion of  $G[v_i]$  or one which is minimal 386  
387 among those satisfying an additional constraint, called *external-minimal permutation completion* (Definition 7 below). 387  
388 This is the reason why our algorithm actually computes not one but two permutation completions of  $G[u] + x$  for each 388  
389 node  $u$  of the modular decomposition tree: one minimal permutation completion and one external-minimal permutation 389  
390 completion. 390

391 **Definition 7 (External(-minimal) permutation completion).** *A permutation completion  $H$  of  $G + x$  is an external* 391  
392 *permutation completion if  $H$  has a permutation model such that the bottom endpoint of segment  $x$  is the leftmost* 392  
393 *among all bottom endpoints of segments of the model. Moreover, if  $H$  is minimal among all external completions then* 393  
394  *$H$  is called an external-minimal permutation completion.* 394

395 Note that since any model can be reversed from left to right or upside-down, in the definition above we can replace 395  
396 “leftmost” by “rightmost” and “bottom” by “top”. 396

397 The notion of *representation* defined below is central in the remaining of the paper. We use it to construct a minimal 397  
398 completion of  $G[u] + x$  from a minimal completion of  $P_u + x$  and minimal and external-minimal completions of 398  
399  $G[v_i] + x$ , for all children  $v_i$  of  $u$ . 399

**Definition 8 (Representation).** Let  $H_u$  be a permutation completion of  $P_u + x$ . A representation  $\mathcal{R}$  of  $H_u$  is a triplet  $(H_u^{mod}, v_{top}, v_{bot})$  or a couple  $(H_u^{mod}, v_{top})$  or a single element  $H_u^{mod}$  such that:

- $H_u^{mod}$  is a permutation model of  $H_u$ , and
- if  $\mathcal{R}$  is a couple or a triplet, then  $v_{top}$  is a vertex of  $P_u$  adjacent to  $x$  in  $H_u$  such that the top endpoint of segment  $v_{top}$  is next to the top endpoint of segment  $x$  in  $H_u^{mod}$ .
- if  $\mathcal{R}$  is a triplet, then  $v_{bot}$  is a vertex of  $P_u$  adjacent to  $x$  in  $H_u$  such that the bottom endpoint of segment  $v_{bot}$  is next to the bottom endpoint of segment  $x$  in  $H_u^{mod}$ .

Observe that a single given permutation model  $H_u^{mod}$  may give rise to several distinct representations. These representations can possibly be of any form, that is a triplet, a couple or a singleton. Note also that when a representation is a triplet, then  $v_{top}$  and  $v_{bot}$  are not required to be distinct. Finally, observe that if there does not exist a vertex satisfying the conditions required for  $v_{top}$  in a permutation model  $H_u^{mod}$ , then  $H_u^{mod}$  cannot give rise to a representation that is a couple or a triplet. Nevertheless, if there exists a vertex satisfying the conditions required for  $v_{bot}$  in  $H_u^{mod}$ , then the model  $\check{H}_u^{mod}$  obtained by flipping the two orders of  $H_u^{mod}$  upside-down gives rise to a representation that is a couple.

Notice moreover that two vertices  $v_{top}^l$  and  $v_{top}^r$  can be candidates for  $v_{top}$  (respectively directly on the left and right of  $x$ ). Since the aim is to construct a minimal completion by shifting endpoints of segment  $x$  inside the parallelograms of  $v_{top}$  and/or  $v_{bot}$ , eventually saving some edges, the side chosen does not matter. Indeed, if we set  $v_{top} = v_{top}^l$  then segment  $x$  will intersect the parallelogram  $v_{top}^r$  and vice-versa. It follows that none of these permutation completions can be a strict subgraph of the other.

The following definition shows how to use a representation of  $H_u$  in order to get a permutation completion of  $G[u] + x$ .

**Definition 9 (Completion resulting from a representation).** Let  $H_u$  be a permutation completion of  $P_u + x$ . Consider a representation  $\mathcal{R}$  of  $H_u$  and assume that we are given, for each node  $v_\alpha \in \{v_{top}, v_{bot}\}$  (if they exist), the neighbourhood  $N'_\alpha$  (resp.  $N_\alpha^{ext}$ ) of  $x$  in a minimal (resp. an external-minimal) permutation completion of  $G[v_\alpha] + x$ . We construct a vertex set  $N'$  (initially empty) as follows. For each  $v_i \in V_u$  adjacent to  $x$  in  $H_u$  and distinct from  $v_{top}$  and  $v_{bot}$ , we add  $V[v_i]$  to  $N'$ . Then, if at least one of  $v_{top}$  and  $v_{bot}$  exists, we distinguish three cases:

1. if  $v_{bot}$  does not exist then we add  $N_{v_{top}}^{ext}$  to  $N'$ .
2. if both  $v_{top}$  and  $v_{bot}$  exist and  $v_{top} = v_{bot}$ , we add  $N'_{v_{top}}$  to  $N'$ .
3. if both  $v_{top}$  and  $v_{bot}$  exist and are not equal, for each  $v_\alpha \in \{v_{top}, v_{bot}\}$ , we add  $N_\alpha^{ext}$  to  $N'$ .

The completion of  $G[u] + x$  resulting from the representation  $\mathcal{R}$  of  $H_u$  is the one obtained from  $G[u] + x$  by filling  $N'$ .

**Lemma 7 (Permutation completion resulting from a representation).** Let  $H_u$  be a permutation completion of  $P_u + x$  and  $\mathcal{R}$  a representation of  $H_u$ . The completion of  $G[u] + x$  resulting from  $\mathcal{R}$  is a permutation completion of  $G[u] + x$ .

**Proof.** Let  $H$  denote the completion of  $G[u] + x$  resulting from  $\mathcal{R}$ . Clearly  $H$  contains  $G[u] + x$  as subgraph, and hence we must show that  $H$  is a permutation graph. We start from the permutation model  $H_u^{mod}$  of  $H_u$  corresponding to our representation and construct a model for  $H$ . Initially, we expand each segment different from  $x$  into a parallelogram without changing the intersections between the geometric objects.

If we are in the first two cases, we can move segment  $x$  inside the parallelogram of  $v_{top}$ . Observe that for all parallelograms distinct from the one of  $v_{top}$ , this does not change whether segment  $x$  intersects these parallelograms or not. Hence, in each parallelogram  $v_i$  different from  $v_{top}$  we put the segments of a model of  $G[v_i]$ . Now in the parallelogram of  $v_{top}$ , we put the segments for the vertices of  $V[v_{top}]$  such that, together with segment  $x$ , they form a model of the completion obtained from  $G[v_{top}] + x$  by filling  $N'_{v_{top}}$ . Therefore we obtain a permutation model for  $H$ . In the third case, we move the top endpoint (resp. bottom endpoint) of segment  $x$  inside the top edge of the parallelogram  $v_{top}$  (resp. inside the bottom edge of the parallelogram  $v_{bot}$ ). Observe that segment  $x$  and the parallelograms still provide a permutation model for  $H_u$ . For any  $v_i \notin \{v_{top}, v_{bot}\}$  we put in the parallelogram of  $v_i$  a permutation model for  $G[v_i]$ . We put the segments for the vertices of  $V[v_{top}]$  such that together with segment  $x$ , they provide a permutation model for  $H[V[v_{top}] \cup \{x\}]$ . Notice that although segment  $x$  has one endpoint outside the parallelogram  $v_{top}$ ,

447  $H[V[v_{top}] \cup \{x\}]$  is an external completion of  $G[v_{top}] + x$ . We proceed similarly with  $v_{bot}$ , therefore obtaining a  
 448 permutation model for graph  $H$ .  $\square$  448

449

450 Finally, we refine the notion of representation for that of an *optimal representation*, based on a minimal permutation  
 451 completion of  $P_u + x$ . The supplementary conditions required for a representation to be optimal ensure that the resulting  
 452 permutation completion is minimal (Theorem 2). 452

453 **Definition 10 (Representation types and optimal representation).** Let  $H_u$  be a minimal permutation completion of  
 454  $P_u + x$ . For each node  $v_i$  of  $P_u$ , let  $N'_i$  be the neighbourhood of  $x$  in a minimal completion of  $G[v_i] + x$ , and let  $N_i^{ext}$   
 455 be the neighbourhood of  $x$  in an external-minimal completion of  $G[v_i] + x$ . 455

456

457 Let  $\mathcal{R}$  be a representation of  $H_u$ . We say that  $\mathcal{R}$  is: 457

- 458 – type-1 if  $\mathcal{R} = (H_u^{mod}, v_{top}, v_{bot})$ ,  $v_{top} \neq v_{bot}$ ,  $N_{top}^{ext} \subsetneq V[v_{top}]$ , and  $N_{bot}^{ext} \subsetneq V[v_{bot}]$ ; 458
- 459 – type-2 if  $\mathcal{R} = (H_u^{mod}, v_{top}, v_{bot})$ ,  $v_{top} = v_{bot}$ , and  $N'_{top} \subsetneq V[v_{top}]$ ; 459
- 460 – type-3 if  $\mathcal{R} = (H_u^{mod}, v_{top})$  and  $N_{top}^{ext} \subsetneq V[v_{top}]$ ; 460
- 461 – type-4 if  $\mathcal{R} = H_u^{mod}$ , where  $H_u^{mod}$  is any model of  $H_u$ . 461

462 We say that  $\mathcal{R}$  is optimal if  $\mathcal{R}$  is type- $i$  for some  $i \in \{1, 2, 3, 4\}$  and there is no type- $j$  representation of  $H_u$  for any  
 463  $j \in \{1, 2, 3, 4\}$  with  $j < i$ . Finally, we also refer to  $H_u$  as type- $i$ . 463

464 **Definition 11 (Completion of  $P_u + x$  contracted from  $H$  and contracted model).** Let  $H$  be a permutation comple-  
 465 tion of  $G$ . The permutation completion of  $P_u + x$  contracted from  $H$  is the graph  $H_u$  obtained from  $P_u$  by adding the  
 466 vertex  $x$  with neighbourhood  $N_{H_u}(x) = \{v_i \in V_u \mid V[v_i] \cap N_H(x) \neq \emptyset\}$ . 466

467

468 A permutation model  $H_u^{mod}$  of  $H_u$  can be obtained from any model  $H^{mod}$  of  $H$  by preserving one vertex  $y_i$  for  
 469 each child  $v_i$  of  $u$  as follows: 469

- 470 – if  $V[v_i] \cap N_H(x) \neq \emptyset$  then choose  $y_i$  adjacent to  $x$ ; 470
- 471 – otherwise choose any vertex of  $V[v_i]$ . 471

472 The permutation model  $H_u^{mod}$  is called the contracted permutation model of  $H_u$  obtained from  $H^{mod}$ . 472

473 **Lemma 8.** Let  $H$  be a permutation completion of  $G + x$  and  $(\pi_1, \pi_2)$  a permutation model for  $H$ . 473

- 474 – If segment  $x$  has exactly one endpoint in the parallelogram of  $v_i$ , then  $H[V[v_i] \cup \{x\}]$  is an external completion  
 475 of  $G[v_i] + x$ . 475
- 476 – If segment  $x$  has no endpoint in the parallelogram of  $v_i$ , then  $V[v_i]$  is either fully hit (contained in  $N_H(x)$ ) or non  
 477 hit (disjoint from  $N_H(x)$ ). 477

478 In particular, there are at most two nodes  $v_i, v_j$  of  $P_u$  such that the corresponding modules  $V[v_i]$  and  $V[v_j]$  of  $G$  are  
 479 partially hit in  $H$ . 479

480 **Proof.** The lemma is a straightforward consequence of the fact that restricting the model of  $H$  to segment  $x$  and the  
 481 segments of the parallelogram embedding  $V[v_i]$  yields a model of  $H[V[v_i] \cup \{x\}]$ . In the first case, the model repre-  
 482 sents an external completion of  $G[v_i] + x$  because only one endpoint of segment  $x$  is in the parallelogram embedding  
 483  $V[v_i]$ . In the second case  $V[v_i]$  is fully hit if segment  $x$  crosses the parallelogram  $v_i$  and non-hit if the two geometric  
 484 objects do not intersect. This implies that if  $v_i$  is partially hit, the parallelogram of  $v_i$  must contain one endpoint of  
 485 segment  $x$ . Since the bases of all parallelograms embedding  $V[v_i]$  are disjoint, it follows that there are at most two  
 486 parallelograms containing one endpoint of  $x$ . Hence, at most two children of  $u$  are partially hit in  $H$ .  $\square$  486

487

488 **Theorem 2.** Let  $H_u$  be a minimal permutation completion of  $P_u + x$  and let  $\mathcal{R}$  be an optimal representation of  $H_u$ .  
 489 The permutation completion  $H$  of  $G[u] + x$  resulting from  $\mathcal{R}$  is minimal. 489

490 **Proof.** Assume for a contradiction that there exists a permutation completion  $H'$  of  $G[u] + x$  such that  $H'$  is a strict 490  
491 subgraph of  $H$ . Let  $H'_u$  be the permutation completion of  $P_u + x$  contracted from  $H'$  (Definition 11). Since  $H'$  is 491  
492 a subgraph of  $H$ ,  $H'_u$  is also a subgraph of  $H_u$  (but not necessarily a strict subgraph). Recall that  $H_u$  is a minimal 492  
493 completion for  $P_u + x$ , and therefore  $H'_u = H_u$ . As a consequence, for every node  $v_i$  such that  $V[v_i]$  is hit in  $H$  (hence 493  
494  $v_i$  is adjacent to  $x$  in  $H_u$ ) the set  $V[v_i]$  is also hit in  $H'$  (because adjacent to  $x$  in  $H'_u$ ). Consequently, and since  $H'$  is 494  
495 a subgraph of  $H$ , if the set  $V[v_i]$  is partially hit in  $H$  then it is also partially hit in  $H'$ . 495

496  
497 Let now  $y$  be a vertex that is adjacent to  $x$  in  $H$  but not in  $H'$ . Let  $v_y$  be the node of  $P_u$  such that  $y \in V[v_y]$ . 497  
498 Observe that, by construction, if  $H_u$  is type- $i$  for  $i \in \{1, 2, 3\}$ , then the node  $v_{top}$  (and  $v_{bot}$  if it exists) satisfies that 498  
499  $V[v_{top}]$  (and  $V[v_{bot}]$  if it exists) is partially hit in  $H$ . 499

500  $H_u$  is type-1 with optimal representation  $(H_u^{mod}, v_{top}, v_{bot})$  and  $v_{bot} \neq v_{top}$ . If  $v_y$  is not in  $\{v_{top}, v_{bot}\}$  then there 500  
501 are three different nodes  $\{v_{top}, v_{bot}, v_y\}$  such that  $V[v_{top}]$ ,  $V[v_{bot}]$  and  $V[v_y]$  are partially hit in  $H'$ , contradicting 501  
502 Lemma 8. Now assume w.l.o.g. that  $v_y = v_{top}$ . By Lemma 8, in the permutation model of  $H'$ , segment  $x$  must have an 502  
503 endpoint in each of the parallelograms of  $v_{top}$  and  $v_{bot}$ , because  $V[v_{top}]$  and  $V[v_{bot}]$  are partially hit. In particular, this 503  
504 model restricted to  $V[v_y] \cup \{x\}$  shows that  $H'[V[v_y] \cup \{x\}]$  is an external completion of  $G[v_y] + x$ . By construction, 504  
505  $H[V[v_y] \cup \{x\}]$  is an external-minimal completion of  $G[v_y] + x$ , contradicting the fact that  $H'[V[v_y] \cup \{x\}]$  is a strict 505  
506 subgraph of  $H[V[v_y] \cup \{x\}]$ . 506

507  $H_u$  is type-2 with optimal representation  $(H_u^{mod}, v_{top}, v_{bot})$  and  $v_{bot} = v_{top}$ . If  $v_y \neq v_{top}$ ,  $V[v_{top}]$  and  $V[v_y]$  are 507  
508 partially hit in  $H'$ . Therefore, by Lemma 8, in a permutation model  $H^{mod'}$  of  $H'$ , segment  $x$  has one endpoint in the 508  
509 parallelogram of  $v_y$  and the other in the parallelogram of  $v_{top}$ . It follows that in the contracted permutation model 509  
510  $H_u^{mod'}$  of  $H'_u = H_u$  obtained from  $H^{mod'}$  (see Definition 11), the segments of  $v_y$  and  $v_{top}$  have endpoints next to 510  
511 the endpoints of  $x$ . Hence  $(H_u^{mod'}, v_{top}, v_y)$  is a type-1 representation for  $H_u$ , contradicting the assumption that  $H_u$  is 511  
512 type-2. If  $v_y = v_{top}$ , then  $H'[V[v_y] \cup \{x\}]$  is a strict subgraph of  $H[V[v_y] \cup \{x\}]$ , contradicting the fact that the latter 512  
513 is a minimal completion of  $G[v_y] + x$ . 513

514  $H_u$  is type-3 with optimal representation  $(H_u, v_{top})$ . If  $v_y \neq v_{top}$ , then, in the completion  $H'$  both  $V[v_{top}]$  and  $V[v_y]$  514  
515 are partially hit. Therefore, by Lemma 8, in a permutation model  $H^{mod'}$  of  $H'$ , segment  $x$  has one endpoint in the 515  
516 parallelogram of  $v_y$  and the other in the parallelogram of  $v_{top}$ . It follows that in the contracted permutation model 516  
517  $H_u^{mod'}$  of  $H'_u = H_u$  obtained from  $H^{mod'}$ , the segments of  $v_y$  and  $v_{top}$  have endpoints next to the endpoints of  $x$ . 517  
518 Hence  $(H_u^{mod'}, v_{top}, v_y)$  is a type-1 representation for  $H_u$ , contradicting the fact that  $H_u$  is type-3. Assume now that 518  
519  $v_y = v_{top}$ . Since  $H'[V[v_y] \cup \{x\}]$  is strictly contained in  $H[V[v_y] \cup \{x\}]$  and  $H[V[v_y] \cup \{x\}]$  is minimal among 519  
520 all external completions of  $G[v_y] + x$ , it follows that  $H'[V[v_y] \cup \{x\}]$  is not an external completion of  $G[v_y] + x$ . 520  
521 Therefore, in a permutation model  $H^{mod'}$  of  $H'$ , segment  $x$  has both endpoints in the parallelogram of  $v_y$ . Then, in the 521  
522 contracted permutation model  $H_u^{mod'}$  of  $H'_u$  obtained from  $H^{mod'}$ , segment  $x$  has both endpoints next to the endpoints 522  
523 of segment  $v_y$ , so  $(H_u^{mod'}, v_y, v_y)$  is a type-2 representation for  $H_u$  — a contradiction. 523

524  $H_u$  is type-4. Recall that  $V[v_y]$  is partially hit in  $H'$ . By Lemma 8, segment  $x$  has an endpoint in the parallelogram of 524  
525  $v_y$  in a model  $H^{mod'}$  of  $H'$ . Consequently, in the contracted permutation model of  $H'_u$  obtained from  $H^{mod'}$ , segment 525  
526  $v_y$  has an endpoint next to an endpoint of segment  $x$ , providing a type- $i$  representation for  $H_u$ ,  $i \in \{1, 2, 3\}$  — a 526  
527 contradiction.  $\square$  527

528  
529 Theorem 3 below is a similar statement for constructing external-minimal completions. In order to state it, we need 529  
530 the notion of *external-optimal representation*. 530

531 **Definition 12 (External representation types and external-optimal representation).** Let  $H_u$  be an external-minimal 531  
532 permutation completion of  $P_u + x$ . For each node  $v_i$  of  $P_u$ , let  $N_i^{ext}$  be the neighbourhood of  $x$  in an external-minimal 532  
533 completion of  $G[v_i] + x$ . 533

534  
535 Let  $\mathcal{R}$  be a representation of  $H_u$ . We say that  $\mathcal{R}$  is: 535

- 536 1. external type-1 if  $\mathcal{R} = (H_u^{mod}, v_{top})$  and  $N_{top}^{ext} \subsetneq V[v_{top}]$ ; 536  
537 2. external type-2 if  $\mathcal{R} = H_u^{mod}$ , where  $H_u^{mod}$  is any model of  $H_u$ . 537

538 We say that  $\mathcal{R}$  is external-optimal if  $\mathcal{R}$  is external type-1 or external type-2 but not external type-1. Finally, we also 538  
539 refer to  $H_u$  as external type-i. 539

540 **Theorem 3.** Let  $H_u$  be an external-minimal permutation completion of  $P_u + x$  and let  $\mathcal{R}$  be an external-optimal 540  
541 representation of  $H_u$ . The permutation completion  $H$  of  $G[u] + x$  resulting from  $\mathcal{R}$  is an external-minimal permutation 541  
542 completion of  $G[u] + x$ . 542

543 **Proof.** From the definition of the permutation completion resulting from a representation (Definition 9 and Lemma 7), 543  
544  $H$  is a permutation completion of  $G[u] + x$ . Moreover, note that since  $\mathcal{R}$  is an external-optimal representation of  $H_u$ , 544  
545 the permutation model built for  $H$  in the proof of Lemma 7 is such that the bottom endpoint of  $x$  is leftmost among all 545  
546 bottom endpoints, which proves that  $H$  is an external permutation completion of  $G[u] + x$ . 546

547 Assume for a contradiction that the external completion  $H$  is not minimal. Then, it strictly contains another external 547  
548 completion  $H'$ . Consider the contracted permutation completion  $H'_u$  of  $P_u + x$  obtained from  $H'$ . From Definition 11, 548  
549  $H'_u$  is an external completion of  $P_u + x$ . Moreover, since  $H'$  is a subgraph of  $H$ , then  $H'_u$  is a subgraph of  $H_u$ . By 549  
550 minimality of  $H_u$ , we have  $H'_u = H_u$ . In particular, for any child  $v_i$  of  $u$  in the modular decomposition,  $V[v_i]$  is hit in 550  
551  $H$  if and only if it is hit in  $H'$ . Consequently if the set  $V[v_i]$  is partially hit in  $H$  then it is also partially hit in  $H'$ . 551  
552

553 As in the proof of Theorem 2, let  $y$  be a neighbour of  $x$  in  $H$  but not in  $H'$ , and let  $v_y$  be the node of  $P_u$  such that 553  
554  $y \in V[v_y]$ . Since  $V[v_y]$  is partially hit in  $H'$ , Lemma 8 implies that segment  $x$  has an endpoint in the parallelogram 554  
555 of  $v_y$  in any model of  $H'$ . If  $v_{top}$  exists and is different from  $v_y$ , since  $v_{top}$  is partially hit in  $H$ , then it is partially hit 555  
556 in  $H'$  and Lemma 8 implies that segment  $x$  has an endpoint in the parallelogram of  $v_{top}$ . This contradicts the fact that 556  
557 there is an external model of  $H'$ . Consider now the case where  $v_{top}$  exists and  $v_{top} = v_y$ . Observe that the external 557  
558 completion  $H'[V[v_{top}] \cup x]$  of  $G[v_{top}] + x$  is strictly contained in the external-minimal completion  $H[V[v_{top}] \cup x]$  of 558  
559  $G[v_{top}] + x$  — a contradiction. 559

560 Consequently,  $v_{top}$  does not exist. segment  $x$  has an endpoint in the parallelogram of  $v_y$  in the model of  $H'$ , but 560  
561 this endpoint cannot be the external one. Recall that the external endpoint of segment  $x$  is bottom left. Hence in the 561  
562 contracted permutation model of  $H'_u$  obtained from the one of  $H'$ , segment  $v_y$  has its top endpoint directly before the 562  
563 top endpoint of  $x$ , next to it (it cannot be after since the two segments intersect). Hence  $H'_u = H_u$  has an external 563  
564 type-1 representation, contradicting the fact that  $v_{top}$  does not exist.  $\square$  564  
565

## 566 4.2 The algorithm 566

567 Our  $\mathcal{O}(n)$  time algorithm computing a minimal permutation completion of  $G+x$  proceeds by a bottom-up computation 567  
568 of two functions for each node  $u$ , namely  $MinIns(u)$  and  $MinInsExt(u)$ . The function  $MinIns(u)$  computes a 568  
569 minimal permutation completion of  $G[u] + x$ , i.e. it returns the neighbourhood  $N'_u = N(x) \cap V[u]$  of  $x$  in this 569  
570 completion. Similarly, the function  $MinInsExt(u)$  computes an external-minimal permutation completion of  $G[u] +$  570  
571  $x$ , i.e. it returns the neighbourhood  $N_u^{ext}$  of  $x$  in such a completion. These functions will only be called on hit nodes, 571  
572 from bottom to top. Therefore when we treat a node  $u$  we can assume that we already have computed the sets  $N'_v$  and 572  
573  $N_v^{ext}$  for each hit child  $v$  of  $u$  in the modular decomposition tree  $T$ . The case when  $u$  is a (hit) leaf is trivial: both 573  
574 functions return as neighbourhood of  $x$  the vertex of  $G$  that labels this leaf. In the remaining of this section we show 574  
575 how given an internal node  $u$  one can determine the type of  $H_u$  and an optimal (resp. external-optimal) representation. 575

576 **Series and parallel nodes.** We first deal with the cases when  $u$  is either a series or a parallel node of the modular 576  
577 decomposition, which turn out to be very similar. 577

578 **Lemma 9.** Algorithm 2 correctly outputs a minimal permutation completion of  $G[u] + x$  when  $u$  is a series or parallel 578  
579 node of the modular decomposition. 579

---

**Algorithm 2:** MinIns( $u$ ): Computing a minimal completion of  $G[u] + x$  for a node  $u$  which is series or parallel.

---

**Input:** a series or parallel node  $u$  of the modular decomposition; for each hit node  $v$  of  $P_u$ , the sets  $N'_v$  and  $N_v^{ext}$

**Output:**  $N'_u$  the set of neighbours of  $x$  in a completion resulting from an optimal representation

```

1  Let  $\{v_1, \dots, v_k\}$  denote the children of  $u$ ;
2  Assume w.l.o.g. that the hit ones are  $\{v_1, \dots, v_l\}$ , for  $l \leq k$ ;
3   $H_u \leftarrow P_u + x$ ;
4  if  $\exists v_a, v_b \in \{v_1, \dots, v_l\}$  s.t.  $v_a \neq v_b$  and  $N_a^{ext} \subsetneq V[v_a]$  and  $N_b^{ext} \subsetneq V[v_b]$  then
5  |    $i \leftarrow 1$ ;
6  |    $v_{top} \leftarrow v_a$ ;
7  |    $v_{bot} \leftarrow v_b$ ;
8  else if  $u$  is parallel and  $l = 1$  and  $N'_1 \subsetneq V[v_1]$  then
9  |    $i \leftarrow 2$ ;
10 |    $v_{top} \leftarrow v_1$ ;
11 |    $v_{bot} \leftarrow v_1$ ;
12 else if  $u$  is series and  $l = k$  and  $\exists v_a \in V_u$  such that  $N'_a \subsetneq V[v_a]$  then
13 |    $i \leftarrow 2$ ;
14 |    $v_{top} \leftarrow v_a$ ;
15 |    $v_{bot} \leftarrow v_a$ ;
16 else if  $\exists v_a \in V_u$  such that  $N_a^{ext} \subsetneq V[v_a]$  then
17 |    $i \leftarrow 3$ ;
18 |    $v_{top} \leftarrow v_a$ ;
19 else
20 |    $i \leftarrow 4$ ;
21 Compute a model  $H_u^{mod}$  of  $H_u$  where  $v_{top}$  and  $v_{bot}$  (if they exist) are next to  $x$  in the top and bottom order respectively;
22 Let  $\mathcal{R}$  denote the type- $i$  representation of  $H_u$  based on  $H_u^{mod}$ , according to the value  $i$  defined above (Definition 10);
23 Compute the set  $N'_u$  of neighbours of  $x$  in the completion resulting from  $\mathcal{R}$  (Definition 9);
24 Return  $N'_u$ ;
```

---

580 **Proof.** First, note that  $H_u = P_u + x$  is already a permutation graph. Moreover, the algorithm exactly follows the order 580  
581 of the mutually exclusive cases of an optimal representation (Definition 10). From Theorem 2, in order to prove that 581  
582 the algorithm is correct, we just have to check that at each step of the case disjunction of the algorithm, the test which 582  
583 is performed determines correctly the type which is assigned by the algorithm. At the same time, we also have to show 583  
584 how to compute a model for  $H_u$  such that  $v_{top}$  and  $v_{bot}$  are next to  $x$ , when they exist. 584  
585 *Type-1.* From Definition 10, it is straightforward to see that the conditions of the first test (Line 4) are necessary so 585  
586 that  $H_u$  is type-1. Let us show that they are sufficient, i.e. we can build a model of  $H_u$  such that  $x$  is next to  $v_{top}$  in 586  
587 the top order and  $x$  is next to  $v_{bot}$  in the bottom order. If  $u$  is a parallel node (Figure 3 left), place all the hit children 587  
588 at the beginning of the top order, ranged from the leftmost  $v_{bot}$  to the rightmost  $v_{top}$ , and place all the non hit children 588  
589 on their right. For the bottom order, use exactly the same order on the children of  $u$ . Finally, place  $x$  at the beginning 589  
590 of the bottom order and between the hit and non hit children on the top order. In this way, segment  $x$  intersects exactly 590  
591 the segments of hit children of  $u$  and  $x$  is next to  $v_{top}$  and  $v_{bot}$  in the top and bottom order respectively. If  $u$  is a series 591  
592 node (Figure 3 right), we proceed slightly differently. We place all the hit children except  $v_{bot}$  at the beginning of the 592  
593 top order,  $v_{top}$  being placed in the rightmost position among them. Then we place all the non hit children on their 593  
594 right. Moreover, we place  $v_{bot}$  in the rightmost position of the top order. For the bottom order, we use the reversed 594  
595 order. Finally, we place  $x$  immediately on the right of  $v_{bot}$  (which is in the leftmost position) on the bottom order and 595  
596 immediately on the right of  $v_{top}$  on the top order. Again, segment  $x$  intersects exactly the segments of hit children of 596  
597  $u$ . Thus, the test of Line 4 is positive if and only if  $H_u$  is type-1, and we can compute a model  $H_u^{mod}$  in this case, as 597  
598 requested at Line 21. 598  
599

600 *Type-2.* Assume the conditions of the test at Line 8 (resp. Line 12) are satisfied. In this case, one can build a model 600  
601 of  $H_u = P_u + x$  from any model of  $P_u$  by placing  $x$  immediately on the left of  $v_1$  (resp.  $v_a$ ) in the top order and 601  
602 immediately on the right of  $v_1$  (resp.  $v_a$ ) in the bottom order. This shows that  $H_u$  is type-2. 602



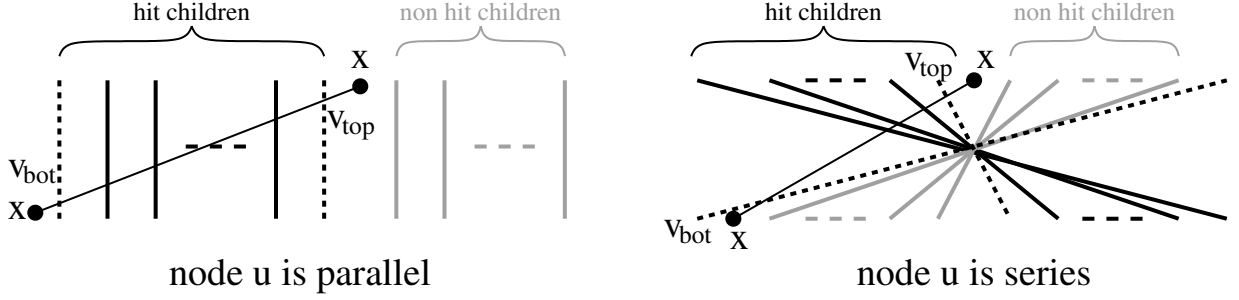


Fig. 3. Parallel and series case for *MinIns*

603 Now, let us consider the case where  $H_u$  is type-2. Since in this case  $x$  is by definition next to  $v_{top}$  in both orders of  
 604 the permutation model of  $H_u$ , it follows that  $v_{top}$  and  $x$  have the same neighbourhood in the rest of  $H_u = P_u + x$ . If  
 605  $u$  is a parallel node, no vertex  $v \in V_u \setminus \{v_{top}\}$  is adjacent to  $x$  in  $P_u + x$ , and  $v_{top}$  is therefore the only vertex of  $P_u$   
 606 adjacent to  $x$ . Hence the conditions of the test at Line 8 are satisfied. If  $u$  is a series node, every vertex  $v \in V_u \setminus \{v_{top}\}$   
 607 is adjacent to  $x$  in  $P_u + x$ , as well as  $v_{top}$  itself by definition of a representation. Thus the conditions of the test at  
 608 Line 12 are satisfied.

609  
 610 *Type-3.* It is straightforward to see that if  $H_u$  is type-3, then the condition of the test at Line 16 is satisfied. Conversely,  
 611 if this condition is satisfied, then withdraw  $v_{bot}$  from the model of  $H_u$  constructed above in the case where  $H_u$  is type-1  
 612 (see Figure 3) i. In the present case, this provides a type-3 model. Moreover, since the tests of Lines 4, 8 and 12 have  
 613 been all negative,  $H_u$  is neither type-1 nor type-2, and it results that  $H_u$  is type-3.

614  
 615 *Type-4.* The final else case of the algorithm deals with all remaining cases, i.e. where  $H_u$  is type-4. Again, withdraw-  
 616 ing  $v_{top}$  and  $v_{bot}$  from the type-1 model constructed above provides a suitable type-4 model. This ends the proof of  
 617 validity of Algorithm 2.  $\square$

618  
 619 The algorithm for function *MinInsExt* is even simpler. We use the same notations as for *MinIns*.

---

**Algorithm 3:** *MinInsExt*( $u$ ): Computing an external-minimal permutation completion of  $G[u] + x$  for a node  $u$  which is series or parallel.

---

**Input:** a *series* or *parallel* node  $u$  of the modular decomposition; for each hit node  $v$  of  $P_u$ , the set  $N_v^{ext}$   
**Output:**  $N_u^{ext}$  the set of neighbours of  $x$  in a completion resulting from an external-optimal representation

- 1 Let  $\{v_1, \dots, v_k\}$  denote the children of  $u$ ;
- 2 Assume w.l.o.g. that the hit ones are  $\{v_1, \dots, v_l\}$ , for  $l \leq k$ ;
- 3  $H_u \leftarrow P_u + x$ ;
- 4 **if**  $\exists v_a \in \{v_1, \dots, v_l\}$  s.t.  $N_a^{ext} \subsetneq V[v_a]$  **then**
- 5      $i \leftarrow 1$ ;
- 6      $v_{top} \leftarrow v_a$ ;
- 7 **else**
- 8      $i \leftarrow 2$ ;
- 9 Compute a model  $H_u^{mod}$  of  $H_u$  where  $v_{top}$  (if it exists) is next to  $x$  in the top order;
- 10 Let  $\mathcal{R}$  denote the external type- $i$  representation of  $H_u$  based on  $H_u^{mod}$ , according to the value  $i$  defined above (Definition 12);
- 11 Compute the set  $N_u^{ext}$  of neighbours of  $x$  in the completion resulting from  $\mathcal{R}$  (Definition 9);
- 12 Return  $N_u^{ext}$ ;

---

620 **Lemma 10.** *Algorithm 3 correctly outputs the neighbourhood of  $x$  in an external-minimal permutation completion of* 620  
621  *$G[u] + x$  when  $u$  is a series or parallel node of the modular decomposition.* 621

622 **Proof.** The proof is very similar to the one of Lemma 9. Observe that  $P_u + x$  is already a permutation graph with 622  
623 an external model (where the bottom endpoint of segment  $x$  is the leftmost one). Such an external model is obtained 623  
624 by withdrawing  $v_{bot}$  from the model built in the proof of Lemma 9 in the case where  $H_u$  is type-1 (see Figure 3). 624  
625 This model also shows that if the condition of the test at Line 4 is satisfied, then  $H_u$  is an external-optimal type-1 625  
626 permutation completion. Conversely, it is straightforward to see from Definition 12 that if  $H_u$  is external type-1, then 626  
627 the condition of the test at Line 4 is satisfied. Thus, the algorithm correctly detects the type of  $P_u + x$ , and the rest 627  
628 follows from Theorem 3.  $\square$  628

629  
630 **Prime nodes** Let us say that two permutation models are *different* if they are not identical apart from left-right of 630  
631 upside-down flipping operation on the model 631

632 **Lemma 11 (see also [3]).** *Let  $H_u$  be a minimal permutation completion of  $P_u + x$  and  $H_u^{mod}$  a permutation model* 632  
633 *for  $H_u$ , where  $u$  is a prime node of the modular decomposition. There is at most one model of  $H_u$  different from  $H_u^{mod}$*  633  
634 *and it can be computed from  $H_u^{mod}$  in  $O(k)$  time, where  $k$  is the number of nodes of  $P_u$ .* 634

635 **Proof.** Since  $P_u$  is a prime graph, Theorem 4 of [3] implies that  $H_u$  has at most one non-trivial module  $M$ . Such 635  
636 a module is either  $M = V_u$  (when  $V_u$  is uniform in  $H_u$ ) or of the form  $M = \{x, v\}$  for some node  $v$  of  $P_u$ . Now 636  
637 consider the given permutation model  $(\pi_1, \pi_2)$  of  $H_u$ . 637

638 If  $V_u$  is a module of  $H_u$ , this can be easily detected. If  $x$  is not adjacent to the (whole) module, it means that its 638  
639 segment has endpoints  $(1, 1)$  or  $(k, k)$  in the model. Assume  $x$  is in position  $(1, 1)$  (the other case is symmetrical). The 639  
640 only other possible model is obtained by moving segment  $x$  completely to the right, in position  $(k, k)$ , and shifting all 640  
641 other segments one slot to the left, from  $(i, j)$  to  $(i - 1, j - 1)$ . If  $x$  is adjacent to the (whole) module, its segment can 641  
642 be in positions  $(1, k)$  or  $(k, 1)$ . Assume w.l.o.g. that  $x$  is in position  $(1, k)$ . Then the only other possibility is to put  $x$  642  
643 in position  $(k, 1)$  and to shift each other segment  $(i, j)$  to  $(i - 1, j + 1)$ . The whole construction can be performed in 643  
644  $O(k)$  time. 644

645 If  $\{x, v\}$  is a module of  $H_u$ , this implies that the two segments are embedded in a parallelogram such that no other 645  
646 segment has an endpoint in this parallelogram. Any other model is obtained by permuting segments  $x$  and  $v$ . Let  $(i, j)$  646  
647 be the position of segment  $x$ . The only possible cases are that  $v$  is in position  $(i - 1, j - 1)$ ,  $(i + 1, j + 1)$ ,  $(i - 1, j + 1)$  647  
648 or  $(i + 1, j - 1)$ . Each of these cases can be detected in constant time, and the shift can be also performed in  $O(1)$  time.  $\square$  648  
649 649

---

**Algorithm 4:** MinIns( $u$ ): Computing a minimal completion of  $G[u] + x$  for a prime node  $u$ .

---

**Input:** a prime node  $u$  of the modular decomposition; for each hit node  $v$  of  $P_u$ , the sets  $N'_v$  and  $N_v^{ext}$

**Output:**  $N'_u$  the set of neighbours of  $x$  in a completion resulting from an optimal representation

- 1 Compute a minimal completion  $H_u$  of  $P_u + x$  as in Section 3 together with a permutation model  $(\pi_1, \pi_2)$ ;
  - 2 Compute, if it exists, the second permutation model  $(\pi'_1, \pi'_2)$  from the first one using Lemma 11;
  - 3 Compute the smallest possible type  $i$  among the two models, together with a type- $i$  representation  $\mathcal{R}$  of  $H_u$ ;
  - 4 Compute the set  $N'_u$  of neighbours of  $x$  in the completion resulting from  $\mathcal{R}$  (Definition 9);
  - 5 Return  $N'_u$ ;
- 

650 **Theorem 4.** *Algorithm 4 correctly computes the neighbourhood of  $x$  in a minimal completion of  $G[u] + x$  when  $u$  is* 650  
651 *a prime node of the modular decomposition.* 651

652 **Proof.** The proof is a direct consequence of Theorem 2, applied to the minimal completion  $H_u$ . Since there are at 652  
653 most two different permutation models for  $H_u$  (Lemma 11) and since our algorithm tries both of them, it correctly 653

654 computes the representation of  $H_u$ . □ 654

655

656 We now give the algorithm computing an external-minimal completion for prime nodes (Algorithm 5). 656

---

**Algorithm 5:** MinInsExt( $u$ ): Computing an external-minimal completion of  $G[u] + x$  for a prime node  $u$ .

---

**Input:** a prime node  $u$  of the modular decomposition; for each hit node  $v_i$  of  $P_u$ , the set  $N_i^{ext}$

**Output:**  $N_u^{ext}$  the set of neighbours of  $x$  in a completion resulting from an external-optimal representation

1 **for** each of the four models  $(\pi_1, \pi_2)$  of  $P_u$  **do**

2     Let  $j$  be the maximum top endpoint of a segment intersecting  $x$  in  $P_u$ ;

3     Let  $v_t$  be the node corresponding to that segment;

4     Insert  $x$  in position  $(j + 0.5, 0.5)$ ;

5 Choose among the four external completions above a minimal one  $H_u$ , which additionally satisfies  $N_t^{ext} \subsetneq V[v_t]$  if possible;

6 Denote  $H_u^{mod}$  the external model computed above for  $H_u$ ;

7 **if**  $N_t^{ext} \subsetneq V[v_t]$  **then**

8      $i \leftarrow 1$ ;

9      $\mathcal{R} \leftarrow (H_u^{mod}, v_t)$ ;

10 **else**

11      $i \leftarrow 2$ ;

12      $\mathcal{R} \leftarrow H_u^{mod}$ ;

13 Compute the set  $N_u^{ext}$  of neighbours of  $x$  in the completion resulting from external type- $i$  representation  $\mathcal{R}$ , according to the value  $i$  defined above (Definition 9);

14 **Return**  $N_u^{ext}$ ;

---

657 **Theorem 5.** Algorithm 5 correctly computes the neighbourhood of  $x$  in an external-minimal completion of  $G[u] + x$  657  
658 when  $u$  is a prime node of the modular decomposition. 658

659 **Proof.** By construction, a model for such a completion is obtained from a model  $(\pi_1, \pi_2)$  of  $P_u$  by inserting  $x$  is 659  
660 position  $(j + 0.5, 0.5)$ , where  $j$  is the rightmost top endpoint among all segments adjacent to  $x$  in  $P_u + x$ . Hence 660  
661 we only need to test among the four possibilities (due to upside-down and left-right flippings of the model  $(\pi_1, \pi_2)$ ) 661  
662 which one produces an external-minimal completion of  $P_u + x$ , and whether segment  $v_i$  with top endpoint  $j$  satisfies 662  
663  $N_i^{ext} \subsetneq V[v_i]$ . In this way we obtain an external-minimal completion  $H_u$  of  $P_u + x$  together with its type and its 663  
664 external-optimal representation (Definition 12). It remains to construct  $N_u^{ext}$  from this representation, as in Defini- 664  
665 tion 9. □ 665

666

### 667 4.3 Efficient implementation of the algorithm 667

668 Recall that our algorithm computes in a bottom-up manner the sets  $N'_u$  (resp.  $N_u^{ext}$ ), that is the sets of neighbours 668  
669 of  $x$  in a minimal (resp. an external-minimal) completion of  $G[u] + x$ . We only need to prove that for each node  $u$ , 669  
670 the corresponding calls to  $MinIns(u)$  and  $MinInsExt(u)$  can be implemented to run in  $O(k)$  time, where  $k$  is the 670  
671 number of children of  $u$ . Actually, during a first sweep the algorithm does not explicitly compute such sets  $N'_u$  and 671  
672  $N_u^{ext}$ . Instead, it only determines: 672

- 673 – the type of the completion (resp. external completion)  $H_u$  of  $P_u + x$ , 673
- 674 – the subset  $N_{H_u}(x)$  of children of  $u$  that are adjacent to  $x$  in  $H_u$ , 674
- 675 – the children  $v_{top}$  and  $v_{bot}$  of  $u$ , if they exist, and 675
- 676 – the size of  $N'_u$  (resp.  $N_u^{ext}$ ). 676

Since from Definition 9  $N'_u$  (resp.  $N_u^{ext}$ ) is computed as disjoint union of sets previously computed for the children of  $u$ , we can obtain the size of  $N'_u$  (resp.  $N_u^{ext}$ ) by simply adding the sizes of such sets. In other words, we add to  $|N'_u|$  (resp.  $|N_u^{ext}|$ ) value  $|V[v_i]|$  if  $v_i \in N_{H_u}(x) \setminus \{v_{top}, v_{bot}\}$  and  $|N'_i|$  or  $|N_i^{ext}|$  otherwise. This part clearly takes  $O(k)$  time. The purpose of determining these sizes is that it allows us to perform the tests  $N'_i \subsetneq V[v_i]$  and  $N_i^{ext} \subsetneq V[v_i]$  in constant time.

The key of the complexity of the first sweep of the algorithm (and of its whole execution as well) lays in determining  $N_{H_u}(x)$  and  $\{v_{top}, v_{bot}\}$  if they exist. Note that all this information can be read easily on a permutation model  $H_u^{mod}$  of  $H_u$ . For the cases where  $u$  is series or parallel, our algorithm computes such a model by a basic manipulation of the set of children of  $u$ . As explained above (Algorithms 2 and 3), this can be done in  $O(k)$  time. For the case where  $u$  is a prime node, the minimal completion  $H_u$  together with its (at most two) models is computed in  $O(k)$  time (Section 3 and Lemma 11). Thus, for every node  $u$  of the modular decomposition  $T$  having  $k$  children, the first sweep of the algorithm will spend  $O(k)$  time on the computation of the information associated to node  $u$ . This gives an  $O(n)$  total computation time for the first sweep.

In a second step, we determine for each node  $u$  of  $T$  different from  $r$  which of  $N'_u$ ,  $N_u^{ext}$ ,  $V[u]$  or  $\emptyset$  is required for the minimal completion determined at the root. To that aim, using the information computed in the first sweep, we can perform a top-down search of the modular decomposition tree  $T$  starting at its root  $r$ .

Note that for the root itself, the set required is always  $N'_r$ . In this search, we colour *grey* the nodes  $u$  for which  $N'_u$  or  $N_u^{ext}$  is required, *black* the nodes for which  $V[u]$  is required and *white* the nodes for which  $\emptyset$  is required. This takes  $O(|T|) = O(n)$  time.

Observe that all the descendants of a black node are necessarily black and that all the descendants of a white node are necessarily white. It follows that the grey nodes form a subtree  $T^{grey}$  of  $T$  which contains the root  $r$ . The set  $N'_r$  of vertices that are neighbours of  $x$  in the completion determined at the root is simply the disjoint union of all the sets  $V[v]$  where  $v$  is a black node whose parent is grey. For each of these nodes  $v$ , we can simply compute  $V[v]$  as a list containing all the leaves of their subtree, which takes  $O(|T_v|)$  time. We then build  $N'_r$  by appending all the lists of the black children of the nodes of  $T^{grey}$ . In total, this takes  $O(|T|) = O(n)$  time, which is also the complexity of the second step. Therefore the whole algorithm to determine the list of vertices adjacent to  $x$  in some minimal permutation completion of  $G + x$  takes  $O(n)$  time. Once we have computed this set  $N'_r(x)$  of neighbours of  $x$  in a minimal completion  $H$  of  $G + x$ , we use the algorithm in [3] to transform the modular decomposition of  $G$  (including the quotient graphs of its prime nodes) into the one of  $H$ . As the algorithm in [3] runs in  $O(n)$  time, this is also the complexity of one incremental step of our algorithm. Altogether, we obtain the following result.

**Theorem 6.** *A minimal permutation completion of an arbitrary graph on  $n$  vertices can be computed in  $O(n^2)$  time.*

## 5 Conclusion

We gave in this article an  $O(n^2)$  time algorithm computing a minimal permutation completion of an arbitrary graph  $G$ . Our approach is incremental: we take vertices one by one, in some order  $(x_1, \dots, x_n)$ , and at step  $i$  we compute a minimal permutation completion  $H_i$  of  $G_i = G[\{x_1, \dots, x_i\}]$  from a minimal permutation completion  $H_{i-1}$  of  $G_{i-1}$ , adding only fill edges incident to  $x_i$ . Each step is performed in  $O(n)$  time thanks to the modular decomposition of  $H_{i-1}$ . Once we have computed the set of fill edges we need to update the modular decomposition of  $H_{i-1}$  into the one of  $H_i$ , which is done using the algorithm of [3]. A natural question is whether we could perform the whole step faster than  $\Theta(n)$ , e.g., in a running time depending on the degree of  $x_i$  in  $G$ , or in  $H_i$ . There are examples (see [3]) showing that when we go from  $H_{i-1}$  to  $H_i$ , the structure of the modular decomposition may differ a lot, even if  $x$  is of small degree. Therefore, improving the overall  $O(n^2)$  complexity of our algorithm for minimal permutation completions probably requires a different approach than the incremental one based on an arbitrary vertex ordering.

## References

1. Bergeron, A., Chauve, C., de Montgolfier, F., Raffinot, M.: Computing common intervals of  $k$  permutations, with applications to modular decomposition of graphs. In: ESA. pp. 779–790. No. 3669 in LNCS (2005)

- 722 2. Burzyn, P., Bonomo, F., Durán, G.: NP-completeness results for edge modification problems. *Discrete Applied Mathematics* 154(13), 1824–1844 (2006), <http://dx.doi.org/10.1016/j.dam.2006.03.031> 722
- 723 154(13), 1824–1844 (2006), <http://dx.doi.org/10.1016/j.dam.2006.03.031> 723
- 724 3. Crespelle, C., Paul, C.: Fully dynamic algorithm for recognition and modular decomposition of permutation graphs. *Algorith-* 724
- 725 *mica* 58(2), 405–432 (2010) 725
- 726 4. Crespelle, C., Perez, A., Todinca, I.: An  $O(n^2)$  time algorithm for the minimal permutation completion problem. In: Mayr, 726
- 727 E.W. (ed.) *Graph-Theoretic Concepts in Computer Science - 41st International Workshop, WG 2015, Garching, Germany, June 17-19, 2015, Revised Papers*. Lecture Notes in Computer Science, vol. 9224, pp. 103–115. Springer (2015) 727
- 728 103–115. Springer (2015) 728
- 729 5. Crespelle, C., Todinca, I.: An  $O(n^2)$ -time algorithm for the minimal interval completion problem. *Theor. Comput. Sci.* 494, 729
- 730 75–85 (2013), <http://dx.doi.org/10.1016/j.tcs.2012.12.031> 730
- 731 6. Heggernes, P.: Minimal triangulations of graphs: A survey. *Discrete Math.* 306(3), 297–317 (2006) 731
- 732 7. Heggernes, P., Mancini, F., Papadopoulos, C.: Minimal comparability completions of arbitrary graphs. *Discrete Applied Math-* 732
- 733 *ematics* 156(5), 705–718 (2008) 733
- 734 8. Heggernes, P., Telle, J.A., Villanger, Y.: Computing minimal triangulations in time  $O(n^{\alpha \log n}) = o(n^{2.376})$ . *SIAM J. Discrete* 734
- 735 *Math.* 19(4), 900–913 (2005) 735
- 736 9. Heggernes, P., Mancini, F.: Minimal split completions. *Discrete Applied Mathematics* 157(12), 2659–2669 (2009), 736
- 737 <http://dx.doi.org/10.1016/j.dam.2008.08.010> 737
- 738 10. Lokshtanov, D., Mancini, F., Papadopoulos, C.: Characterizing and computing minimal cograph completions. *Discrete Applied* 738
- 739 *Mathematics* 158(7), 755–764 (2010), <http://dx.doi.org/10.1016/j.dam.2009.01.016> 739
- 740 11. Mancini, F.: *Graph Modification Problems Related to Graph Classes*. Ph.D. thesis, University of Bergen, Norway (2008) 740
- 741 12. Möhring, R.H.: Algorithmic aspect of the substitution decomposition in optimization over relations, set systems and boolean 741
- 742 functions. *Annals of Operations Research* 4, 195–225 (1985) 742
- 743 13. Möhring, R.H., Radermacher, F.: Substitution decomposition for discrete structures and connections with combinatorial opti- 743
- 744 mization. *Annals of Discrete Mathematics* 19, 257–356 (1984) 744
- 745 14. Ohtsuki, T., Mori, H., Kashiwabara, T., Fujisawa, T.: On minimal augmentation of a graph to obtain an interval graph. *Journal* 745
- 746 *of Computer and System Sciences* 22(1), 60–97 (1981) 746
- 747 15. Ohtsuki, T.: A fast algorithm for finding an optimal ordering for vertex elimination on a graph. *SIAM J. Comput.* 5(1), 133–145 747
- 748 (1976) 748
- 749 16. Rapaport, I., Suchan, K., Todinca, I.: Minimal proper interval completions. *Information Processing Letters* 5, 195–202 (2008) 749
- 750 17. Rose, D.J., Tarjan, R.E., Lueker, G.S.: Algorithmic aspects of vertex elimination on graphs. *SIAM J. Comput.* 5(2), 266–283 750
- 751 (1976), <http://dx.doi.org/10.1137/0205021> 751
- 752 18. Yannakakis, M.: Computing the minimum fill-in is NP-complete. *SIAM J. on Algebraic and Discrete Methods* 2(1), 77–79 752
- 753 (1981), <http://dx.doi.org/10.1137/0602010> 753