Synthesis of Quasi-interpretations

Guillaume Bonfante, Jean-Yves Marion, Jean-Yves Moyen, Romain Péchoux

Synthesis of Quasi-interpretations

G. Bonfante [a] J.-Y. Marion [a] J.-Y. Moyen [b] R. Péchoux [a]

[a] *Loria, Calligramme project, B.P. 239, 54506 Vandœuvre-lès-Nancy Cedex, France, and École Nationale Supérieure des Mines de Nancy, INPL, France.*

[b] *Loria, Calligramme project, B.P. 239, 54506 Vandœuvre-lès-Nancy Cedex, France, and Université Henri Poincaré Nancy I, France.*

Abstract

This paper presents complexity results by showing that the synthesis of **Max-Poly** quasi-interpretations over $\mathbb{R}^+$ is decidable in exponential time with fixed polynomial degrees and fixed max-degree and that the synthesis of **Max-Plus** quasi-interpretations over $\mathbb{R}^+$ is NPTIME-complete with fixed multiplicative degrees and fixed max-degree. Quasi-interpretations are a tool that allows to control resources like the runtime, the runspace or the size of a result in a program execution. Quasi-interpretations assign to each program symbol a numerical function which is compatible with the computational semantics and roughly speaking provide an upper bound on sizes of intermediate values computed.

The synthesis problem is to find a quasi-interpretation for a given program. We show that this problem is decidable in exponential time for the class of **Max-Poly** assignments over real numbers with fixed polynomial degree and max-degree. The class **Max-Poly** contains the projections, max, addition, multiplication operations and is closed by composition. This class is broad enough to cover a lot of practical algorithms.

Then we consider the class of **Max-Plus** of assignments which consists of projections, addition, the max operation and is closed by composition. We extend the work of Amadio on the synthesis of **Max-Plus** quasi-interpretation over real numbers by establishing that it is NPTIME-complete with fixed multiplicative degree and max-degree.

Email addresses: bonfante@loria.fr (G. Bonfante), Jean-Yves.Marion@loria.fr (J.-Y. Marion), moyen@loria.fr (J.-Y. Moyen), pechoux@loria.fr (R. Péchoux).

Preprint submitted to LCC 2005 *15 April 2005*

1 Introduction

Quasi-interpretations provide a method to perform a static analysis of the complexity of first order functional programs. It assigns to each function symbol of a program a monotonic function that ranges over non-negative real numbers and which admits a polynomial bound. A quasi-interpretation gives an upper bound on the values computed along the execution of a program. By combining quasi-interpretations with termination proofs, we capture major complexity classes [6–8,16]. The quasi-interpretation applies also to byte-code resource verification [2] and to reactive programming [3]. For that purposes we are motivated in finding a quasi-interpretation for a given program, if there is one. The reader may consult the submitted survey [7] which presents quasi-interpretations.

From a practical point of view, the interest of this approach lies on the fact that (i) the quasi-interpretation method applies to a broad class of programs, (ii) the resource analysis is made off-line, and (iii) the resource certificates can be sent with the program in the mobile code context. A first prototype has been presented at Rule workshop [17] which shows the feasibility of our approach. A key challenge is how to automatically synthesize quasi-interpretations. Over natural numbers or rational numbers, the synthesis problem is undecidable because Robinson [18] proved that the arithmetic of rational numbers is undecidable by giving an arithmetical definition of the integers in the rationals. However, there is some interest to find smaller classes of assignments for which it is easier to determine whether or not there are quasi-interpretations. For this reason, Amadio [1] considers the **Max-Plus** assignments as candidates to determine quasi-interpretations over rational numbers. The **Max-Plus** class is the class of **Max-Plus** assignments which consists in projections, the addition, the max operation, and which is closed by composition. Amadio [1] establishes that the synthesis problem is NPTIME-hard for **Max-Plus** assignments with bounded coefficients in $\mathbb{N}$ and bounded max-degree and NPTIME-complete for **Max-Plus**-multilinear assignments (assignments whose coefficients are in $\{0, 1\}$) of bounded max-degree.

In this paper, we suggest using quasi-interpretations over non-negative real numbers. We show that the synthesis problem of the **Max-Poly** class of assignments is decidable in exponential time, as a consequence of Tarski's theorem [19]. Whereas the general quantifier elimination procedure over reals has a complexity doubly exponential in the number of quantifier alternations, our problem remains exponential since we have only one alternation. The **Max-Poly** class of assignments is the smallest set of functions which contains projections, the addition, the multiplication, the max operation, and which is closed by composition. Actually, it appears that the class of **Max-Poly** quasi-interpretations is sufficient for daily programs. The complexity of the decision procedure is high, but we think that it is tractable because the quantification is bounded and the degree of the polynomials is at most 2 in practical cases. We demonstrate that over real numbers the synthesis problem remains NPTIME-complete for **Max-Plus** assignments of bounded degree and bounded max-degree. This result extends the one of Amadio and differs in the following respect. The quasi-interpretation domain is the set $\mathbb{R}$ of real numbers and

not the set $\mathbb{Q}$ of rational numbers. One should have expected another result because of the analogy with linear programming. Recall that over $\mathbb{R}$, linear programming is PTIME-complete and NPTIME-complete over $\mathbb{N}$. The proof used a different argument than the ones of [1]. Indeed, we cannot say that if $x + y \leq 1$ then we always have $x = 1$ and $y = 0$ or the inverse over $\mathbb{R}$, and this kind of argument is used in [1] when x and y range over $\mathbb{N}$.

The paper has the following organization. The next section defines a first order functional programming language. The section 3 introduces the notion of quasi-interpretation and call back to mind the main properties of quasi-interpretations. The section 4 concerns the decidability of the synthesis problem for the class of **Max-Poly**. Section 5 is devoted to the results of NPTIME-hardness and NPTIME-completeness over **Max-Plus**.

2 First order functional programming

Throughout the following discussion, we consider three disjoint sets $\mathcal{X}, \mathcal{F}, \mathcal{C}$ of variables, function symbols and constructors.

2.1 Syntax of programs

Definition 1 *The sets of terms and the rules are defined in the following way:*

(Constructor terms)	$\mathcal{T}(\mathcal{C}) \ni v$	$::=$	$\mathbf{c} \mid \mathbf{c}(v_1, \cdots, v_n)$
(terms)	$\mathcal{T}(\mathcal{C}, \mathcal{F}, \mathcal{X}) \ni t$	$::=$	$\mathbf{c} \mid x \mid \mathbf{c}(t_1, \cdots, t_n) \mid \mathtt{f}(t_1, \cdots, t_n)$
(patterns)	$\mathcal{P} \ni p$	$::=$	$\mathbf{c} \mid x \mid \mathbf{c}(p_1, \cdots, p_n)$
(rules)	$\mathcal{D} \ni d$	$::=$	$\mathtt{f}(p_1, \cdots, p_n) \rightarrow t$

where $x \in \mathcal{X}$, $\mathtt{f} \in \mathcal{F}$, and $\mathbf{c} \in \mathcal{C}$. We shall use a typewriter font for function symbols and a bold face font for constructors.

Definition 2 *A program is a quadruple $\mathtt{f} = \langle \mathcal{X}, \mathcal{C}, \mathcal{F}, \mathcal{E} \rangle$ such that $\mathcal{E}$ is a finite set of $\mathcal{D}$-rules. Each variable in the right-hand side of a rule also appears in the left hand side of the same rule. We distinguish among $\mathcal{F}$ a main function symbol whose name is given by the program name $\mathtt{f}$.*

Throughout, we consider orthogonal programs which is a sufficient condition in order to be confluent. Following Huet [11], the program rules satisfy both conditions: (i) Each rule $\mathtt{f}(p_1, \cdots, p_n) \rightarrow t$ is left-linear, that is a variable appears only once in $\mathtt{f}(p_1, \cdots, p_n)$, and (ii) there are no two left hand-sides which are overlapping.

$$\frac{\mathbf{c} \in \mathcal{C} \quad t_i \downarrow w_i}{\mathbf{c}(t_1, \cdots, t_n) \downarrow \mathbf{c}(w_1, \cdots, w_n)} \textit{ (Constructor)}$$

$$\frac{t_i \downarrow w_i \quad \mathtt{f}(p_1, \cdots, p_n) \rightarrow r \in \mathcal{E} \quad \sigma \in \mathfrak{S} \quad p_i\sigma = w_i \quad r\sigma \downarrow w}{\mathtt{f}(t_1, \cdots, t_n) \downarrow w} \textit{ (Function)}$$

Fig. 1. Call by value semantics with respect to a program $\langle \mathcal{X}, \mathcal{C}, \mathcal{F}, \mathcal{E} \rangle$

2.2 Semantics

The domain of the computed functions is the constructor algebra $\mathcal{T}(\mathcal{C})$. A substitution σ is a mapping from variables to terms. We say that it is a constructor substitution when the range of σ is $\mathcal{T}(\mathcal{C})$. We note $\mathfrak{S}$ the set of these constructor substitutions.

We consider a call by value semantics which is displayed in Figure 1. The meaning of $t \downarrow w$ is that t evaluates to the constructor term w. The program $\mathtt{f}$ computes a partial function $[\![\mathtt{f}]\!] : \mathcal{T}(\mathcal{C})^n \rightarrow \mathcal{T}(\mathcal{C})$ defined as follows. For all $v_i \in \mathcal{T}(\mathcal{C}), [\![\mathtt{f}]\!](v_1, \cdots, v_n) = w$ iff $\mathtt{f}(v_1, \cdots, v_n) \downarrow w$. Otherwise $[\![\mathtt{f}]\!](v_1, \cdots, v_n)$ is undefined.

3 Quasi-interpretations

The central question of this paper is the synthesis of quasi-interpretation. The notion of quasi-interpretation is related to the notion of polynomial interpretations for proving termination of term rewriting systems introduced by Lankford [12] over $\mathbb{N}$ and by Dershowitz [9] over $\mathbb{R}$ (see also recent works like [13] of Lucas). After that [5] establishes for the first time interpretation as a useful tool to ensure complexity bounds, quasi-interpretations have been introduced by Bonfante [4], Marion [14,15] and Marion-Moyen [16] .

The set of non-negative real numbers is noted $\mathbb{R}^+$.

Definition 3 (Assignment) *An assignment of a symbol $b \in \mathcal{F} \bigcup \mathcal{C}$ whose arity is n is a function $(\!|b|\!) : (\mathbb{R}^+)^n \rightarrow \mathbb{R}^+$ such that:*

(Subterm) $(\!|b|\!)(X_1, \cdots, X_n) \geq X_i$ *for all* $1 \leq i \leq n$.

(Weak Monotonicity) $(\!|b|\!)$ *is increasing (not necessarily strictly) with respect to each variable.*

We extend an assignment $(\!|-|\!)$ to terms canonically. Given a term t with n variables, the assignment $(\!|t|\!)$ is a function $(\mathbb{R}^+)^n \to \mathbb{R}^+$ defined by the rules:

$$(\!|b(t_1, \cdots, t_n)|\!) = (\!|b|\!)((\!|t_1|\!), \cdots, (\!|t_n|\!))$$
$$(\!|x|\!) = X$$

where X is a fresh variable ranging over reals.

Given two functions $P : (\mathbb{R}^+)^n \to \mathbb{R}^+$ and $Q : (\mathbb{R}^+)^m \to \mathbb{R}^+$ such that $n \geq m$, we say that $P \geq Q$ iff $\forall X_1, \ldots, X_n : P(X_1, \ldots, X_n) \geq Q(X_1, \ldots, X_m)$.

There are some well-known and useful consequences of such definitions. We have $(\!|s|\!) \geq (\!|t|\!)$ if t is a subterm of s. And, for every substitution σ, $(\!|s|\!) \geq (\!|t|\!)$ implies that $(\!|s\sigma|\!) \geq (\!|t\sigma|\!)$.

Definition 4 (Quasi-interpretation) *A program assignment $(\!|-|\!)$ is an assignment of each program symbol. An assignment $(\!|-|\!)$ of a program is a quasi-interpretation if for each rule $l \to r$,*

$$(\!|l|\!) \geq (\!|r|\!)$$

Example 5

Given a list l of tally natural numbers, reverse(l) reverses the elements of l. The constructor set is $\mathcal{C} = \{\mathbf{nil}, \mathbf{cons}\}$.

$$\begin{aligned} \mathrm{rev}(\mathbf{nil}, z) &\to z \\ \mathrm{rev}(\mathbf{cons}(x, y), z) &\to \mathrm{rev}(y, \mathbf{cons}(x, z)) \\ \mathrm{reverse}(l) &\to \mathrm{rev}(l, \mathbf{nil}) \end{aligned}$$

It admits the following quasi-interpretation.

- $(\!|\mathbf{nil}|\!) = 0$
- $(\!|\mathbf{cons}|\!)(X, Y) = X + Y + 1$
- $(\!|\mathrm{rev}|\!)(X, Y) = X + Y$
- $(\!|\mathrm{reverse}|\!)(X) = X$

Definition 6 *Let* $\mathbf{c}$ *be a constructor of arity* $n > 0$*. An assignment of* $\mathbf{c}$ *is* additive *if*

$$(\!|\mathbf{c}|\!)(X_1, \cdots, X_n) = \sum_{i=1}^{n} X_i + \alpha \qquad\qquad \alpha \geq 1$$

An assignment is additive if each assignment of constructor is additive.

Definition 7 *An assignment $(\!|-|\!)$ is said to be polynomial if for each symbol $b \in \mathcal{F} \bigcup \mathcal{C}$, $(\!|b|\!)$ is a function bounded by a polynomial. A quasi-interpretation $(\!|-|\!)$ is polynomial if the assignment $(\!|-|\!)$ is polynomial.*

All along, we shall always consider additive polynomial quasi-interpretations and consider programs which admit additive and polynomial quasi-interpretations. So, we often omit the adjectives additive and polynomial. When we write "quasi-interpretation", we always mean "additive polynomial quasi-interpretation".

3.1 Characterizing polynomial space and time bounded computation

The combination of the quasi-interpretation method with termination tools and particularly term orderings characterizes complexity classes starting from PTIME and PSPACE. We briefly list the main characterizations. The interested reader may consult the corresponding paper [8] in order to have details.

Definition 8 *A RPO^{QI}-program is a program that (i) admits a quasi-interpretation and (ii) which terminates by Recursive Path Ordering.*

Definition 9 *A PPO^{QI}-program is a program that (i) admits a quasi-interpretation and (ii) which terminates by Product Path Ordering.*

Theorem 10 (Bonfante, Marion, Moyen [6]) *The set of functions computed by RPO^{QI}-programs is exactly the set of functions computable in* polynomial space.

Theorem 11 (Marion, Moyen [16]) *The set of functions computed by PPO^{QI}-programs with an quasi-interpretation is exactly the set of functions computable in* polynomial time.

4 Decidability of the synthesis problem for Max-Poly

In this section, we shall see that finding quasi-interpretations over reals is solvable. It is a consequence of Tarski's Theorem [19]. For this, we shall consider a logical formulation of the hypotheses of quasi-interpretation.

Definition 12 *The* synthesis problem *is as follows:*

inputs: *A program* $\mathtt{f}$.

problem: *Is there an assignment* $(\!|-|\!)$ *which is a quasi-interpretation for* $\mathtt{f}$ *?*

Definition 13 *The class of* ***Max-Poly*** *functions contains constant functions ranging over real numbers projections,* max*, addition, multiplication and is closed by composition.*

Before proceeding to the main discussion, it is convenient to have a normal representation of functions in **Max-Poly**.

Proposition 14 (Normalization) *Any* ***Max-Poly*** *function Q can be written*

$$Q(X_1, \ldots, X_n) = \max(P_1(X_1, \ldots, X_n), \ldots, P_k(X_1, \ldots, X_n))$$

where each P_i is a polynomial.

PROOF. This is due to the fact that max is distributive with $+$ and $\times$ over the reals. □

Definition 15 (Degrees) *We say that the* max*-degree of Q is k and the degree of Q is the maximum degree of the polynomials $P_1, \ldots, P_k$, also noted $\times$-degree. We recall that the degree of $X_1^{i_1} X_2^{i_2} ... X_n^{i_n}$ is $\sum_{j=1}^{n} i_j$.*

The quasi-interpretation of Example 5 belongs to the class **Max-Poly**.

Definition 16 *A* ***Max-Poly*** *assignment belongs to the class* ***(k,d)-Max-Poly*** *if its $\times$-degree and its* max*-degree are respectively bounded by constants d and k.*

Theorem 17 *Given constants k and d, the synthesis problem for* ***(k,d)-Max-Poly*** *assignments is decidable in exponential time in the size of the program.*

PROOF. A sketch of the proof is given here. The complete proof is in the appendix A. First we try to encode the synthesis problem into the synthesis of a first order formula. This transformation is done by eliminating the max operator. For example, a formula of the form $\max(p_1, \cdots, p_n) \geq \max(q_1, \cdots, q_m)$ is equivalent to $\wedge_{j=1}^{m} \vee_{i=1}^{n} p_i \geq q_j$. After that step, we try to eliminate quantifiers from a prenex form of our first order formula. This result due to Tarski [19] is known to have a complexity doubly exponential in the number of alternations of quantifiers. Since this number is bounded by a constant in the synthesis problem, we obtain the decidability of our problem with an exponential time complexity. □

Remark 18 *In practice, each program appears to admit a* ***Max-Poly*** *quasi-interpretation with low degrees, usually no more than* 2 *for the degree of polynomials, the arity of* max *being bounded by 2.*

5 Synthesis of Max-Plus quasi-interpretations

There are at least two reasons to consider the synthesis problem for subclasses of **Max-Poly** functions. First, the procedure above has a great complexity. Second, it does not ensure that the assignment is optimal. Indeed, as [8] shows it, restriction over quasi-interpretations may lead to some lower bounds on the complexity of a program. For instance, Amadio [1] considered the max-plus algebra over rational numbers. Amadio established that the synthesis of **Max-Plus** quasi-interpretation is in NPTIME-hard and NPTIME-complete in the case of multi-linear assignments.

We demonstrate that it is NPTIME-complete for **Max-Plus** assignments over $\mathbb{R}^+$ under some slight restrictions over degrees.

Definition 19 *The class of* ***Max-Plus***$\{\boldsymbol{K}\}$ *assignments contains constant functions ranging over* $\boldsymbol{K}$*, a subset of reals and is closed by projections,* max*, addition and composition and must verify that each constructor assignment is* additive*.*

Definition 20 *We define the* $+$*-degree of a polynomial to be its maximal multiplicative coefficient.*

The following result is due to R. Amadio [1]:

Theorem 21 *The synthesis problem for* ***Max-Plus***$\{\boldsymbol{K}\}$ *assignments of bounded* $+$*-degree and bounded* max*-degree and with variables ranging over* $\mathbb{Q}^+$ *is* NPTIME*-hard if* $\boldsymbol{K} = \mathbb{N}$ *and* NPTIME*-complete if* $\boldsymbol{K} = \{0, 1\}$*.*

Now we are going to demonstrate that it is NPTIME-complete for **Max-Plus** assignments over $\mathbb{R}^+$ (under some slight restrictions over degrees). The proof of NPTIME-hardness is largely inspired by Amadio's proof which reduces a 3-CNF satisfiability problem into a synthesis problem for a system of rules. However it differs for technical reasons: The corresponding rules allow the author to encode every disjunction using properties available over natural numbers, but no longer available over reals. For example, the following property is required by Amadio: $\sum_{i=1}^{n} \alpha_i = 1 \Rightarrow (\exists j$ such that $\alpha_j = 1$ and $\forall k \neq j\ \alpha_k = 0)$. In the proof we add new rules to ensure a correct encoding over reals.

Theorem 22 *The synthesis problem for* ***Max-Plus***$\{\mathbb{R}^+\}$ *assignments of bounded* $+$*-degree and bounded* max*-degree and with variables ranging over* $\mathbb{R}^+$ *is* NPTIME*-complete.*

PROOF. A sketch of the proof is given here. The complete proof is in the appendix B. We procede by reducing a 3-CNF problem into a synthesis problem for **Max-Plus**. The reduction follows Amadio [1]. The main difference is that this property: $\sum_{i=1}^{n} \alpha_i = 1 \Rightarrow (\exists j$ such that $\alpha_j = 1$ and $\forall k \neq j\ \alpha_k = 0)$ no longer holds over reals or rationals. However it is required in Amadio's proof to encode literals into a synthesis problem. Indeed a function symbol $\mathtt{f}$ having a quasi-interpretation $(\!|\mathtt{f}|\!) = \alpha_1 X_1 + \alpha_2 X_2$ with $(\alpha_1 = 1 \wedge \alpha_2 = 2) \vee (\alpha_1 = 2 \wedge \alpha_2 = 1)$ is associated to each literal x. Thus we have to find a new encoding. We eliminate this problem by adding new rules:

$$\mathtt{f}(x_1, \ldots, x_n) \rightarrow \mathtt{f}(\mathtt{f}(x_1, \ldots, x_n), \ldots, \mathtt{f}(x_1, \ldots, x_n))$$

$$\mathtt{f}(\mathbf{d}(x_1, \mathbf{0}), \ldots \mathbf{d}(x_n, \mathbf{0})) \rightarrow \mathtt{f}(\mathbf{d}(x_1, \mathtt{f}(\mathbf{0}, \ldots \mathbf{0})), \ldots \mathbf{d}(x_n, \mathtt{f}(\mathbf{0}, \ldots, \mathbf{0})))$$

in the reduction. They give sufficient conditions on the considered assignments to allow the encoding of a 3-CNF formula in a synthesis problem. Finally, we show that the problem belongs to NPTIME by showing that we can eliminate the max operator. Thus we obtain a problem of linear programming. The result follows since linear programming is PTIME-complete over reals. $\square$

References

[1] R. Amadio. Synthesis of max-plus quasi-interpretations. Research Report LIF, 2004. Fundamenta Informaticae, 65(1–2), 2005. `http://cmi.univ-mrs.fr/~amadio/qsynthesis-rr.ps.gz`.

[2] R. Amadio, S. Coupet-Grimal, S. Dal-Zilio, and L. Jakubiec. A functional scenario for bytecode verification of resource bounds. In Jerzy Marcinkowski and Andrzej Tarlecki, editors, *Computer Science Logic, 18th International Workshop, CSL 13th Annual Conference of the EACSL, Karpacz, Poland*, volume 3210 of *Lecture Notes in Computer Science*, pages 265–279. Springer, 2004.

[3] R. Amadio and S. Dal-Zilio. Resource control for synchronous cooperative threads. In *CONCUR*, volume 3170 of *Lecture Notes in Computer Science*, pages 68–82. Springer, 2004.

[4] G. Bonfante. *Constructions d'ordres, analyse de la complexité*. Thèse, Institut National Polytechnique de Lorraine, 2000.

[5] G. Bonfante, A. Cichon, J.-Y. Marion, and H. Touzet. Complexity classes and rewrite systems with polynomial interpretation. In *Computer Science Logic, 12th International Workshop, CSL'98*, volume 1584 of *Lecture Notes in Computer Science*, pages 372–384, 1999.

[6] G. Bonfante, J.-Y. Marion, and J.-Y. Moyen. On lexicographic termination ordering with space bound certifications. In Dines Bjørner, Manfred Broy, and Alexandre V. Zamulin, editors, *Perspectives of System Informatics, 4th International Andrei Ershov Memorial Conference, PSI 2001, Akademgorodok, Novosibirsk, Russia, Ershov Memorial Conference*, volume 2244 of *Lecture Notes in Computer Science*. Springer, Jul 2001.

[7] G. Bonfante, J.-Y. Marion, and J.-Y. Moyen. Quasi-interpretation a way to control resources. *Submitted to Theoretical Computer Science*, 2005. `http://www.loria.fr/~moyen/appsemTCS.ps`.

[8] G. Bonfante, J.-Y. Marion, and J.-Y. Moyen. Quasi-interpretation and small space bounds. In *16th International Conference on Rewriting Techniques and Applications*, Nara, Japan, April 2005.

[9] N. Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, pages 69–115, 1987.

[10] J. Heintz, M.-F. Roy, and P. Solerno. Sur la complexité du principe de Tarski-Seidenberg. *Bulletin de la S.M.F., tome 118*, pages 101–126, 1990.

[11] G. Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27(4):797–821, 1980.

[12] D.S. Lankford. On proving term rewriting systems are noetherien. Technical report, 1979.

[13] S. Lucas. Polynomials over the reals in proofs of termination: from theory to practice. *RAIRO Theoretical Informatics and Applications*, 2005. to appear.

[14] J.-Y. Marion. *Complexité implicite des calculs, de la théorie à la pratique*. Habilitation à diriger les recherches, Université Nancy 2, 2000.

[15] J.-Y. Marion. Analysing the implicit complexity of programs. *Information and Computation*, 183:2–18, 2003.

[16] J.-Y. Marion and J.-Y. Moyen. Efficient first order functional program interpreter with time bound certifications. In Michel Parigot and Andrei Voronkov, editors, *Logic for Programming and Automated Reasoning, 7th International Conference, LPAR 2000, Reunion Island, France*, volume 1955 of *Lecture Notes in Computer Science*, pages 25–42. Springer, Nov 2000.

[17] J.-Y. Moyen. System Presentation: An analyser of rewriting systems complexity. In Mark van den Brand and Rakesh Verma, editors, *Electronic Notes in Theoretical Computer Science*, volume 59. Elsevier Science Publishers, 2001. Workshop RULE, accessible `http://www.elsevier.com/gej-ng/31/29/23/89/33/37/59.4.011.ps`.

[18] J. Robinson. Definability and decision problems in arithmetic. In *The Journal of Symbolic Logic, Volume 14*, pages 98–114. 1949.

[19] A. Tarski. *A Decision Method for Elementary Algebra and Geometry, 2nd ed.* University of California Press, 1951.

A Decidability of the synthesis problem for Max-Poly

Proof of Theorem 17:

Given constants k and d, the synthesis problem for **(k,d)-Max-Poly** assignments is decidable in exponential time in the size of the program.

PROOF. Suppose we have a program $\langle \mathcal{X}, \mathcal{C}, \mathcal{F}, \mathcal{E} \rangle$. We have to show that the problem of synthesis for **Max-Poly** reduces to the satisfiability of a first-order formula.

Notation 23 *Given an n-ary symbol, its assignments of max-degree k and $\times$-degree d have the form:*

$$(\!|\mathtt{f}|\!)(X_1, \ldots, X_n) = \max(P[\mathtt{f}, 1](X_1, \ldots, X_n), \ldots, P[\mathtt{f}, k](X_1, \ldots, X_n))$$

where $P[\mathtt{f}, i]$ are polynomials of degree at most d. In other words,

$$P[\mathtt{f}, i](X_1, \ldots, X_n) = \sum a[\mathtt{f}, i, j_1, \ldots, j_n] X_1^{j_1} \times \cdots \times X_n^{j_n}$$

with $1 \leq i \leq k$ and $1 \leq \ell \leq n$ in $0 \leq j_\ell \leq d$, $j_\ell \in \mathbb{N}$.

So, for all symbols, we introduce new (logical) variables $a[\mathtt{f}, i, j_1, \ldots, j_n]$.

Definition 24 *[Subterm property] Suppose that $\mathtt{f}$ is a n-ary symbol, the subterm property can be expressed by the formula:*

$$S[\mathtt{f}] \triangleq \bigwedge_{j=1}^{n} S[\mathtt{f}, j]$$

with

$$S[\mathtt{f}, j] \triangleq \forall X_1, \ldots, X_n \geq 0 : \bigvee_{i=1}^{k} P[\mathtt{f}, i](X_1, \ldots, X_n) \geq X_j$$

Definition 25 *[Weak monotonicity] Take $\mathtt{f}$ as above. Recalling the definition of* max*, weak monotonicity is expressed by:*

$$M[\mathtt{f}] \triangleq \forall X_1, \ldots, X_n : \forall\, 0 \leq Y_1 \leq X_1, \ldots, 0 \leq Y_n \leq X_n :$$
$$\textstyle\bigwedge_{j=1..k} \bigvee_{i=1..n} P[\mathtt{f}, i](X_1, \ldots, X_n) \geq P[\mathtt{f}, j](Y_1, \ldots, Y_n)$$

Definition 26 *[Additivity of constructors] Take* $\mathbf{c}$ *a constructor symbol. To express the additivity, consider the formula:*

$$\begin{aligned} C[\mathbf{c}] \triangleq\ & a[\mathbf{c}, 1, 0, \ldots, 0] \geq 1 \quad \wedge \\ & \textstyle\bigwedge_{\Sigma j_\ell = 1} a[\mathbf{c}, 1, j_1, \ldots, j_n] = 1 \quad \wedge \\ & \textstyle\bigwedge_{\Sigma j_\ell > 1} a[\mathbf{c}, 1, j_1, \ldots, j_n] = 0 \quad \wedge \\ & \textstyle\bigwedge_{i=2..k, j_1, \ldots, j_n} a[\mathbf{c}, i, j_1, \ldots, j_n] = 0 \end{aligned}$$

Definition 27 *The size $|t|$ of a term t is the number of symbols in t.*

Proposition 28 *Given an assignment $(\!|-|\!)$ of max-degree k and $\times$-degree d and given a term t of size m, the max-degree of $(\!|t|\!)$ is at most k^m and its $\times$-degree is bounded by d^m.*

PROOF. The two results are obtained by induction on the size of t. □

Notation 29 *Propositions 14 and 28 show that the expressions $P[\mathtt{f}, i]$ can be extended to terms. In other words, we write*

$$(\!|t|\!)(X_1, \ldots, X_n) = \max(P[t, 1](X_1, \ldots, X_n), \ldots, P[t, k'](X_1, \ldots, X_n))$$

with $k' \leq k^{|t|}$ and the $\times$-degree of t is bounded by $d^{|t|}$, if the assignment is of max-degree k and $\times$-degree d.

Definition 30 *[Quasi-interpretation] Now consider a **Max-Poly** assignment $(\!|-|\!)$ of a program $\mathtt{f}$ whose max-degree is k. Take a rule $l \to r$ and define*

$$Q[l \to r] \triangleq \forall X_1, \ldots X_p \geq 0 : \bigwedge_{j=1..l} \bigvee_{i=1..n} P[l, i](X_1, \ldots, X_p) \geq P[r, j](X_1, \ldots, X_p)$$

with $n \leq k^{|l|}$ and $l \leq k^{|r|}$ (cf. proposition 28).

Proposition 31 *The first order formula $Q[l \to r]$ is true iff $(\!|l|\!) \geq (\!|r|\!)$.*

Using Definitions 24,25,26,30, we introduce the first order formula [1]

$$\begin{aligned} F[\langle \mathcal{X}, \mathcal{C}, \mathcal{F}, \mathcal{E} \rangle] \triangleq \exists_{p \in \mathcal{C} \cup \mathcal{F}, 1 \leq i \leq k, 0 \leq j_1, \ldots, j_n \leq d}\, a[p, i, j_1, \ldots, j_n] : \\ \textstyle\bigwedge_{\mathbf{c} \in \mathcal{C}} C[\mathbf{c}] \quad \wedge \\ \textstyle\bigwedge_{\mathbf{g} \in \mathcal{F}} (S[\mathbf{g}] \wedge M[\mathbf{g}]) \quad \wedge \\ \textstyle\bigwedge_{l \to r \in \mathcal{E}} Q[l \to r] \end{aligned}$$

After a renaming of variables involved in the different inequalities: we can skolemize such a formula. We obtain a new one with an alternation of one block of existencial quantifiers for the coefficients of polynomials and one block of universal quantifiers for the variables. Now we state a quantifier elimination theorem over reals taken from [10] relying on the of Tarski principle and proving our decidability result:

Theorem 32 *Suppose we have a cancellation ring $\boldsymbol{K}$ included in a closed real field $\boldsymbol{R}$ and we have a formula ϕ composed of inequalities and equalities of polynomials with coefficients in $\boldsymbol{K}$ including existencial and universal quantifiers binding variables over $\boldsymbol{R}$. Moreover suppose that ϕ is under prenex form with m the number of*

[1] n is the maximal arity of a symbol.

alternations of quantifiers blocks and D the sum of the $\times$-degree of every polynomial involved in ϕ. Let n be the number of variables and L the size of the boolean combination. There exists an algorithm of complexity $O(L)D^{n^m}$ that computes an equivalent quantifier-free formula.

Since we consider the multiplicative coefficients $a[f, i, j_1, \ldots, j_n]$ to be variables, the new multiplicative coefficients are all equal to 1. As a consequence we can take $\mathbf{K}$ to be $\mathbb{Q}$. Let ϕ be the prenex form of $F[\langle \mathcal{X}, \mathcal{C}, \mathcal{F}, \mathcal{E} \rangle]$, we know by hypothesis that the $\times$-degree of polynomials is bound by a constant d and the max-degree is bound by a constant k. As a consequence $D \leq d \times L$. The number m of alternations of quantifiers in ϕ being equal to 1, it remains to find bounds on L and n which are respectively the size of the boolean formula and the number of variables. One can remark that finding a bound for L is the same as finding the number of inequalities added in ϕ. Let i be the maximal arity of a function, Δ the cardinal of $\mathcal{C} \cup \mathcal{F}$ and α the maximal size of a rule.

- Definition 24 introduces at most $\Delta \times i \times k$ inequalities and adds at most $\Delta \times i$ universaly quantified variables.
- Definition 25 introduces at most $\Delta \times k^2$ inequalities and adds at most $2i \times \Delta$ new universaly quantified variables.
- Definition 26 introduces at most $\Delta \times k \times i^{d+1}$ inequalities and adds 0 new universaly quantified variables. Indeed we observe that $\#\{(j_1, ..., j_n) | \sum_{l=1}^{n} j_l \leq d\} \leq n^{d+1}$.
- Definition 30 introduces at most $\Delta \times k^{\alpha}$ inequalities and adds at most $\Delta \times i$ new universaly quantified variables.
- the last formula above introduces 0 inequalities and at most $\Delta \times k \times i^{d+1}$ existentially quantified variables.

Finally, if P is the size of the program we have $L = O(\Delta \times (i \times k + k^2 + k \times i^{d+1} + k^{\alpha})) = O(P \times k^P)$ for k large enough and $n = O(\Delta \times (4i + k \times i^{d+1})) = O(P^{d+2})$. Thus, since $D \leq d \times L$ and applying the theorem 32, we can eliminate the quantifiers with a procedure whose complexity is in $O(P \times k^P) \times (O(P \times k^P))^{O(P^{d+2})^{O(1)}}$. Since the elimination of quantifiers consists in eliminating variables, we obtain a result to our synthesis problem, if there is one. To conclude we have demonstrated that the synthesis problem over **Max-Poly** for bounded $\times$-degree and max-degree is decidable in exponential time in the size of the program. $\square$

B NPTIME-completeness of the synthesis problem for Max-Plus

Proof of Theorem 22: The synthesis problem for **Max-Plus** $\{\mathbb{R}^+\}$ assignments of bounded $+$-degree and bounded max-degree and with variables ranging over $\mathbb{R}^+$ is NPTIME-complete.

PROOF. The proof of the theorem is inspired by Amadio's proof of previous theorem but differs in the sense that Amadio used the following constraints on natural numbers $\sum_{i=1}^{n} \alpha_i = 1 \Rightarrow (\exists j$ such that $\alpha_j = 1$ and $\forall k \neq j\ \alpha_k = 0)$. Such a constraint is no more available in $\mathbb{R}^+$ and we have to find more rules to compensate this fact. It is divided in two steps. First, to prove the NPTIME-hardness of the problem, we encode every 3-CNF problem in a **Max-Plus** $\{\mathbb{R}^+\}$ synthesis problem. Then, we prove its NPTIME-completeness by showing that the verification problem is NPTIME in **Max-Plus** $\{\mathbb{R}^+\}$. Throughout the following proof, every **Max-Plus** $\{\mathbb{R}^+\}$ assignment will be written $\max_{i\in I}(\sum_{j=1}^{n} \alpha_{i,j}X_j + a_i)$, as a consequence of Proposition 14, with the size of I bounded by the max-degree and $\alpha_{i,j}$, a_i in $\mathbb{R}^+$ bounded by the degree. □

Proposition 33 *Given an assignment* $\max_{i\in I}(\sum_{j=1}^{n} \alpha_{i,j}X_j + a_i)$*, we have*

(1) for all $j \leq n$*, there is* $i \in I$ *such that* $\alpha_{i,j} \geq 1$ *(Subterm Property)*

(2) for all $i \in I$*,* $\sum_{j=1}^{n} \alpha_{i,j} \geq 0$ *(Weak Monotonicity Property)*

B.1 NPTIME-*hardness*

Proposition 34 *[Forcing interpretation] Given a function symbol* $\mathtt{f}$*, we can find rules such that one of the two following conditions applies:*

(1) $(\!|f|\!)(X_1, \ldots, X_n) = \max(X_1, \ldots, X_n)$

(2) $(\!|f|\!)(X_1, \ldots, X_n) = \max_{i\in I}(\sum_{j=1}^{n} \alpha_{i,j}X_j)$ *whatever* $\alpha_{i,j}$ *are.*

PROOF. We begin by (1). Let us consider the rule:

$$e_1 \equiv \mathtt{f}(x_1, \ldots, x_n) \rightarrow \mathtt{f}(\mathtt{f}(x_1, \ldots, x_n), \ldots, \mathtt{f}(x_1, \ldots, x_n)) \equiv e_1^{'}$$

A quasi-interpretation of this rule must verify $(\!|e_1|\!) \geq (\!|e_1^{'}|\!)$. This condition implies that $(\!|e_1|\!) \geq (\!|e_1^{'}|\!) = \max_{i\in I}((\sum_{j=1}^{n} \alpha_{i,j})(\!|e_1|\!) + a_i)$. As a consequence, for all $i \in I$, $\sum_{j=1}^{n} \alpha_{i,j} \leq 1$. Due to Proposition 33, we have for all j, a k for which $\alpha_{k,j} \geq 1$. With the previous inequality, this implies $\alpha_{k,i} = 0$ for all $i \neq j$.

Thus, the interpretation has the form

$$(\!|\mathtt{f}|\!) = \max(X_1 + a_{i_1}, \ldots, X_n + a_{i_n}, \sum_{j=1}^{n} \alpha_{i,j}X_j + a_i)$$

with $\sum_{j=1}^{n} \alpha_{i,j} \leq 1$. For sufficiently large X, we have $(\!|e_1|\!)(X, 0, \ldots, 0) = X + a_{i_1}$, the subterm property shows that $a_{i_1} \geq 0$. But, at the same time, due to the rule, we have $(\!|e_1|\!)(X, 0, \ldots, 0) = X + a_{i_1} \geq (\!|e_1'|\!)(X, 0, \ldots, 0) \geq X + 2a_{i_1}$ which implies

$a_{i_1} \leq 0$. The weak monotonicity property of Proposition 33 implies that $a_{i_1} = 0$. The same can be said for every other a_{i_k}. So, the interpretation is actually of the form $(\!|\mathtt{f}|\!) = \max(X_1, \ldots, X_n, \sum_{j=1}^{n} \alpha_{i,j} X_j + a_i)$ with $\sum_{j=1}^{n} \alpha_{i,j} \leq 1$. Now, we add the rule:

$$e_2 \equiv \mathtt{f}(\mathbf{d}(x_1, \mathbf{0}), \ldots \mathbf{d}(x_n, \mathbf{0})) \rightarrow \mathtt{f}(\mathbf{d}(x_1, \mathtt{f}(\mathbf{0}, \ldots \mathbf{0})), \ldots \mathbf{d}(x_n, \mathtt{f}(\mathbf{0}, \ldots, \mathbf{0}))) \equiv e_2^{'}$$

With the second rule, $(\!|-|\!)$ has to verify: $(\!|e_2|\!) = \max_{i \in I}((\sum_{j=1}^{n} \alpha_{i,j})(X + a^d) + a_i) \geq \max_{i \in I}((\sum_{j=1}^{n} \alpha_{i,j})(X + a^d + \max_{k \in I}(a_k)) + a_i) = (\!|e_2^{'}|\!)$ when $X_1 = \ldots = X_n = X$. Let $\sum_{j=1}^{n} \alpha_{l,j} = \max_{i \in I}(\sum_{j=1}^{n} \alpha_{i,j})$. For X large enough and since $\sum_{j=1}^{n} \alpha_{l,j} = 1$ and $a_l = 0$, we have: $X + a^d \geq X + a^d + \max_{k \in I}(a_k)$ this is equivalent to $a_k = 0 \ \forall k \in I$. Finally, we obtain that:

$$(\!|\mathtt{f}|\!) = \max(\max(X_1, \ldots, X_n), \max(\textstyle\sum_{j=1}^{n} \alpha_{i,j} X_j)) \text{ with } \sum_{j=1}^{n} \alpha_{i,j} \leq 1.$$

Since $\forall X_1, \ldots, \forall X_n \in \mathbb{R}^+$, $\max(X_1, \ldots, X_n) \geq \max(\sum_{j=1}^{n} \alpha_{i,j} X_j)$:

$$(\!|\mathtt{f}|\!) = \max(X_1, \ldots, X_n)$$

To prove condition (2) of the proposition, we add a further rule:

$$\mathtt{m}(\mathbf{c}(x_1, \ldots, x_n)) \rightarrow \mathbf{c}(\mathtt{g}(\mathbf{0}, \ldots, \mathbf{0}), \mathbf{0}, \ldots, \mathbf{0})$$

with $\mathtt{m}$ a function such that $(\!|\mathtt{m}(x)|\!) = X$ (such a function exists according to the previous rules). The corresponding assignment has to verify that $a^c + \sum_{j=1}^{n} X_j \geq a^c + \max_{k \in I}(a_k)$ which is equivalent to $a_k = 0 \ \forall k \in I$. Thus, $(\!|\mathtt{g}|\!) = \max_{i \in I}(\sum_{j=1}^{n} \alpha_{i,j} X_j)$ and (2) is proved. $\Box$

Remark 35 *We can force the equality of the additive coefficients of constructor assignments by adding the following rules:*

$$m(\mathbf{c}(x, \boldsymbol{0}, \ldots, \boldsymbol{0})) \rightarrow \mathbf{d}(x, \boldsymbol{0}, \ldots, \boldsymbol{0})$$
$$m(\mathbf{d}(x, \boldsymbol{0}, \ldots, \boldsymbol{0})) \rightarrow \mathbf{c}(x, \boldsymbol{0}, \ldots, \boldsymbol{0})$$

Theorem 36 *The synthesis problem for* ***Max-Plus*** $\{\mathbb{R}^+\}$ *assignments of bounded degree is* NPTIME*-hard.*

PROOF. As a consequence of the previous proposition, we can force a function $\mathtt{f}$ of arity 2 to have the quasi-interpretation below:

$$(\!|\mathtt{f}|\!) = \max_{i \in I}(\alpha_{i,1} X_1 + \alpha_{i,2} X_2)$$

Throughout the following proof, we consider (cf. previous Remark) that constructors have the same additive coefficient k in their respective assignment and we define α_j as the $\max_{i \in I}(\alpha_{i,j})$ for $j \in \{1, 2\}$ and α as the $\max_{i \in I}(\alpha_{i,1} + \alpha_{i,2})$. Trivially, we have that $\alpha_1 + \alpha_2 \geq \alpha$. Now, adding the rule:

$$\mathtt{f}(\mathbf{d}(x_1, \mathbf{0}), \mathbf{d}(x_2, \mathbf{0})) \rightarrow \mathbf{d}(\mathtt{f}(x_1, \mathbf{0}), \mathtt{f}(\mathbf{0}, x_1))$$

For $X_1 = X_2$, we obtain that $\alpha(X_1 + k) \geq k + (\alpha_1 + \alpha_2)X_1$. As a consequence and for X_1 large enough, $\alpha = \alpha_1 + \alpha_2$. A direct result is that:

$$(\!|\mathtt{f}|\!) = \alpha_1 X_1 + \alpha_2 X_2$$

since $\exists j \in I$ such that $\alpha_{j,1} = \alpha_1$ and $\alpha_{j,2} = \alpha_2$ which implies that $\forall X_1, X_2 \in \mathbb{R}^+,\ \forall i \in I\ \alpha_{j,1}X_1 + \alpha_{j,2}X_2 \geq \alpha_{i,1}X_1 + \alpha_{i,2}X_2$. Now, we are able to prove that we can associate rules to $\mathtt{f}$ to constrain α_1 and α_2 to the following condition

$$(\alpha_1 = 1 \wedge \alpha_2 = 2) \vee (\alpha_1 = 2 \wedge \alpha_2 = 1)$$

By virtue of the subterm condition, $\alpha_1, \alpha_2 \geq 1$. We add the rules below:

$$\begin{aligned} \mathtt{f}(\mathbf{c}(x_1), \mathbf{c}(x_2)) &\to \mathbf{c}(\mathbf{c}(\mathbf{c}(\mathbf{0}))) \\ \mathtt{m}(\mathbf{c}(\mathbf{c}(\mathbf{c}(x)))) &\to \mathtt{f}(\mathbf{c}(\mathbf{0}), \mathbf{c}(\mathbf{0})) \end{aligned}$$

They imply that $\alpha_1 + \alpha_2 = 3$. Now, adding the rule:

$$\mathtt{m}(\mathbf{c}(\mathbf{c}(x))) \to \mathtt{f}(\mathtt{f}(\mathbf{0}, \mathbf{c}(\mathbf{0})), \mathbf{0})$$

We obtain that $2k + x \geq \alpha_1\alpha_2 k$. In other words, $2 \geq \alpha_1\alpha_2$. Since, $\alpha_1 = 3 - \alpha_2$, we want the following inequality to be satsified $\alpha_1^2 - 3\alpha_1 + 2 \geq 0$ with $2 \geq \alpha_1 \geq 1$. The only solutions satifying this inequality are $(\alpha_1 = 1 \wedge \alpha_2 = 2) \vee (\alpha_1 = 2 \wedge \alpha_2 = 1)$. What we were expecting. Now we want to encode the satisfiability of a 3-SAT problem under 3-CNF in a synthesis problem. For that purpose, we associate to each literal x_i appearing in a 3-CNF formula ϕ a function $\mathtt{f}_i$ such that $(\!|\mathtt{f}_i|\!) = \alpha_1^i X_1 + \alpha_2^i X_2$. The table below shows the different values taken by the interpretation $(\!|\mathtt{f}_i|\!)$ in function of its coefficients and of its input arguments:

Coefficients of $(\!\|\mathtt{f}_i\|\!)$: (α_1^i, α_2^i)	Arguments: (x_1, x_2)	Computed interpretation: $(\!\|\mathtt{f}_i\|\!)$
(1,2)	(c(0),0)	k
(1,2)	(0,c(0))	2k
(2,1)	(c(0),0)	2k
(2,1)	(0,c(0))	k

We suppose that the constant k (respectively $2k$) is the encoding for the truth value **True** (respectively **False**). We want the evaluation of a literal to the truth value **True** (respectively **False**) to be encoded by the evaluation of the function $\mathtt{f}_i$ quasi-interpretation in $X_1 + 2X_2$ (respectively $2X_1 + X_2$). Given a disjunction D of the formula ϕ, there are two basic cases in the encoding of its first literal:

- If the first literal of D is x_i, we associate the arguments $(\mathbf{c}(\mathbf{0}), \mathbf{0})$ to the function $\mathtt{f}_i$. In this case, we have $(\!|\mathtt{f}_i(\mathbf{c}(\mathbf{0}), \mathbf{0})|\!) = \alpha_1 k$. As a result, $(\!|\mathtt{f}_i|\!)$ will correspond to the truth value **True** if and only if $\alpha_1 = 1$, in other words, if and only if $(\!|\mathtt{f}_i|\!) = X_1 + 2X_2$.

- If the first literal of D is $\neg x_i$, we associate the arguments $(\mathbf{0}, \mathbf{c}(\mathbf{0}))$ to the function $\mathtt{f}_i$. In this case, we have $(\!|\mathtt{f}_i(\mathbf{c}(\mathbf{0}), \mathbf{0})|\!) = \alpha_2 k$. As a result, $(\!|\mathtt{f}_i|\!)$ will correspond to the truth value **True** if and only if $\alpha_2 = 1$, in other words, if and only if $(\!|\mathtt{f}_i|\!) = 2X_1 + X_2$.

In order to synthesize the previous ideas, we define $arg(x, D)$ to be the arguments of the function encoding x in the disjunction D:

$$arg(x, D) = \begin{cases} (\mathbf{c}(\mathbf{0}), \mathbf{0}) \text{ if } x \text{ appears in } D \\ (\mathbf{0}, \mathbf{c}(\mathbf{0})) \text{ if } \neg x \text{ appears in } D \end{cases}$$

$(\!|\mathtt{f}(arg(x, D))|\!)$ is equal to k if $(\!|\mathtt{f}|\!)$ corresponds to **True** and x appears in D or if $(\!|\mathtt{f}|\!)$ corresponds to **False** and $\neg x$ appears in D. $(\!|\mathtt{f}(arg(x, D))|\!)$ is equal to $2k$ if $(\!|\mathtt{f}|\!)$ corresponds to **True** and $\neg x$ appears in D or if $(\!|\mathtt{f}|\!)$ corresponds to **False** and x appears in D. Now, it remains to encode the disjunctions. For that purpose, we need a function $\mathtt{s}_n$ of arity n whose quasi-interpretation verifies:

$$(\!|\mathtt{s}_n|\!) = \alpha_1 X_1 + \cdots + \alpha_n X_n$$

This quasi-interpretation is obtained by adding the n following rules:

$$\mathtt{m}(\mathbf{c}(x)) \to \mathbf{c}(\mathtt{s}_n(\mathbf{0}, \ldots, \mathbf{0}, x, \mathbf{0}, \ldots \mathbf{0}))$$

with x at the i-th position in the right member for $i = 1..n$. Finally, we associate the rule to every disjunction D of the formula ϕ containing the literals x_i, x_j and x_k:

$$\mathtt{m}(\mathbf{c}(\mathbf{c}(\mathbf{c}(\mathbf{c}(\mathbf{c}(x)))))) \to \mathtt{s}_3(\mathtt{f}_i(arg(x_i, D), \mathtt{f}_j(arg(x_j, D), \mathtt{f}_k(arg(x_k, D))$$

whose interpretation is:

$$5k + X \geq (\!|\mathtt{f}_i(arg(x_i, D))|\!) + (\!|\mathtt{f}_j(arg(x_j, D))|\!) + (\!|\mathtt{f}_k(arg(x_k, D))|\!)$$

At most 2 of the quasi-interpretations of the literals are evaluated to $2k$. In other words, it forces at least one of the function quasi-interpretation to be evaluated to k (which corresponds to the truth value **True**). It remains to encode all the disjunctions of our problem. To conclude, we have encoded a 3-CNF problem (which is known to be NPTIME-hard) into a synthesis problem over **Max-Plus** $\{\mathbb{R}^+\}$. □

B.2 NPTIME-*completeness*

Since our problem is NPTIME-hard, it remains to prove that it is NPTIME-complete (under some restrictions) by showing that it belongs to NPTIME. Our restrictions concern degrees: we only consider assignments whose max-degree d and $\times$-degree k are bounded polynomially in the size of the program. This condition is not too restrictive since the majority of relevant programs admitting a quasi-interpretation in **Max-Plus** $\{\mathbb{R}^+\}$ falls in.

Theorem 37 *[Verification] Suppose that we have a program p and of an assignment* $(\!|-|\!)$ *in* ***Max-Plus*** $\{\mathbb{R}^+\}$*, we can check in time polynomial in the max-degree d and in the degree if* $(\!|-|\!)$ *is a quasi-interpretation of p.*

PROOF. Given p a program and $(\!|-|\!)$ an assignment. Let A be the maximum size of a right-hand-side term of a rule (we suppose that A is bounded without restriction). For every rule $\mathtt{f}(p_1, \ldots, p_n) \to e$, we can calculate $(\!|\mathtt{f}(p_1, \ldots, p_n)|\!)$ and $(\!|e|\!)$ polynomialy with respect to A and d. Consequently, it remains to verify that the quasi-interpretation inequality holds. The number of such comparisons to do corresponds to the number of rules and is bounded polynomialy in the size of the program. We note:

$$(\!|\mathtt{f}|\!)(X_1, \ldots, X_n) = \max_{i \in I}(\sum_{j=1}^{n} \alpha_{i,j} X_j + a_i)$$
$$(\!|e|\!) = \max_{k \in L}(\sum_{j=1}^{m} \beta_{k,j} X_j + b_k)$$

$\{X_1, \ldots, X_m\}$ being the set of variables ranging over $\mathbb{R}^+$ and corresponding to the set $\{x_1, \ldots, x_m\}$ of variables occuring in $\mathtt{f}(p_1, \ldots, p_n)$. By virtue of the max-degree definition and thanks to the proposition 28, we have $\sharp I \leq d$ and $\sharp L \leq d^{|e|} \leq d^A$ The verification problem is to prove that the following inequality holds:

$$\max_{i \in I}(\sum_{j=1}^{n} \alpha_{i,j} (\!|p_j|\!) + a_i) \geq \max_{k \in L}(\sum_{j=1}^{m} \beta_{k,j} X_j + b_k)$$

Suppose that r is the number of rules in our program, we eliminate the max in the right-hand-side of the inequality. Since $\sharp L \leq d^A$, we obtain at most $r \times d^A$ inequalities of the shape:

$$\max_{i \in I}(\sum_{j=1}^{n} \alpha_{i,j} (\!|p_j|\!) + a_i) \geq \sum_{j=1}^{m} \beta_{k,j} X_j + b_k$$

Now we have to prove the previous inequality in polynomial time. It is satisfied in $\mathbb{R}^+$ if, a contrario, there are no $X_1, \ldots, X_m$ in $\mathbb{R}^+$ such that:

$$\max_{i \in I}(\sum_{j=1}^{n} \alpha_{i,j} (\!|p_j|\!) + a_i) < \sum_{j=1}^{m} \beta_{k,j} X_j + b_k$$

So we obtain at most $r \times d^A$ inequalities systems of the following shape:

$$\left\{ \sum_{j=1}^{n} \alpha_{i,j} (\!|p_j|\!) + a_i < \sum_{j=1}^{m} \beta_{k,j} X_j + b_k, i \in I \right\}$$

Since $\sharp I \leq d$, the size of each system is bounded by d. Then we can solve each system in time polynomial in d thanks to linear programming and find a solution, if there is one. In this case, the assignment is not a quasi-interpretation. By iterating at most $r \times d^A$ times this resolution of systems, we have shown that verification problem for assignment of bounded degree is polynomial in the size of the program and in the max-degree. □