Polynomials over the reals are safe for program interpretations

Guillaume Bonfante, Florian Deloup, Antoine Henrot

Polynomials over the reals are safe for program interpretations

Guillaume Bonfante[1], Florian Deloup[2] and Antoine Henrot[3]

1- Université de Nancy - LORIA, Nancy, France,
2- Université Paul Sabatier, Toulouse - IMT, France
3- Université de Nancy - IECN, Nancy, France

Abstract. In the field of implicit computational complexity, we are considering in this paper the fruitful branch of interpretation methods. Due to their good intensional properties, they have been widely developped. Among usual issues is the synthesis problem which has been solved by the use of Tarski's decision procedure, and consequently interpretations are usually chosen over the reals rather than over the integers. Doing so, one cannot use anymore the (good) properties of the natural (well-) ordering of $\mathbf{N}$ employed to bound the complexity of programs. We show that, actually, polynomials over the reals benefit from some properties that allows their safe use for complexity. We illustrate this by two characterizations, one of PTIME and one of PSPACE.

Among studies in rewriting are the noticeable work concerning termination. This is now a largely and thoroughly studied field and very elegant methods have been proposed that cover a large spectrum of algorithms (see for instance [8]). These studies can be refined to get a characterization of the complexity of first order programs. This has been one of the main successful approaches of implicit computational complexity, see for instance the early work [2, 4].

To prove termination by interpretation over a well-founded ordering seems rather natural and such interpretation methods have been introduced in the 70's (see [16, 15]). Lankford describes interpretations as monotone Σ-algebras with domain of interpretation being the natural numbers with their usual ordering. The fact that this ordering is well-founded gives immediately the well-foundedness of the rewriting relation. Based on Kruskal's Theorem, Dershowitz showed in [7] that the well-foundedness of the domain of interpretation is not necessary whenever the interpretations are chosen monotonic and have the subterm property.

One of the main interesting points about choosing of real numbers rather than natural numbers is that we get (at least from a theoretical point of view) a procedure to verify the validity of an interpretation of a program by Tarski's decomposition procedure [25]. But furthermore, we can even give a semi-algorithm to compute interpretations: indeed, for a given choice of the degree of the interpretations, finding the coefficients of these polynomial becomes decidable. Actually, since the problem of the verification or the synthesis (up to some degree) can be stated by a first order formula with two alternations of quantifiers,

following Roy et al. [1], the complexity of these algorithms is exponential with respect to the size of the program. To obtain a faster procedure, we have used in the CROCUS tool –developped by the first author– the (sufficient) criterion of Hong and Jakuš [13].

A second good point is that the use of reals (as opposed to integers) enlarges the set of rewriting systems that have an interpretation, as shown recently by Lucas [18].

The main concern of this work is to show that the structure of polynomials over the reals has an important role from the point of view of complexity. Our thesis is that, in the field of complexity, due to Stengle's Positivstellensatz [24], polynomials over the reals can safely replace polynomials over the integers. To say it more precisely, one may recover both derivational complexity (up to a polynomial) and size bounds on terms as applications of the Positivstellensatz. Moreover, this can be done in a constructive way.

Given a strict interpretation for a Term Rewriting System (TRS), it follows immediately that for any rewriting step $s \to t$, we have $(\!|s|\!) > (\!|t|\!)$. If one takes the interpretation on natural numbers (as they were introduced by Lankford [16]), this can be used to give a bound on the derivation height. Thus, Hofbauer and Lautemann have shown in [12] that the derivation height is bounded by a double exponential. However, their argument uses deeply the fact that the interpretation of a term is itself a bound on the derivation height:

$$\mathtt{dh}(t) \leq (\!|t|\!). \tag{1}$$

Indeed, suppose $t_1 \to t_2 \to \cdots \to t_n$, then $(\!|t_1|\!) > (\!|t_2|\!) > \cdots > (\!|t_n|\!)$. On natural numbers, this means that $n \leq (\!|t_1|\!)$. Such a proof does not hold with real numbers.

Equation (1) comes from the fact that $(\!|t|\!) \geq (\!|u|\!) + 1$ for terms $t \to u$. This fact itself is due to a) $(\!|\ell|\!) > (\!|r|\!)$ implies that

$$(\!|\ell|\!) \geq (\!|r|\!) + 1 \tag{2}$$

and b) that for all $x_i > y_i$:

$$(\!|f|\!)(x_1, \ldots, x_i, \ldots, x_n) - (\!|f|\!)(x_1, \ldots, y_i, \ldots, x_n) \geq x_i - y_i. \tag{3}$$

The two inequalities (2), (3) do not hold in general for real interpretations. To recover the good properties holding with natural numbers, people have enforced the inequalities on terms. For instance [18, 22] suppose the existence of some real $\delta > 0$ such that $(\!|\ell|\!) \geq (\!|r|\!) + \delta$. We prove that this is not necessary.

The mathematical structure of polynomials over the reals has also been considered by Shkaravska et al [23] with the intention to compute the size of terms appearing in computations.

One of our two characterization use dependency pairs. In this vein, we mention here the work of Hirokawa and Moser [10], and, in the same spirit, Lucas and Peña in [20] made some investigation on the tools of rewriting to tackle the complexity of first order functional programs. In particular, they provide a certified compilation procedure to the JVM.

1 Preliminaries

We suppose that the reader has familiarity with first order rewriting. We briefly recall the context of the study, essentially to fix the notations. Dershowitz and Jouannaud's survey [9] of rewriting is a good entry point.

1.1 Syntax of programs

All along, $\mathcal{X}$ will denote a set of variables, $\mathcal{C}$ a (finite) signature of constructor symbols and $\mathcal{F}$ a (finite) signature of function symbols.

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

$$\begin{array}{lllll}
\textit{(Constructor terms)} & \mathcal{T}(\mathcal{C}) \ni u & ::= & \mathbf{c} \mid \mathbf{c}(u_1, \cdots, u_n) \\
\textit{(terms)} & \mathcal{T}(\mathcal{C}, \mathcal{F}, \mathcal{X}) \ni t & ::= & \mathbf{c} \mid x \mid \mathbf{c}(t_1, \cdots, t_n) \mid \mathtt{f}(t_1, \cdots, t_n) \\
\textit{(patterns)} & \mathcal{P} \ni p & ::= & \mathbf{c} \mid x \mid \mathbf{c}(p_1, \cdots, p_n) \\
\textit{(D-rules)} & \mathcal{D} \ni d & ::= & \mathtt{f}(p_1, \ldots, p_n) \to t
\end{array}$$

where $x \in \mathcal{X}$, $\mathtt{f} \in \mathcal{F}$, and $\mathbf{c} \in \mathcal{C}$.

We shall use a type writer font for function symbols and a bold face font for constructors. Finally, we use $\boldsymbol{u}$ to denote a (finite) sequence of terms $u_1, \ldots, u_n$. The size of a term is the number of symbols occuring in the term. It is written $|t|$.

Definition 2. *A program is a quadruplet $\mathtt{f} = \langle \mathcal{X}, \mathcal{C}, \mathcal{F}, \mathcal{E} \rangle$ 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 $\mathtt{f}$.*

The set of rules induces a rewriting relation $\to$. The relation $\xrightarrow{*}$ is the reflexive and transitive closure of $\to$. We write $t_0 \to^n t_n$ the fact that $t_0 \to t_1 \cdots \to t_n$. One defines the derivation height for a term t as the maximal length of a derivation:

$$\mathtt{dh}(t) = \max\{n \in \mathbf{N} \mid \exists v : t \to^n v\}.$$

Moreover, we suppose programs to be confluent. This is achieved by the following syntactic restriction due to Huet [14]: (i) Each rule $\mathtt{f}(p_1, \cdots, p_n) \to t$ 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. Such programs are called orthogonal.

A substitution σ is a mapping from variables to terms. We say that it is a constructor substitution when the range of σ is $\mathcal{T}(\mathcal{C})$.

Orthogonal programs define a class of deterministic first order functional programs. The domain of the computed functions is the constructor term algebra $\mathcal{T}(\mathcal{C})$. The program $\mathtt{f}$ computes a partial function $[\![\mathtt{f}]\!] : \mathcal{T}(\mathcal{C})^n \to \mathcal{T}(\mathcal{C})$ defined as follows. For every $u_1, \cdots, u_n \in \mathcal{T}(\mathcal{C}), [\![\mathtt{f}]\!](u_1, \cdots, u_n) = v$ iff $\mathtt{f}(u_1, \cdots, u_n) \xrightarrow{*} v$. Otherwise, it is undefined and $[\![\mathtt{f}]\!](u_1, \cdots, u_n) = \bot$.

Definition 3 (Call-tree). *Suppose we are given a program* $\langle \mathcal{X}, \mathcal{C}, \mathcal{F}, \mathcal{E} \rangle$. *Let* $\rightsquigarrow$ *be the relation*

$$(f, t_1, \ldots, t_n) \rightsquigarrow g(u_1, \ldots, u_m) \Leftrightarrow f(t_1, \ldots, t_n) \to C[g(v_1, \ldots, v_m)] \xrightarrow{*} C[g(u_1, \ldots, u_m)]$$

where f *and* g *are defined symbols, and* $t_1, \ldots, t_n, u_1, \ldots, u_m$ *are constructor terms. Given a term* $f(t_1, \ldots, t_n)$*, the relation* $\rightsquigarrow$ *defines a tree whose root is* $(f, t_1, \ldots, t_n)$ *and* η' *is a daughter of* η *iff* $\eta \rightsquigarrow \eta'$*. The size of a call-tree is the number of nodes it contains.*

When we do not make a distinction between constructors and function symbols, we speak of Term Rewriting System (TRS). We present them as 3-tuple $(\mathcal{X}, \Sigma, R)$ where Σ is the signature and R is the set of rules.

1.2 Interpretations of programs

Given a signature Σ, a Σ-algebra on the domain A is a mapping $(\!|-|\!)$ which associates to every n-ary symbol $f \in \Sigma$ an n-ary function $(\!|f|\!) : A^n \to A$. A Σ-algebra can be extended to terms by

- $(\!|x|\!) = 1_A$, that is the identity on A, for $x \in \mathcal{X}$,
- $(\!|f(t_1, \ldots, t_m)|\!) = \mathrm{comp}((\!|f|\!), (\!|t_1|\!), \ldots, (\!|t_m|\!))$ where comp is the composition of functions.

The interpretation $(\!|t|\!)$ of a term t with n variables is then a function $A^n \to A$.

An interpretation for a rewriting system (Σ, R) is an order-preserving mapping $(\!|-|\!)$ from $(\mathcal{T}(\Sigma), \xrightarrow{+})$ to some (partially) ordered set $(A, >)$. From now on, we restrict our attention to the case where $(A, >)$ is the set of nonnegative real numbers $\mathbf{R}^+$ with the usual ordering and $(\!|-|\!)$ has the structure of a Σ-algebra:

Definition 4. *A polynomial strict interpretation of a rewriting system* (Σ, R) *is given by a* Σ*-algebra* $(\!|-|\!)$ *such that:*

1. *for all symbol* f*, the interpretation* $(\!|f|\!)$ *is a monotonic polynomial, that is if* $x_i > x_i'$*, then*

$$(\!|f|\!)(x_1, \ldots, x_n) > (\!|f|\!)(x_1, \ldots, x_i', \ldots, x_n),$$

2. *for all symbol* f*, the interpretation* $(\!|f|\!)$ *verifies the (weak) sub-term property, that is* $(\!|f|\!)(x_1, \ldots, x_n) \geq x_i$ *with* $i \in 1..n$*,*

3. *for all rules* $\ell \to r$*,* $(\!|\ell|\!) > (\!|r|\!)$*.*

Example 1. Consider the system

$$\begin{pmatrix} \mathbf{A}(\mathbf{B}(x)) \to \mathbf{B}(\mathbf{B}(\mathbf{A}(x))) \\ \mathbf{c}(\mathbf{A}(x)) \to \mathbf{A}(\mathbf{A}(\mathbf{c}(x))) \end{pmatrix}$$

For this system, we define the interpretation $(\!|\mathbf{A}|\!)(x) = 3(x + 2)$, $(\!|\mathbf{B}|\!)(x) = x + 1$ and $(\!|\mathbf{c}|\!)(x) = x^2 + 1$.

Sup-interpretation have been introduced by Marion and Pechoux in [21]. We give a slight variant of their definition. In [21], the last inequality refers to the size of normal forms. We prefered to have a more uniform definition.

Definition 5. *A (polynomial) sup-interpretation of a rewriting system (Σ, R) is given by a Σ-algebra $(\!|-|\!)$ such that:*

1. *$(\!|f|\!)$ is a weakly monotonic polynomial, that is if $x_i \geq x_i'$, then*

$$(\!|f|\!)(x_1, \ldots, x_n) \geq (\!|f|\!)(x_1, \ldots, x_i', \ldots, x_n),$$

2. *for all constructor terms $t_1, \ldots, t_n$, we have the inequality $(\!|f(t_1, \ldots, t_n)|\!) \geq (\!|[\![f]\!](t_1, \ldots, t_n)|\!)$.*

Lemma 1. *Suppose that we are given a Σ-algebra for which interpretations of symbols f is bounded by some polynomials. Then, for all closed terms, $(\!|t|\!) \leq 2^{2^{O(|t|)}}$.*

Proof. By induction on terms. A proof (for natural numbers, but this has no consequences here) can be found in [2].

Definition 6. *An interpretation of a program is said to be additive if for all constructor $\mathbf{c}$, we have $(\!|\mathbf{c}|\!)(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i + c_{\mathbf{c}}$ for some constant $c_{\mathbf{c}} > 0$.*

2 Positivstellensatz and applications

In this subsection, we introduce a deep mathematical result, the Positivstellensatz. Then we give some applications to polynomial interpretations. They will be key points of the Theorems 4 and 6 in our analysis of the role of reals in complexity (§3).

Let $n > 0$. Denote by $\mathbf{R}[x_1, \ldots, x_n]$ the $\mathbf{R}$-algebra of polynomials with real coefficients. Denote by $(\mathbf{R}^+)^n = \{x = (x_1, \ldots, x_n) \in \mathbf{R}^n \mid x_1, \ldots, x_n > 0\}$ the first quadrant. Since we need to consider only the $\mathbf{R}$-algebra of polynomial functions $(\mathbf{R}^+)^n \to \mathbf{R}$, it will be convenient to identify the two spaces. In particular throughout this section, all polynomial functions are defined on $(\mathbf{R}^+)^n$.

2.1 Preliminary results

Definition 7. *A polynomial $P \in \mathbf{R}[x_1, \ldots, x_n]$ is said* over-homogeneous of level *d if there exists $A, \alpha > 0$ such that for any $\lambda > \alpha$ and any $x \in (\mathbf{R}^+)^n$, $||x|| > A \implies P(\lambda x) \geq \lambda^d P(x)$.*

Remark. This notion is essentially independent of the chosen norm on $\mathbf{R}^n$. We say that P is (A, α)-over-homogeneous when we need to specify the bounds (A, α). In this case, the bounds depend on the choice of the norm $||\cdot||$ on $\mathbb{R}^n$.

Lemma 2. *Let $Q \in \mathbf{R}[x_1, \ldots, x_n]$ an over-homogeneous polynomial of level $d \geq 1$ such that there is some $r > A$ such that*

$$\inf_{||x||=r} Q(x_1, \ldots, x_n) > 0$$

For any $C > 0$, there is $D > 0$ such that $||x|| > D \implies Q(x) > C$.

Proof. Let us write $\alpha = \inf_{||x||=r} Q(x) > 0$. One observes that $Q(\lambda x) \geq \lambda^d Q(x) \geq \lambda^d \alpha > C$ for all $\lambda > (C/\alpha)^{1/d} = D$.

Lemma 3. *Let $Q \in \mathbf{R}[x_1, \ldots, x_n]$ an over-homogeneous polynomial of level $d \geq 1$.*

(i) If there is $x \in (\mathbf{R}^+)^n$ such that $P(x) \geq 0$ then $P(\lambda\ x) \geq 0$ for all $\lambda \geq 0$; furthermore, the same statement holds with all inequalities replaced by strict inequalities.
(ii) If there is $x \in (\mathbf{R}^+)^n$ such that $P(x) < 0$ then $P(\lambda\ x) < 0$ for all $\lambda > 0$;

Proof. (i): $P(\lambda\ x) \geq \lambda^d\ P(x) \geq 0$. (ii): arguing by contradiction, suppose there is $\lambda_0 > 0$ such that $P(\lambda_0\ x) > 0$. Then by (i), $P(\lambda x) > 0$ for all $\lambda > 0$. Hence $P(x) > 0$, which is a contradiction.

Lemma 4. *Given $P \in \mathbf{R}[x_1, \ldots, x_n]$, a over-homogeneous polynomial of level $d \geq 2$ such that $\forall x \in (\mathbf{R}^+)^n, ||x|| > B \implies P(x) \geq 0$ for some constant C. Suppose that Q is over-homogeneous of level k with $1 \leq k < d$. Then, $P + Q$ is over-homogeneous of level k.*

Proof. Suppose that P is (A, α)-over-homogeneous and Q is (B, β)-over-homogeneous. Let $C = \max(A, B)$. For $||x|| > C$, $\lambda > \max(1, \alpha, \beta)$,

$$\begin{aligned} P(\lambda x) + Q(\lambda x) &\geq \lambda^d P(x) + \lambda^k Q(x) \\ &\geq \lambda^k P(x) + \lambda^k Q(x) \quad \text{since } P(x) \geq 0 \\ &= \lambda^k (P + Q)(x) \end{aligned}$$

Given a polynomial $P \in \mathbf{R}[x_1, \ldots, x_n]$, we can decompose it in homogeneous components $P = P_d + P_{d-1} + \cdots P_0$ with each P_i of degree i. Furthermore we note $P_{\geq k}$ the polynomial $\sum_{i=k}^{d} P_i$.

Lemma 5. *Let $P \in \mathbf{R}[x_1, \ldots, x_n]$ of total degree greater than 1. Suppose that there is some $C > 0$ such that for all $k \geq 0$, and all $x \in (\mathbf{R}^+)^n$ with $||x|| > C$, we have $P(x) > 0$. Then, $P_{\geq k}$ is over-homogeneous of level k for $1 \leq k \leq d$ and $P_{\geq k}(x) > 0$ with $||x|| > C'$ for some C'.*

Proof. By descending induction on $k \leq d$. For the base case, $P_{\geq d} = P_d$ is homogeneous, hence over-homogeneous, of level d. Moreover, P_d has the sign of P for $||x|| > C$ ([1]) hence P_d is nonnegative for $||x|| > C$. Since each P_i is homogeneous (and consequently over-homogeneous) of level i, the induction step is a direct consequence of Lemma 4.

[1] This comes from the equality $P(\lambda x) = \lambda^d P_d(x) + R(\lambda x)$ with $\deg_\lambda(R) < d$.

Theorem 1. *Given a polynomial $P \in \mathbf{R}[x_1, \ldots, x_n]$ such that*

(i) $\forall x_1, \ldots, x_n \geq 0 : P(x_1, \ldots, x_n) > \max(x_1, \ldots, x_n)$,

then, there exist $A \geq 0$ such that $P(x_1, \ldots, x_n) > x_1 + \cdots + x_n$ whenever $||x|| > A$.

Proof. Let $d = \deg(P)$. Without loss of generality, we may assume that $P(0) = P_0 = 0$. For $d = 1$, the theorem is easily seen to hold. Assume that $d > 1$. Note that (i) implies that $P(x) > 0$ if $x \neq 0$ (x is assumed to lie in $(\mathbf{R}^+)^n$ as usual). Applying Lemma 5, we see that $P_{\geq k}$ is positive and over-homogeneous of level k for any $1 \leq k \leq d$.

Lemma 6. *If P is positive, over-homogeneous of level $k \geq 2$ and satisfies the hypothesis (i), then for any $K > 0$, there exists $A = A_K > 0$ such that $P(x_1, \ldots, x_n) > K(x_1 + \cdots + x_n)$ whenever $||x|| > A$. In particular, the theorem holds.*

Proof of the Lemma. Let $x = (x_1, \ldots, x_n) \in (\mathbf{R}^+)^n$, let $\lambda \in \mathbf{R}^+$ and let $X = \lambda\, x$. We have

$$\begin{aligned} P(X) = P(\lambda\ x) \geq \lambda^k P(x) &\geq \lambda^k \max(x_1, \ldots, x_n) \\ &\geq \lambda^k\ \frac{x_1 + \cdots + x_n}{n} \\ &= \lambda^{k-1} \frac{X_1 + \cdots + X_n}{n}. \end{aligned}$$

Since $k \geq 2$, there exists $A \geq 0$ such that for any $\lambda > A$, $\frac{\lambda^{k-1}}{n} > K > 0$. This proves the lemma.

In particular the theorem holds if P has no homogeneous component P_1 of degree 1. We now complete the proof when $P_1(x) = a_1 x_1 + \cdots + a_n x_n \neq 0$.

Lemma 7. *$P_{\geq 2} = P - P_1$ satisfies the hypothesis (i).*

Proof of the lemma. Argue by contradiction and suppose it does not. There exists then $x = (x_1, \ldots, x_n) \in (\mathbf{R}^+)^n$ such that $P_{\geq 2}(x) - x_i \leq 0$ for some fixed $1 \leq i \leq n$. Since $P_{\geq 2}(x)$ is over-homogeneous, Lemma 4 ensures that the polynomial $Q(x) = P_{\geq 2}(x) - x_i$ is over-homogeneous of level 1. By Lemma 3, $Q(\lambda x) = P_{\geq 2}(\lambda\ x) - \lambda\ x_i \leq 0$ for any $\lambda \geq 0$. Since $P_{\geq 2}$ is positive, the map $\lambda \mapsto P_{\geq 2}(\lambda x)$ is a positive polynomial of degree d. Thus

$$\lim_{\lambda \to +\infty} P_{\geq 2}(\lambda x) - \lambda x_i = +\infty,$$

which is a contradiction.

Apply Lemma 6 to $P_{\geq 2}$ with $K = 1 + \max(|a_1|, \ldots, |a_n|)$. We obtain $P_{\geq 2}(x) > K(x_1 + \cdots + x_n)$. Hence

$$P(x) = P_{\geq 2}(x) + P_1(x) > (K + a_1)x_1 + \cdots + (K + a_n)x_n > x_1 + \cdots + x_n.$$

This achieves the proof of the theorem.

Corollary 1. *Let $P(x_1, \ldots, x_n)$ be a polynomial with the hypotheses of Theorem 1. There is a constant A such that for any subset $I \subseteq \{1..n\}$, such that*

$$x_1, \ldots, x_n \geq 0,\ x_i \geq A,\ \textit{for all } i \in I \implies P(x_1, \ldots, x_n) > \sum_{i \in I} x_i.$$

Proof. Use the norm $||x|| = \max(|x_1|, \ldots, |x_n|)$.

Theorem 2. *Given a polynomial $P \in \mathbf{R}[x_1, \ldots, x_n]$ such that*

(i) $\forall x_1 \geq 0, \ldots, x_n \geq 0 : P(x_1, \ldots, x_n) > \max(x_1, \ldots, x_n)$,

(ii) $\forall x_1 \geq 0, \ldots, x_n \geq 0 : \dfrac{\partial P}{\partial x_i}(x_1, \ldots, x_n) > 0$ *for all* $i \leq n$,

then, there exist $A > 0$ such that for any $\Delta > 0$, we have $P(x_1, \ldots, x_i + \Delta, \ldots, x_n) > P(x_1, \ldots, x_n) + \Delta$ whenever $||x|| > A$.

Proof. For simplicity we take $n = 2$ (the reader will readily extend the argument to the general case) and we make an analysis by case. Consider the case where $P(x, y) = P_0(y) + xP_1(y)$. Let us consider $\dfrac{\partial P}{\partial x}(x, y) = P_1(y)$. First , we prove that $P_1(y) \geq 1$ for all $y \geq 0$. Ad absurdum, suppose that $P_1(y) < 1$. Then, take $x > \dfrac{P_0(y)}{1 - P_1(y)} > 0$. Due to (i), we have

$$P(x, y) - x = P_0(y) + x(P_1(y) - 1) > 0. \tag{4}$$

Since $P_1(y) - 1 < 0$ and $x > \dfrac{P_0(y)}{1 - P_1(y)} > 0$, we have

$$P_0(y) + x(P_1(y) - 1) < P_0(y) + \frac{P_0(y)}{1 - P_1(y)} \times (P_1(y) - 1) = 0$$

which contradicts (4).

In particular, for y large enough, $P_1(y) > 1$. The conclusion follows by the mean value inequality.

Consider now $\dfrac{\partial P}{\partial y}(x, y) = P_0'(y) + xP_1'(y)$. Suppose $P_1'(y_0) < 0$ for some $y_0 > 0$. Then

$$\lim_{x \to \infty} \frac{\partial P}{\partial y}(x, y_0) = \lim_{x \to \infty} P_0'(y_0) + xP_1(y_0) = -\infty.$$

This contradicts (ii). So, $P_1'(y) \geq 0$. Now, $P(0, y) = P_0(y) > y$ shows that $P_0'(y) - 1 \geq 0$ for y sufficiently large, say $y > A$. Then we have $\dfrac{\partial P}{\partial y}(x, y) = P_0'(y) + xP_1'(y) > 1$. We conclude as before.

The case $P(x, y) = Q_0(x, y) + yQ_1(x, y)$ is symmetric. So the remaining case is when $\deg_x(P) \geq 1$ and $\deg_y(P) \geq 1$. Thus $Q := \dfrac{\partial P}{\partial x}$ is positive and has total

degree greater than 2. According to Lemma 5, $Q_{\geq 1}$ is positive over-homogeneous. Hence Lemma 2 applies to $Q_{\geq 1}$ with $C = 1 + \left|\frac{\partial P}{\partial x}(0,0)\right|$. Hence there is some $D > 0$ such that for $||(x,y)|| > D$,

$$\frac{\partial P}{\partial x}(x,y) = Q_{\geq 1}(x,y) + Q(0,0) > 1 + C + \frac{\partial P}{\partial x}(0,0) > 1.$$

We conclude with the mean value inequality.

Corollary 2. *With the hypotheses of Theorem 2, there is a bound B such that for all $x_1, \ldots, x_n \geq 0$, if $x_i > B$, then, $P(x_1, \ldots, x_i + \Delta, \ldots, x_n) > P(x_1, \ldots, x_n) + \Delta$.*

Proof. Apply Theorem 2 with the norm $||x|| = \max(|x_1|, \ldots, |x_n|)$.

2.2 Positivstellensatz and applications

Theorem 3 (Positivstellensatz, Stengle [24]). *Suppose that we are given polynomials $P_1, \ldots, P_m \in \mathbf{R}[x_1, \ldots, x_k]$, the following two assertions are equivalent:*

1. $\{x_1, \ldots, x_k : P_1(x_1, \ldots, x_k) \geq 0 \wedge \cdots \wedge P_m(x_1, \ldots, x_k) \geq 0\} = \emptyset$
2. $\exists Q_1, \ldots, Q_m : -1 = \sum_{i \leq m} Q_i P_i$ *where each Q_i is a sum of squares of polynomials (and so is positive and monotonic).*

Moreover, these polynomials $Q_1, \ldots, Q_i$ can effectively computed. We refer the reader to the work of Lombardi, Coste and Roy [17, 5]. As a consequence, all the constructions given below can be actually (at least theoretically) computed.

It will be convenient to derive from the Positivstellensatz a proposition useful for our applications.

Proposition 1. *Let $(P_i)_{i \in I}$ and $(Q_j)_{j \in J}$ two finite families of polynomials in $\mathbf{R}[x_1, \ldots, x_n]$. Suppose that $\max_{i \in I} P_i(x) > \max_{j \in J} Q_j(x)$ for all $x \in (\mathbf{R}^+)^n$. Then there exists a polynomial $R \in \mathbf{R}[x_1, \ldots, x_n]$ such that*

$$R(x) > 0 \quad \textit{and} \quad \max_{i \in I} P_i(x) \geq \max_{j \in J} Q_j(x) + \frac{1}{R(x)}, \quad \textit{for all } x \in (\mathbf{R}^+)^n.$$

Proof. First we prove the result when J is a singleton. Suppose that $\max_{i \in I} P_i(x) > Q(x)$ for all $x \in (\mathbf{R}^+)^n$. Let $s \in I$. Define

$$D_s = \{x \in (\mathbf{R}^+)^n \ : \ P_s(x) \geq P_i(x) \text{ for all } i \neq s\}.$$

For $x \in D_s$, $P_s(x) = \max_{i \in I} P_i(x) > Q(x)$. Therefore the set

$$\left\{ \begin{array}{l} x_1 \geq 0, \ldots, x_n \geq 0, \\ P_s(x) - P_i(x) \geq 0, \text{for all} i \neq s, \\ Q(x) - P_s(x) \geq 0 \end{array} \right\}$$

is empty. The Positivstellensatz yields positive monotonic polynomials $(T_k)_{1\leq k\leq n}$, $(U_i)_{i\neq s}$ and V_s such that

$$\sum_k U_k(x)x_k + \sum_{i\neq s} T_i(x)(P_s(x) - P_i(x)) + V_s(x)(Q(x) - P_s(x)) = -1.$$

Hence for $x \in D_s$,

$$(P_s(x) - Q(x))V_s(x) = 1 + \sum_{i\neq s} T_i(x)(P_s(x) - P_i(x)) \geq 1.$$

Thus

$$\forall x \in D_s,\ \max_{i\in I} P_i(x) - Q(x) \geq \frac{1}{V_s(x)}.$$

Hence, setting $R(x) = \sum_{s\in I} V_s(x) > 0$, we have

$$\forall x \in (\mathbf{R}^+)^n,\ \max_{i\in I} P_i(x) - Q(x) \geq \frac{1}{R(x)}.$$

Next, we treat the general case. The previous argument yields for each $j \in J$ a positive monotonic polynomial R_j such that $\max_{i\in I} P_i(x) - Q_j(x) \geq \frac{1}{R_j(x)}$ for all $x \in (\mathbf{R}^+)^n$. Set $R(x) = \sum_{j\in J} R_j(x)$. For any $j \in J$,

$$\max_{i\in I} P_i(x) \geq Q_j(x) + \frac{1}{R(x)}.$$

Hence $\max_{i\in I} P_i(x) \geq \max_{j\in J} Q_j(x) + \frac{1}{R(x)}$.

We give a first application.

Proposition 2. *Suppose that a TRS (Σ, R) admits an interpretation $(\!|-|\!)$ over* Max-Poly *such that for all rules $\ell \to r$, we have $(\!|\ell|\!) > (\!|r|\!)$. There is a positive, monotonic polynomial P such that for any rule $\ell \to r$, we have $(\!|\ell|\!)(x_1, \ldots, x_k) - (\!|r|\!)(x_1, \ldots, x_k) \geq \dfrac{1}{P(x_1, \ldots, x_k)}$.*

Proof. For each symbol ℓ, there is a finite family of polynomials $(P_i)_{i\in I}$ such that $(\!|\ell|\!)(x_1, \ldots, x_n) = \max_{i\in I} P_i(x_1, \ldots, x_n)$. Therefore, if $(\!|\ell|\!) > (\!|r|\!)$, Proposition 1 applies: there is a positive monotonic polynomial $P_{\ell\to r}$ such that such that $(\!|\ell|\!)(x_1, \ldots, x_k) - (\!|r|\!)(x_1, \ldots, x_k) \geq \dfrac{1}{P_{\ell\to r}(x_1, \ldots, x_k)}$. Since there are only finitely many rules, we can take $P = \sum_{\ell\to r\in R} P_{\ell\to r}$.

Proposition 2 has an important consequence. Since, in a derivation all terms have an interpretation bounded by the interpretation of the first term, there is a minimal decay for each rule of the derivation.

Proposition 3. *Suppose that a TRS (Σ, R) admits a strict interpretation $(\!|-|\!)$ over* Max-Poly*. For all $A > 0$, the set of terms $\{t \in \mathcal{T}(\Sigma) \mid (\!|t|\!) < A\}$ is finite.*

Proof. For all symbols $f \in \Sigma$, we have $(\!|f|\!)(x_1, \ldots, x_n) > x_i$ for all i. By Proposition 2, there is a polynomial P such that $(\!|f|\!)(x_1, \ldots, x_n) \geq x_i + \dfrac{1}{P(x_1, \ldots, x_n)}$. Take a term $f(t_1, \ldots, t_n)$ such that $(\!|f(t_1, \ldots, t_n)|\!) < A$.

$$\begin{aligned}(\!|f(t_1, \ldots, t_n)|\!) &\geq (\!|t_i|\!) + \frac{1}{P((\!|t_1|\!), \ldots, (\!|t_n|\!))} \\ &\geq (\!|t_i|\!) + \frac{1}{P(A, \ldots, A)}\end{aligned}$$

where the second inequality is due to the sub-term property together with the monotonicity of P. Consequently, the height of a term t with $(\!|t|\!) < A$ is bounded by $A \times P(A, \ldots, A)$. There are only finitely many such terms.

Proposition 4. *Suppose that a TRS (Σ, R) admits a strict interpretation $(\!|-|\!)$ over Poly. There is a real $A > 0$ and a positive, monotonic polynomial P such that for all $x_1, \ldots, x_n \geq 0$, if $x_{i_1}, \ldots, x_{i_k} > A$, then for all symbol f, we have $(\!|f|\!)(x_1, \ldots, x_n) \geq x_{i_1} + \cdots + x_{i_k} + \dfrac{1}{P((\!|f|\!)(x_1, \ldots, x_n))}$.*

Proof. Due to Corollary 1, there is a bound A such that for all $x_1, \ldots, x_n \geq 0$, if $x_{i_1}, \ldots, x_{i_k} > A$, then for all symbol f, we have $(\!|f|\!)(x_1, \ldots, x_n) > x_{i_1} + \cdots + x_{i_k}$. Applying Theorem 3, we get Q_f such that $(\!|f|\!)(x_1, \ldots, x_n) \geq x_{i_1} + \cdots + x_{i_k} + \dfrac{1}{Q_f(x_1, \ldots, x_n)} \geq x_{i_1} + \cdots + x_{i_k} + \dfrac{1}{Q_f((\!|f|\!)(x_1, \ldots, x_n), \ldots, (\!|f|\!)(x_1, \ldots, x_n))}$. It is then routine to get a uniform polynomial wrt to all symbols.

3 The role of reals in complexity

We have now all the tools to prove that reals can safely replace integers from a complexity point of view. This is illustrated first by the following theorem.

Theorem 4. *Functions computed by programs with an additive interpretation (over the reals) are exactly* Ptime *functions.*

The rest of the section is devoted to the proof of the Theorem. The main difficulty of the proof is that inequalities as given by the preceding section only hold for sufficiently large values. So, the main issue is to split "small" terms (and "small rewriting steps") from "large" ones. The Positivstellensatz gives us the arguments for the large terms (Lemma 8), Lemmas 9,10 show that there are not too many small steps. Lemma 12 describe how small steps and big steps alternate.

From now on, we suppose we are given a program with an additive strict interpretation over polynomials.

Lemma 8. *There is a polynomial P and a real $A > 0$ such that for all step $\ell\sigma \to r\sigma$ with $(\!|r\sigma|\!) > A$, then, for all context C, we have $(\!|C[\ell\sigma]|\!) \geq (\!|C[r\sigma]|\!) + \frac{1}{P((\!|\ell\sigma|\!))}$.*

Proof. From Proposition 2, we have a polynomial P such that $(\!|\ell|\!) - (\!|r|\!) > \frac{1}{P}$. From 2, we have a uniform bound A such that for all symbol f, $f(x_1, \ldots, x_i + \Delta, \ldots, x_n) \geq f(x_1, \ldots, x_n) + \Delta$ if $x_i > A$. The Lemma is then obtained by induction on the context C. For the base case, we have $(\!|\ell\sigma|\!) \geq (\!|r\sigma|\!) + \frac{1}{P((\!|t_1|\!), \ldots, (\!|t_n|\!))}$ where $t_1, \ldots, t_n$ are in the range of σ. By sub-term property, $(\!|t_i|\!) \leq (\!|\ell\sigma|\!)$, so that $(\!|\ell\sigma|\!) \geq (\!|r\sigma|\!) + \frac{1}{P((\!|\ell\sigma|\!), \ldots, (\!|\ell\sigma|\!))}$. Using A as defined above, the induction step is immediate (by the sub-term property).

Definition 8. *Given a real $A > 0$, we say that the A-size of a closed term t is the number of subterms u of t (including itself) such that $(\!|u|\!) > A$. We note $|t|_A$ the A-size of t.*

Lemma 9. *There is a constant A and a polynomial Q for which $|t|_A \leq Q((\!|t|\!))$ for all closed terms t. For all $B > A$, $|t|_B \leq |t|_A \leq Q((\!|t|\!))$.*

Proof. Take A and P as given by Proposition 4. Consider one term t, we note $a = \frac{1}{P((\!|t|\!))}$. By induction on sub-terms u of t, we show that $|u|_A \leq (\!|u|\!)/a$. If $(\!|u|\!) \leq A$, then $|u|_A = 0 \leq (\!|t|\!)/a$. Otherwise, $(\!|u|\!) > A$. Let us write $u = f(u_1, \ldots, u_n)$, and say that $u_{i_1}, \ldots, u_{i_k}$ verify $(\!|u_{i_j}|\!) > A$. Then,

$$\begin{aligned}
|u|_A &= 1 + |u_{i_1}|_A + \cdots + |u_{i_k}|_A \\
&\geq (a + (\!|u_{i_1}|\!) + \cdots + (\!|u_{i_k}|\!))/a \text{ (by induction)} \\
&\geq (\frac{1}{P((\!|u|\!))} + (\!|u_{i_1}|\!) + \cdots + (\!|u_{i_k}|\!))/a \text{ (sub-term property)} \\
&\geq (\!|f(u_1, \ldots, u_n)|\!)/a \text{ (due to Proposition 4).}
\end{aligned}$$

As a consequence, $|t|_A \leq (\!|t|\!) \times P((\!|t|\!))$.

For $A > 0$, we say that $t = C[\ell\sigma] \to C[r\sigma] = u$ is an A-step whenever $(\!|r\sigma|\!) > A$. We note such a rewriting step $t \to_{>A} u$. Otherwise, it is an $\leq A$-step, and we note it $t \to_{\leq A} u$. We use the usual $*$ notation for transitive closure. In case we restrict the relation to the call by value strategy[2], we add "cbv" as a subscript. Take care that an $\to_{\leq A}$-normal form is not necessarily a normal form for $\to$.

Lemma 10. *There is a constant A and a polynomial P such that for all terms t, any call by value derivation $t \to^*_{\leq A, cbv} u$ has length less than $P((\!|t|\!))$.*

[2] Innermost in the present context.

Proof. Take A and P as in the Lemma above. Let S be the finite set of terms with interpretation smaller than A. According to Proposition 2, the set S is finite. Then, we can define a to be the maximal derivation length of terms in S. Let d be the maximal arity of a term. By induction on terms, we show that the derivation length $t \to^*_{\leq A,\mathrm{cbv}} u$ of $\leq A$-steps is bounded by $\max(|t|_A \times (d+1) \times (a+1), a)$.

- If $(\!|t|\!) \leq A$, then, by definition of a, we have $\mathtt{dh}(t) \leq a = \max(|t|_A \times (d+1) \times (a+1), a)$.
- Otherwise, let us write $t = f(t_1, \ldots, t_n)$. According to $\leq A$-call by value, consider the rewriting $f(t_1, \ldots, t_n) \to^m_{\leq A,\mathrm{cbv}} f(u_1, \ldots, u_n)$. Let us note $i_1, \ldots, i_k$ indices for which $(\!|t_{i_j}|\!) > A$, we have by induction

$$\begin{aligned} m &\leq (n-k) \times a + (d+1) \times (a+1) \times \sum_{j=1}^{k} |t_{i_j}|_A \\ &\leq d \times a + (d+1) \times (a+1) \times \sum_{j=1}^{k} |t_{i_j}|_A. \end{aligned}$$

  Consider $f(u_1, \ldots, u_n)$, if $f(u_1, \ldots, u_n)$ is not a $\leq A$-normal form, since the u_i are $\leq A$-normal form, a $\leq A$-step is of the form $f(u_1, \ldots, u_n) \to u$ with $(\!|u|\!) \leq A$. In which case, the derivation length of u is bounded by a. To sum up, the derivation length is then bounded by $1 + a + d \times (a+1) + (d+1) \times (a+1) \times \sum_{j=1}^{k} |t_{i_j}|_A = (d+1) \times (a+1) \times |t|_A$.

Using the Lemma above, $|t|_A \leq P((\!|t|\!))$, and consequently, the derivation length is bounded by $(d+1) \times (a+1) \times P((\!|t|\!))$.

Lemma 11. *For constructor terms, we have $(\!|t|\!) \leq \Gamma \times |t|$ for some constant Γ.*

Proof. Take $\Gamma = \max\{\frac{1}{\gamma_c} \mid (\!|c|\!)(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i + \gamma_c\}$. By induction on terms.

Lemma 12. *Let us suppose we are given an additive program with interpretation in **Poly**. For a given function symbol f, there is a strategy such that for all constructor terms $t_1, \ldots, t_n$, the derivation length of $f(t_1, \ldots, t_n)$ is bounded by $Q(\max(|t_1|, \ldots, |t_n|))$ where Q is a polynomial.*

Proof. Let us consider A and P_0 as defined in Lemma 10 and B and P_1 as defined in Lemma 8. We define $C = \max(A, B)$. Let us consider the strategy as introduced above: rewrite as long as possible the according to $\to_{\leq C,\mathrm{cbv}}$, and then, apply an C-step. That is, we have $t_1 \to^*_{\leq C,\mathrm{cbv}} t'_1 \to_{>C,\mathrm{cbv}} t_2 \to^*_{\leq C,\mathrm{cbv}} t'_2 \to^*$. In Lemma 10, we have seen that there are at most $(d+1) \times a \times P_0((\!|t_i|\!))$ steps in the derivation $t_i \to^*_{\leq C,\mathrm{cbv}} t'_i$. From Lemma 8, we can state that there are at most $(\!|t_1|\!) \times P_1((\!|t_1|\!))$ such C-steps. Consequently, the derivation length is bounded by $(\!|t_1|\!) \times P_1((\!|t_1|\!)) \times (1 + (d+1) \times a \times P_0((\!|t_1|\!)))$ since $(\!|t_i|\!) < (\!|t_1|\!)$.

Consider now a function symbol $f \in \mathcal{F}$, from Lemma 11, $(\!|f(t_1, \ldots, t_n)|\!) = (\!|f|\!)((\!|t_1|\!), \ldots, (\!|t_n|\!)) \leq (\!|f|\!)(\Gamma \max(|t_1|, \ldots, |t_n|), \ldots, \Gamma \max(|t_1|, \ldots, |t_n|))$. The conclusion is immediate.

Proof (Theorem 4). With the strategy defined above, we have seen that the derivation length of a term $f(t_1, \ldots, t_n)$ is polynomial wrt to $\max(|t_1|, \ldots, |t_n|)$. The computation can be done in polynomial time due to dal Lago and Martini, see [6], together with the fact that the normal form has polynomial size (Lemma 11). For the converse part, we refer the reader to [2] where a proof that PTIME programs can be computed by functional programs with strict interpretations over the integers. This proof can be safely used in the present context.

3.1 Dependency Pairs with polynomial interpretation over the reals

Termination by Dependency Pairs is a general method introduced by Arts and Giesl [26]. It puts into light recursive calls.

Suppose $f(t_1, \ldots, t_n) \to C[g(u_1, \ldots, u_n)]$ is a rule of the program. Then, $(F(t_1, \ldots, t_n), G(u_1, \ldots, u_n))$ is a dependency pair where F and G are new symbols associated to f and g respectively. $S(\mathcal{C}, \mathcal{F}, R)$ denotes the program thus obtained by adding these rules. The dependency graph links dependency pairs $(u, v) \to (u', v')$ if there is a substitution σ such that $\sigma(v) \xrightarrow{*} \sigma(u)$ and termination is obtained when there is no cycles in the graph. Since the definition of the graph involves the rewriting relation, its computation is undecidable. In practice, one gives an approximation of the graph which is bigger. Since this is not the issue here, we suppose that we have a procedure to compute this supergraph which we call the dependency graph.

Theorem 5. *[Arts,Giesl [26]] A TRS $(\mathcal{C}, \mathcal{F}, R)$ is terminating iff there exists a well-founded weakly monotonic quasi-ordering $\geq$, where both $\geq$ and $>$ are closed under substitution, such that*

- *$\ell \geq r$ for all rules $\ell \to r$,*
- *$s \geq t$ for all dependency pairs (s, t) on a cycle of the dependency graph and*
- *$s > t$ for at least one dependency pair on each cycle of the graph.*

It is natural to use the polynomial orderings presented above for the quasi-ordering and the ordering of terms. However, the ordering $>$ is not well-founded on $\mathbf{R}$, so that system may not terminate. Here is such an example.

Example 2. Consider the non terminating system:

$$\begin{pmatrix} f(0) \to 0 \\ f(x) \to f(\mathbf{s}(x)) \end{pmatrix}$$

Take $(\!|0|\!) = 1$, $(\!|\mathbf{s}|\!)(x) = x/2$. The system has a unique dependency pair $F(x) \to F(\mathbf{s}(x))$ for which we can give the interpretation $(\!|F|\!)(x) = x+1$.[3] Take $(\!|f|\!)(x) = x$.

[3] The interpretation is correct since for all terms t, $(\!|t|\!) > 0$.

One way to avoid these infinite descent is to force the inequalities over reals to be of the form $P(x_1, \ldots, x_n) \geq Q(x_1, \ldots, x_n) + \delta$ for some $\delta > 0$ (see for instance Lucas's work [19]). Doing so, one gets a well-founded ordering on reals. We propose an alternative approach to that problem, keeping the original ordering of $\mathbf{R}$.

Definition 9. *A weak polynomial*[4] *algebra for a signature Σ consists of monotone polynomials $(\!|f|\!)$ for all symbols in Σ.*

Definition 10. *A $\mathbf{R}$-DP-interpretation for a program $P = (\mathcal{C}, \mathcal{F}, R)$ is weak polynomial algebra $(\!|-|\!)$ for $S(P)$ such that*

1. *there is $\delta > 0$ such that for each n-ary constructor $\mathbf{c}$ with $n > 0$, for all $x_1, \ldots, x_n \geq 0$, we have $(\!|\mathbf{c}|\!)(x_1, \ldots, x_n) \geq \delta$,*
2. *$(\!|\ell|\!) \geq (\!|r|\!)$ for $\ell \to r \in R$,*
3. *$(\!|s|\!) \geq (\!|r|\!)$ for $(s, r) \in DP(R)$,*
4. *for each dependency pair (s, t) in a cycle, $(\!|s|\!) > (\!|r|\!)$ holds.*

The main difference with say [22] is that we do not ask for the existence of some δ such that $(\!|s|\!) \geq (\!|r|\!) + \delta$ in the last equation. To simplify the proof of Theorem 6, we took $(\!|s|\!) > (\!|t|\!)$ for all dependency pairs in a cycle, and not for only one. We make the conjecture that the theorem holds, even in the standard case: for each cycle, there is a dependency pair (s, t) such that $(\!|s|\!) > (\!|r|\!)$.

Lemma 13. *Suppose that a polynomial P is weakly monotonic in every argument on $(\mathbf{R}^+)^n$. Suppose that it is not constant wrt some variable. Then for any arbitrarily small $\delta > 0$, there is a polynomial P_δ such that for all $x_1 \geq \delta, \ldots, x_n \geq \delta$, if $x_i \geq P_\delta(A)$ for some i, then $P(x_1, \ldots, x_n) \geq A$.*

Proof. Consider first that $n = 1$. We write $P(x) = ax^k + R(x)$ with k being the degree of P. There is $B > 0$ such that for all $x > B$, we have $P(x) \geq a/2x^k$. Then, define $P_\delta(A) = 1 + B + \dfrac{2}{a} \times A$. We observe

$$\begin{aligned} P(x) &\geq P(P_\delta(A)) \text{ by monotonicity} \\ &\geq \frac{a}{2}(P_\delta(A))^k \text{ since } P_\delta(A) > B \\ &\geq \frac{a}{2} P_\delta(A) \text{ since } P_\delta(A) > 1 \\ &\geq A \text{ by definition of } P_\delta(A) \end{aligned}$$

For a polynomial with $n + 1$ variable $P(x_1, \ldots, x_{n+1})$. Let us consider one of the variables. Wlog, we take it to be the last one: x_{n+1}. Then, for an arbitrary small δ, P is not constant wrt x_{n+1}. And, for all $x_1, \ldots, x_n \geq \delta$, we have $P(x_1, \ldots, x_n, x_{n+1}) \geq P(\delta, \ldots, \delta, x_{n+1})$ by monotonicity. Define $Q(x_{n+1}) = P(\delta, \ldots, \delta, x_{n+1})$, and we come back to the case with one variable.

[4] L-polynomial algebra in the terminology of Lucas [19].

Theorem 6. *A program with a* $\mathbf{R}$*-DP-interpretation is strongly terminating. Moreover, its derivation height is bounded by* $2^{2^{2^{O(n)}}}$ *with* n *the size of the input. This bound is tight.*

Proof. Since, in the present terms, the hypothesis of the Theorem 5 do not hold, we come back to its proof and show that we have an extra-ingredient to get the termination property. Actually, we give a presentation of the proof by means of call-tree. Since the size of a call tree gives an upper bound on the derivation height, we get a direct more proof for our Theorem.

Let us consider the call-tree of a term $f(t_1, \ldots, t_n)$ for some constructor terms $t_1, \ldots, t_n$. The size of the call tree bounds the derivation height of the term $f(t_1, \ldots, t_n)$. Suppose that we prove that there is a polynomial P such that the height of the call tree is polynomial wrt to the interpretation of $F(t_1, \ldots, t_n)$. Since the branching of the call-tree is bounded by R the maximal size of the right hand side of rules, the size of the call-tree is bounded by $R^{P(\llparenthesis f(t_1,\ldots,t_n) \rrparenthesis)}$. But, since $\llparenthesis F(t_1, \ldots, t_n) \rrparenthesis \leq 2^{2^{O(|F(t_1,\ldots,t_n)|)}}$ by Lemma 1, we have $P(\llparenthesis F(t_1, \ldots, t_n) \rrparenthesis) \leq P(2^{2^{O(\sum_{i=1}^{n} |t_i|+1)}}) = 2^{2^{O(\max_{i=1}^{n} |t_i|)}}$. And, the conclusion follows.

So, it remains to prove that the existence of such a polynomial P. This is done by induction on the rank of symbols. But, first, consider a rule $f(\boldsymbol{p_i})\sigma \to C[g(\boldsymbol{e_i})]\sigma$ with f and g of same rank. Then, we have $\llparenthesis F(\boldsymbol{p_i}) \rrparenthesis > \llparenthesis G(\boldsymbol{e_i}) \rrparenthesis$. Notice that Proposition 2 applies in the present context for dependency pairs of same rank, so that we can state that $\llparenthesis F(\boldsymbol{p_i}) \rrparenthesis - \llparenthesis G(\boldsymbol{e_i}) \rrparenthesis > \dfrac{1}{P(x_1, \ldots, x_n)}$ where x_i are the variables of $F(\boldsymbol{p_i})$. Wlog, we can suppose (possibly by padding arguments) that P is common to all the dependency pairs of equal rank.

Now, let's work on the induction. Actually, the base case draw the shape of the proof.

Base case Let us consider a symbol f of minimal rank and some constructor terms $t_1, \ldots, t_n$ and finally, let $P = f_1(\boldsymbol{u_1}), \ldots, f_k(\boldsymbol{u_k})), \ldots$ be a path in the call tree of $f(t_1, \ldots, t_n)$. From hypothesis (4) of Definition 10, we can state that $\llparenthesis F_1(\boldsymbol{u_1}) \rrparenthesis > \llparenthesis F_2(\boldsymbol{u_2}) \rrparenthesis > \cdots$.

Given a dependency pair given by $\ell = h(\boldsymbol{p_i}) \to C[g(\boldsymbol{e_i})] = r$ with h and g of equal rank, we extract the sequence $P_\psi = (f_{\psi(i)}(\boldsymbol{u_{\psi(i)}}))_{i \in \mathbf{N}}$ from P such that

- $f_{\psi(i)} = h$ and
- $\ell\sigma = f_{\psi(i)}(\boldsymbol{u_{\psi(i)}}) \to C[f_{\psi(i)+1}(\boldsymbol{u_{\psi(i)+1}})] = r$.

Wlog, we can suppose that $\llparenthesis H(\boldsymbol{p_i}) \rrparenthesis$ varies with $x_1, \ldots, x_m$ and is constant wrt $x_{m+1}, \ldots, x_n$. Let us consider a (possibly empty) set $\theta \subseteq \{1..m\}$. To simplify the readability of the proof, we suppose $\theta = \{k..m\}$.

We extract the sequence $P_\varphi = (f_{\varphi(i)}(\boldsymbol{u_{\varphi(i)}}))_{i \in \mathbf{N}}$ from P_ψ such that

1. for all $j \leq k$, $\llparenthesis \sigma(x_j) \rrparenthesis \neq 0$,
2. for all $j \geq k$, $\llparenthesis \sigma(x_j) \rrparenthesis = 0$,

where σ is the substitution mentioned in the construction of P_ψ.

Observe that $(\!|H(\boldsymbol{p_i})|\!)$ varies with $x_1, \ldots, x_k$. And that $(\!|\sigma(x_i)|\!) \geq \delta$ for all $i \leq k$. Consequently, from Lemma 13, we get a polynomial Q such that $(\!|\sigma(x_i)|\!) \leq Q((\!|f_i(\boldsymbol{u_i})|\!))$. But, then, $(\!|H(\boldsymbol{p_i})|\!) \geq (\!|G(\boldsymbol{e_i})|\!) + \dfrac{1}{P(Q((\!|F_i(\boldsymbol{u_i})|\!)), \ldots, Q((\!|F_i(\boldsymbol{u_i})|\!)))} \geq (\!|G(\boldsymbol{e_i})|\!) + \dfrac{1}{P(Q((\!|F_1(\boldsymbol{u_1})|\!)), \ldots, Q((\!|F_1(\boldsymbol{u_1})|\!)))}$.

As a conclusion, we have a polynomial bound on the initial subsequence restricted to the redex of a chosen rule and a particular choice of variable whose interpretation is 0. Since the number of dependency pairs is finite, since the number of choice for the variables is finite (bounded by 2^D where D is the maximal number of variables in a rule), since only (finitely many) constants have interpretation equal to 0, we can say that the length of the sequence P is bounded by $N \times 2^D \times |\mathcal{C}| \times P(Q((\!|F_1(\boldsymbol{u_1})|\!)), \ldots, Q((\!|F_1(\boldsymbol{u_1})|\!)))$.

Induction step Suppose f has a higher rank. A path of a call-tree can be decomposed in a sub-path of nodes with function of rank of f, and symbols of smaller rank. The depth of symbols of rank of f can be treated as it has been done in the base case. For symbols of lower rank, one employs the induction. The depth of symbols of different ranks sums, and we get a call-tree of polynomial depth in the interpretation of the initial term.

The bound is tight as shown by the next example.

Example 3. The Quantified Boolean Formula (QBF) problem is PSPACE complete. It consists in determining the validity of a boolean formula with quantifiers over propositional variables. Without loss of generality, we restrict formulae to $\neg, \vee, \exists$. QBF problem is solved by the following program.

$$
\begin{array}{rrr}
\mathtt{not}(\mathbf{tt}) \to \mathbf{ff} & \mathtt{not}(\mathbf{ff}) \to \mathbf{tt} & \\
\mathtt{or}(\mathbf{tt}, y) \to \mathbf{tt} & \mathtt{or}(x, \mathbf{tt}) \to \mathbf{tt} & \mathtt{or}(\mathbf{ff}, \mathbf{ff}) \to \mathbf{ff} \\
\mathbf{0} = \mathbf{0} \to \mathbf{tt} & \mathbf{s}(x) = \mathbf{0} \to \mathbf{ff} & \\
\mathbf{0} = \mathbf{s}(y) \to \mathbf{ff} & \mathbf{s}(x) = \mathbf{s}(y) \to x = y & \\
\mathtt{in}(x, \varepsilon) \to \mathbf{ff} & \mathtt{in}(x, \mathbf{cons}(a, l)) \to \mathtt{or}(x = a, \mathtt{in}(x, l)) &
\end{array}
$$

$$
\begin{array}{rl}
\mathtt{verify}(\mathbf{Var}(x), t) & \to \mathtt{in}(x, t) \\
\mathtt{verify}(\mathbf{Not}(\varphi), t) & \to \mathtt{not}(\mathtt{verify}(\varphi, t)) \\
\mathtt{verify}(\mathbf{Or}(\varphi_1, \varphi_2), t) & \to \mathtt{or}(\mathtt{verify}(\varphi_1, t), \mathtt{verify}(\varphi_2, t)) \\
\mathtt{verify}(\mathbf{Exists}(n, \varphi), t) & \to \mathtt{or}(\mathtt{verify}(\varphi, \mathbf{cons}(n, t)), \mathtt{verify}(\varphi, t)) \\
\mathtt{qbf}(\varphi) & \to \mathtt{verify}(\varphi, \varepsilon)
\end{array}
$$

They admit the following interpretation :

$$
\begin{aligned}
(\!|0|\!) = (\!|\varepsilon|\!) &= 1\\
(\!|\mathbf{s}|\!)(x) = (\!|\mathbf{Not}|\!)(x) = (\!|\mathbf{Var}|\!)(x) &= x+1\\
(\!|\mathbf{cons}|\!)(x,y) = (\!|\mathbf{Or}|\!)(x,y) = (\!|\mathbf{Exists}|\!)(x,y) &= x+y+1\\
(\!|\mathtt{or}|\!)(x) = (\!|\mathtt{not}|\!)(x) = (\!|\mathtt{qbf}|\!)(x) &= 1\\
(\!|=|\!)(x,y) = (\!|\mathtt{in}|\!)(x,y) = (\!|\mathtt{verify}|\!)(x,y) &= 1\\
(\!|\mathrm{NOT}|\!)(x) &= x\\
(\!|\mathrm{OR}|\!)(x,y) = (\!|\mathrm{EQ}|\!)(x,y) &= \max(x,y)\\
(\!|\mathrm{IN}|\!)(x,y) &= x+y\\
(\!|\mathrm{VERIFY}|\!)(x,y) &= 2\times x+y+1\\
(\!|\mathrm{QBF}|\!)(x) &= 2x+1
\end{aligned}
$$

It is well known that the derivation height of the QBF is exponential wrt the number of nested **Exists** symbols. Since the following program builds an input of double exponential depth, we get the triple exponential bound on the derivation height.

$$
\begin{aligned}
\mathtt{add}(\mathbf{0},y) &\to y\\
\mathtt{mult}(\mathbf{0},y) &\to \mathbf{0}\\
\mathtt{dexp}(\mathbf{0}) &\to \mathbf{s}(\mathbf{s}(\mathbf{0}))\\
\mathrm{e}(0) &\to \mathbf{tt}\\
\mathrm{main}(x) &\to \mathtt{qbf}(\mathrm{e}(x))
\end{aligned}
\qquad
\begin{aligned}
\mathtt{add}(\mathbf{s}(x),y) &\to \mathbf{s}(\mathtt{add}(x,y))\\
\mathtt{mult}(\mathbf{s}(x),y) &\to \mathtt{add}(y,\mathtt{mult}(x,y))\\
\mathtt{dexp}(\mathbf{s}''(x)) &\to \mathtt{mult}(\mathtt{dexp}(x),\mathtt{dexp}(x))\\
\mathrm{e}(\mathbf{s}(n)) &\to \mathbf{Exists}(n,\mathrm{e}(n))
\end{aligned}
$$

with interpretations:

$$
\begin{aligned}
(\!|\mathbf{0}|\!) &= 0\\
(\!|\mathbf{s}''|\!)(x) &= (x+3)^2\\
(\!|e|\!)(x) &= 3x \qquad (\!|\mathrm{E}|\!)(x) = x\\
(\!|\mathrm{main}|\!)(x) &= (\!|\mathtt{qbf}(\mathrm{e}(x))|\!) \qquad (\!|\mathrm{MAIN}|\!)(x) = 6x+1\\
(\!|\mathtt{add}|\!)(x,y) &= x+y \qquad (\!|\mathrm{ADD}|\!)(x,y) = x+y\\
(\!|\mathtt{mult}|\!)(x,y) &= x\times y \qquad (\!|\mathrm{MUL}|\!)(x,y) = (x+1)(y+1)\\
(\!|\mathtt{dexp}|\!)(x) &= x+2 \qquad (\!|\mathrm{DEXP}|\!)(x) = x
\end{aligned}
$$

We observe that the term $\mathrm{main}(\underbrace{\mathbf{s}''\cdots\mathbf{s}''}_{n \text{ times } \mathtt{q}}\mathbf{0}) \to^* \mathbf{Exists}(s^{2^n}(\mathbf{0},\ldots,\mathbf{Exists}(\mathbf{0},\mathbf{tt})\cdots)$ gives the triple exponential derivation height lower bound. We provide in the technical report an interpretation for the system.

It is not clear whether there are some programs with **R**-DP-interpretation which do not admit L-weak interpretation. Nevertheless, there is at least one good point for **R**-DP-interpretation: the logical formulation of the synthesis of **R**-DP-interpretation involves less alternation of quantifiers. Suppose that we are given program and a fixed degree, the logical formula corresponding to $(\!|\ell|\!) > (\!|r|\!)$ is $\forall x_1,\ldots,x_n > 0 : (\!|\ell|\!)(x_1,\ldots,x_n) > (\!|r|\!)(x_1,\ldots,x_n)$. While for expressing $>_\delta$,

we have to write $\exists \delta > 0 : \forall x_1, \ldots, x_n > 0 : (\!|\ell|\!)(x_1, \ldots, x_n) > (\!|r|\!)(x_1, \ldots, x_n) + \delta$. Since the complexity of the QED procedure depends drastically on the number of alternation, we may hope to get thus a better procedure.

With respect to complexity, the proof of Theorem 6 gives us some insight on the cost of a computation. Consider programs with a **R**-DP-interpretation such that any constructor $\mathbf{c}$ has an interpretation of the form $k_{\mathbf{c}} + \sum_i x_i$, in other words, an additive **R**-DP-interpretation in [22] terminology. There is a constant K such that for all contructor terms t, $(\!|t|\!) \leq K.|t|$, so that $(\!|f(t_1, \ldots, t_n)|\!)$ is polynomially bounded wrt the size of the (constructor) terms t_i.[5] The proof shows then that the nesting of function calls is itself bounded polynomially. If, furthermore the interpretation of capital functions verify the sub-term property, due to Lemma 13, we can state that the size of arguments remain polynomial. This is an other formulation (and a slightly more general one) of the hypothesis of "bounded recursion call" which can be found in [22]. So that computations can be performed within Pspace. Since polynomial time can be done with such systems (cf. [2]), and QBF can be simulated, it is then clear that the following Theorem holds:

Theorem 7. *Functions computed by programs*

- *with additive* **R***-DP-interpretations*
- *the interpretation of capital symbols F has the sub-term property*

are exactly Pspace *computable functions.*

Proof. The completeness comes from the example of the QBF, plus the fact that all functions in polynomial time can be easily computed by such programs. It is then routine to plug all these programs together.

Conclusion

If one goes back to the two characterization of complexity classes presented in this paper, one sees that we essentially use two arguments: a) interpretations with the subset properties provide a polynomial bound wrt the interpretation of the initial interpretation, and b) the size of terms is polynomial wrt their interpretation.

As a consequence, our result can be used in other context such as proofs of termination by matrix interpretations [11]. Potentially, any system dealing with decreasing chain of (interpreted) could be treated.

References

1. Saugata Basu, Richard Pollack, and Marie.-Françoise Roy. *Algorithms in real algebraic geometry.* Springer, Berlin Heidelberg New York, 2003.

[5] See [3] for a proof.

2. Guillaume Bonfante, Adam Cichon, Jean-Yves Marion, and Hélène Touzet. Algorithms with polynomial interpretation termination proof. *J. Funct. Program.*, 11(1):33–53, 2001.
3. Guillaume Bonfante, Jean-Yves Marion, and Jean-Yves Moyen. Quasi-interpretations: a way to control resources. *Theoretical Computer Science*, 2009. to appear.
4. Adam Cichon and Jean-Yves Marion. The light lexicographic path ordering. Technical report, Loria, 2000. Workshop Rule.
5. Michel Coste, Henri Lombardi, and Marie-Françoise Roy. Dynamical method in algebra: Effective nullstellensätze. *Annals of Pure and Applied Logic*, 111:203256, 2001.
6. Ugo dal Lago and Simone Martini. Derivational complexity is an invariant cost model. In *Foundational and Practical Aspects of Resource Analysis (FOPARA '09)*, 2009.
7. Nachum Dershowitz. A note on simplification orderings. *Information Processing Letters*, pages 212–215, 1979.
8. Nachum Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, pages 69–115, 1987.
9. Nachum Dershowitz and Jean-Pierre Jouannaud. *Handbook of Theoretical Computer Science vol.B*, chapter Rewrite systems, pages 243–320. 1990.
10. Nao Hirokawa and Georg Moser. Automated complexity analysis based on the dependency pair method. In Alessandro Armando, Peter Baumgartner, and Gilles Dowek, editors, *IJCAR*, volume 5195 of *Lecture Notes in Computer Science*, pages 364–379. Springer, 2008.
11. Dieter Hofbauer. Proving termination with matrix interpretations. In *Proceedings of the 17th International Conference on Rewriting Techniques and Applications, RTA-06*, volume 4098 of *Lecture Notes in Computer Science*, pages 328–342, Seattle, USA, 2006. Springer-Verlag.
12. Dieter Hofbauer and Clemens Lautemann. Termination proofs and the length of derivations. *Lecture Notes in Computer Science*, 355:167–177, 1988.
13. Hoon Hong and Dalibor Jakus. Testing positiveness of polynomials. *J. Autom. Reasoning*, 21(1):23–38, 1998.
14. Gérard Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27(4):797–821, 1980.
15. Gérard Huet and Derek Oppen. Equations and rewrite rules: A survey. In R. V. Book, editor, *Formal Language Theory: Perspectives and Open Problems*, pages 349–405. AP, 1980.
16. D.S. Lankford. On proving term rewriting systems are noetherien. Technical report, 1979.
17. Henri Lombardi. Effective real nullstellensatz and variants. In *Effective Methods in Algebraic Geometry*, volume 94, page 263288. Mora T., Traverso C. Birkhaüser, 1991.
18. Salvador Lucas. On the relative power of polynomials with real, rational, and integer coefficients in proofs of termination of rewriting. *Appl. Algebra Eng., Commun. Comput.*, 17(1):49–73, 2006.
19. Salvador Lucas. Practical use of polynomials over the reals in proofs of termination. In *PPDP '07: Proceedings of the 9th ACM SIGPLAN international conference on Principles and practice of declarative programming*, pages 39–50, New York, NY, USA, 2007. ACM.
20. Salvador Lucas and Ricardo Peña. Rewriting techniques for analysing termination and complexity bounds of safe programs. In *LOPSTR 08*, pages 43–57, 2008.

21. Jean-Yves Marion and Romain Péchoux. Resource analysis by sup-interpretation. In *FLOPS*, volume 3945 of *Lecture Notes in Computer Science*, pages 163–176. Springer-Verlag, 2006.
22. Jean-Yves Marion and Romain Péchoux. Characterizations of polynomial complexity classes with a better intensionality. In Sergio Antoy and Elvira Albert, editors, *PPDP*, pages 79–88. ACM, 2008.
23. Olha Shkaravska, Marko van Eekelen, and Ron van Kesteren. Polynomial size analysis of first-order shapely functions. *CoRR*, abs/0902.2073, 2009.
24. Gilbert Stengle. A nullstellensatz and a positivstellensatz in semialgebraic geometry. *Mathematische Annalen*, 207(2):87–97, 1973.
25. Alfred Tarski. *A Decision Method for Elementary Algebra and Geometry*. University of California Press, 1951. 2nd edition.
26. Arts Thomas and Jürgen Giesl. Termination of term rewriting using dependency pairs. *Theoretical Computer Science*, 236:133–178, 2000.