RPO constraint solving is in NP

Paliath Narendran[1*], Michael Rusinowitch[2], and Rakesh Verma[3**]

[1] Institute of Programming and Logics, Department of Computer Science, State University of New York at Albany, Albany, NY 12222, USA
dran@cs.albany.edu

[2] LORIA, 615, rue du jardin botanique, BP 101, 54602 Villers les Nancy cedex, France
rusi@loria.fr

[3] Dep. of Comp. Science, University of Houston, Houston, TX 77204, USA
rmverma@cs.uh.edu

Abstract. A new decision procedure for the existential fragment of ordering constraints expressed using the recursive path ordering is presented. This procedure is nondeterministic and checks whether a set of constraints is solvable over the *given* signature, i.e., the signature over which the terms in the constraints are defined. It is shown that this nondeterministic procedure runs in polynomial time, thus establishing the membership of this problem in the complexity class NP for the first time.

1 Introduction

Reduction orderings have been developed for testing the termination of term rewrite systems. Among these orderings the Recursive Path Ordering (RPO) [3] is one of the most used since it gives simple syntactic conditions for orienting equations as rules. An optimal way to apply RPO is to orient an equation $u = v$ as $u \rightarrow v$ if for all ground substitutions θ, $u\theta \succ_{rpo} v\theta$. When the precedence is total this amounts to checking the unsatisfiability of the *ordering constraint* $v \succ_{rpo} u \vee u \sim_{rpo} v$. For instance, $h(f(x,x), f(y,y), k(z)) = h(f(x,y), f(x,y), z)$ cannot be oriented with the standard version of RPO. However the *constrained* version allows for an immediate left-to-right orientation.

Many extensions of RPO have been designed in order to orient equations of the kind above [16, 9, 13, 17]. Since these extensions (in the multiset case) coincide with RPO on ground terms for total precedence, *constrained* RPO can be considered as subsuming them: if two terms are comparable with any of these extensions they are indeed comparable with *constrained* RPO.

Symbolic constraints are also widely used in automated deduction for narrowing the search space ([6, 11, 15]). They are commonly formulas to be interpreted in a Herbrand universe (a term algebra). Deduced formulas inherit constraints

* Research supported in part by the NSF grants CCR-9712396 and INT-9401087

** Research supported in part by NSF grant CCR-9732186

from their antecedents. These constraints reduce the number of potential instantiations of the formulas they label. For instance, unification or equational constraints prevent redundant deductions by forbidding equality replacements in subformulas that have been frozen as constraints. Ordering constraints based on RPO and its variants allow us to express theorem-proving strategies such as ordered resolution, ordered paramodulation [11, 15] and unfailing completion [7]. Note that the completeness of the above strategies only assumes that the ordering can be extended to a simplification ordering that is total on ground terms. Although RPO is not total on ground terms it can be extended easily in such a way.

An important breakthrough in this area was the proof by H. Comon that the *Lexicographic Path Ordering* (LPO) constraint solving problem is decidable [1]. Subsequently R. Nieuwenhuis [14] showed that the problem is NP-complete. In the case of RPO, the decidability of constraint solving was proved by J.P. Jouannaud and M. Okada [8]. In fact, they gave an algorithm for a general ordering combining both RPO and LPO. The RPO case is also considered by [10] where a polynomial bound on the depths of the terms in a solution is derived. This is the only upper bound that has been given for RPO constraint solving so far, as far as we know; however, it does not help us show that the solvability problem is in NP. In this paper we show it to be in NP for the first time, and therefore NP-complete since it is NP-hard by [14]. Unlike [8] our algorithm does not use any predecessor function. Such a function creates big terms in the solving process and it is unlikely that the algorithm in [8] could be modified into a NP one.

Note also that the proof that LPO constraint solving is NP cannot be adapted for RPO since it relies heavily on the fact that a non-natural term is a subterm of its successor, which is not the case for RPO.

Given a total precedence on function symbols, our non-deterministic algorithm for solving a constraint C is very simple: The first step is standard ([1]) and amounts to guessing a linear ordering L on the set of subterms of C. In the second step we introduce a quadratic number of new variables in the chain L to get L'. Finally we check the so-called *feasibility of gaps* where a gap in a chain is a subsystem of the chain with non-variable terms only occurring at its ends. A gap is feasible if it is entailed by former (earlier) gaps (in the chain). If all gaps of L' are feasible and L' implies C then the algorithm returns "satisfiable".

2 Basic Definitions

We assume that F is a finite set of function symbols given with their arity. $T(F,V)$ is the set of finite terms built on F and an alphabet V of (first-order) *variable symbols*. $\equiv$ denotes syntactic equality of terms. $T(F)$ is the set of terms which do not contain any variables. A multiset over a set X is a function M from X to the natural numbers. Any ordering $>$ on X can be extended to an ordering $\gg$ on finite multisets over X as follows: $M \gg N$ if a) $M \neq N$ and b) whenever $N(x) > M(x)$ then $M(y) > N(y)$ for some $y > x$. Note that if $>$ is well-founded so is $\gg$.

Recursive Path Ordering (*rpo*) as originally defined by Dershowitz [3] (sometimes referred to as *multiset path ordering* or *mpo* to distinguish it from *lpo*) is defined as follows:

Given a finite total precedence $\succ_F$ on functions,

$$s = f(s_1, \ldots, s_m) \succ_{rpo}^{F} g(t_1, \ldots, t_n) = t$$

if and only if one of the following holds:

1. $f \succ_F g$ and $s \succ_{rpo}^{F} t_i$ for all $1 \leq i \leq n$
2. $f = g$, $m = n$ and
$$\{s_1, \ldots, s_m\} \ggcurly_{rpo}^{F} \{t_1, \ldots, t_m\}$$
3. There exists a j, $1 \leq j \leq m$, such that $s_j \succ_{rpo}^{F} t$ or $s_j \sim_{rpo} t$.

where $\sim_{rpo}$ is defined as "permutatively equal," or, in other words, the terms are treated as *unordered* trees. The multiset extension of $\succ_{rpo}$ used above, namely $\ggcurly_{rpo}$, is defined in terms of this equivalence.

An *RPO constraint solving problem* is a quantifier-free first-order formula constructed from terms and the binary predicate symbols "$\succ$" and "$=$". The formula is interpreted in $T(F)$, with $\succ$ as $\succ_{rpo}$ and $=$ as $\sim_{rpo}$.

Note that the smallest ground term is also the smallest constant a. Let f be the lowest nonconstant function symbol. Let $\{a_k \succ_F a_{k-1} \succ_F \ldots \succ_F a_1\}$ be the possibly empty set of constants smaller than f and let $Const = \{a_k \succ_F a_{k-1} \succ_F \ldots \succ_F a_1 \succ_F a\}$. For readability we present in the body of the paper the case where f is of arity 2 and in the Appendix we give the modifications needed for handling the general case.

Let g be the smallest symbol greater than f; then the first non-successor term w.r.t. $\succ_{rpo}$ is $g(a, \ldots, a_{arity(g)})$. We assume in the following that g has arity 2. The proof only needs trivial change in all other cases (including arity 0 or 1).

The set of ground terms up to the permutative equivalence, $T(F)/\sim_{rpo}$, is well-ordered by $\succ_{rpo}$. It has been proved (see e.g. [10]) that $\circ$, defined below, is a successor function on this well-ordered set:

$$\circ(s) = \begin{cases} a_{i+1} & \text{if } s = a_i \text{ and } i < k, \\ f(a, a) & \text{if } s = a_k, \\ f(a, s) & \text{if } s = h(s_1, \ldots, s_n) \text{ and } h \succ_F f, \\ f(a, \circ(s_2)) & \text{if } s = f(s_1, s_2) \text{ and } s_1 \sim_{rpo} s_2, \\ f(\circ(s_1), s_2) & \text{if } s = f(s_1, s_2) \text{ and } s_1 \prec_{rpo} s_2. \\ f(s_1, \circ(s_2)) & \text{if } s = f(s_1, s_2) \text{ and } s_2 \prec_{rpo} s_1. \end{cases}$$

Note that $\circ$ increases the depth of a term by at most 1. For instance one has $\circ(f(f(a, a), f(a, a))) = f(a, f(a, f(a, a)))$.

3 Example

We consider a precedence $a < f < g$ with f and g as above and the problem:

$$s \equiv f(g(g(u,u), g(v,v)), u) \prec t \equiv f(g(g(u,v), g(u,v)), f(a,a))$$

Guessing a linear ordering between subterms allows us to get for instance:

$$a \prec u = v \prec f(a,a) \prec g(a,a) \prec g(u,u) = g(u,v) = g(v,v) \prec g(g(u,v), g(u,v)) = \\ g(g(u,u), g(v,v)) \prec s \prec t$$

In the following we shall call such a system a *simple* system: the equality relations are compatible; also any inequality that can be decomposed according to the RPO definition is entailed from inequalities to its left. However this simple system is not satisfiable since no term can be inserted between a and $f(a,a)$: we say in this case that the gap $a \prec u \prec f(a,a)$ is not feasible. A satisfiable simple system could have been obtained if we had instead guessed $u = a$.

4 Linear and simple systems

The first step of our procedure follows the same strategy as [1]. Given an *RPO-constraint solving problem* we transform it, in non-deterministic polynomial time, into a *linear system,* defined as follows, by first guessing the relations between all subterms occurring in C:

Definition 4.1. A *linear system* L is a linear order of the form

$$a \# \ldots \# g(a,a) \# \ldots$$

which is closed under subterm and where each symbol $\#$ is either $\prec$ or $=$. In other words, if t is in the system then every subterm s of t has to occur to the left of it. For ease of exposition, we assume that the terms a and $g(a,a)$ are in L.

Note that any solution of C satisfies some linear system extracted from C. Therefore satisfiability of C is reduced to checking that some linear system extracted from C is satisfiable and entails C. Entailment can be verified easily by simple evaluation of C using direct information from the linear system. We show now how to check the satisfiability of a linear system.

At the expense of polynomial-time work we can assume that

- Terms in a linear system will be always represented in such a way that the arguments to each function are given in ascending order in terms of $\prec$.

We further assume that L is *rpo-closed,* which is defined as follows:

A relation $\succ$ on $T(F,V)$ is *rpo-closed* if (i) $a \succ s$ is not in the relation for any term s and (ii) for every $f(s_1, \ldots, s_m) \succ^* g(t_1, \ldots, t_n)$ one of the following holds:

1. $f \succ_F g$ and $s \succ^* t_i$ for all $1 \leq i \leq m$
2. $f = g$, $m = n$ and
$$\{s_1, \ldots, s_m\} \succ\!\!\succ^* \{t_1, \ldots, t_m\}$$
3. There exists a j, $1 \leq j \leq m$, such that $s_j \succ^* t$ or $s_j \equiv t$.

where $\succ\!\!\succ^*$ is the multiset extension of $\succ^*$, the transitive closure of $\succ$.

It can be checked in polynomial time that a linear system is rpo-closed.

We first concentrate on the equational part $eq(L)$ of a linear system L. We do not want to solve it explicitly since replacing every variable by its solution may generate exponential size terms. We rather adopt a technique similar to the *DAG solved form* approach by considering the equivalence relation generated by all equalities in L, and denoted by $=_L$.

Definition 4.2. *A linear system L is* eq-feasible *if:*

1. *two terms with different root function symbols are not equivalent modulo $=_L$.*
2. *a term and a proper subterm of it are not equivalent modulo $=_L$.*
3. *$f(s_1, \ldots, s_n) =_L f(t_1, \ldots, t_n)$ implies $s_i =_L t_i$ for $i = 1, \ldots, n$*

It is clear that a satisfiable L is eq-feasible. The last requirement is satisfied because the arguments are ordered w.r.t. $\prec$. We also have:

Proposition 4.1. *It can be checked in polynomial time that a linear system is eq-feasible. If L is eq-feasible then every ground substitution whose domain is the set of variables in L is a solution of $eq(L)$.*

This proposition shows that once the system is eq-feasible we only need to show existence of solutions for the inequalities.

Definition 4.3. *A* simple system *is an eq-feasible linear system.*

Given a simple system S we shall "identify" a term with its equivalence class modulo $=_S$. For instance a variable is a class that only contains variables. The root symbol of a non-variable term is the root symbol of any non-variable term of its class. The arguments of a non-variable term are the classes of the arguments of any non-variable term of its class (recall that arguments are ordered w.r.t. $\prec$).

Definition 4.4. *A* gap *in a system is a subchain with non-variable terms at both ends and only variables in the middle.*

5 Membership in NP

Let S be an rpo-closed simple system

$$a \prec \ldots \prec g(a, a) \prec \ldots$$

The function π denotes the position of a term on S from the left. Thus $\pi(a) = 0$.

Let i be an integer. An i-expansion of S between positions j and k is an introduction of new variables $z_1, \ldots, z_i$ between j and k.

By a *natural number term* we mean a term t such that $t \prec g(a,a)$. We use N to denote the set of all natural number terms. The numerical value ν of a natural number term is defined as follows: $\nu(a) = 0$ and, for $s \neq a$, $\nu(s) = k$ if s is the k^{th} successor of a.

Note that computing the set of $\nu(b)$ for all ground terms b occurring in a simple system S is polynomial in $|S|$.

The following easy result will be very useful in the proof. It can be checked by induction on $f(t_1,t_2)$ w.r.t. $\prec_{rpo}$.

Fact: Given natural number terms $f(s_1,s_2) \prec_{rpo} f(t_1,t_2)$, we have $\nu(f(t_1,t_2)) - \nu(f(s_1,s_2)) = \nu(s_2) - \nu(s_1) + \nu(t_1) + 1/2[(\nu(t_2)+\nu(s_2)+2)(\nu(t_2)-\nu(s_2)-1)]+1$.

Lemma 5.1. *Let t_1 and t_2 be two terms in a simple system S =*

$$a \prec \ldots \prec t_1 \prec \ldots \prec t_2 \prec \ldots$$

and let θ be a solution of S. Let $\delta = \pi(t_2) - \pi(t_1)$. Then $\circ^{\delta}(\theta(t_1)) \preceq \theta(t_2)$.

For the sake of exposition we often use the phrase "k terms are possible between s and t" to mean $\circ^k(s) \prec t$.

Lemma 5.2. *Let t_1 and t_2 be two terms in a simple system S =*

$$a \prec \ldots t_1 \ldots t_2 \ldots \prec g(a,a)$$

and let θ be a solution of S. Let $\Delta = \nu(\theta(t_2)) - \nu(\theta(t_1)) - 1$ and $\delta = \pi(t_2) - \pi(t_1) - 1$. Then, for all $k < (\Delta - \delta)$ there is a k-expansion S' of S between $\pi(t_1)$ and $\pi(t_2)$ such that θ can be extended to a solution of S'.

A more general statement is the following:

Lemma 5.3. *Let t_1 and t_2 be two terms in a simple system S =*

$$a \prec \ldots \prec t_1 \prec \ldots \prec t_2 \prec \ldots$$

and let θ be a solution of S. Let $\delta = \pi(t_2) - \pi(t_1) - 1$. Then, for all $m > \delta$ the following holds: if $\circ^m(\theta(t_1)) \prec \theta(t_2)$, then there is an $(m-\delta)$-expansion S' of S between $\pi(t_1)$ and $\pi(t_2)$ such that θ can be extended to a solution of S'.

Definition 5.1. *A gap $G = (s \prec x_1 \prec \ldots \prec x_n \prec t)$ is* rigid *if and only if none of the x_i's occur in t. G is* elastic *if and only if it is not rigid, i.e., if x_i occurs in t for some $1 \leq i \leq n$.*

Definition 5.2. *A rigid gap $G = (s \prec x_1 \prec \ldots \prec x_n \prec t)$ in a simple system S is* feasible *if and only if one of the following holds:*

(a) *s and t are ground natural-number terms, s is a constant, and $\nu(t) - \nu(s) - 1 \geq n$.*

(b) *$s = f(s_1, s_2)$, $t = f(t_1, t_2)$, $t \prec g(a, a)$ and*

$$(\pi(s_2) - \pi(s_1)) + \pi(t_1) + \frac{(\pi(t_2) + \pi(s_2) + 2)(\pi(t_2) - \pi(s_2) - 1)}{2} \geq n$$

(c) *The root symbol of t is some $h \succ f$*

(d) *$g(a, a) \prec s$, $s = f(s_1, s_2)$, $t = f(t_1, t_2)$, and either*

  (i) *$s_2 = t_2$ and $\pi(t_1) - \pi(s_1) \geq n + 1$, or*

  (ii) *$\pi(t_2) - \pi(s_2) > 1$, or*

  (iii) *$\pi(t_2) - \pi(s_2) = 1$ and $(\pi(s_2) - \pi(s_1)) + \pi(t_1) \geq n + 1$*

(e) *$g(a, a) \preceq s$, $s = h(s_1, s_2)$, $t = f(t_1, t_2)$, $h \succ f$, and $\pi(t_1) \geq n$. (h could be the same as g.)*

Note that cases (a) and (b) deal with the case when the gap needs to be filled with natural number terms.

Lemma 5.4. *Let $S = a \prec \ldots \prec g(a, a)$ be a simple system and θ be a solution of S. Let $G = (s \prec x_1 \prec \ldots \prec x_n \prec t)$ be a rigid gap in S. Then there is a k-expansion ($k \leq n$) S' of S between 0 and $\pi(s)$ such that*

(a) G is feasible in S', and

(b) θ can be extended to a solution of S'.

Proof: If s and t are ground natural-number terms then any solution θ of S satisfies $\nu(\theta(t)) - \nu(\theta(s)) - 1 = \nu(t) - \nu(s) - 1 \geq n$. Hence G is feasible in S and we can take S for S'. Otherwise, $s = f(s_1, s_2), t = f(t_1, t_2)$ and $\nu(\theta(t)) - \nu(\theta(s)) - 1$ is equal to

$$(\nu(\theta(s_2)) - \nu(\theta(s_1))) + \nu(\theta(t_1)) + \frac{(\nu(\theta(t_2)) + \nu(\theta(s_2)) + 2)(\nu(\theta(t_2)) - \nu(\theta(s_2)) - 1)}{2}$$

Since θ is a solution we have $\nu(\theta(t)) - \nu(\theta(s)) - 1 \geq n$. Assume that the following expression E_π

$$(\pi(s_2) - \pi(s_1)) + \pi(t_1) + \frac{(\pi(t_2) + \pi(s_2) + 2)(\pi(t_2) - \pi(s_2) - 1)}{2}$$

is less than n. We can show that by adding new variables to S we can raise the values of π so that E_π gets bigger than n. Assume for instance that $\pi(t_2) - \pi(s_2) > 1$. If $\pi(s_2) - \pi(s_1)$ (resp. $\pi(t_1)$, $\pi(t_2)$, $\pi(s_2)$, $\pi(t_2) - \pi(s_2)$) is less than $\nu(\theta(s_2)) - \nu(\theta(s_1))$ (resp. $\nu(\theta(t_1))$, $\nu(\theta(t_2))$, $\nu(\theta(s_2))$, $\nu(\theta(t_2)) - \nu(\theta(s_2))$) then by adding a new variable between s_2 and s_1 (resp. before t_1, before t_2, before s_2, between t_2 and s_2) we get a new system S' with function π' such that $E_{\pi'} > E_\pi$. More precisely, the extra variable should always be added between two non-variable terms s', t' delimiting a subinterval of m variables ($m \geq 0$) with $\nu(\theta(s')) - \nu(\theta(t')) - 1 > m$. Note that the only situation where we cannot extend the system by a new variable is when $\pi(s_2) - \pi(s_1) = \nu(\theta(s_2)) - \nu(\theta(s_1))$, $\pi(t_1) = \nu(\theta(t_1))$, $\pi(t_2) = \nu(\theta(t_2))$, $\pi(s_2) = \nu(\theta(s_2))$ and $\pi(t_2) - \pi(s_2) = \nu(\theta(t_2)) - \nu(\theta(s_2))$. However in this case $E_\pi \geq n$ and no extension is needed. This shows that by adding at most n variables to S we get a new system S' such that $E_{\pi'} \geq n$ and therefore the gap G is feasible in S'. When $\pi(t_2) - \pi(s_2) \leq 1$ the reasoning is similar. □

Lemma 5.5. *Let S be a simple system and and θ be a solution of S. Let $G = (s \prec x_1 \prec \ldots \prec x_n \prec t)$ be a rigid gap in S where $g(a,a) \preceq s$. Then there is a k-expansion ($k \leq n$) S' of S between 0 and $\pi(s)$ such that*

(a) G is feasible in S', and

(b) θ can be extended to a solution of S'.

Proof: If the root symbol of t is some $h \succ f$, then G is feasible in S and we may take S for S'. Otherwise let $t = f(t_1, t_2)$.

Let us assume that $s = f(s_1, s_2)$. If $\theta(s_2) = \theta(t_2)$ then ($s_2 = t_2$, and) $\circ^n(\theta(s_1)) \prec \theta(t_1)$. Hence we can insert at most n new variables between t_1 and s_1 such that G becomes feasible in the extended system S' and θ can be extended to a solution of S'. If $\circ(\theta(s_2)) = \theta(t_2)$ then for some ordinals k, k', $\circ^k(\theta(s_1)) = \theta(s_2)$ and $\circ^{k'}(a) = \theta(t_1)$. If one of the ordinals, say k, is not finite then we can add n new variables between s_1 and s_2 to get a new system S' such that $\pi(s_1) - \pi(s_2) \geq n$. If both k, k' are finite then we must have $k + k' \geq n$ for θ to be a solution. By adding variables between s_1 and s_2 and before t_1 we get again a suitable system S'. If $\circ(\theta(s_2)) \prec_{rpo} \theta(t_2)$ then it is always possible to introduce a new variable such that $\pi(t_2) - \pi(s_2) > 1$.

Suppose the root symbol of s is some $h \succ f$. The case $s \prec t_2$ is impossible: if t_2 is a non-variable term $s \prec \ldots \prec t$ is not a gap; if t_2 is a variable $s \prec \ldots \prec t$ is not rigid. So the only possible case is when $s = t_2$. Since θ is a solution, if $n > 0$ then $\circ^n(\theta(s)) = f(\circ^{n-1}(a), s) \prec_{rpo} \theta(f(t_1, s))$. Hence $\circ^{n-1}(a) \prec_{rpo} \theta(t_1)$ and there is a way to insert n new variables before t_1 such that G gets feasible in the extended system S' and θ can be extended to a solution of S'. □

Lemma 5.6. *Let S be a simple system and $G = (s \prec x_1 \prec \ldots \prec x_n \prec t)$ be a feasible rigid gap in S. Let S' be such that $S = S' \cup G \cup S''$, where every term in S' is below s and every term in S'' is above t in the ordering. Then S' is solvable if and only if $S' \cup G$ is solvable.*

Proof: Let θ be a solution of S'. Note that $\theta(s)$ and $\theta(t)$ are both ground terms, since G is a rigid gap. Suppose $S' \cup G$ is not solvable. Then it must be that there is a $k \leq n$ such that $\circ^k(\theta(s)) = \theta(t)$. Hence the root symbol of t must be f; let $t = f(t_1, t_2)$ for some $t_1 \preceq t_2$. We now consider different cases depending on the root symbol of s and also where the gap is situated—in the natural number part or to the right of $g(a, a)$.

$\mathbf{root(s) = h \neq f}$: Thus $\circ^k(\theta(s)) = f(\bar{q}, \theta(s))$ where $\nu(\bar{q}) = k-1$. In other words, $\theta(t_1) = \bar{q}$ and hence $\nu(\theta(t_1)) = k - 1$. But since the gap is feasible, by case (e) in the definition of feasibility it must be that $\pi(t_1) \geq n$. Since $t_1 \prec f(t_1, t_2)$, t_1 belongs to S'. Hence, due to Lemma 5.1, it is impossible that θ is a solution to S', since $k \leq n$.

$\mathbf{s = f(s_1, s_2) \prec t \prec g(a, a)}$: Since $\theta(s)$ and $\theta(t)$ are natural numbers, $\nu(\theta(t)) - \nu(\theta(s)) - 1$ is equal to:

$$(\nu(\theta(s_2)) - \nu(\theta(s_1))) + \nu(\theta(t_1)) + \frac{(\nu(\theta(t_2)) + \nu(\theta(s_2)) + 2)(\nu(\theta(t_2)) - \nu(\theta(s_2)) - 1)}{2}$$

Since gap G is feasible, by Lemma 5.1, it is not hard to show that this value must be at least n.

$\mathbf{g(a, a) \prec s = f(s_1, s_2) \prec t}$: Clearly $g(a, a) \preceq s_2 \preceq t_2$. If $s_1 \equiv s_2$ then $\circ(\theta(s)) = f(a, \circ(\theta(s_2)))$. Otherwise, i.e., if $s_1 \prec s_2$, then $\circ(\theta(s)) = f(\circ(\theta(s_1)), \theta(s_2))$. We now split into cases based on the relationship between s_2 and t_2.

$s_2 \equiv t_2$: Then it must be that $\circ^k(\theta(s_1)) = \theta(t_1)$. Hence the feasibility condition (d)(i), which applies to this gap, could not have been fulfilled.

$\theta(t_2) = \circ(\theta(s_2))$: Here condition (d)(iii) is what ensures feasibility. By Lemma 5.1, the number of terms possible between $\theta(s_2)$ and $\theta(s_1)$ must be at least $\pi(s_2) - \pi(s_1)$. Similarly, the number of terms possible between $\theta(t_1)$ and a must be at least $\pi(t_1)$. Hence it is not hard to see that $\theta(t)$ cannot be equal to $\circ^k(\theta(s))$ for any $k \leq n$.

$\circ(\theta(s_2)) \prec_{rpo} \theta(t_2)$: Note that $\theta(s) \prec_{rpo} f(a, \circ(\theta(s_2)))$. Since $g(a, a) \prec_{rpo} \theta(s_2)$, it is impossible that $\circ^m(f(a, \circ(\theta(s_2)))) \equiv \theta(t)$ for any m. □

Lemma 5.7. *Let S be a simple system and $G = (s \prec x_1 \prec \ldots \prec x_n \prec t)$ be an elastic gap in S. Let S' be such that $S = S' \cup G \cup S''$, where every term in S' is below s and every term in S'' is above t in the ordering. Then S' is solvable if and only if $S' \cup G$ is solvable.*

Proof: Let θ be a solution of S' and let i be the largest index such that x_i occurs in t. We shall extend θ to a solution of $S' \cup G$. Therefore we define: $\theta(x_j) = \circ^j(\theta(s))$ for $1 \leq j \leq i-1$. In the following we show that a ground term v can be selected such that:

$$\theta(s) \prec_{rpo} v \preceq_{rpo} \circ^{n-i}(v) \prec_{rpo} \theta_v(t)$$

where $\theta_v = \theta \cup \{x_i/v\}$. This will be sufficient for proving the lemma since the variables between x_i and t do not occur in t. We shall take $v = f(a, \circ^{n-i}(\theta(x)))$ where $x = x_{i-1}$ if $i - 1 > 0$ and s otherwise.

case $t \prec g(a,a)$: In this case we have $t \equiv f(t_1, t_2)$ with $t_1 \preceq t_2$ for some terms t_1, t_2 in S. Observe that $x_i \equiv t_1$ or $x_i \equiv t_2$ otherwise x_i would belong to a gap in S'.

subcase $x_i \equiv t_1 \equiv t_2$: Note that for any $u \prec_{rpo} g(a,a)$

$$\nu(f(u,u)) - \nu(u) = \nu(u) + 1$$

Now since $\nu(v) > n - i$, we have $\nu(f(v,v)) - \nu(v) > n - i$. Hence $\circ^{n-i}(v) \prec_{rpo} \theta_v(t)$.

subcase $x_i \equiv t_2 \not\equiv t_1$: If t_1 is not a variable then $t_1 \preceq s$ in S'. If t_1 is a variable then this variable occurs before x_i in S', by the choice of x_i. Hence, $\theta(t_1)$ is a ground term. Since

$$\circ^{n-i}(v) = f(\circ^{n-i}(a), \circ^{n-i}(\theta(x))) \prec_{rpo} f(\theta(t_1), f(a, \circ^{n-i}(\theta(x))))$$

we conclude again $\circ^{n-i}(v) \prec_{rpo} \theta_v(t)$.

subcase $x_i \equiv t_1 \not\equiv t_2$: Then $x \prec t_2$ in S'. Since $t_2 \prec t$, t_2 has to be a variable occurring between x_i and t. But this is impossible since x_i is the maximal variable to occur in t.

case $g(a,a) \preceq s \prec \ldots \prec t$: Let $t \equiv h(t_1, t_2)$. As above we have that $x_i \equiv t_2$. Note also that $\theta_v(t)$ is ground by the choice of x_i. We discuss according to the root symbol h of t:

subcase $h \neq f$: Since $g(a,a) \preceq_{rpo} \theta(x)$,

$$\circ^{n-i}(v) = f(\circ^{n-i}(a), \circ^{n-i}(\theta(x))) \prec_{rpo} \theta(h(t_1, x_i))$$

subcase $h = f$:

$$\circ^{n-i}(v) = f(\circ^{n-i}(a), \circ^{n-i}(\theta(x))) \prec_{rpo} \theta(f(t_1, x_i))$$

since $\circ^{n-i}(a) \prec_{rpo} \circ^{n-i}(\theta(x)) \prec_{rpo} \theta(x_i)$. □

A simple system S is *gap-consistent* if and only if every rigid gap in it is feasible.

Lemma 5.8. *Let S be a simple system and $G = (s \prec x_1 \prec \ldots \prec x_n \prec t)$ be the rightmost gap in S. Let S' be such that $S = S' \cup G \cup S''$, where every term in S' is below s and every term in S'' is above t in the ordering. If G is elastic, then S is solvable if and only if S' is solvable.*

Lemma 5.9. *Every gap-consistent simple system is solvable.*

From the proofs of Lemmas 5.6 and 5.7, it is clear that we can derive a polynomial depth-bound for a solution of a gap-consistent system. Together with the main theorem below this gives us another derivation of the result of [10].

Theorem 5.1. *A simple system S is solvable iff it admits a gap-consistent extension S' of size $t*(v+1)$ where v is the number of variables in S and t is the number of non-variable terms in S.*

Proof: We prove the "only if" part by induction on t. The base case is when $t = 2$. Then the simple system S is of the form

$$a \prec x_1 \prec \ldots \prec x_k \prec g(a,a) \prec x_{k+1} \prec \ldots \prec x_v$$

which is clearly gap-consistent by condition (c) in the definition. Thus we can take $S' = S$. Otherwise $S = S_i \cup G \cup S_u$ where $G = (s \prec \ldots \prec t)$ is the rightmost gap of S. Let θ be a solution of S, k ($k \leq v$) be the number of variables in G, and t_i (resp. t_u) be the number of non-variable terms in S_i (resp. S_u). If G is elastic, then by Lemma 5.8 S is solvable iff S_i is solvable, and we apply the induction hypothesis to S_i. Now assume that G is rigid. By Lemmas 5.4 and 5.5 there is a k-expansion S_i' of S_i such that G is feasible in $S_i' \cup G \cup S_u$ and this system has a solution extending θ. Let us apply the induction hypothesis to S_i': There is an expansion S_i'' of S_i' of size bounded by $t_i * (v+1)$ which is gap-consistent and admits a solution extending θ. Now $S_i'' \cup G \cup S_u$ is also gap-consistent with a solution extending θ and has size bounded by: $t_i * (v+1) + k + t_u$ which is less than $t * (v+1)$. □

Example: Consider the simple system

$$a \prec f(a,a) \prec g(a,a) \prec u \prec v \prec f(u,u) \prec x_1 \prec x_2 \prec f(f(a,a),v)$$

The rightmost gap $(f(u,u) \prec x_1 \prec x_2 \prec f(f(a,a),v))$ is not feasible unless $\pi(v) - \pi(u) > 1$, since (d)(ii) is the only condition that is useful here. (Condition (d)(iii) will not work since $\pi(f(a,a)) = 1$.) If we introduce a new variable w between u and v, the system becomes gap-consistent. Note that the gap $(g(a,a) \prec u \prec w \prec v \prec f(u,u))$ is elastic.

6 Conclusion

In this paper, we have given a novel, non-deterministic decision procedure for ordering constraints expressed using the recursive path ordering. This procedure requires only polynomial-time for verification after the initial guessing phase. Hence, we established the membership of this problem in the complexity class NP. Since the problem is NP-hard, our result implies that it is NP-complete. We are also able to derive the depth-bound of [10] through a significantly different approach. Note that in the case of extended signatures the result can be obtained much more easily as in [14] since every rpo-closed eq-feasible linear system is satisfiable.

Extension to RPO with status is not very difficult because the successor function only depends on the status of f (see [8]). Hence the method outlined by Nieuwenhuis [14] can be adapted here. Details will be given in the full paper.

References

1. H. Comon. Solving inequations in term algebras. In *Proc. 5th IEEE Symp. Logic in Computer Science (LICS),* Philadelphia, June 1990, 62–69.
2. H. Comon and R. Treinen. The first-order theory of lexicographic path orderings is undecidable. *Theoretical Computer Science* 176(1-2): 67–87, 1997.
3. N. Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science* 17(3): 279–301, 1982.
4. N. Dershowitz. Termination of rewriting. *Journal of Symbolic Computation* 3(1):69–115, Feb. 1987.
5. N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 243–309. North-Holland, 1990.
6. H. Ganzinger, R. Nieuwenhuis, and P. Nivela. The Saturate System, 1995. Software and documentation available from: `www.mpi-sb.mpg.de`.
7. J. Hsiang and M. Rusinowitch. On word problems in equational theories. In T. Ottmann, editor, *14th International Colloquium on Automata, Languages and Programming (ICALP),* volume 267 of *Lecture Notes in Computer Science*, pages 54–71, Karlsruhe, Germany, July 1987. Springer-Verlag.
8. J.-P. Jouannaud and M. Okada. Satisfiability of systems of ordinal notations with the subterm ordering is decidable. In *18th International Colloquium on Automata, Languages and Programming (ICALP),* volume 510 of *Lecture Notes in Computer Science*, pages 455–468, Madrid, Spain, July 1991. Springer-Verlag.
9. D. Kapur, P. Narendran and G. Sivakumar. A path ordering for proving termination of term rewriting systems. In H. Ehrig (ed.), *10th CAAP,* volume 185 of *Lecture Notes in Computer Science*, pages 173–187, Berlin, March 1985.
10. D. Kapur and G. Sivakumar. Maximal Extensions of Simplification Orderings. In: Proceedingsof the 15^{th} conference on the *Foundations of Software Technology and Theoretical Computer Science (FSTTCS-15),* (P.S. Thiagarajan, ed.) Bangalore, India, December 1995, *Lecture Notes in Computer Science* 1026, pages 225–239, Springer-Verlag.
11. C. Kirchner, H. Kirchner, and M. Rusinowitch. Deduction with symbolic constraints. *Revue Française d'Intelligence Artificielle*, 4(3):9–52, 1990. Special issue on automatic deduction.
12. M.S. Krishnamoorthy and P. Narendran. Note on recursive path ordering. *Theoretical Computer Science* 40: 323–328, 1985.
13. P. Lescanne. On the recursive decomposition ordering with lexicographical status and other related orderings. *J. of Automated Reasoning* 6 (1) 39–49 (1990).
14. R. Nieuwenhuis. Simple LPO constraint solving methods. *Inf. Process. Lett.* 47(2):65–69, Aug. 1993.
15. R. Nieuwenhuis and A. Rubio. Theorem proving with ordering constrained clauses. In D. Kapur, editor, Proceedings of 11th Conf. on Automated Deduction, Saratoga Springs, NY, 1992, volume 607 of *Lecture Notes in Artificial Intelligence,* pages 477–491, June 1992, Springer-Verlag.

16. D. Plaisted. A recursively defined ordering for proving termination of term rewriting systems. Technical Report R-78-943, U. of Illinois, Dept of Computer Science, 1978.
17. J. Steinbach. Extensions and comparison of simplification orderings. Proceedings of 3rd RTA, Chapel Hill, NC, *Lecture Notes in Computer Science,* 355, 434–448, 1989, Springer-Verlag.

Appendix

We now consider the case when the arity of f, the lowest function symbol, is greater than 2. For ease of presentation we define gaps between ordered tuples of terms. An *ordered tuple* is a tuple $(s_1, \ldots, s_n)$ where $s_1 \preceq \ldots \preceq s_n$. Given natural number terms (or ordered tuples of natural number terms) s and t, let $[s, t]$ denote the gap between s and t including both. For instance, $[a, f(a, a, a)]$ $= 2$ if we assume f is ternary and a is the only constant below f in the ordering. The following lemma can be shown:

Lemma 6.1. *Let $p_1, \ldots, p_n, q_1, \ldots, q_n$ be ground natural number terms, where $p_i \preceq p_{i+1}$ and $q_i \preceq q_{i+1}$ for $1 \leq i < n$. Then*

$$[(p_1, \ldots, p_n), (q_1, \ldots, q_n)] = 1 + \sum_{i=1}^{n} \binom{\nu(q_i) + i - 1}{i} - \binom{\nu(p_i) + i - 1}{i}$$

Proof: By induction using the measure (length of the tuple, $\nu(q_n) - \nu(p_n)$). □

Corollary 6.1. *Let $p_1, \ldots, p_n$, $q_1, \ldots, q_n$ be ground natural number terms, where $n > 1$ and $p_i \preceq p_{i+1}$ and $q_i \preceq q_{i+1}$ for $1 \leq i < n$. If $\nu(q_n) - \nu(p_n) \geq 2$, then $[(p_1, \ldots, p_n), (q_1, \ldots, q_n)] \geq \nu(q_n) + 2$.*

Proof: The worst-case scenario is where $p_1 = \ldots = p_n$, $q_1 = \ldots = q_{n-1} = a$ and $\nu(q_n) = \nu(p_n){+}2$. Then $[(p_n, \ldots, p_n), (a, \ldots, a, q_n)] = 2 + \binom{\nu(p_n)+1+n}{n} - \binom{\nu(p_n)+n}{n}$ $= 2 + \binom{\nu(p_n)+n}{n-1} \geq 2 + \nu(p_n) + 2$. □

Corollary 6.2. *Let $p_1, \ldots, p_n$, $q_1, \ldots, q_n$ be ground natural number terms, where $n > 1$ and $p_i \preceq p_{i+1}$ and $q_i \preceq q_{i+1}$ for $1 \leq i < n$. If $\nu(q_n) - \nu(p_n) = 1$, then $[(p_1, \ldots, p_n), (q_1, \ldots, q_n)] \geq \nu(p_n) - \nu(p_1) + \nu(q_{n-1}) + 2$.*

Condition (b) can now be changed to

(b′) $s = f(s_1, \ldots, s_m)$, $t = f(t_1, \ldots, t_m)$, $t \prec g(a, a)$ and

$$\sum_{i=1}^{m} \binom{\pi(t_i) + i - 1}{i} - \binom{\pi(s_i) + i - 1}{i} > n$$

Note that this can be checked in polynomial time since m, $\pi(s_i)$, etc. are effectively given in unary.

Let us now consider condition (d). Let $p = f(p_1, \ldots, p_m)$, $q = f(q_1, \ldots, q_m)$ be ground terms and $g(a, a) \prec p$. Let $k \in \{1, \ldots, m\}$ be the largest number such that $p_k \prec q_k$. The gap between p and q is infinite if and only if $q_k \succ g(a, a)$ and one of the following holds:

(i) $k = 1$ and the gap between p_1 and q_1 is infinite
(ii) $k > 1$ and $q_k \succ \circ(p_k)$
(iii) $k > 1$ and $g(a,a) \preceq q_{k-1}$
(iv) $p_2 \prec p_k$
(v) $k > 1$ and the gap between p_1 and p_2 is infinite

It is easy to see that when none of the above 5 conditions hold, *either* $k = 1$ and the gap between p_1 and q_1 is finite, *or* it has to be that $q_k = \circ(p_k)$, $p_2 = \ldots = p_k$, the gap between p_1 and p_2 is finite and $q_1, \ldots, q_{k-1}$ are natural number terms. In the latter case the gap between p and q is

$$\text{(gap between } p_1 \text{ and } p_2) + [(a, a, \ldots, a), (q_1, \ldots, q_{k-1})]$$

Thus the new condition in place of (d) is

(d′) $s = f(s_1, \ldots, s_m)$, $t = f(t_1, \ldots, t_m)$, $t \prec g(a,a)$. Let $k \in \{1, \ldots, m\}$ be the largest number such that $s_k \prec t_k$. Now either

- **(i)** $t_k \prec g(a,a)$ and $\sum_{i=1}^{k} \binom{\pi(t_i)+i-1}{i} - \binom{\pi(s_i)+i-1}{i} > n$, or
- **(ii)** $k = 1$ and $\pi(t_k) - \pi(s_k) > n$, or
- **(iii)** $k > 1$, $g(a,a) \preceq t_k$ and $\pi(t_k) - \pi(s_k) > 1$, or
- **(iv)** $k > 1$ and $g(a,a) \preceq t_{k-1}$, or
- **(v)** $g(a,a) \preceq t_k$ and $s_2 \prec s_k$, or
- **(vi)** $k > 1$ and $\pi(s_2) - \pi(s_1) + \sum_{i=1}^{k-1} \binom{\pi(t_i)+i-1}{i} > n$

In condition (e), the root symbol of $s = h(s_1, \ldots, s_{arity(h)})$ is greater than f in the ordering. The only case where the gap between s and t is finite is when $t = f(t_1, \ldots, t_m)$, $t_m = s$ and $t_1, \ldots, t_{m-1}$ are natural number terms.

Hence condition (e) can be reformulated as

(e′) $g(a,a) \preceq s, s = h(s_1, \ldots, s_{arity(h)}), t = f(t_1, \ldots, t_m), h \succ f$ and either

- **(i)** $\sum_{i=1}^{m-1} \binom{\pi(t_i)+i-1}{i} \geq n$, or
- **(ii)** $g(a,a) \preceq t_i$ for some $1 \leq i \leq m-1$