The First-Order Theory of Lexicographic Path Orderings is Undecidable

Hubert Comon [1]

CNRS and LRI, Bat. 490, Université de Paris Sud, F-91405 ORSAY cedex, France, comon@lri.lri.fr

Ralf Treinen [1,2]

Laboratoire de Recherche en Informatique (LRI), Bat. 490, Université de Paris Sud, F-91405 ORSAY cedex, France; and German Research Center for Artificial Intelligence (DFKI), Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany, treinen@lri.fr

We show, under some assumption on the signature, that the $\exists^*\forall^*$ fragment of the theory of a lexicographic path ordering is undecidable, both in the partial and in the total precedence cases. Our result implies in particular that the simplification rule of ordered completion is undecidable.

1 Introduction

The *recursive path orderings* (short *rpo*) are orderings on terms introduced by N. Dershowitz. They are the most popular orderings used for proving the termination of term rewriting systems (see [4] for a survey). The reason for the usefulness of these orderings lies in their stability properties: if $s >_{\mathrm{rpo}} t$, then, for every context C, $C[s] >_{\mathrm{rpo}} C[t]$ (this is the *monotonicity* property) and, assuming that variable symbols are incomparable with any other term (except themselves), $s >_{\mathrm{rpo}} t$ implies that $s\sigma >_{\mathrm{rpo}} t\sigma$ for any substitution σ. These two stability properties are important because, when they hold, proving the termination of a rewrite system amounts to proving that every left hand side of a rule is strictly larger than the corresponding right hand side. A classical problem in term rewriting systems is however the impossibility of orienting an

[1] Supported by the ESPRIT working group CCL.
[2] Supported by The German Bundesminister für Bildung, Wissenschaft, Forschung und Technologie (ITW-9105).

equation such as $x+y=y+x$ without losing termination. Several approaches have been proposed since the early 80's to overcome this problem. One of the most interesting is to orient the equation, depending on which instance of it is applied. In other words, if $\gg$ is a total monotonic ordering on terms, then we may see $s=t$ as the two *constrained rules* $s \to t \mid s > t$ (read: s rewrites to t if $s > t$) and $t \to s \mid t > s$, which translate into classical rewriting as the set of all $s\sigma \to t\sigma$ such that $s\sigma \gg t\sigma$ and the set of all $t\sigma \to s\sigma$ such that $t\sigma \gg s\sigma$. This allows one to use ordered strategies, even in presence of equations which are not uniformly orientable. A similar approach was used for the *unfailing completion* [8] and was described in its full generality in [13] where also the completeness of a set of deduction rules is proved. This powerful (yet simple) approach however requires *constraint solving techniques* for ordering constraints that are built over the $>$ symbol, which is interpreted as a monotonic ordering on ground terms, typically a recursive path ordering.

Basically, there are two forms of recursive path orderings: rpo with *multiset status*, which was the original definition of rpo by Dershowitz, and rpo with *lexicographic status*, also called by its more popular name *lexicographic path ordering* (short: *lpo*), and there also mixed forms (see [4] for a survey). In this paper we are concerned with the lexicographic path ordering, in Section 6 we will discuss shortly why our result does not transfer to the case of rpo with multiset status.

The constraints which have to be solved depend on the deduction rules that are used on constrained equations. At least the existential fragment of the theory of the ordering must be decidable. Furthermore, the question of decidability of the $\exists^*\forall^*$ fragment is also of great importance to constrained deduction. Indeed, one problem with constrained equational reasoning is to define simplification rules (which are essential in rewriting techniques). Such a simplification rule could be defined as follows:

$$\frac{s \to t \mid c \qquad u \to v \mid c'}{u \to v \mid c' \qquad s[v]_p = t \mid c' \wedge s|_p = u}$$

$$\text{If } T(F) \models \forall Var(s) \exists Var(u).c \to (s|_p = u \wedge c')$$

Here, $s|_p$ is the subterm of s at position p, $s[v]_p$ denotes the term obtained from s by replacing $s|_p$ by v, and $T(F)$ is the first-order logic structure of ground terms. This rule is called "total simplification" in [10]; it can be read as: "the rule $s \to t \mid c$ is simplified by the rule $u \to v \mid c'$ at position p in s if, for all instances of $s \to t$ that satisfy the constraint c, there is an instance of $u \to v$ which satisfies c' and which reduces $s|_p$".

The case of a total lexicographic path ordering has been investigated by H. Comon and its existential fragment has been shown decidable [2]. This fragment is actually NP-complete, as shown by R. Nieuwenhuis [12]. The existential fragment of the theory of any total recursive path ordering is actually decidable [9]. On the other side, R. Treinen has shown that the full first-order theory (actually the $\exists^*\forall^*\exists^*\forall^*$ fragment) of the theory of a *partial* recursive path ordering is undecidable [15]. This leaves as open questions the existential fragment of a partial recursive path ordering on the one hand, and the first-order theory of a total recursive path ordering on the other hand. These problems were listed as `Problem 24` in the lists of open problems in rewriting theory in [6] and further in [7]. A partial answer to the first question has been given by A. Boudet and H. Comon: the positive existential fragment of the theory of tree embedding is decidable [1]. The second problem remained open up to now. We answer this question here, showing that the $\exists^*\forall^*$ fragment of a lexicographic path ordering is undecidable, both in the total and in the partial cases. This improves Treinen's result for the partial case by reducing the number of quantifier alternations of the undecidable fragment. Furthermore, as an application, we show that this implies the undecidability of the above simplification rule.

The undecidability proof follows the ideas developed by R. Treinen in [15]: we encode the Post Correspondence Problem (PCP) thanks to a direct simulation of sequences. The general idea is to express as a first-order formula that a term is a "Post sequence", i.e. that every subsequence is either empty or obtained by one step of the PCP problem. Note that the universal quantification over subsequences is essential here. In [15], this is achieved using the fact that there are two incomparable symbols in $\mathcal{F}$. In this case, sequences can be coded in such a way that the predicate "s is a subsequence of t" can (roughly) be expressed as "s is a maximal term smaller than t (w.r.t. $\leq_{\mathrm{lpo}}$) following a certain pattern." In the case of a total ordering, however, this technique can not be applied since every finite set has a greatest element. We need here another trick: sequences are encoded the other way around ("upside down" if we compare with [15]) which allows to express the "subsequence relation". This last part is the most difficult part of our proof.

The paper is organized as follows: in Section 2 we state precisely the problem and establish (or recall) some properties of the lexicographic path ordering. In Section 3 we explain the top level structure of the proof, reducing undecidability of our problem to the problem of expressing some properties in the theory of $\geq_{\mathrm{lpo}}$. In Section 4, which is the heart of the paper, we show how to construct the formulas satisfying the requirements given in Section 3. In Section 5 we show the undecidability of the simplification rule and conclude in Section 6. In particular, we summarize the hypotheses we used on the signature and discuss various possible extensions.

2 The Problem

2.1 The Main Theorem

In this section, we define precisely the setting and present the main theorem. We use mainly the notations of [5]. Terms are built from an alphabet F of function symbols each of which is associated with a fixed arity. Typical elements of F are $f, g, h, k, 0$. In addition, we use variable symbols out of a set X. The set of terms built over some subset $G \subseteq F$ is written $T(G)$, and we write $T(G, X)$ for the set of terms built over G and X.

Assuming an ordering $\geq_F$ on F (called *precedence on* F), the *lexicographic path ordering* $\geq_{\mathrm{lpo}}$ on $T(F)$ is defined as follows.

Definition 2.1 (lexicographic path ordering, [4]) *For all $f, g \in F$ and $s_1, \ldots, s_n, t_1, \ldots, t_m \in T(F)$ we define $f(s_1, \ldots, s_n) >_{\mathrm{lpo}} g(t_1, \ldots, t_m)$ iff one of the following holds:*

- *$s_i \geq_{\mathrm{lpo}} g(t_1, \ldots, t_m)$ for some i*
- *$f >_F g$ and $f(s_1, \ldots, s_n) >_{\mathrm{lpo}} t_i$ for all $i = 1, \ldots, m$*
- *$f = g$ and the two following properties are satisfied:*
  - *$f(s_1, \ldots, s_n) >_{\mathrm{lpo}} t_i$ for all $i = 1, \ldots, m$ and*
  - *there is an $i \in \{1, \ldots, n\}$ such that $s_1 = t_1 \wedge \ldots \wedge s_{i-1} = t_{i-1}$ and $s_i >_{\mathrm{lpo}} t_i$.*

In this definition (and in the rest of the paper) we use the standard notational derivations of $s \geq_{\mathrm{lpo}} t$: $s >_{\mathrm{lpo}} t$ is an abbreviation for $s \geq_{\mathrm{lpo}} t$ and $s \neq t$, $t \leq_{\mathrm{lpo}} s$ stands for $s \geq_{\mathrm{lpo}} t$, $t \not\leq_{\mathrm{lpo}} s$ means that $t \leq_{\mathrm{lpo}} s$ does not hold, etc.

The following properties of $\geq_{\mathrm{lpo}}$ can be found in the literature (see the survey of N. Dershowitz [4]).

Proposition 2.2 *The relation $\geq_{\mathrm{lpo}}$ defined on $T(F)$*

- *is an ordering, i.e. it is reflexive, antisymmetric and transitive.*
- *is monotonic, i.e. $f(s_1, \ldots, s_n) \geq_{\mathrm{lpo}} f(t_1, \ldots, t_n)$ whenever $s_i \geq_{\mathrm{lpo}} t_i$ for all $i = 1, \ldots, n$,*
- *has the subterm property, i.e. $s >_{\mathrm{lpo}} t$ whenever t is a proper subterm of s.*
- *is total whenever $\geq_F$ is total.*

If we know that $h(\bar{t}) >_{\mathrm{lpo}} k(\bar{s})$, then we can in general not tell from the head symbols which case of Definition 2.1 applies. For instance if $f >_F > b >_F a$, then we have to use the first case of Definition 2.1 to prove $f(a, f(b, b)) >_{\mathrm{lpo}} f(b, b)$, but we can not prove this if we first decompose by the third case. Hence, to decompose an inequality, we have in general to consider different possibilities. In case of unary head symbols, however, we can decompose deterministically:

Proposition 2.3 *If $h(t) >_{\text{lpo}} h(s)$, then $t >_{\text{lpo}} s$.*

Proof: Let $h(t) >_{\text{lpo}} h(s)$. If the last case of Definition 2.1 applies, then $t >_{\text{lpo}} s$ must hold. If the first case of Definition 2.1 applies, then $t \geq_{\text{lpo}} h(s) >_{\text{lpo}} s$, where the second inequality holds by the subterm property. □

We define the language $\mathcal{L}$ as the set of all first-order predicate logic formulae built on the two binary predicates $=$ and $\geq$. The $\exists^*\forall^*$-fragment of $\mathcal{L}$, written $\Sigma_2(\mathcal{L})$, is defined as the set of formulae of the special form

$$\exists x_1, \ldots, x_n \forall y_1, \ldots, y_n P$$

where P is a Boolean combination of atoms $s = t$ and $s \geq t$. For a given precedence $\geq_F$ on F, the formulae of $\mathcal{L}$ are interpreted in the domain of (ground) terms $T(F)$ where $=$ is the (syntactic) equality between terms and $\geq$ is the lexicographic path ordering generated $\geq_F$. We write such a model as $\mathcal{A}_{F,\geq_F}$ or shortly as $\mathcal{A}$, when F and $\geq_F$ are clear. Our concern is to show that, under certain conditions on F and $\geq_F$, it is undecidable whether $\mathcal{A}_{F,\geq_F} \models \phi$ holds for given $\phi \in \Sigma_2(\mathcal{L})$. Our assumption on the set F and the precedence is

$$\left.\begin{array}{l} F \text{ is a finite set of function symbols containing at least} \\ - \text{ a constant } 0 \text{ which is minimal among the constants,} \\ - \text{ a binary function } f \text{ which is minimal in } F - \{0\}, \\ - \text{ a unary function symbol } g \text{ which is minimal in } \{h \mid h >_F f\}. \end{array}\right\} \quad (1)$$

This assumption includes both partial and total orderings. We do not require that $f \geq_F 0$. Note that there might be non-constant functions symbols smaller than 0, and constants greater than f. These restrictions are further discussed in Section 6.

Theorem 2.4 (Main Theorem) *For any set F of function symbols and precedence $\geq_F$ satisfying (1), it is undecidable whether for given formula $\phi \in \Sigma_2(\mathcal{L})$ we have $\mathcal{A}_{F,\geq_F} \models \phi$.*

2.2 Consequences of the Assumption on the Precedence

Before we begin with the proof we list some consequences of our assumption on the precedence.

Proposition 2.5 *The term 0 is minimal, that is there is no term t with $0 >_{\text{lpo}} t$.*

Proof: Assume that $0 >_{\mathrm{lpo}} t$. The term t must contain a constant a, hence we get $0 >_{\mathrm{lpo}} t \geq_{\mathrm{lpo}} a$ by the subterm property (Proposition 2.2). This contradicts the minimality of 0 among the constants. $\square$

Lemma 2.6 *Let $t \in T(F)$ and $u \in T(\{0, f\})$. If $t <_{\mathrm{lpo}} u$, then $f(0,t) \leq_{\mathrm{lpo}} u$.*

Hence, $f(0,t)$ can be seen as a successor of t as far as comparison to terms consisting only of 0 and f is concerned. A complete characterization of the successor function in the context of *total* lpos has been given in [2].

Proof: Let $t <_{\mathrm{lpo}} u$. We proceed by induction on the size of u. By Proposition 2.5, u can not be 0. Hence $u = f(u_1, u_2)$ for some $u_1, u_2 \in T(\{f, 0\})$. First, observe that $0 <_{\mathrm{lpo}} u$, $0 \leq_{\mathrm{lpo}} u_1$ and $0 \leq_{\mathrm{lpo}} u_2$ since 0 is a subterm of u_1 and of u_2. Let $t = h(\overline{t})$. There are three cases:

$h = 0$. Since $0 \leq_{\mathrm{lpo}} u_1$ and $0 \leq_{\mathrm{lpo}} u_2$, we get $f(0,0) \leq_{\mathrm{lpo}} f(u_1, u_2)$ from the monotonicity property (Proposition 2.2).

$h = f$. Let $t = f(t_1, t_2)$. We have to show that

$$f(0, f(t_1, t_2)) \leq_{\mathrm{lpo}} f(u_1, u_2) \quad \text{where} \quad f(t_1, t_2) <_{\mathrm{lpo}} f(u_1, u_2)$$

If the first case of Definition 2.1 applies to $t <_{\mathrm{lpo}} u$, we have to consider two cases:

- If $t \leq_{\mathrm{lpo}} u_1$, then $f(0,t) \leq_{\mathrm{lpo}} f(0, u_1) \leq_{\mathrm{lpo}} f(u_1, u_2)$. The first inequality follows from the monotonicity property of $\leq_{\mathrm{lpo}}$. The second inequality holds since either $u_1 = 0$ and $0 \leq_{\mathrm{lpo}} u_2$, or $0 <_{\mathrm{lpo}} u_1$ and $u_1 <_{\mathrm{lpo}} f(u_1, u_2)$ by the monotonicity property of $\leq_{\mathrm{lpo}}$.
- If $t \leq_{\mathrm{lpo}} u_2$, then $f(0,t) \leq_{\mathrm{lpo}} f(0, u_2) \leq_{\mathrm{lpo}} f(u_1, u_2)$ by the monotonicity property of $\leq_{\mathrm{lpo}}$, and since $0 \leq_{\mathrm{lpo}} u_1$.

If the last case of Definition 2.1 applies to $t <_{\mathrm{lpo}} u$, there are again two cases:

- If $u_1 = 0$, then since $t_1 < 0$ is not possible by Proposition 2.5, we have $t_1 = 0$ and $t_2 <_{\mathrm{lpo}} u_2$. We apply the induction hypothesis to $t_2 <_{\mathrm{lpo}} u_2$ and obtain $f(0, t_2) \leq_{\mathrm{lpo}} u_2$. Hence, $f(0, f(0, t_2)) \leq_{\mathrm{lpo}} f(0, u_2)$ by monotonicity.
- If $u_1 \neq 0$, then in fact $0 <_{\mathrm{lpo}} u_1$. From the assumption that $f(t_1, t_2) <_{\mathrm{lpo}} f(u_1, u_2)$ it follows that $f(0, f(t_1, t_2)) <_{\mathrm{lpo}} f(u_1, u_2)$ by Definition 2.1.

$h \notin \{f, 0\}$. By (1), this means $h \not\leq_F f$. By the lpo definition, $t \leq_{\mathrm{lpo}} u_1$ or $t \leq_{\mathrm{lpo}} u_2$, but equality does not hold because the top symbols of the terms are different. Hence, by the induction hypothesis, we have $f(0,t) \leq_{\mathrm{lpo}} u_1$ or $f(0,t) \leq_{\mathrm{lpo}} u_2$. The claim follows from the subterm property of $\leq_{\mathrm{lpo}}$. $\square$

Lemma 2.7 *Let $t, u \in T(\{0, f, g\})$, then t and u are comparable w.r.t. $\geq_{\mathrm{lpo}}$.*

Proof: We use induction on the sum of the sizes of t and u. If $u = 0$ or $t = 0$ then t, u are comparable since, for all $s \in T(\{g, f, 0\})$, $s \geq_{\mathrm{lpo}} 0$, by the subterm property. There are now three cases (up to permutation):

- If $s = g(s_1)$ and $t = g(t_1)$, then, by the induction hypothesis, $s_1 \geq_{\mathrm{lpo}} t_1$ (resp. $t_1 \geq_{\mathrm{lpo}} s_1$). Hence, $s \geq_{\mathrm{lpo}} t$ (resp. $t \geq_{\mathrm{lpo}} s$) holds.
- If $s = g(s_1)$ and $t = f(t_1, t_2)$, then, by the induction hypothesis, s and t_1 (resp. t_2) are comparable. If $t_1 \geq_{\mathrm{lpo}} s$ (resp. $t_2 \geq_{\mathrm{lpo}} s$), then $t >_{\mathrm{lpo}} s$. Otherwise, $s >_{\mathrm{lpo}} t_1$ and $s >_{\mathrm{lpo}} t_2$ and, $s >_{\mathrm{lpo}} t$ follows from the fact that $g >_F f$.
- Assume now that $t = f(t_1, t_2)$ and $u = f(u_1, u_2)$. By induction hypothesis, t_1 and u_1 are comparable, hence we can assume without loss of generality that $t_1 \geq_{\mathrm{lpo}} u_1$.

  If $t_1 = u_1$, then by the induction hypothesis $t_2 \geq_{\mathrm{lpo}} u_2$ or $t_2 \leq_{\mathrm{lpo}} u_2$. Hence, $t \geq_{\mathrm{lpo}} u$ or $t \leq_{\mathrm{lpo}} u$ holds.

  If $t_1 >_{\mathrm{lpo}} u_1$, then by the induction hypothesis, $t >_{\mathrm{lpo}} u_2$ or $t \leq_{\mathrm{lpo}} u_2$. In the former case, $t >_{\mathrm{lpo}} u$ holds, and $t <_{\mathrm{lpo}} u$ in the latter case. □

3 Coding the Post Correspondence Problem

In this section we present the overall framework that we employ in the reduction of the Post Correspondence Problem to the theory of a lexicographic path ordering. We will explain the difference to the method developed in [15] at the end of this section.

3.1 The Post Correspondence Problem

Definition 3.1 (Post Correspondence Problem, [14]) *An instance P of the Post Correspondence Problem over the alphabet $\{a, b\}$ is a finite set of the form $\{(p_i, q_i) \mid 1 \leq i \leq m; p_i, q_i \in \{a, b\}^+\}$. A sequence $((l_i, r_i))_{i=1,\ldots,n}$ with $l_i, r_i \in \{a, b\}^*$ is a solution of P if $l_1 = r_1 = \epsilon$, $l_n = r_n \neq \epsilon$ and for every $i < n$ there is a $j_i \leq m$ such that $l_{i+1} = l_i p_{j_i}$ and $r_{i+1} = r_i q_{j_i}$.*

If $((l_i, r_i))_{i=1,\ldots,n}$ is a solution of P, we say that (l_{i+1}, r_{i+1}) is constructed from (l_i, r_i) in one P-step. Our definition of a solution is slightly different from most of the literature, where the index sequence $j_1, \ldots, j_{n-1}$ would be considered as solution. Solvability of an instance of the Post Correspondence Problem is one of the most famous undecidable problems [14].

3.2 Coding the Construction Steps

In this subsection, we define formulae $\underline{\mathtt{i}}(x)$, $\underline{\mathtt{f}}(x)$ and $x\,\underline{\mathtt{s}}\,x'$ such that

(i) $x\,\underline{\mathtt{s}}\,x'$ defines a well-founded relation on $\mathcal{A}$, that is there is no infinite sequence $t_1, t_2, \ldots$ of ground terms with $\mathcal{A} \models t_i\,\underline{\mathtt{s}}\,t_{i+1}$ for all i;
(ii) the relation defined by $\underline{\mathtt{s}}$ is contained in $<_{\text{lpo}}$, that is if $\mathcal{A} \models t_i\,\underline{\mathtt{s}}\,t_{i+1}$, then $t_i <_{\text{lpo}} t_{i+1}$.

In the next subsection, we show how to construct a formula $\underline{\mathtt{solvable}_{\underline{\mathtt{i}},\underline{\mathtt{s}},\underline{\mathtt{f}}}}$ such that $\mathcal{A} \models \underline{\mathtt{solvable}_{\underline{\mathtt{i}},\underline{\mathtt{s}},\underline{\mathtt{f}}}}$ holds if and only if there is a sequence $(t_1, \ldots, t_n) \in \mathcal{A}^*$ with $\mathcal{A} \models \underline{\mathtt{i}}(t_1)$, $\mathcal{A} \models \underline{\mathtt{f}}(t_n)$ and $\mathcal{A} \models t_i\,\underline{\mathtt{s}}\,t_{i+1}$ for every $i < n$.

Having such a $\underline{\mathtt{solvable}_{\underline{\mathtt{i}},\underline{\mathtt{s}},\underline{\mathtt{f}}}}$ at hand, we can encode the solvability of an instance $P = \{(p_i, q_i) \mid i = 1, \ldots, n\}$ of the Post Correspondence Problem over an alphabet $\{a, b\}$. The idea is to define a representation of pairs of strings, such that $\mathcal{A} \models \underline{\mathtt{i}}(t)$ if t represent (ϵ, ϵ), $\mathcal{A} \models \underline{\mathtt{f}}(t)$ if t represents some (w, w) with $w \neq \epsilon$, and $\mathcal{A} \models t\,\underline{\mathtt{s}}\,t'$ if t' represents a pair which is constructed in one P-step from the pair represented by t.

The two above conditions on the relation defined by $\underline{\mathtt{s}}$ will be used at two different stages of the proof. We will use the well-foundedness of $\underline{\mathtt{s}}$ in this section only. Here, the well-foundedness of the relation is essential for the finiteness of the sequence. The second condition, that the relation defined by $\underline{\mathtt{s}}$ be contained in $<_{\text{lpo}}$, will not be used for the overall framework but only in the next section to prove the properties of the auxiliary formulae. We will not use the fact that this second property implies that $\underline{\mathtt{s}}$ is also well-founded in the "reverse direction".

First we define an injective coding function cw: $\{a, b\}^* \to T(F)$ by

$$\begin{aligned} \mathrm{cw}(\epsilon) &= 0 \\ \mathrm{cw}(wa) &= f(0, \mathrm{cw}(w)) \\ \mathrm{cw}(wb) &= f(f(0,0), \mathrm{cw}(w)) \end{aligned}$$

For instance, $\mathrm{cw}(ba) = f(0, f(f(0,0), 0))$. In the following, we will often identify a string with its term representation and write w instead of $\mathrm{cw}(w)$. For every fixed word $v \in \{a, b\}^*$ we can now easily define a formula $x = x' \cdot \underline{v}$ with the property that for all $w \in \{a, b\}^*$ and $t \in \mathcal{A}$, we have

$$\mathcal{A} \models t = \mathrm{cw}(w) \cdot \underline{v} \quad \text{iff} \quad t = \mathrm{cw}(wv)$$

For instance, the formula $x = x' \cdot \underline{ba}$ is $x = f(0, f(f(0,0), x'))$.

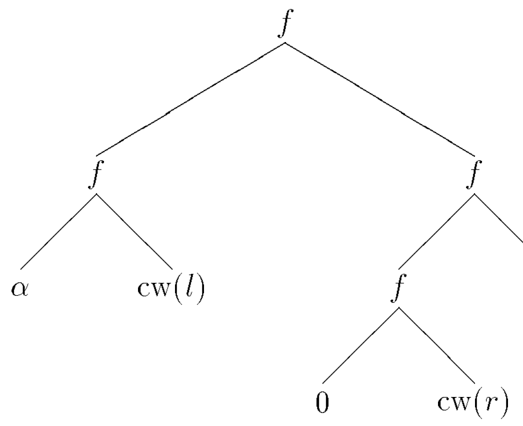

Fig. 1. A representation of (l, r).

A first attempt to code pairs of words could be to map (l, r) to the term $f(\mathrm{cw}(l), \mathrm{cw}(r))$. With this approach, the relation defined by $\underline{\mathtt{s}}$ would be contained in $<_{\mathrm{lpo}}$, but it would not be well-founded. For this reason, we add a "counter" to the representation which is decremented by $\underline{\mathtt{s}}$ (hence, we now have a representation relation rather than a function, since the counter can take any value). Note, however, that with the definition $f(\mathrm{cw}(l), f(\mathrm{cw}(r), c))$ the relation defined by $\underline{\mathtt{s}}$ is no longer contained in $<_{\mathrm{lpo}}$, as the reader easily verifies. Hence, we take another approach and code a pair (l, r) as the term

$$f(f(\alpha, \mathrm{cw}(l)), f(f(0, \mathrm{cw}(r)), c))$$

where $\alpha = f(f(0,0),0)$ and where c is the counter mentioned before (see Figure 1). We now define

$$\begin{aligned}
\underline{\mathtt{i}}(x) &:= \exists z. x = f(f(\alpha, 0), f(f(0,0), z)) \\
\underline{\mathtt{f}}(x) &:= \exists x_l. x = f(f(\alpha, x_l), f(f(0, x_l), 0)) \wedge x_l \neq 0 \\
x \,\underline{\mathtt{s}}\, x' &:= \exists x_l, x_r, z, x'_l, x'_r, z'.\ x = f(f(\alpha, x_l), f(f(0, x_r), z)) \\
&\qquad\qquad\qquad \wedge\ x' = f(f(\alpha, x'_l), f(f(0, x'_r), z')) \\
&\qquad\qquad\qquad \wedge\ z = f(0, z') \\
&\qquad\qquad\qquad \wedge\ f(f(0, x_r), z) < x' \\
&\qquad\qquad\qquad \wedge \bigvee_{(p,q)\in P} (x'_l = x_l \cdot \underline{p} \wedge x'_r = x_r \cdot \underline{q})
\end{aligned}$$

The first two lines in the definition of $x \,\underline{\mathtt{s}}\, x'$ match x and x' with the pattern of Figure 1. The third line decrements the counter, and the last line says that one P-construction step has been performed. The forth line is needed for the proof of Lemma 3.2.

Lemma 3.2 *If* $\mathcal{A} \models t \,\underline{\mathtt{s}}\, t'$, *then* $t <_{\mathrm{lpo}} t'$.

Proof: By the first two lines of the definition of $t\,\underline{\mathtt{s}}\,t'$, we know that

$$t = f(f(\alpha, t_l), f(f(0, t_r), u)) \quad \text{and} \quad t' = f(f(\alpha, t'_l), f(f(0, t'_r), u'))\,.$$

Furthermore, by the last line of the definition of $t\,\underline{\mathtt{s}}\,t'$, $t_l <_{\mathrm{lpo}} t'_l$ since t_l is a proper subterm of t'_l, hence $f(\alpha, t_l) <_{\mathrm{lpo}} f(\alpha, t'_l)$. The claim follows, since $f(f(0, t_r), u) <_{\mathrm{lpo}} t'$ by the forth line of the definition of $t\,\underline{\mathtt{s}}\,t'$. □

Lemma 3.3 *$\underline{\mathtt{s}}$ defines a well founded relation on $\mathcal{A}$, that is there is no infinite sequence $t_1, t_1, \ldots$ of ground terms with $\mathcal{A} \models t_i\,\underline{\mathtt{s}}\,t_{i+1}$ for every i.*

Proof: This follows immediately from the fact that by the third line of the definition of $t\,\underline{\mathtt{s}}\,t'$, the "counter-component" is *de*creasing with respect to the subterm relation. □

Lemma 3.4 *An instance P of the Post Correspondence Problem has a solution if and only if there is a sequence $(t_1, \ldots, t_n) \in \mathcal{A}^*$ with $\mathcal{A} \models \underline{\mathtt{i}}(t_1)$, $\mathcal{A} \models \underline{\mathtt{f}}(t_n)$ and $\mathcal{A} \models t_i\,\underline{\mathtt{s}}\,t_{i+1}$ for every $i < n$.*

Proof: Any such sequence $(t_1, \ldots, t_n)$ obviously exhibits a solution to P. On the other hand, let $(l_1, r_1), \ldots, (l_n, r_n)$ be a solution of P. We define the sequence $(t_1, \ldots, t_n)$ by

$$t_i = f(f(\alpha, \mathrm{cw}(l_i)), f(f(0, \mathrm{cw}(r_i)), f^{n-i}(0)))$$

where we take the inductive definition

$$\begin{aligned} f^0(0) &:= 0 \\ f^{n+1}(0) &:= f(0, f^n(0)) \end{aligned}$$

Now, every two consecutive elements of the sequence are in the relation $\underline{\mathtt{s}}$, as the reader easily verifies. For the verification of the third line of the definition of $t\,\underline{\mathtt{s}}\,t'$ note that, by the definition of the coding function cw, $f(0, \mathrm{cw}(w)) <_{\mathrm{lpo}} f(\alpha, \mathrm{cw}(v))$ for all $v, w \in \{a, b\}^*$. □

3.3 Coding Solvability

In this subsection we present the top level of the definition of $\underline{\mathtt{solvable}}_{\underline{\mathtt{i}},\underline{\mathtt{s}},\underline{\mathtt{f}}}$ which expresses the solvability of an instance of the Post Correspondence Problem. The construction of $\underline{\mathtt{solvable}}_{\underline{\mathtt{i}},\underline{\mathtt{s}},\underline{\mathtt{f}}}$ uses some subformulas which will be defined in the next section. The requirements on these subformulas used for the correctness proof of the coding are stated. The subformulas will be defined and the respective requirements will be proven in the next section.

The intended meaning of the subformulas is as follows. $\underline{\texttt{construction}}_{\underline{\texttt{s}},\underline{\texttt{f}}}\, y$ will express the fact that y can be interpreted as a sequence $(t_1, \ldots, t_n)$ with $\mathcal{A} \models \underline{\texttt{f}}(t_n)$ and $\mathcal{A} \models t_i\, \underline{\texttt{s}}\, t_{i+1}$ for every $i < n$. The formulae $\underline{\texttt{head}}$, $\underline{\texttt{s}}$ and $\underline{\texttt{i}}$ will be defined in Section 4. $x\, \underline{\texttt{head}}\, y$ is intended to express that x is the head of the list y, $(x, y')\, \underline{\texttt{finseg}}\, y$ is intended to express that the sequence with head x and tail y' is a final segment of y and $\underline{\texttt{nonempty}}\, y$ will express that the list y has a head.

Now we can define

$$\begin{aligned}
&\underline{\texttt{solvable}}_{\underline{\texttt{i}},\underline{\texttt{s}},\underline{\texttt{f}}} := \\
&\qquad \exists x, y.\ \underline{\texttt{i}}(x) \wedge \underline{\texttt{construction}}_{\underline{\texttt{s}},\underline{\texttt{f}}}\, y \wedge \exists y' (x, y')\, \underline{\texttt{finseg}}\, y \\
&\underline{\texttt{construction}}_{\underline{\texttt{s}},\underline{\texttt{f}}}\, y := \\
&\qquad \forall x, y'.\ (x, y')\, \underline{\texttt{finseg}}\, y \rightarrow \\
&\qquad \{\underline{\texttt{f}}(x) \vee (\underline{\texttt{nonempty}}\, y' \wedge \forall x'. x'\, \underline{\texttt{head}}\, y' \rightarrow x\, \underline{\texttt{s}}\, x')\}
\end{aligned}$$

We have to verify that $\mathcal{A} \models \underline{\texttt{solvable}}_{\underline{\texttt{i}},\underline{\texttt{s}},\underline{\texttt{f}}}$ if and only if P has a solution. The two following lemmata show what needs to be done in order to prove this equivalence. We define

$$\mathit{Seq} := \{(t_1, \ldots, t_n) \in T(\{0, f\})^* \mid \mathcal{A} \models \underline{\texttt{f}}(t_n), \mathcal{A} \models t_i\, \underline{\texttt{s}}\, t_{i+1} \text{ for all } i < n\}$$

Lemma 3.5 *Let $ct \colon \mathit{Seq} \rightarrow \mathcal{A}$ such that for all $t, u \in \mathcal{A}$ and $s \in \mathit{Seq}$ we have*

$$\mathcal{A} \models \underline{\texttt{nonempty}}\, ct(s) \textit{ iff } s \neq () \tag{2}$$

$$\mathcal{A} \models t\, \underline{\texttt{head}}\, ct(s) \textit{ iff } s = cons(t, s') \textit{ for some } s' \in \mathit{Seq} \tag{3}$$

$$\begin{aligned}\mathcal{A} \models (t, u)\, \underline{\texttt{finseg}}\, ct(s) &\textit{ iff } u = ct(s') \textit{ for some } s' \in \mathit{Seq} \\ &\textit{and } cons(t, s') \textit{ is a final segment of } s\end{aligned} \tag{4}$$

If P has a solution, then $\mathcal{A} \models \underline{\texttt{solvable}}_{\underline{\texttt{i}},\underline{\texttt{s}},\underline{\texttt{f}}}$.

Proof: This follows directly from Lemma 3.4. □

Lemma 3.6 *Suppose that the following statements hold:*

$$\mathcal{A} \models \forall y. \underline{\texttt{nonempty}}\, y \rightarrow \exists x. x\, \underline{\texttt{head}}\, y \tag{5}$$

$$\begin{aligned}\mathcal{A} \models \forall x, x', y, y'. &(x, y')\, \underline{\texttt{finseg}}\, y \wedge x'\, \underline{\texttt{head}}\, y' \wedge x\, \underline{\texttt{s}}\, x' \\ &\rightarrow \exists y''. (x', y'')\, \underline{\texttt{finseg}}\, y\end{aligned} \tag{6}$$

If $\mathcal{A} \models \underline{\texttt{solvable}}_{\underline{\texttt{i}},\underline{\texttt{s}},\underline{\texttt{f}}}$, then P has a solution.

Proof: Suppose that $\mathcal{A} \models \underline{\texttt{construction}}_{\underline{\texttt{s}},\underline{\texttt{f}}}\, u$. We will show that whenever $\mathcal{A} \models (t, u')\, \underline{\texttt{finseg}}\, u$, then there is a sequence $t_1, \ldots, t_n \in \mathcal{A}^*$ such that $t = t_1$,

$\mathcal{A} \models \underline{\mathtt{f}}(t_n)$ and $\mathcal{A} \models t_i \,\underline{\mathtt{s}}\, t_{i+1}$ for all $i < n$. We proceed by induction on the relation $\underline{\mathtt{s}}$ which is well founded by Lemma 3.3. If $\mathcal{A} \models \underline{\mathtt{f}}(t)$, then we can take the sequence to be (t), and we are done. Otherwise, $\mathcal{A} \models \underline{\mathtt{nonempty}}\, u'$ holds. By (5), there is an t' with $\mathcal{A} \models t' \,\underline{\mathtt{head}}\, u'$. From the definition of $\underline{\mathtt{construction}}_{\underline{\mathtt{s}},\underline{\mathtt{f}}}\, y$ we get that $\mathcal{A} \models t \,\underline{\mathtt{s}}\, t'$. Hence, by (6), there is a u'' such that $\mathcal{A} \models (t', u'') \,\underline{\mathtt{finseg}}\, u$. Now we can apply the induction hypothesis on t', which yields the claim. By Lemma 3.4, P has a solution. □

The number of quantifier alternations of the formula $\underline{\mathtt{solvable}}_{\underline{\mathtt{i}},\underline{\mathtt{s}},\underline{\mathtt{f}}}$ depends of course on the quantifier prefix in the subformulas. The reader easily checks that $\underline{\mathtt{solvable}}_{\underline{\mathtt{i}},\underline{\mathtt{s}},\underline{\mathtt{f}}}$ has the quantifier prefix $\exists^*\forall^*$ (that is the best we can get with this approach) if and only if

$\underline{\mathtt{i}}(x)$	has quantifier prefix $\exists^*\forall^*$,
$x \,\underline{\mathtt{s}}\, x'$	has quantifier prefix $\forall^*$,
$\underline{\mathtt{f}}(x)$	has quantifier prefix $\forall^*$,
$\underline{\mathtt{nonempty}}\, y$	has quantifier prefix $\forall^*$,
$x \,\underline{\mathtt{head}}\, y$	has quantifier prefix $\exists^*$,
$(x, y') \,\underline{\mathtt{finseg}}\, y$	has quantifier prefix $\exists^*$.

The formula $\underline{\mathtt{i}}(x)$ is already in the required form, but for $x \,\underline{\mathtt{s}}\, x'$ and $\underline{\mathtt{f}}(x)$ we have to find equivalent formulae in the $\forall^*$-fragment. For the case of $\underline{\mathtt{f}}(x)$, this can be achieved with the quantifier elimination method of [3]. An equivalent universal form of $\underline{\mathtt{f}}(x)$ is

$$
\begin{aligned}
&\bigwedge_{h \neq f} \forall \bar{u}, v_1, v_2, v_3 \Big(x \neq h(\bar{u}) \wedge x \neq f(h(\bar{u}), v_1) \wedge x \neq f(v_1, h(\bar{u})) \\
&\qquad\qquad \wedge x \neq f(f(v_1, v_2), f(h(\bar{u}), v_3)) \Big) \\
&\wedge \forall v_1, v_2, v_3, v_4, v_5 \Big(x = f(f(v_1, v_2), f(f(v_3, v_4), v_5)) \rightarrow \\
&\qquad\qquad v_1 = \alpha \wedge v_2 = v_4 \wedge v_3 = 0 \wedge v_5 = 0 \Big)
\end{aligned}
$$

By the first two lines, x is of the form $f(f(v_1, v_2), f(f(v_3, v_4), v_5))$. Since $v_1, \ldots, v_5$ are completely determined by the value for x, we can now use an universal quantifier to state further properties about these variables.

The method of [3] does not apply to formulae involving inequations. In case of $x \,\underline{\mathtt{s}}\, x'$, however, we can nevertheless find an equivalent universal formula. Intuitively speaking, this is possible since all the variables in the inequation $f(f(0, x_r), z) < x'$ are either free (the variable x') or are existentially quantified and "completely defined" by the equations (the variables x_r, z). The universal form of $x \,\underline{\mathtt{s}}\, x'$ is given in Appendix A.

The main difference to the method of [15] lies in the representation of pairs of strings in the first-order structure under consideration, and in the definition of $\underline{\mathtt{s}}$. As explained in the beginning of this section, an essential property of $\underline{\mathtt{s}}$ is well-foundedness. In [15], we could define $\underline{\mathtt{s}}$ in such a way that $v \,\underline{\mathtt{s}}\, w$ holds iff v is constructed from w in one P-construction step. In most of the applications shown in [15], this implies immediately the well-foundedness of $\underline{\mathtt{s}}$.

The situation is different in this paper. As we already mentioned, we will need the property that $\underline{\mathtt{s}}$ is contained in $<_{\mathrm{lpo}}$. With all natural representations of words, $v <_{\mathrm{lpo}} w$ does *not* hold if v is constructed from w in one P-step. On the other hand, it is not difficult to ensure that $w <_{\mathrm{lpo}} v$ holds in this case. Hence, we decided to use a "reversed" definition of $\underline{\mathtt{s}}$ with the property that $v \,\underline{\mathtt{s}}\, w$ holds iff w is constructed from v in one P-construction step.

As a consequence, we have to regain well-foundedness of $\underline{\mathtt{s}}$, since there might well be infinite sequences $v_0 <_{\mathrm{lpo}} v_1 <_{\mathrm{lpo}} v_2 <_{\mathrm{lpo}} \cdots$. Hence, we introduced an additional representation of pairs of strings (in [15], pairs where "hard-wired" in the formulae $\underline{\mathtt{s}}$, $\underline{\mathtt{finseg}}$, $\underline{\mathtt{i}}$ etc.), and equipped the representation of pairs with a "counter" which is decreased along $\underline{\mathtt{s}}$.

4 The Undecidability Proof

Following the method presented in Section 3, we will now define the predicates $\underline{\mathtt{nonempty}}\, y$, $x \,\underline{\mathtt{head}}\, y$, $(x, y') \,\underline{\mathtt{finseg}}\, y$ and the coding function ct and verify the conditions 2, 3, 4, 5, and 6. This completes the proof of Theorem 2.4.

4.1 Definition of the Coding Function

We code a sequence $(t_1, \ldots, t_n) \in Seq$ as

$$\mathrm{ct}(t_1, \ldots, t_n) = f(g(t_1), f(g(t_1), \ldots, f(g(t_n), 0) \ldots))$$

(see Figure 2). The term 0 encodes the empty sequence.

4.2 Accessing the Greatest Element of a List

Before we give the complete definition of the predicates, we first define some intermediate formulae and show some of their properties. The purpose is to have, in the presentation of a list $(t_1, \ldots, t_n)$ as defined in Subsection 4.1,

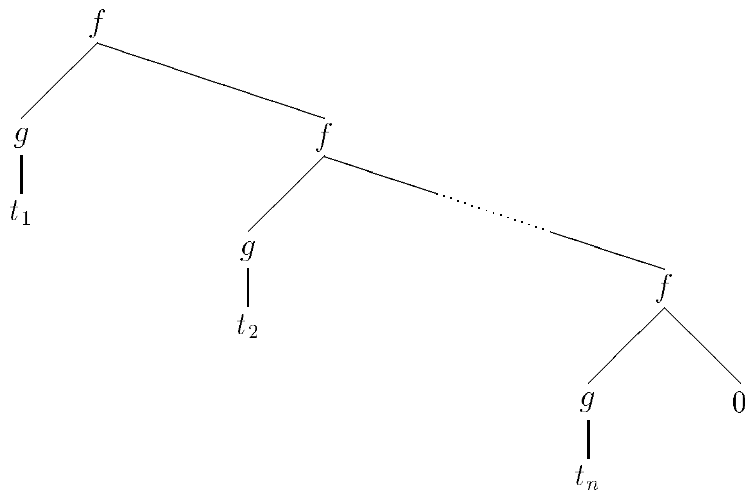

Fig. 2. The term $\mathrm{ct}((t_1, \ldots, t_n))$.

access to the last element t_n. Note that the last element might occur as an arbitrarily deep subterm in the coding. First, we define

$$\phi_1(x, y) := f(g(x), g(x)) \geq y > g(x)$$

The following lemma explains its meaning:

Lemma 4.1 *Let $\mathcal{A} \models \phi_1(t, u)$. Then*

(i) $g(t)$ is a subterm of u
(ii) for every subterm $g_0(\bar{v})$ of u with $g_0 \not<_F g$, we have $g(t) \geq_{\mathrm{lpo}} g_0(\bar{v})$.

Intuitively, $\phi_1(t, u)$ means that t is the greatest subterm of u which is headed by a symbol not smaller than g. Especially, $g(t)$ is the greatest g-headed subterm of u.

Proof: For the second claim let $g_0(\bar{v})$ be a subterm of u with $g_0 \not<_F g$. By the subterm property and since $f \neq g_0$ (since $g >_F f$), the first inequality of $\phi_1(t, u)$ yields $f(g(t), g(t)) >_{\mathrm{lpo}} g_0(\bar{v})$. Now, since $g_0 \not\leq_F f$, we have $g(t) \geq_{\mathrm{lpo}} g_0(\bar{v})$ by the definition of $\geq_{\mathrm{lpo}}$.

For proving that $g(t)$ is a subterm of u, we use an induction on the structure of $u = h(u_1, \ldots, u_n)$. There are three cases:

$h = 0$. This can not occur, since the second inequation of $\phi_1(t, u)$, $0 >_{\mathrm{lpo}} g(t)$, contradicts Proposition 2.5.

$h = f$. The second inequality of $\phi_1(t, u)$, $f(u_1, u_2) >_{\mathrm{lpo}} g(t)$, yields $u_1 \geq_{\mathrm{lpo}} g(t)$ or $u_2 \geq_{\mathrm{lpo}} g(t)$. If $u_1 = g(t)$ or $u_2 = g(t)$, then the claim is proven.

Otherwise, the first inequality of $\phi_1(t,u)$, $f(g(t),g(t)) \geq_{\mathrm{lpo}} f(u_1,u_2)$, yields $g(t) >_{\mathrm{lpo}} u_1$ and $f(g(t),g(t)) >_{\mathrm{lpo}} u_2$. Since this contradicts $u_1 >_{\mathrm{lpo}} g(t)$, $u_2 >_{\mathrm{lpo}} g(t)$ must hold. Hence, by the induction hypothesis, $g(t)$ is a subterm of u_2 and consequently of u.

$h \notin \{f, 0\}$. Hence $h \not\leq_F f$. The first inequation of $\phi_1(t,u)$, $f(g(t),g(t)) >_{\mathrm{lpo}} u$, yields $g(t) \geq_{\mathrm{lpo}} u$ which contradicts the second inequation of $\phi_1(t,u)$, $u >_{\mathrm{lpo}} g(t)$. Hence, this case cannot occur. □

Corollary 4.2 *For every term u, if $\mathcal{A} \models \exists x.\phi_1(x,u)$ then there is a unique term $gs(u)$ such that $\mathcal{A} \models \phi_1(gs(u),u)$.*

If we want to ensure the existence of an x such that $\mathcal{A} \models \phi_1(x,u)$ we have to assume more hypotheses on u. Let

$$\begin{aligned} \psi(y) = \quad & g(0) < y < g(g(0)) \\ & \wedge\ \forall x.y \neq g(x) \\ & \wedge\ \forall x.(y \not\leq f(g(x),g(x)) \wedge y > g(x)) \to y > g(f(0,x)) \\ & \wedge\ \forall x.\ y > g(x) \to \bigwedge_{h \notin \{f,0\}} x \not\geq h(0,\ldots,0) \end{aligned} \tag{7}$$

Lemma 4.3 *Let $u \in T(F)$. Then $\mathcal{A} \models \psi(u) \to \exists x.\phi_1(x,u)$.*

Proof: Let $\mathcal{A} \models \psi(u)$. From the inequality $u <_{\mathrm{lpo}} g(g(0))$, we infer that every symbol in u is 0 or is equal to or smaller than g. From this and the fact that $g(0) <_{\mathrm{lpo}} u$ we infer that u contains at least one occurrence of g.

Hence, there is a subterm $g(w)$ of u. From the last part of ψ and the subterm property of $\geq_{\mathrm{lpo}}$, for any subterm $g(w)$ of u, $w \in T(\{f,0\})$. Then, by Lemma 2.7, there is a term $w_0 = \max\{w \mid g(w) \text{ subterm of } u\}$.

We show that $\mathcal{A} \models \phi_1(w_0,u)$. We have of course $u \geq_{\mathrm{lpo}} g(w_0)$. Moreover, u is not equal to $g(w_0)$ by the second part of $\psi(u)$. Assume that $u \not\leq_{\mathrm{lpo}} f(g(w_0),g(w_0))$. By the third part of $\psi(u)$, this means that $u >_{\mathrm{lpo}} g(f(0,w_0))$. Hence, there is a subterm $g(v)$ of u which $v \geq_{\mathrm{lpo}} f(0,w_0)$. By the maximality of w_0, we get $w_0 \geq_{\mathrm{lpo}} v \geq_{\mathrm{lpo}} f(0,w_0)$. This is a contradiction to the subterm property, hence $u \leq_{\mathrm{lpo}} f(g(w_0),g(w_0))$ holds. □

Lemma 4.4 *For all sequences $s = (t_1,\ldots,t_n) \in Seq$ with $n \geq 1$, we have $\mathcal{A} \models \psi(ct(s))$.*

Proof: The formula $\psi(\mathrm{ct}(s))$ consists of four parts.

(i) $\mathcal{A} \models g(0) < \mathrm{ct}(s) < g(g(0))$. This follows immediately from the definition of $<_{\mathrm{lpo}}$.

(ii) $\mathcal{A} \models \forall x.\mathrm{ct}(s) \neq g(x)$ since $\mathrm{ct}(s) = f(g(t_1),u)$ for some u.

(iii) $\mathcal{A} \models \forall x.(\mathrm{ct}(s) \not\leq f(g(x), g(x)) \wedge \mathrm{ct}(s) > g(x)) \rightarrow \mathrm{ct}(s) > g(f(0, x))$. If $\mathrm{ct}(s) >_{\mathrm{lpo}} g(t)$, then, for some i, $t_i \geq_{\mathrm{lpo}} t$, hence $t \in T(\{f, 0\})$ and, by Lemma 2.7, $\mathrm{ct}(s) >_{\mathrm{lpo}} f(g(t), g(t))$. Then $t_i >_{\mathrm{lpo}} t$ holds for some i. By Lemma 2.6, this implies $t_i \geq_{\mathrm{lpo}} f(0, t)$. Hence, $\mathrm{ct}(s) >_{\mathrm{lpo}} g(t_i) \geq_{\mathrm{lpo}} g(f(0, t))$.

(iv) If $\mathrm{ct}(s) >_{\mathrm{lpo}} g(t)$, then $t_i \geq_{\mathrm{lpo}} t$ for some i. This implies, by minimality of f that $t \in T(\{f, 0\})$. This proves the last part of $\psi(\mathrm{ct}(s))$. □

Corollary 4.5 *For all sequences $s = (t_1, \ldots, t_n) \in Seq$ with $n \geq 1$, we have $\mathcal{A} \models \phi_1(t_n, ct(s))$.*

Proof: By Lemma 4.4, $\mathcal{A} \models \psi(\mathrm{ct}(s))$. By Lemma 4.3, there is a t with $\mathcal{A} \models \phi_1(t, \mathrm{ct}(s))$. By Lemma 4.1, t must be equal to t_n. □

4.3 Definition of the Predicates

We are now ready to give the missing definitions:

$$
\begin{aligned}
(x, y')\, \underline{\texttt{finseg}}\, y :=\; & \big(\phi_1(x, y) \wedge y' = 0\big) \vee \\
& \exists w. f(g(x), f(g(x), y')) > y \geq f(g(x), y') \wedge \\
& \qquad g(w) > g(x) \wedge \phi_1(w, y) \\
x\, \underline{\texttt{head}}\, y :=\; & \exists y'. y = f(g(x), y') \wedge (y' = 0 \vee \exists w.(x < w \wedge \phi_1(w, y))) \\
\underline{\texttt{nonempty}}\, y :=\; & \forall \overline{u}, u'. \bigwedge_{f' \neq f} y \neq f'(\overline{u}) \wedge \bigwedge_{g' \neq g} y \neq f(g'(\overline{u}), u') \\
& \wedge \psi(y) \\
& \wedge \forall x, y'.(y = f(g(x), y') \\
& \qquad \rightarrow (y' = 0 \vee \forall w.(\phi_1(w, y) \rightarrow x < w)))
\end{aligned}
$$

All parts of the predicate <u>`finseg`</u> will be used in in the proof of Property 4, and also later in the proof of Property 6.

With regard to Property 3, it would be sufficient to define x <u>`head`</u> y as $\exists y'. y = f(g(x), y')$. The second part of the predicate <u>`head`</u> is needed in the proof of Property 6.

Note that the first conjunct of the predicate <u>`nonempty`</u> is equivalent to the formula $\exists x, y'.(y = f(g(x), y'))$. Since <u>`nonempty`</u> is required to be $\forall^*$-formula (see the discussion at the end of Section 3), we use the universal form instead of the straightforward existential form. Again, this would be sufficient with regard to Property 2 alone, but we need the last two conjuncts for the proof of Property 5.

Lemma 4.6 *Property (4) holds.*

Proof: We have to prove for all $(t_1, \dots, t_n) \in \mathit{Seq}$ the equivalence

$$\begin{aligned} &\mathcal{A} \models (t, u')\, \underline{\texttt{finseg}}\, \mathrm{ct}(t_1, \dots, t_n) \\ \Longleftrightarrow\, &\text{exists } i \leq n \text{ with } t = t_i \text{ and } u' = \mathrm{ct}(t_{i+1}, \dots, t_n). \end{aligned}$$

where it is understood that $(t_{n+1}, \dots, t_n)$ is the empty sequence.

For the direction from left to right we have to consider two cases.

If $\mathcal{A} \models \phi_1(t, \mathrm{ct}(t_1, \dots, t_n)) \wedge u' = 0$, then $n \geq 1$ and $t = t_n$ by Corollaries 4.5 and 4.2.

Otherwise, there is an $r \in \mathcal{A}$ such that

$$\begin{aligned} \mathcal{A} \models\, & f(g(t), f(g(t), u')) > \mathrm{ct}(t_1, \dots, t_n) \geq f(g(t), u') \\ & \wedge g(r) > g(t) \wedge \phi_1(r, \mathrm{ct}(t_1, \dots, t_n)) \end{aligned}$$

By Corollaries 4.5 and 4.2, $r = t_n$ holds. Now, $g(r) >_{\mathrm{lpo}} g(t)$, hence $t_n >_{\mathrm{lpo}} t$ by Proposition 2.3. Since $t_n >_{\mathrm{lpo}} t$, there is a smallest index i such that $t_i \geq_{\mathrm{lpo}} t$. Hence, $t_{i'} \not\geq_{\mathrm{lpo}} t$ for all $i' < i$. Using the lpo rules and Proposition 2.3, $\mathrm{ct}(t_1, \dots, t_n) \geq_{\mathrm{lpo}} f(g(t), u')$ is simplified into $\mathrm{ct}(t_i, \dots, t_n) \geq_{\mathrm{lpo}} f(g(t), u')$, hence $\mathrm{ct}(t_i, \dots, t_n) >_{\mathrm{lpo}} u'$.

Since $t \not\geq_{\mathrm{lpo}} t_n$, there is a smallest index j such that $t \not\geq_{\mathrm{lpo}} t_j$. Furthermore, since $f(g(t), f(g(t), u')) >_{\mathrm{lpo}} \mathrm{ct}(t_1, \dots, t_n)$, it follows from the subterm property that $f(g(t), f(g(t), u')) >_{\mathrm{lpo}} \mathrm{ct}(t_j, \dots, t_n)$. Since by construction $t \not\geq_{\mathrm{lpo}} t_j$, this inequality is equivalent to $u' \geq_{\mathrm{lpo}} \mathrm{ct}(t_j, \dots, t_n)$. Together we have

$$\mathrm{ct}(t_i, \dots, t_n) >_{\mathrm{lpo}} u' \geq_{\mathrm{lpo}} \mathrm{ct}(t_j, \dots, t_n)$$

and hence $i < j$. By our construction of j this means $t \geq_{\mathrm{lpo}} t_i$. On the other hand we have $t_i \geq_{\mathrm{lpo}} t$, hence $t = t_i$. Using the definition of an lpo, we can now simplify

$$\begin{aligned} & f(g(t_i), f(g(t_i), u')) >_{\mathrm{lpo}} \mathrm{ct}(t_1, \dots, t_n) \\ & \Rightarrow^* f(g(t_i), f(g(t_i), u')) >_{\mathrm{lpo}} \mathrm{ct}(t_i, \dots, t_n) \\ & \Rightarrow f(g(t_i), u') >_{\mathrm{lpo}} \mathrm{ct}(t_{i+1}, \dots, t_n) \\ & \Rightarrow u' \geq_{\mathrm{lpo}} \mathrm{ct}(t_{i+1}, \dots, t_n) \end{aligned}$$

On the other hand, we have

$$\begin{aligned} \mathrm{ct}(t_1,\ldots,t_n) \geq_{\mathrm{lpo}} f(g(t_i),u') &\Rightarrow^* \mathrm{ct}(t_i,\ldots,t_n) \geq_{\mathrm{lpo}} f(g(t_i),u') \\ &\Rightarrow \mathrm{ct}(t_{i+1},\ldots,t_n) \geq_{\mathrm{lpo}} u' \end{aligned}$$

Hence, $u' = \mathrm{ct}(t_{i+1},\ldots,t_n)$.

For the direction from right to left we only have to check that

$$\mathcal{A} \models \phi_1(t_n, \mathrm{ct}(t_1,\ldots,t_n))$$

(this is Corollary 4.5), and that for $i < n$ we have

$$\begin{aligned} \mathcal{A} \models \exists w.\, & f(g(t_i), f(g(t_i), \mathrm{ct}(t_{i+1},\ldots,t_n))) > \mathrm{ct}(t_1,\ldots,t_n) \\ & \geq f(g(t_i), \mathrm{ct}(t_{i+1},\ldots,t_n)) \\ & \wedge g(w) > g(t_i) \\ & \wedge \phi_1(w, \mathrm{ct}(t_1,\ldots,t_n)) \end{aligned}$$

This is easily proven for the choice $w = t_n$. □

Lemma 4.7 *Property (2) holds.*

Proof: For the implication from left to right, assume $\mathcal{A} \models \underline{\texttt{nonempty}}\,\mathrm{ct}(s)$. We have to show that then $s \neq ()$. Note that the formula

$$\forall \overline{u}.\ \bigwedge_{f' \neq f} y \neq f'(\overline{u})$$

implies in particular that $y \neq 0$, hence the sequence is not empty.

For the implication from right to left, assume that $s \neq ()$. We have to show that $\mathcal{A} \models \underline{\texttt{nonempty}}\,\mathrm{ct}(s)$. We split this proof into three parts corresponding respectively to the three conjuncts in the formula $\underline{\texttt{nonempty}}\,y$.

- When s is not empty, $\mathrm{ct}(s) = f(g(t_1), u)$ for some u. Hence the first part of the formula is valid:

$$\mathcal{A} \models \forall \overline{u}, u'.\ \bigwedge_{f' \neq f} \mathrm{ct}(s) \neq f'(\overline{u}) \wedge \bigwedge_{g' \neq g} \mathrm{ct}(s) \neq f(g'(\overline{u}), u')$$

- $\mathcal{A} \models \psi(\mathrm{ct}(s))$ has been proven in Lemma 4.4.
- For the last part of the formula let $\mathrm{ct}(s) = f(g(t_1), u)$. If $u = 0$, then the formula holds. Otherwise, u must be of the form $f(g(t_2), v)$ with $t_2 >_{\mathrm{lpo}} t_1$.

For all w such that $\phi_1(w, \mathrm{ct}(s))$ holds, $g(w) \geq_{\mathrm{lpo}} g(t_2) >_{\mathrm{lpo}} g(t_1)$ thanks to Lemma 4.1. As a consequence, $w >_{\mathrm{lpo}} t_1$ holds by Proposition 2.3. □

Lemma 4.8 *Property (3) holds.*

Proof: For the implication from left to right, assume that $\mathcal{A} \models t\,\underline{\texttt{head}}\,\mathrm{ct}(s)$. We have to show that $s = cons(t, s')$ for some $s' \in Seq$. Indeed, by definition of $x\,\underline{\texttt{head}}\,y$, we must have $\mathcal{A} \models \exists y'.\mathrm{ct}(s) = f(g(t), y')$ which means that $s = (t, t_2, \ldots, t_n)$ and $s' = \mathrm{ct}(t_2, \ldots, t_n)$.

For the other direction, let $s = (t_1, \ldots, t_n) \in Seq$. We have to show that $\mathcal{A} \models t_1\,\underline{\texttt{head}}\,\mathrm{ct}(s)$. Indeed, $\mathrm{ct}(s) = f(g(t_1), u)$ for some u. If $u = 0$, then the claim is proven. Otherwise, $t_n >_{\mathrm{lpo}} t_1$ and $\mathcal{A} \models \phi_1(t_n, \mathrm{ct}(s))$ by Corollary 4.5. □

Note that actually some parts of the definitions of $x\,\underline{\texttt{head}}\,y$ and $\underline{\texttt{nonempty}}\,y$ have not been used so far. They will be exploited when proving Property 6.

4.5 Proof of the Conditions of Lemma 3.6

We are left to prove Properties 6 and 5, which is the subject of the next two lemmas.

Lemma 4.9 *Property (5) holds.*

Proof: We have already seen that the first part of the formula $\underline{\texttt{nonempty}}\,u$ implies that there are t, u such that $u = f(g(t), u')$ If $u' = 0$, then we are done. Otherwise, since $\mathcal{A} \models \psi(u)$ there is by Lemma 4.3 a t' with $\mathcal{A} \models \phi_1(t', u)$. From the last part of $\underline{\texttt{nonempty}}\,u$ it follows that $\mathcal{A} \models t < t'$. □

Lemma 4.10 *Property (6) holds.*

Proof: Assume that $(t, u')\,\underline{\texttt{finseg}}\,u$ and $t'\,\underline{\texttt{head}}\,u'$ and $t\,\underline{\texttt{s}}\,t'$ hold. We have to show that $(t', u'')\,\underline{\texttt{finseg}}\,u$ holds for some u''.

Since $\mathcal{A} \not\models t'\,\underline{\texttt{head}}\,0$, $u' \neq 0$ holds and $\mathcal{A} \models (t, u')\,\underline{\texttt{finseg}}\,u$ implies that

$$\mathcal{A} \models \exists w. f(g(t), f(g(t), u')) > u \geq f(g(t), u') \wedge g(w) > g(t) \wedge \phi_1(w, u) \quad (8)$$

holds. Moreover, by definition of $t'\,\underline{\texttt{head}}\,u'$ we have that for some u''

$$\mathcal{A} \models u' = f(g(t'), u'') \wedge (u'' = 0 \vee \exists w'.\phi_1(w', u') \wedge t' < w') \quad (9)$$

Note that, by (8), $\mathrm{gs}(u)$ exists. We shall show that

$$\begin{aligned}\mathcal{A} \models & (u'' = 0 \wedge t' = \mathrm{gs}(u)) \\ & \vee (f(g(t'), f(g(t'), u'')) > u \geq f(g(t'), u'') \wedge g(\mathrm{gs}(u)) > g(t') \\ & \quad \wedge \phi_1(\mathrm{gs}(u), u)\end{aligned}$$

There are two cases:

$t' = \mathbf{gs}(u)$. If $u'' = 0$, then the claim is proven.

Otherwise, assume that $u'' \neq 0$. Then by (9), $\mathrm{gs}(u')$ exists and $t' <_{\mathrm{lpo}} \mathrm{gs}(u')$. From (8) and Lemma 4.1, we know that $u \geq_{\mathrm{lpo}} f(g(t), u') >_{\mathrm{lpo}} u' \geq_{\mathrm{lpo}} g(\mathrm{gs}(u'))$. By the lpo rules, there must be a subterm $h(\bar{r})$ of u with $h \not<_F g$ and $h(\bar{r}) \geq_{\mathrm{lpo}} g(\mathrm{gs}(u'))$. By the second part of Lemma 4.1, this means $g(\mathrm{gs}(u)) \geq_{\mathrm{lpo}} h(\bar{r}) \geq_{\mathrm{lpo}} g(\mathrm{gs}(u'))$, hence $\mathrm{gs}(u) \geq_{\mathrm{lpo}} \mathrm{gs}(u')$ by Proposition 2.3. This contradicts $t' = \mathrm{gs}(u) <_{\mathrm{lpo}} \mathrm{gs}(u')$, hence the case $u'' \neq 0$ can not occur.

$t' \neq \mathbf{gs}(u)$. Note that $\phi_1(\mathrm{gs}(u), u)$ holds by (8). We have to prove three inequalities

(i) $\mathcal{A} \models f(g(t'), f(g(t'), u'')) > u$. From (8), (9) and from $t' >_{\mathrm{lpo}} t$ (since $\mathcal{A} \models t \,\underline{\mathtt{s}}\, t'$ and by Lemma 3.2), we get

$$f(g(t'), f(g(t'), u'')) = f(g(t'), u') >_{\mathrm{lpo}} f(g(t), f(g(t), u')) >_{\mathrm{lpo}} u\,.$$

(ii) $\mathcal{A} \models u \geq f(g(t'), u'')$. From (8) and (9) we get

$$u \geq_{\mathrm{lpo}} f(g(t), u') = f(g(t), f(g(t'), u'')) >_{\mathrm{lpo}} f(g(t'), u'')\,.$$

(iii) $\mathcal{A} \models g(\mathrm{gs}(u)) > g(t')$. By (8) and (9), $u >_{\mathrm{lpo}} g(t')$ holds. Hence, there is a subterm $g_0(\bar{v})$ of u with $g_0 \not<_F g$ and $g_0(\bar{v}) \geq g(t')$. This implies, by Lemma 4.1, $g(\mathrm{gs}(u)) \geq_{\mathrm{lpo}} g(t')$. Since we assumed $t' \neq \mathrm{gs}(u)$ in the case distinction, $g(\mathrm{gs}(u)) >_{\mathrm{lpo}} g(t')$ follows.

□

Theorem 2.4, reconsidered: *Let F contain (at least) one binary symbol f, one unary symbol g and one constant 0. The $\exists^*\forall^*$ fragment of the theory of a lexicographic path ordering extending a precedence in which 0 is a minimal constant, f is minimal in $F - \{0\}$ and g is a minimal symbol greater than f is undecidable.*

Proof: For every instance P of the Post Correspondence Problem, we can construct the sentence $\underline{\mathtt{solvable}}_{\underline{\mathtt{i}},\underline{\mathtt{s}},\underline{\mathtt{f}}}$, which belongs to the $\exists^*\forall^*$-fragment. Lemma 3.5, Lemma 4.6, Lemma 4.7 and Lemma 4.8 show that $T(F) \models \underline{\mathtt{solvable}}_{\underline{\mathtt{i}},\underline{\mathtt{s}},\underline{\mathtt{f}}}$ if P is solvable. If P is solvable, then by Lemma 3.6, Lemma 4.9 and Lemma 4.10, $T(F) \models \underline{\mathtt{solvable}}_{\underline{\mathtt{i}},\underline{\mathtt{s}},\underline{\mathtt{f}}}$ holds. Since solvability of an instance of the Post Correspondence Problem is undecidable, so is validity of $\exists^*\forall^*$-sentences in $T(F)$. □

5 Undecidability of the simplification rule

Recall the simplification rule given in the introduction, which corresponds to the "total simplification rule" of [10].

$$\frac{s \to t \mid c \qquad u \to v \mid c'}{u \to v \mid c' \qquad s[v]_p = t \mid c' \wedge s|_p = u}$$

$$\text{If } T(F) \models \forall Var(s) \exists Var(u).c \to (s|_p = u \wedge c')$$

When writing a constrained rule like $s \to t \mid c$, it is understood that $Var(c) \subseteq Var(s) \cup Var(t)$. We consider the constraint system consisting of constraints of the form $\exists y_1, \ldots, y_n.b$ where b is Boolean combination of equalities and inequalities.

Theorem 5.1 *For any set F of function symbols and precedence $\geq_F$ satisfying (1), the set of instances of the simplification rule is undecidable. This also holds when c is instantiated to be $\top$.*

Proof: We reduce the validity problem in $\mathcal{A}$ of $\forall^*\exists^*$-sentences to the decision problem of the set of instances of the simplification rule. Note that the set of $\forall^*\exists^*$-sentences which are valid in $\mathcal{A}$ is (up to equivalence transformations) the complement of the set of $\exists^*\forall^*$-sentences valid in $\mathcal{A}$, and hence is undecidable by Theorem 2.4. Let $\forall x_0, \ldots, x_n \exists y_0, \ldots, y_m.\phi$ be given. This sentence is obviously equivalent to

$$\begin{aligned} \forall x_0, \ldots, x_n \exists z_0, \ldots, z_n, y_0, \ldots, y_m.\, z_0 = x_0 \wedge \ldots \wedge z_n = x_n \\ \wedge\, \phi[z_0/x_0, \ldots, z_n/x_n] \end{aligned} \tag{10}$$

where $z_0, \ldots, z_n$ are fresh distinct variables. We use the abbreviations

$$\begin{aligned} F(\bar{x}) &= f(x_0, f(\ldots f(x_n, 0)\ldots)) \\ F(\bar{z}) &= f(z_0, f(\ldots f(z_n, 0)\ldots)) \\ \phi' &= \phi[z_0/x_0, \ldots, z_n/x_n] \end{aligned}$$

Now, (10) is equivalent to

$$\forall x_0, \ldots, x_n \exists z_0, \ldots, z_n, y_0, \ldots, y_m.\, \phi' \wedge F(\bar{z}) = F(\bar{x})$$

This sentence is valid in $\mathcal{A}$ if and only if

$$\frac{F(\bar{x}) \to 0 \mid \top \qquad F(\bar{z}) \to 0 \mid \phi'}{F(\bar{z}) \to 0 \mid \phi' \qquad 0 = 0 \mid \phi' \wedge F(\bar{x}) = F(\bar{z})}$$

is an instance of the simplification rule. □

6 Concluding Remarks

We proved the undecidability of the $\exists^*\forall^*$ fragment of lexicographic path orderings over finite signatures. This proof assumes some weak hypotheses on the precedence. Choosing 0 as a minimal constant is not a restriction. The main restrictions are

(i) among the minimal symbols of $F \setminus \{0\}$ w.r.t. $\geq_F$, there should be a (at least) binary one (which we called f);
(ii) among the minimal symbols larger than f there should be a non-constant one (which we called g).

Indeed, if there is a minimal symbol h in $\mathcal{F} \setminus \{0\}$ whose arity is, say, 3, we can for example code g and f as:

$$f(x, y) \stackrel{\text{def}}{=} h(0, x, y); \qquad\qquad g(x) \stackrel{\text{def}}{=} h(h(0, 0, 0), 0, x).$$

Note that, in such a case, Assumption (ii) above is no longer used: the proof applies to one constant and one ternary function symbol. Similarly, g needs not to be unary: "at least unary" is sufficient.

We conjecture that Assumption (ii) above can be removed, at the price of some additional coding, which we avoid here for sake of simplicity. The idea of the coding would be to map $T(\{0, f, g\})$ into $T(\{0, h\})$ where h is binary, while preserving the ordering relation. For example, we could define $f(x, y) \stackrel{\text{def}}{=} h(0, h(x, y))$ and $g(x) \stackrel{\text{def}}{=} h(h(0, 0), x)$. Actually, this particular mapping does not work. Some additional work has to be done in order to cope with several "overlappings" of $g(x)$ into $f(x, y)$ or of $f(x, y)$ into itself. This did not occur in the ternary symbol case above because of the "flatness" of the coding. We believe that the coding is still possible, though tedious.

However, Assumption (i) cannot be removed easily. Actually, the decidability of the first-order theory of a total lexicographic path ordering on a signature containing only unary symbols and constants remains open. Our method cannot be applied in this case, because we have no means by which we could encode sequences.

Similarly, our method cannot be applied directly to recursive path orderings with multiset status. Indeed, Lemma 4.6 does not hold: we took advantage of the fact that

$$x > x' \models f(x,y) > f(x',y') \leftrightarrow f(x,y) > y'$$

which does not hold for multiset status. Moreover, this property is important since this is the way we "go down" in the terms, retrieving subterms.

On the positive side, our method might be applied for proving undecidability of confluence of ordered rewrite systems (see [11]) which use a lexicographic path ordering. Indeed, strong ground confluence of such systems is expressed using a $\forall^*\exists^*$ sentence over $\geq_{\mathsf{lpo}}$. But there are still difficulties because in the problem, as it is stated in [11], the constraints only consist in single inequalities $l > r$ for each rule $l \to r$. It is possible to encode any quantifier-free formula over $\geq_{\mathsf{lpo}}$ into a single inequation, using additional function symbols. However, we would need existential quantifications in the constraints. This can only be achieved through rules which introduce new variables. But then, we get only inequalities in which existentially quantified variables are all on the same side of the inequality, which is not sufficient for our purpose.

References

[1] A. Boudet and H. Comon. About the theory of tree embedding. In M. C. Gaudel and J.-P. Jouannaud, editors, *4th International Joint Conference on Theory and Practice of Software Development*, Lecture Notes in Computer Science, vol. 668, pages 376–390, Orsay, France, Apr. 1993. Springer-Verlag.

[2] H. Comon. Solving symbolic ordering constraints. *International Journal of Foundations of Computer Science*, 1(4):387–411, 1990.

[3] H. Comon and P. Lescanne. Equational problems and disunification. *Journal of Symbolic Computation*, 7:371–425, 1989.

[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] N. Dershowitz, J.-P. Jouannaud, and J. Klop. Open problems in term rewriting. In R. V. Book, editor, *4th International Conference on Rewriting Techniques

and Applications, Lecture Notes in Computer Science, vol. 488, pages 445–456, Como, Italy, Apr. 1991. Springer-Verlag.

[7] N. Dershowitz, J.-P. Jouannaud, and J. Klop. More problems in rewriting. In C. Kirchner, editor, *5th International Conference on Rewriting Techniques and Applications*, Lecture Notes in Computer Science, vol. 690, pages 468–487, Montreal, Canada, June 1993. Springer-Verlag.

[8] J. Hsiang and M. Rusinowitch. On word problems in equational theories. In T. Ottmann, editor, *14th International Colloquium on Automata, Languages and Programming*, Lecture Notes in Computer Science, vol. 267, pages 54–71, Karlsruhe, Germany, July 1987. Springer-Verlag.

[9] J.-P. Jouannaud and M. Okada. Satisfiability of systems of ordinal notations with the subterm property is decidable. In J. L. Albert, B. Monien, and M. R. Artalejo, editors, *18th International Colloquium on Automata, Languages and Programming*, Lecture Notes in Computer Science, vol. 510, pages 455–468, Madrid, Spain, 1991. Springer-Verlag.

[10] 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.

[11] R. Nieuwenhuis. A new ordering constraint solving method and its applications. Research Report MPI-I-92-238, Max-Planck- Institut für Informatik, Saarbrücken, Feb. 1993.

[12] R. Nieuwenhuis. Simple LPO constraint solving methods. *Inf. Process. Lett.*, 47(2):65–69, Aug. 1993.

[13] R. Nieuwenhuis and A. Rubio. Theorem proving with ordering constrained clauses. In D. Kapur, editor, *11th International Conference on Automated Deduction*, Lecture Notes in Computer Science vol. 607, pages 477–491, Saratoga Springs, NY, June 1992. Springer-Verlag.

[14] E. L. Post. A variant of a recursively unsolvable problem. *Bulletin of the AMS*, 52:264–268, 1946.

[15] R. Treinen. A new method for undecidability proofs of first order theories. *Journal of Symbolic Computation*, 14(5):437–458, Nov. 1992.

A The Universal Form of $x \underline{\leq} x'$

In Section 3, $x \underline{\leq} x'$ has been defined in form of a $\exists^*$-formula. We give here an equivalent definition as a $\forall^*$-formula (see also the explanation given with the universal formulation of $\underline{\mathtt{f}}(x)$, Section 3.

$$\begin{aligned}
& \bigwedge_{h \neq f} \forall \bar{u}, v_1, v_2, v_3 \Big(&& x \neq h(\bar{u}) \wedge x \neq f(h(\bar{u}), v_1) \\
& && \wedge x \neq f(v_1, h(\bar{u})) \\
& && \wedge x \neq f(f(v_1, v_2), f(h(\bar{u}), v_3))\Big) \\
\wedge & \bigwedge_{h \neq f} \forall \bar{u}, v_1, v_2, v_3 \Big(&& x' \neq h(\bar{u}) \wedge x' \neq f(h(\bar{u}), v_1) \\
& && \wedge x' \neq f(v_1, h(\bar{u})) \\
& && \wedge x' \neq f(f(v_1, v_2), f(h(\bar{u}), v_3))\Big) \\
\wedge & \forall v_1, v_2, v_3, v_4, v_5, v_1', v_2', v_3', v_4', v_5' \Big(&& x = f(f(v_1, v_2), f(f(v_3, v_4), v_5)) \\
& && \wedge x' = f(f(v_1', v_2'), f(f(v_3', v_4'), v_5')) \\
& && \rightarrow v_1 = \alpha \wedge v_3 = 0 \wedge v_5 = f(0, v_5') \\
& && \wedge v_1' = \alpha \wedge v_3' = 0 \\
& && \wedge f(f(0, v_4), v_5) < x' \\
& && \wedge \bigvee_{(p,q) \in P} (v_2' = v_2 \cdot \underline{p} \wedge v_4' = v_4 \cdot \underline{q})\Big)
\end{aligned}$$