Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

Research
Report
RR-93-42

The First-Order Theory of Lexicographic Path Orderings is Undecidable

Hubert Comon, Ralf Treinen

September 1993

Deutsches Forschungszentrum für Künstliche Intelligenz
GmbH

Postfach 20 80
67608 Kaiserslautern, FRG
Tel.: (+49 631) 205-3211/13
Fax: (+49 631) 205-3210

Stuhlsatzenhausweg 3
66123 Saarbrücken, FRG
Tel.: (+49 681) 302-5252
Fax: (+49 681) 302-5341

Deutsches Forschungszentrum für Künstliche Intelligenz

The German Research Center for Artificial Intelligence (Deutsches Forschungszentrum für Künstliche Intelligenz, DFKI) with sites in Kaiserslautern and Saarbrücken is a non-profit organization which was founded in 1988. The shareholder companies are Atlas Elektronik, Daimler-Benz, Fraunhofer Gesellschaft, GMD, IBM, Insiders, Mannesmann-Kienzle, SEMA Group, Siemens and Siemens-Nixdorf. Research projects conducted at the DFKI are funded by the German Ministry for Research and Technology, by the shareholder companies, or by other industrial contracts.

The DFKI conducts application-oriented basic research in the field of artificial intelligence and other related subfields of computer science. The overall goal is to construct *systems with technical knowledge and common sense* which - by using AI methods - implement a problem solution for a selected application area. Currently, there are the following research areas at the DFKI:

- Intelligent Engineering Systems
- Intelligent User Interfaces
- Computer Linguistics
- Programming Systems
- Deduction and Multiagent Systems
- Document Analysis and Office Automation.

The DFKI strives at making its research results available to the scientific community. There exist many contacts to domestic and foreign research institutions, both in academy and industry. The DFKI hosts technology transfer workshops for shareholders and other interested groups in order to inform about the current state of research.

From its beginning, the DFKI has provided an attractive working environment for AI researchers from Germany and from all over the world. The goal is to have a staff of about 100 researchers at the end of the building-up phase.

Friedrich J. Wendl
Director

The First-Order Theory of Lexicographic Path Orderings is Undecidable

Hubert Comon, Ralf Treinen

DFKI-RR-93-42

This work has been supported by the ESPRIT working group CCL. The results of this paper have been obtained while the second author was visiting LRI, Univ. Paris-Sud. The second author has been supported by The Bundesminister für Forschung und Technologie (FKZ ITW-9105).

The First-Order Theory of Lexicographic Path Orderings is Undecidable*

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

Ralf Treinen
German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany
treinen@dfki.uni-sb.de

Abstract

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

*This work has been supported by the ESPRIT working group CCL. The results of this paper have been obtained while the second author was visiting LRI, Univ. Paris-Sud. The second author has been supported by The Bundesminister für Forschung und Technologie (FKZ ITW-9105).

Contents

1 Introduction

The *recursive path orderings* 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 $s >_{rpo} t$, then, for every context C, $C[s] >_{rpo} C[t]$ (this is the *monotonicity* property) and, assuming that variable symbols are uncomparable with any other term (except themselves), $s >_{rpo} t$ implies that $s\sigma >_{rpo} t\sigma$ 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 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 ones 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* $s \rightarrow t \mid s > t$ and $t \rightarrow s \mid t > s$ which are respectively interpreted as the set of all $s\sigma \rightarrow t\sigma$ such that $s\sigma \gg t\sigma$ and the set of all $t\sigma \rightarrow s\sigma$ such that $t\sigma \gg s\sigma$. This allows to use ordered strategies, even in presence of equations that are not uniformly orientable. A similar approach was used for the *unfailing completion* [8] and was described in its full generality in [13] where the completeness of a set of deduction rules is also 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.

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. The case of a total lexicographic path ordering has been considered 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 $\forall^*\exists^*\forall^*\exists^*$ 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 $\forall^*\exists^*$ fragment of a total lexicographic path ordering is undecidable. This also improves Treinen's result for the partial case by reducing the number of quantor alternations of the undecidable fragment.

The question of decidability of $\forall^*\exists^*$ fragment of a total lexicographic path ordering is 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:

$$\frac{s \to t \mid c \quad u \to v \mid c'}{u \to v \mid c' \qquad s[v]_p = t \mid c' \wedge s|_p = u} \quad \text{If } T(F) \models \forall Var(s) \exists Var(u).c \Rightarrow (s|_p = u \wedge c')$$

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 $s|_p$". This rule requires to solve a formula in the $\forall^*\exists^*$ fragment of the ordering. Actually it is equivalent to the $\forall^*\exists^*$ fragment: as a consequence of our undecidability result, we get that the applicability of the above simplification rule is undecidable.

The undecidability proof follows the ideas developed by R. Treinen in [15]: we encode the Post Correspondence Problem thanks to a direct simulation of sequences. The coding is not very difficult. However, the formula which expresses the main property of the coding, though simple, is not starightforward.

2 Statement of the problem

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. All these symbols are assumed to have arity 0. The set of terms built over F only is written $T(F)$. We write $T(F, X)$ insteadt of $T(F \cup X)$.

Assuming an ordering $\geq_F$ on F, the *lexicographic path ordering* $\geq_{lpo}$ on $T(F)$ is defined as follows (see e.g. [4]).

$f(s_1, \ldots, s_n) >_{lpo} g(t_1, \ldots, t_m)$ iff one of the following holds:

$s_i \geq_{lpo} g(t_1, \ldots, t_n)$ for some i

$f >_F g$ and $f(s_1, \ldots, s_n) >_{lpo} t_i$ for all $i = 1, \ldots, m$

$f = g$ and the two following properties are satisfied:

- $f(s_1, \ldots, s_n) >_{lpo} t_i$ for all $i = 1, \ldots, m$
- there is an index $i \in \{1, \ldots, n\}$ such that $s_1 = t_1 \wedge \ldots \wedge s_{i-1} = t_{i-1}$ and $s_i >_{lpo} t_i$

In this definition (and in the following) we use $s >_{lpo} t$ as an abbreviation for $s \geq_{lpo} t$ and $s \neq t$.

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

Proposition 2.1 *The relation $\geq_{lpo}$ defined on $T(F)$ is an ordering. Moreover:*

- *it is monotonic, i.e.* $f(s_1,\ldots,s_n) \geq_{lpo} f(t_1,\ldots,t_n)$ *whenever* $s_i \geq_{lpo}$ *for all* $i = 1,\ldots,n$,
- *it has the subterm property: if* t *is a strict subterm of* s*, then* $s >_{lpo} t$.

We consider the logical language built on the two binary predicates $=$ and $\geq$. More precisely, we consider $\forall^*\exists^*$ formulas, which can be written $\forall\overline{x}, \exists\overline{y}.P$ where P is a Boolean combination (using connectives $\wedge, \vee, \neg$) of expressions $s = t$ and $s \geq t$ where $s, t \in T(F, X)$. Let $\mathcal{L}$ be this logical language.

The formulas 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 by some precedence $\geq_F$ on 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 $\mathcal{A} \models \phi$ is undecidable for $\phi \in \mathcal{L}$.

We assume here that F is any finite set of function symbols. $\geq_F$ is any ordering on F such that 0 is a constant (symbol of arity 0) which is minimal among the constant symbols. f is a binary function symbol and it is minimal in $F - \{0\}$. g is a minimal symbol larger than f. For convenience, we assume that g is unary, but it can be actually any non-constant function symbol.

Lemma 2.2 *For every* $t \in T(F)$ *and every* $u \in T(\{f, 0\})$*, if* $t <_{lpo} u$*, then* $f(0,t) \leq_{lpo} u$.

Proof

Let $t <_{lpo} u$. We proceed by induction on $|t| + |u|$, the total number of function symbols occurring in t and u. The term u cannot be 0 since t contains a constant a and, by the subterm property of $\geq_{lpo}$, $u >_{lpo} t \geq_{lpo} a$. This contradicts the minimality of 0 among the constants of F. Hence $u = f(u_1, u_2)$. First, observe that

$$f(0, u_1) \leq_{lpo} f(u_1, u_2) \quad \text{and} \quad f(0, u_2) \leq_{lpo} f(u_1, u_2) \tag{1}$$

The first inequality holds because either $u_1 = 0$ and this follows from $u_2 \geq_{lpo} 0$, or else $u_1 >_{lpo} 0$ and this follows from $f(u_1, u_2) >_{lpo} u_1$. The second inequality holds since $0 \leq_{lpo} u_1$.

Let $t = h(\overline{t})$. According to the assumptions on the precedence, there are three cases: $h \not\leq_{lpo} f$, $h = 0$ and $h = f$.

$h \not\leq f$ By the lpo definition, $t \leq_{lpo} u_1$ or $t \leq_{lpo} u_2$ and the equality does not hold because the top symbols of the terms are distinct. Hence, by induction hypothesis, we have $t \leq_{lpo} f(0, u_1)$ or $t \leq_{lpo} f(0, u_2)$. Form (1) we get that, in any case, $f(0,t) \leq_{lpo} f(u_1, u_2)$.

$h = 0$ The inequality is obvious since $u_1 \geq_{lpo} 0$ and $u_2 \geq_{lpo} 0$.

$h = f$ Let $t = f(f_1, t_2)$. If $t \leq_{lpo} u_1$ or $t \leq_{lpo} u_2$, then $f(0,t) \leq_{lpo} f(0, u_1)$ or $f(0,t) \leq_{lpo} f(0, u_2)$. In both cases, the claim is a consequence of (1).

If $u_1 \neq 0$, then in fact $u_1 >_{lpo} 0$ since u consists only of the symbols 0 and f. From the assumption that $f(u_1, u_2) >_{lpo} f(t_1, t_2)$ it follows that $f(u_1, u_2) >_{lpo} f(0, f(t_1, t_2))$.

Otherwise, either $0 = u_1 = t_1$ and $u_2 >_{lpo} t_2$ or else $0 = u_1 >_{lpo} t_1$ and $u >_{lpo} t_2$. The second case can not occur, since $0 >_{lpo} t_1$ contradicts the minimality of 0 among the constants.

In the first case, we apply the induction hypothesis to $t_2 <_{lpo} u_2$ and obtain $f(0, t_2) \leq_{lpo} u_2$. Hence, $f(0, t) = f(0, f(0, t_2)) \leq_{lpo} f(0, u_2) = u$. □

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. The frame presented here is a modification of the method presented in [15].

An instance P of the Post Correspondence Problem [14] over the alphabet $\{a, b\}$ is a given by a finite set of the form $\{(p_i, q_i) \mid 0 \leq i \leq m; p_i, q_i \in \{a, b\}^+\}$. P is solvable if there is a sequence $i_1 \dots i_n \in \{1, \dots, m\}^+$ such that $p_{i_1} \cdots p_{i_n} = q_{i_1} \cdots q_{i_n}$. Solvability of an instance of the Post Correspondence Problem is one of the most famous undecidable problems [14].

Let $\mathcal{I}(x)$, $\mathcal{F}(x)$ and $\mathcal{S}_P(x, x')$ be formulae such that $\mathcal{S}_P(x, x')$ defines a well-founded ordering on $\mathcal{A}$, that is there no infinite sequence $t_0, t_1, \dots$ of ground terms with $\mathcal{A} \models \mathcal{S}_P(t_i, t_{i+1})$ for all i. We show how to construct a formula $\underline{\texttt{solvable}}_{\mathcal{I},\mathcal{S}_P,\mathcal{F}}$ such that $\mathcal{A} \models \underline{\texttt{solvable}}_{\mathcal{I},\mathcal{S}_P,\mathcal{F}}$ holds if and only if there is a sequence $(t_0, \dots, t_n) \in \mathcal{A}^*$ with $\mathcal{A} \models \mathcal{I}(t_0)$, $\mathcal{A} \models \mathcal{F}(t_n)$ and $\mathcal{A} \models \mathcal{S}_P(t_i, t_{i+1})$ for every $i < n$.

Having such a $\underline{\texttt{solvable}}_{\mathcal{I},\mathcal{S}_P,\mathcal{F}}$ at hand, we can encode the solvability of an instance $P = \{(p_i, q_i) \mid i = 1, \dots, n\}$ of the Post correspondence problem over an alphabet $\{a, b\}$. First note that there is a straightforward representation of strings over the alphabet $\{a, b\}$ as terms in $T(F)$ as follows:

- the empty string is represented as the term 0,
- the function $\lambda w.cons(a, w)$ on words corresponds to the function $\lambda x.f(0, x)$ on terms,
- the function $\lambda w.cons(b, w)$ on words corresponds to the function $\lambda x.f(f(0, 0), x)$ on terms.

This induces an injective (but not surjective) representation function cw: $\{a, b\}^* \to T(F)$. For instance, $\text{cw}(ab) = f(0, f(f(0, 0), 0))$. In the following, we will often identify a string with its term representation and write w instead of $\text{cw}(w)$. For every fixed word $v \in \{a, b\}^*$ we can now easily define a formula $x = \underline{\texttt{prefix}}_v x'$ with the property that for all $w \in \{a, b\}^*$ and $t \in \mathcal{A}$, we have

$$\mathcal{A} \models t = \underline{\texttt{prefix}}_v \text{cw}(w) \quad \text{iff} \quad t = \text{cw}(append(v, w))$$

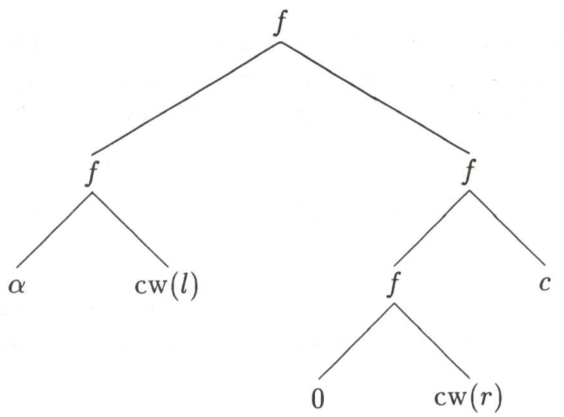

Figure 1: The term $\mathrm{cw}(l, r)$.

In the Post Correspondence Problem, we have in fact to consider sequences of *pairs* of strings. A first attempt of a pairing function could be to map (l, r) to the term $f(\mathrm{cw}(l), \mathrm{cw}(r))$. With this approach, we can not have both Lemma 3.4 and Lemma 3.5. We therefore take an other 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 a term which serves an index in a sequence of pairs (see Figure 1). We now define

$$\begin{aligned}
\mathcal{I}(x) &:= \exists z. x = f(f(\alpha, 0), f(f(0,0), z)) \\
\mathcal{F}(x) &:= \exists x_l. x = f(f(\alpha, x_l), f(f(0, x_l), 0)) \wedge x_l \neq 0 \\
\mathcal{S}_P(x, x') &:= \exists x_l, x_r, z, x_l', x_r', z'. \quad x = f(f(\alpha, x_l), f(f(0, x_r), z)) \\
&\qquad \wedge x' = f(f(\alpha, x_l'), f(f(0, x_r'), z')) \\
&\qquad \wedge z = f(0, z') \wedge f(f(0, x_r), z) < x' \\
&\qquad \wedge \bigvee_{(p,q) \in P} (x_l' = \underline{\mathtt{prefix}}_p x_l \wedge x_r' = \underline{\mathtt{prefix}}_q x_r)
\end{aligned}$$

where $\alpha = f(f(0, 0), 0)$.

Lemma 3.3 *An instance P of the Post Correspondence Problem has a solution if and only if there is a sequence $(t_0, \ldots, t_n) \in \mathcal{A}^*$ with $\mathcal{A} \models \mathcal{I}(t_0)$, $\mathcal{A} \models \mathcal{F}(t_n)$ and $\mathcal{A} \models \mathcal{S}_P(t_i, t_{i+1})$ for every $i < n$.*

Proof

Any such sequence $(t_0, \ldots, t_n)$ obviously exhibits a solution to P. On the other hand, let $(l_0, r_0), \ldots, (l_n, r_n)$ be a solution of P, where $l_0 = r_0 = \mathrm{cw}(\epsilon)$ and $l_n = r_n \neq \mathrm{cw}(\epsilon)$. We define the sequence $(t_0, \ldots, t_n)$ 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 $\mathcal{S}_P$, as the reader easily verifies. Note that, by the definition of the coding function cw, $f(\alpha, \mathrm{cw}(v)) >_{lpo} f(0, \mathrm{cw}(w))$ for all $v, w \in \{a, b\}^*$. □

The following lemma will be used in Section 4.

Lemma 3.4 *If $\mathcal{A} \models \mathcal{S}_P(t, t')$, then $t <_{lpo} t'$.*

Proof

By the definition of $\mathcal{S}_P(t, t')$, 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 definition of the Post Correspondence Problem, $t_l' >_{lpo} t_l$ and $t_r' >_{lpo} t_r$. Hence, $f(\alpha, t_l') >_{lpo} f(\alpha, t_l)$. The claim follows, since $t' >_{lpo} f(f(0, t_r), u)$ by definition of $\mathcal{S}_P(t, t')$. □

Lemma 3.5 *$\mathcal{S}_P$ defines a well founded relation on $\mathcal{A}$, that is there is no infinite sequence $t_0, t_1, \ldots$ of ground terms with $\mathcal{A} \models \mathcal{S}_P(t_i, t_{i+1})$ for every i.*

Proof

This follows immediately form the fact that the "z-component" is *de*creasing with respect to the subterm relation. □

The construction of $\underline{\texttt{solvable}}_{\mathcal{I},\mathcal{S}_P,\mathcal{F}}$ uses some subformulas. $\underline{\texttt{construction}}_{\mathcal{S}_P,\mathcal{F}}\, y$ will express the fact that y can be interpreted as a sequence $(t_0, \ldots, t_n)$ with $\mathcal{A} \models \mathcal{F}(t_n)$ and $\mathcal{A} \models \mathcal{S}_P(t_i, t_{i+1})$ for every $i < n$. $x\, \underline{\texttt{head}}\, y$ is intended to express that x is the head of the list y, $(x, y')\, \underline{\texttt{sub}}\, y$ is intended to express that the sequence with head x and tail y' is a subsequence of y and $\underline{\texttt{nonempty}}\, y$ will express that the list y has a head.

Now we can define

$$\begin{aligned} \underline{\texttt{solvable}}_{\mathcal{I},\mathcal{S}_P,\mathcal{F}} &:= \exists x, y. \mathcal{I}(x) \wedge \underline{\texttt{construction}}_{\mathcal{S}_P,\mathcal{F}}\, y \wedge \exists y'(x, y')\, \underline{\texttt{sub}}\, y \\ \underline{\texttt{construction}}_{\mathcal{S}_P,\mathcal{F}}\, y &:= \forall x, y'.\, (x, y')\, \underline{\texttt{sub}}\, y \rightarrow \\ &\qquad \{\mathcal{F}(x) \vee (\underline{\texttt{nonempty}}\, y' \wedge \forall x'.x'\, \underline{\texttt{head}}\, y' \rightarrow \mathcal{S}_P(x, x'))\} \end{aligned}$$

We have to verify that $\mathcal{A} \models \underline{\texttt{solvable}}_{\mathcal{I},\mathcal{S}_P,\mathcal{F}}$ 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

$$\text{Seq} := \{(t_0, \ldots, t_n) \in \mathcal{A}^* \mid \mathcal{A} \models \mathcal{F}(t_n) \text{ and } \mathcal{A} \models \mathcal{S}_P(t_i, t_{i+1}) \text{ for all } i < n\}$$

Lemma 3.6 *Let ct: Seq $\to \mathcal{A}$ such that for all $t, u \in \mathcal{A}$ and $s \in$ Seq we have*

$$\begin{aligned}
\mathcal{A} \models \underline{\texttt{nonempty}}\, ct(s) \quad & \textit{iff} \quad s \neq () && (2)\\
\mathcal{A} \models t\, \underline{\texttt{head}}\, ct(s) \quad & \textit{iff} \quad s = cons(t, c') \textit{ for some } c' \in Seq && (3)\\
\mathcal{A} \models (t, u)\, \underline{\texttt{sub}}\, ct(s) \quad & \textit{iff} \quad u = ct(s') \textit{ for some } s' \in Seq \\
& \qquad \textit{and } cons(t, s') \textit{ is a subsequence of } s && (4)
\end{aligned}$$

If P has a solution, then $\mathcal{A} \models \underline{\texttt{solvable}}_{\mathcal{I},\mathcal{S}_P,\mathcal{F}}$.

Proof

This follows directly from Lemma 3.3. □

Lemma 3.7 *Suppose that the following statements hold:*

$$\begin{aligned}
& \forall y. \underline{\texttt{nonempty}}\, y \to \exists x. x\, \underline{\texttt{head}}\, y && (5)\\
& \forall x, x', y, y'. (x, y')\, \underline{\texttt{sub}}\, y \wedge x'\, \underline{\texttt{head}}\, y' \wedge \mathcal{S}_P(x, x') \to \exists y''. (x', y'')\, \underline{\texttt{sub}}\, y && (6)
\end{aligned}$$

If $\mathcal{A} \models \underline{\texttt{solvable}}_{\mathcal{I},\mathcal{S}_P,\mathcal{F}}$, then P has a solution.

Proof

Suppose that $\mathcal{A} \models \underline{\texttt{construction}}_{\mathcal{S}_P,\mathcal{F}}\, b$. Using (5), (6) and Lemma 3.5, we show that if $\mathcal{A} \models (t, u')\, \underline{\texttt{sub}}\, u$, then there is a sequence $t_0, \ldots, t_n \in \mathcal{A}^*$ such that $t = t_0$, $\mathcal{A} \models \mathcal{F}(t_n)$ and $\mathcal{A} \models \mathcal{S}_P(t_i, t_{i+1})$ for all $i < n$. We proceed by induction on the relation $\mathcal{S}_P$ which is well founded by Lemma 3.5. If $\mathcal{A} \models \mathcal{F}(t)$, then we can take the sequence to be (t), and we are done. Otherwise, $\mathcal{A} \models \underline{\texttt{nonempty}}\, u$ holds. By (5), there is an t' with $\mathcal{A} \models t'\, \underline{\texttt{head}}\, u$. From the definition of $\underline{\texttt{construction}}_{\mathcal{I},\mathcal{S}_P,\mathcal{F}}\, y$ we get that $\mathcal{A} \models \mathcal{S}_P(t, t')$. Hence, by (6), there is a u'' such that $\mathcal{A} \models (t', u'')\, \underline{\texttt{sub}}\, u$. Now we can apply the induction hypothesis on t', which yields the claim. By Lemma 3.3, P has a solution. □

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

$\mathcal{I}(x)$	has quantifier prefix $\exists^*\forall^*$,	$\underline{\texttt{nonempty}}\, y$	has quantifier prefix $\forall^*$,
$\mathcal{S}_P(x, x')$	has quantifier prefix $\forall^*$,	$x\, \underline{\texttt{head}}\, y$	has quantifier prefix $\exists^*$,
$\mathcal{F}(x)$	has quantifier prefix $\forall^*$,	$(x, y')\, \underline{\texttt{sub}}\, y$	has quantifier prefix $\exists^*$.

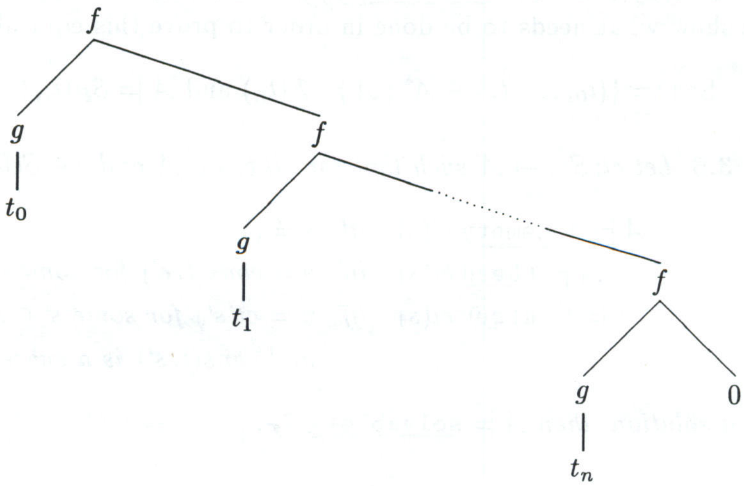

Figure 2: The term $\mathrm{ct}((t_0, \ldots, t_n))$.

The formula $\mathcal{I}(x)$ is already in the required form, but for $\mathcal{S}_P(x, x')$ and $\mathcal{F}(x)$ we have to find equivalent formulae in the $\forall^*$-fragment. This can be achieved with the quantifier elimination method of [3]. For the case of $\mathcal{S}_P(x, x')$ we have to extend this to inequalities. We illustrate this extension only with a simpler example: $\exists y, y'.x = f(y, y') \wedge y < y'$ is equivalent to

$$\forall \bar{u}. \bigwedge_{g \neq f} x \neq g(\bar{u}) \wedge \forall y, y'.x = f(y, y') \rightarrow y < y' .$$

4 The undecidability proof

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

4.1 The coding function

In this section we provide the missing definitions of the coding function and the predicates and prove Lemma 3.6. A sequence $(t_0, \ldots, t_n) \in \mathrm{Seq}$ will be coded as

$$\mathtt{ct}(t_0, \ldots, t_n) = f(g(t_0), f(g(t_1), \ldots, f(g(t_n), 0) \ldots))$$

which is depicted in Figure 2. The empty sequence will be encoded as 0.

Before we give the complete definition of the predicates, we first define an intermediate formula

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

The following lemma explains its meaning:

Lemma 4.8 *Let $\mathcal{A} \models \phi_1(t,u)$. Then*

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

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 $f(g(t),g(t)) >_{lpo} g_0(\bar{v})$. Now, since $g_0 \not\leq_F f$, we have $g(t) \geq_{lpo} g_0(\bar{v})$ by definition of $\geq_{lpo}$.

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

- If $h \not\leq_F f$, then decomposing $f(g(t),g(t)) >_{lpo} u$ according to the lpo definition, we get $g(t) \geq_{lpo} u$ which contradicts $u >_{lpo} g(t)$. Hence, this case cannot occur.
- If $h <_F f$, then, for some u_i, $f(g(t),g(t)) >_{lpo} u_i \geq_{lpo} g(t)$. There are two cases:
  - $u_i = g(t)$. In this case, $g(t)$ is a subterm of u.
  - $u_i >_{lpo} g(t)$. By induction hypothesis, $g(t)$ is a subterm of u_i. Hence, $g(t)$ is a subterm of u.
- If $h = f$, then the second inequality of $\phi_1(t,u)$ yields $u_1 \geq_{lpo} g(t)$ or $u_2 \geq_{lpo} g(t)$. If for some i equality holds, then the claim is proven. Otherwise, the first inequality of $\phi_1(t,u)$ yields $g(t) >_{lpo} u_1$ and $f(g(t),g(t)) >_{lpo} u_2$. Since this contradicts $u_1 >_{lpo} g(t)$, $u_2 >_{lpo} g(t)$ must hold. Hence, by induction hypothesis, $g(t)$ is a subterm of u_2 and consequently of u. □

Corollary 4.9 *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 $\mathcal{A} \models \exists x.\phi_1(x,u)$ we have to assume more hypotheses on u. More precisely, let

$$\begin{array}{rcl}\psi(y) & = & g(0) < y < g(g(0)) \\ & & \wedge \forall x. y \neq g(x) \\ & & \wedge \forall x. y > f(g(x),g(x)) \rightarrow y > g(f(0,x)) \\ & & \wedge \forall \bar{x}. \displaystyle\bigwedge_{\substack{f_1,f_2 \in F\setminus\{0\} \\ f_1 \not<_F f_2, f_2 \not\leq_F f_1}} \neg(y > f_1(\overline{x}) \wedge y > f_2(\overline{x'}))\end{array}$$

Lemma 4.10 *Let $u \in T(F)$. Then $\mathcal{A} \models \psi(u) \rightarrow \exists x.\phi_1(x,u)$.*

Proof

From the inequality $g(g(0)) >_{lpo} u$, we infer that every symbol in u is equal to or smaller than g. From this and the fact that $g(0) <_{lpo} u$ we infer that u contains at least one occurrence of g.

By the last part of $\psi(u)$, u cannot contain two uncomparable function symbols. This means in particular that all subterms of u are comparable w.r.t. $\geq_{lpo}$. Since u contains at least one occurrence of g, there is a greatest term w such that $g(w)$ is a subterm of u. We will show that $\mathcal{A} \models \phi_1(w, u)$.

We have of course $u \geq_{lpo} g(w)$. Moreover, u is not equal to $g(w)$ by the second part of $\psi(u)$. Now, if $f(g(w), g(w)) \not\geq_{lpo} u$, we have $u >_{lpo} f(g(w), g(w))$ since all symbols of w and u are comparable w.r.t. $>_F$. From the third part of $\psi(u)$ we conclude that $u >_{lpo} g(f(0, w))$. This contradicts the maximality of w, Hence $f(g(w), g(w)) \geq_{lpo} u$. □

Lemma 4.11 *For all sequences $s = (t_0, \ldots, t_n)$ with $n \geq 1$, we have $\mathcal{A} \models \psi(ct(s))$.*

Proof

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

1. $\mathcal{A} \models g(0) < \mathrm{ct}(s) < g(g(0))$. This follows immediately from the definition of $<_{lpo}$.
2. $\mathcal{A} \models \forall x.\mathrm{ct}(s) \neq g(x)$ since $\mathrm{ct}(s) = f(g(t_0, u))$.
3. $\mathcal{A} \models \forall x.\mathrm{ct}(s) > f(g(x), g(x)) \rightarrow \mathrm{ct}(s) > g(f(0, x))$. If $\mathrm{ct}(s) >_{lpo} f(g(t), g(t))$, then $t_i >_{lpo} t$ holds for some i. By Lemma 2.2, this implies $t_i \geq_{lpo} f(0, t)$. Hence, $\mathrm{ct}(s) >_{lpo} g(t_i) \geq_{lpo} g(f(0, t))$.
4. Every term smaller than $\mathrm{ct}(s)$ contains only symbols smaller than or equal to g. By our assumption on the precedence, all these symbols are comparable. This proves the last part of $\psi(\mathrm{ct}(s))$. □

Corollary 4.12 *For all sequences $s = (t_0, \ldots, t_n)$ with $n \geq 1$, we have $\mathcal{A} \models \phi_1(t_n, ct(s))$.*

Proof

By Lemma 4.11, $\mathcal{A} \models \psi(\mathrm{ct}(s))$. By Lemma 4.10, there is a t with $\mathcal{A} \models \phi(t, \mathrm{ct}(s))$. By Lemma 4.8, t must be equal to t_n. □

Now, let (x, y') <u>sub</u> y be the formula (this is the main trick):

$$\big(\phi_1(x, y) \wedge y' = 0\big) \vee \exists w.f(g(x), f(g(x), y')) > y \geq f(g(x), y') > g(w) > g(x) \wedge \phi_1(w, y)$$

Lemma 4.13 *Property (4) holds.*

Proof

We have to prove for all $(t_0, \ldots, t_n) \in \text{Seq}$:

$$\mathcal{A} \models (t, u') \,\underline{\text{sub}}\, \text{ct}(t_0, \ldots, t_n) \quad \Longleftrightarrow \quad \text{exists } i \leq n \text{ with } t = t_i \text{ and } y' = \text{ct}(t_{i+1}, \ldots, t_n).$$

It is understood that $(t_{n+1}, \ldots, t_n)$ is the empty sequence. First, note that $t_n >_{lpo} \ldots >_{lpo} t_0$ by definition of Seq and Lemma 3.4. This implies in particular that $\text{gs}(\text{ct}(t_0, \ldots, t_n)) = t_n$.

For the direction from left to right we have to consider two cases. If $\mathcal{A} \models \phi_1(t, \text{ct}(t_0, \ldots, t_n)) \wedge u' = 0$, then $t = t_n$ and the claim is proven.

Otherwise,

$$\mathcal{A} \models f(g(t), f(g(t), u')) > \text{ct}(t_0, \ldots, t_n) \geq f(g(t), u') > g(r) > g(t) \wedge \phi_1(r, \text{ct}(t_0, \ldots, t_n))$$

holds for some $r \in \mathcal{A}$. By Lemma 4.8, in fact $r = t_n$. Now, $\mathcal{A} \models g(r) > g(t)$, hence $t_n >_{lpo} t$. Let i be the smallest index such that $t_i \geq_{lpo} t$. Such an i exists since $t_n >_{lpo} t$. Hence, $t_{i'} \not\geq_{lpo} t$ for all $i' < i$. Using the lpo rules, $\text{ct}(t_0, \ldots, t_n) \geq_{lpo} f(g(t), u')$ is simplified into $\text{ct}(t_i, \ldots, t_n) \geq_{lpo} f(g(t), u')$, hence $\text{ct}(t_i, \ldots, t_n) >_{lpo} u'$.

Now let j be the smallest index such that $t \not\geq_{lpo} t_j$. Note that j is well defined since $t \not\geq_{lpo} t_n$. Since $f(g(t), f(g(t), u')) >_{lpo} \text{ct}(t_0, \ldots, t_n)$, it follows that $f(g(t), f(g(t), u')) >_{lpo} \text{ct}(t_j, \ldots, t_n)$. Since by construction $t \not\geq_{lpo} t_j$, this inequality is equivalent to $u' \geq_{lpo} \text{ct}(t_j, \ldots, t_n)$. Together we have

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

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

$$\begin{aligned} f(g(t_i), f(g(t_i), u')) >_{lpo} \text{ct}(t_0, \ldots, t_n) \quad &\Rightarrow^* \quad f(g(t_i), f(g(t_i), u')) >_{lpo} \text{ct}(t_i, \ldots, t_n) \\ &\Rightarrow \quad f(g(t_i), u') >_{lpo} \text{ct}(t_{i+1}, \ldots, t_n) \\ &\Rightarrow \quad u' \geq_{lpo} \text{ct}(t_{i+1}, \ldots, t_n) \end{aligned}$$

On the other hand, we have

$$\begin{aligned} \text{ct}(t_0, \ldots, t_n) \geq_{lpo} f(g(t_i), u') \quad &\Rightarrow^* \quad \text{ct}(t_i, \ldots, t_n) \geq_{lpo} f(g(t_i), u') \\ &\Rightarrow \quad \text{ct}(t_{i+1}, \ldots, t_n) \geq_{lpo} u' \end{aligned}$$

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

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

$$\begin{aligned} \mathcal{A} \models \exists w. \quad & f(g(t_i), f(g(t_i), \text{ct}(t_{i+1}, \ldots, t_n))) > \text{ct}(t_0, \ldots, t_n) \\ & \geq f(g(t_i), \text{ct}(t_{i+1}, \ldots, t_n)) > g(w) > g(t_i) \\ & \wedge \phi_1(w, \text{ct}(t_0, \ldots, t_n)) \end{aligned}$$

for all $i < n$ and that $\mathcal{A} \models \phi_1(t_n, \mathrm{ct}(t_0, \ldots, t_n))$. Both properties follow from $t_n >_{lpo} \ldots >_{lpo} t_0$ and the definition of $\geq_{lpo}$. (For the first property, we choose $w = t_n$). □

Now we define

$$\begin{array}{rcl} x\,\underline{\mathtt{head}}\,y & := & \exists y'.y = f(g(x), y') \wedge \exists w.(x < w \wedge \phi_1(w, y)) \vee y' = 0 \\ \underline{\mathtt{nonempty}}\,y & := & \forall \overline{u}, u'. \bigwedge_{f' \neq f} y \neq f'(\overline{u}) \wedge \bigwedge_{g' \neq g} y \neq f(g'(\overline{u}), u') \wedge \psi(y) \\ & & \wedge \forall x, y'.(y = f(g(x), y') \rightarrow (y' = 0 \vee \forall w.(\phi_1(w, y) \rightarrow x < w))) \end{array}$$

Lemma 4.14 *Properties (2) and (3) are satisfied with the above definitions.*

Proof

We have to prove 4 implications

1. If $\mathcal{A} \models \underline{\mathtt{nonempty}}\,\mathrm{ct}(s)$ then $s \neq \emptyset$. Let us first note that

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

   is logically equivalent to

$$\exists x, y'.y = f(g(x), y')$$

   (thanks to the results of [3] for example). This means that the sequence is indeed non-empty.

2. If $s \neq \emptyset$, then $\mathcal{A} \models \underline{\mathtt{nonempty}}\,\mathrm{ct}(s)$. We split this proof in three parts corresponding respectively to the three parts in the formula $\underline{\mathtt{nonempty}}\,y$.

   - When s is not empty, $\mathrm{ct}(s) = f(g(t_0), u)$ 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 4.11.
   - For the last part of the formula let $\mathrm{ct}(s) = f(g(t_0), u)$. If $u = 0$, then the formula holds. Otherwise, u must be of the form $f(g(t_1), v)$ with $t_1 >_{lpo} t_0$. By construction, $g(0) <_{lpo} u <_{lpo} g(g(0))$. Furthermore, for all w such that $\phi_1(w, \mathrm{ct}(s))$ holds, $g(w) \geq_{lpo} g(t_1) >_{lpo} g(t_0)$ thanks to Lemma 4.8. As a consequence, $w >_{lpo} t_0$ holds.

   Hence, in all cases, $\mathcal{A} \models \underline{\mathtt{nonempty}}\,\mathrm{ct}(s)$.

3. If $\mathcal{A} \models t\,\underline{\mathtt{head}}\,\mathrm{ct}(s)$ then $s = cons(t, s')$ for some $s' \in Seq$. Indeed, by definition of $x\,\underline{\mathtt{head}}\,y$, we must have $\mathcal{A} \models \exists y'.\mathrm{ct}(s) = f(g(t), y')$ which means that $s = (t, t_1, \ldots, t_n)$ and $s' = \mathrm{ct}(t_1, \ldots, t_n)$.

4. If $s = cons(a, c')$ for some $c' \in Seq$, then $\mathcal{A} \models a\,\underline{\mathtt{head}}\,\mathrm{ct}(s)$. Indeed, $\mathrm{ct}(s) = f(g(a), u)$ for some u. If $u = 0$, then the claim is proven. Otherwise, $t_n >_{lpo} a$ and $\mathcal{A} \models \phi_1(t_n, \mathrm{ct}(s))$ by Corollary 4.12. □

Note that actually some parts of the definitions of $x\,\underline{\mathtt{head}}\,y$ and $\underline{\mathtt{nonempty}}\,y$ are unnecessary for this lemma. However, they will be useful for proving property 6.

4.2 Properties of the predicates

In this subsection we prove Lemma 3.7. We are left to prove properties 6 and 5, which is the subject of the next two lemmas.

Lemma 4.15 *Property (5) holds.*

Proof

We have already seen that the first part of the formula $\underline{\mathtt{nonempty}}\,u$ 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 4.10 a t' with $\mathcal{A} \models \phi_1(t', u)$. From the last part of $\underline{\mathtt{nonempty}}\,u$ it follows that $\mathcal{A} \models t < t'$. □

Lemma 4.16 *Property (6) holds.*

Proof

Assume that $(t, u')\,\underline{\mathtt{sub}}\,u$ and $t'\,\underline{\mathtt{head}}\,u'$ and $\mathcal{S}_P(t, t')$ hold. $\mathcal{A} \not\models t'\,\underline{\mathtt{head}}\,0$, hence $\mathcal{A} \models (t, u')\,\underline{\mathtt{sub}}\,u$ implies that

$$\mathcal{A} \models \exists w. f(g(t), f(g(t), u')) > u \geq f(g(t), u') > g(w) > g(t) \wedge \phi_1(w, u) \tag{7}$$

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

$$\mathcal{A} \models u' = f(g(t'), u'') \wedge (u'' = 0 \vee \exists w'.\phi_1(w', u') \wedge t' < w') \tag{8}$$

Moreover, by Lemma 3.4, $\mathcal{A} \models \mathcal{S}_P(t, t')$ implies that $t' >_{lpo} t$.

We shall show that

$$\mathcal{A} \models (u'' = 0 \wedge t' = \mathrm{gs}(u)) \vee (f(g(t'), f(g(t'), u'')) > u \geq f(g(t'), u'') > g(\mathrm{gs}(u)) > g(t')$$

Note that, by (7) and (8), $\mathrm{gs}(u)$ and $\mathrm{gs}(u')$ exist. There are two cases:

$t' = \mathbf{gs}(u)$. From (7) and Lemma 4.8, we know that $u \geq_{lpo} f(g(t), u') >_{lpo} u' \geq_{lpo} g(\mathrm{gs}(u'))$. By the lpo rules, there must be a subterm $h(\bar{r})$ of u with $h \not<_F g$ and $h(\bar{r}) \geq_{lpo} g(\mathrm{gs}(u'))$. By the second part of Lemma 4.8, this means $g(\mathrm{gs}(u)) \geq_{lpo} h(\bar{r}) \geq_{lpo} g(\mathrm{gs}(u'))$, hence $\mathrm{gs}(u) \geq_{lpo} \mathrm{gs}(u')$. This contradicts $t = \mathrm{gs}(u) <_{lpo} \mathrm{gs}(u')$, hence $u'' = 0$ follows from (8).

$t' \neq \text{gs}(u)$. We have to prove four inequalities

1. $\mathcal{A} \models f(g(t'), f(g(t'), u'')) > u$. From (8) and form $t' >_{lpo} t$, we get

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

2. $\mathcal{A} \models u \geq f(g(t'), u'')$. From the assumptions, we get

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

3. $\mathcal{A} \models f(g(t'), u'') > g(\text{gs}(u))$. Indeed, $\mathcal{A} \models f(g(t), f(g(t'), u'')) > g(\text{gs}(u))$ which simplifies, since by (7) $t <_{lpo} \text{gs}(u)$, to $f(g(t'), u'') \geq g(\text{gs}(u))$. The two terms cannot be equal since they have distinct head symbols.

4. $\mathcal{A} \models g(\text{gs}(u)) > g(t')$. Let $u = r[g_1(\bar{v}_1), \ldots, g_n(\bar{v}_n)]$ where all symbols occurring in r are strictly smaller than g and $g_1, \ldots, g_n$ are not strictly smaller than g. In other words, $g_1(\bar{v}_1), \ldots, g_n(\bar{v}_n)$ are the maximal subterms of u headed by symbols which are not smaller than g. By Lemma 4.8, $g(\text{gs}(u)) \geq_{lpo} g_i(\bar{v}_i)$ for every i. On the other hand, $u >_{lpo} g(t')$. Now, using the lpo rules, this means that there is some j such that $g_j(\bar{v}_j) \geq_{lpo} g(t')$. Altogether, $g(\text{gs}(u)) \geq_{lpo} g(t')$. But, by hypothesis, the equality does not hold. Hence $g(\text{gs}(u)) >_{lpo} g(t')$. □

Theorem 4.17 *Let F contain (at least) one binary symbol f, one unary symbol g and one constant 0. The $\forall^*\exists^*$ 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

The reduction of the Post Correspondence Problem to the validity of $\forall^*\exists^*$ formula is established on the on hand by Lemma 3.6, Lemma 4.14 and Lemma 4.13, and on the other hand by Lemma 3.7, Lemma 4.15 and Lemma 4.16. □

5 Undecidability of the simplification rule

Let us recall the simplification rule given in introduction and which corresponds to the "total simplification rule" of [10].

$$\frac{s \to t \mid c \quad u \to v \mid c'}{u \to v \mid c' \qquad s[v]_p = t \mid c' \wedge s|_p = u} \quad \text{If } T(F) \models \forall Var(s) \exists Var(u). c \Rightarrow (c' \wedge s|_p = u)$$

When writing a constrained rule like $s \to t \mid c$, it is understood that $Var(c) \subseteq Var(s, t)$. 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 5.18 *Under the same assumption on the signature than in Theorem 4.17, 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 of a $\forall^*\exists^*$ sentence $\forall x_0, \ldots, x_n \exists y_0, \ldots, y_m. c$ to the problem of determining instances of the simplification rule. This sentence is obviously equivalent to

$$\forall x_0, \ldots, x_n \exists z_0, \ldots, z_n, y_0, \ldots, y_m. z_0 = x_0 \wedge \ldots \wedge z_n = x_n \wedge c[z_0/x_0, \ldots, z_n/x_n] \quad (9)$$

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

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

Now, (9) is equivalent to

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

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

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

is an instance of the simplification rule. □

6 Concluding Remarks

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

1. 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);
2. among the minimal symbols larger than f the should be a non-constant one (which we called g).

We believe that assumption 2 above can be removed, at the price of some additional coding, which we avoid here for sake of simplicity. However, condition 1 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. Note however that we are able to prove the undecidability of the $\forall^*\exists^*\forall^*$ fragment of the theory when assumption 1 is weakened to" "there is at least one non-unary and non constant function symbol". Indeed, using a larger fragment of the theory, we do not need lemma 2.2 which is the only place where we use minimality hypotheses on f.
Similarly, our method cannot be applied directly to multiset path orderings. Indeed, lemma 4.13 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 path orderings. 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 $\geq_{lpo}$. 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 $\geq_{lpo}$ 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 *Proc. CAAP 93*, 1993. LNCS 668.

[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. Book, editor, *Proc. 4th Rewriting Techniques and Applications, Como, LNCS 488*, 1991.

[7] N. Dershowitz, J.-P. Jouannaud, and J. Klop. More problems in rewriting. In *Proc. 5th Rewriting Techniques and Applications, Montréal, LNCS 690*, 1993.

[8] J. Hsiang and M. Rusinowitch. On word problems in equational theories. In *Proc. of 14th ICALP Karlsruhe*, July 1987.

[9] J.-P. Jouannaud and M. Okada. Satisfiability of systems of ordinal notations with the subterm property is decidable. In *Proc. 18th Int. Coll. on Automata, Languages and Programming, Madrid, LNCS 510*, 1991.

[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, D-66123 Saarbrücken, Germany. Feb. 1993.

[12] R. Nieuwenhuis. Simple LPO-constraint solving methods. Research Report LSI–93–9–R, Departament de Llenguatges i sistemes informàtics, Universitat Politècnica de Catalunya, 1993. To appear in Information Processing Letters.

[13] R. Nieuwenhuis and A. Rubio. Theorem proving with ordering constrained clauses. In D. Kapur, editor, *Proc. 11th Int. Conf. on Automated Deduction, Saratoga Springs, NY, LNCS 607*. Springer-Verlag, June 1992.

[14] E. 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.

Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

DFKI
-Bibliothek-
PF 2080
67608 Kaiserslautern
FRG

DFKI Publikationen

Die folgenden DFKI Veröffentlichungen sowie die aktuelle Liste von allen bisher erschienenen Publikationen können von der oben angegebenen Adresse bezogen werden.
Die Berichte werden, wenn nicht anders gekennzeichnet, kostenlos abgegeben.

DFKI Publications

The following DFKI publications or the list of all published papers so far can be ordered from the above address.
The reports are distributed free of charge except if otherwise indicated.

DFKI Research Reports

RR-92-43
Christoph Klauck, Jakob Mauss: A Heuristic driven Parser for Attributed Node Labeled Graph Grammars and its Application to Feature Recognition in CIM
17 pages

RR-92-44
Thomas Rist, Elisabeth André: Incorporating Graphics Design and Realization into the Multimodal Presentation System WIP
15 pages

RR-92-45
Elisabeth André, Thomas Rist: The Design of Illustrated Documents as a Planning Task
21 pages

RR-92-46
Elisabeth André, Wolfgang Finkler, Winfried Graf, Thomas Rist, Anne Schauder, Wolfgang Wahlster: WIP: The Automatic Synthesis of Multimodal Presentations
19 pages

RR-92-47
Frank Bomarius: A Multi-Agent Approach towards Modeling Urban Traffic Scenarios
24 pages

RR-92-48
Bernhard Nebel, Jana Koehler:
Plan Modifications versus Plan Generation: A Complexity-Theoretic Perspective
15 pages

RR-92-49
Christoph Klauck, Ralf Legleitner, Ansgar Bernardi:
Heuristic Classification for Automated CAPP
15 pages

RR-92-50
Stephan Busemann:
Generierung natürlicher Sprache
61 Seiten

RR-92-51
Hans-Jürgen Bürckert, Werner Nutt:
On Abduction and Answer Generation through Constrained Resolution
20 pages

RR-92-52
Mathias Bauer, Susanne Biundo, Dietmar Dengler, Jana Koehler, Gabriele Paul: PHI - A Logic-Based Tool for Intelligent Help Systems
14 pages

RR-92-53
Werner Stephan, Susanne Biundo:
A New Logical Framework for Deductive Planning
15 pages

RR-92-54
Harold Boley: A Direkt Semantic Characterization of RELFUN
30 pages

RR-92-55
John Nerbonne, Joachim Laubsch, Abdel Kader Diagne, Stephan Oepen: Natural Language Semantics and Compiler Technology
17 pages

RR-92-56
Armin Laux: Integrating a Modal Logic of Knowledge into Terminological Logics
34 pages

RR-92-58
Franz Baader, Bernhard Hollunder:
How to Prefer More Specific Defaults in Terminological Default Logic
31 pages

RR-92-59
Karl Schlechta and David Makinson: On Principles and Problems of Defeasible Inheritance
13 pages

RR-92-60
Karl Schlechta: Defaults, Preorder Semantics and Circumscription
19 pages

RR-93-02
Wolfgang Wahlster, Elisabeth André, Wolfgang Finkler, Hans-Jürgen Profitlich, Thomas Rist: Plan-based Integration of Natural Language and Graphics Generation
50 pages

RR-93-03
Franz Baader, Berhard Hollunder, Bernhard Nebel, Hans-Jürgen Profitlich, Enrico Franconi: An Empirical Analysis of Optimization Techniques for Terminological Representation Systems
28 pages

RR-93-04
Christoph Klauck, Johannes Schwagereit: GGD: Graph Grammar Developer for features in CAD/CAM
13 pages

RR-93-05
Franz Baader, Klaus Schulz: Combination Techniques and Decision Problems for Disunification
29 pages

RR-93-06
Hans-Jürgen Bürckert, Bernhard Hollunder, Armin Laux: On Skolemization in Constrained Logics
40 pages

RR-93-07
Hans-Jürgen Bürckert, Bernhard Hollunder, Armin Laux: Concept Logics with Function Symbols
36 pages

RR-93-08
Harold Boley, Philipp Hanschke, Knut Hinkelmann, Manfred Meyer: COLAB: A Hybrid Knowledge Representation and Compilation Laboratory
64 pages

RR-93-09
Philipp Hanschke, Jörg Würtz: Satisfiability of the Smallest Binary Program
8 Seiten

RR-93-10
Martin Buchheit, Francesco M. Donini, Andrea Schaerf: Decidable Reasoning in Terminological Knowledge Representation Systems
35 pages

RR-93-11
Bernhard Nebel, Hans-Juergen Buerckert: Reasoning about Temporal Relations: A Maximal Tractable Subclass of Allen's Interval Algebra
28 pages

RR-93-12
Pierre Sablayrolles: A Two-Level Semantics for French Expressions of Motion
51 pages

RR-93-13
Franz Baader, Karl Schlechta: A Semantics for Open Normal Defaults via a Modified Preferential Approach
25 pages

RR-93-14
Joachim Niehren, Andreas Podelski, Ralf Treinen: Equational and Membership Constraints for Infinite Trees
33 pages

RR-93-15
Frank Berger, Thomas Fehrle, Kristof Klöckner, Volker Schölles, Markus A. Thies, Wolfgang Wahlster: PLUS - Plan-based User Support Final Project Report
33 pages

RR-93-16
Gert Smolka, Martin Henz, Jörg Würtz: Object-Oriented Concurrent Constraint Programming in Oz
17 pages

RR-93-17
Rolf Backofen: Regular Path Expressions in Feature Logic
37 pages

RR-93-18
Klaus Schild: Terminological Cycles and the Propositional μ-Calculus
32 pages

RR-93-20
Franz Baader, Bernhard Hollunder: Embedding Defaults into Terminological Knowledge Representation Formalisms
34 pages

RR-93-22
Manfred Meyer, Jörg Müller: Weak Looking-Ahead and its Application in Computer-Aided Process Planning
17 pages

RR-93-23
Andreas Dengel, Ottmar Lutzy: Comparative Study of Connectionist Simulators
20 pages

RR-93-24
Rainer Hoch, Andreas Dengel: Document Highlighting — Message Classification in Printed Business Letters
17 pages

RR-93-25
Klaus Fischer, Norbert Kuhn: A DAI Approach to Modeling the Transportation Domain
93 pages

RR-93-26
Jörg P. Müller, Markus Pischel: The Agent Architecture InteRRaP: Concept and Application
99 pages

RR-93-27
Hans-Ulrich Krieger:
Derivation Without Lexical Rules
33 pages

RR-93-28
Hans-Ulrich Krieger, John Nerbonne, Hannes Pirker: Feature-Based Allomorphy
8 pages

RR-93-33
Bernhard Nebel, Jana Koehler:
Plan Reuse versus Plan Generation: A Theoretical and Empirical Analysis
33 pages

RR-93-34
Wolfgang Wahlster:
Verbmobil Translation of Face-To-Face Dialogs
10 pages

RR-93-35
Harold Boley, François Bry, Ulrich Geske (Eds.):
Neuere Entwicklungen der deklarativen KI-Programmierung — *Proceedings*
150 Seiten
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

RR-93-36
Michael M. Richter, Bernd Bachmann, Ansgar Bernardi, Christoph Klauck, Ralf Legleitner, Gabriele Schmidt: Von IDA bis IMCOD: Expertensysteme im CIM-Umfeld
13 Seiten

RR-93-38
Stephan Baumann: Document Recognition of Printed Scores and Transformation into MIDI
24 pages

RR-93-40
Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, Werner Nutt, Andrea Schaerf:
Queries, Rules and Definitions as Epistemic Statements in Concept Languages
23 pages

RR-93-41
Winfried H. Graf: LAYLAB: A Constraint-Based Layout Manager for Multimedia Presentations
9 pages

RR-93-42
Hubert Comon, Ralf Treinen:
The First-Order Theory of Lexicographic Path Orderings is Undecidable
9 pages

DFKI Technical Memos

TM-91-14
Rainer Bleisinger, Rainer Hoch, Andreas Dengel:
ODA-based modeling for document analysis
14 pages

TM-91-15
Stefan Busemann: Prototypical Concept Formation
An Alternative Approach to Knowledge Representation
28 pages

TM-92-01
Lijuan Zhang: Entwurf und Implementierung eines Compilers zur Transformation von Werkstückrepräsentationen
34 Seiten

TM-92-02
Achim Schupeta: Organizing Communication and Introspection in a Multi-Agent Blocksworld
32 pages

TM-92-03
Mona Singh:
A Cognitiv Analysis of Event Structure
21 pages

TM-92-04
Jürgen Müller, Jörg Müller, Markus Pischel, Ralf Scheidhauer:
On the Representation of Temporal Knowledge
61 pages

TM-92-05
Franz Schmalhofer, Christoph Globig, Jörg Thoben:
The refitting of plans by a human expert
10 pages

TM-92-06
Otto Kühn, Franz Schmalhofer: Hierarchical skeletal plan refinement: Task- and inference structures
14 pages

TM-92-08
Anne Kilger: Realization of Tree Adjoining Grammars with Unification
27 pages

TM-93-01
Otto Kühn, Andreas Birk: Reconstructive Integrated Explanation of Lathe Production Plans
20 pages

TM-93-02
Pierre Sablayrolles, Achim Schupeta:
Conlfict Resolving Negotiation for COoperative Schedule Management
21 pages

TM-93-03
Harold Boley, Ulrich Buhrmann, Christof Kremer:
Konzeption einer deklarativen Wissensbasis über recyclingrelevante Materialien
11 pages

DFKI Documents

D-92-19
Stefan Dittrich, Rainer Hoch: Automatische, Deskriptor-basierte Unterstützung der Dokumentanalyse zur Fokussierung und Klassifizierung von Geschäftsbriefen
107 Seiten

D-92-21
Anne Schauder: Incremental Syntactic Generation of Natural Language with Tree Adjoining Grammars
57 pages

D-92-22
Werner Stein: Indexing Principles for Relational Languages Applied to PROLOG Code Generation
80 pages

D-92-23
Michael Herfert: Parsen und Generieren der Prolog-artigen Syntax von RELFUN
51 Seiten

D-92-24
Jürgen Müller, Donald Steiner (Hrsg.): Kooperierende Agenten
78 Seiten

D-92-25
Martin Buchheit: Klassische Kommunikations- und Koordinationsmodelle
31 Seiten

D-92-26
Enno Tolzmann: Realisierung eines Werkzeugauswahlmoduls mit Hilfe des Constraint-Systems CONTAX
28 Seiten

D-92-27
Martin Harm, Knut Hinkelmann, Thomas Labisch: Integrating Top-down and Bottom-up Reasoning in COLAB
40 pages

D-92-28
Klaus-Peter Gores, Rainer Bleisinger: Ein Modell zur Repräsentation von Nachrichtentypen
56 Seiten

D-93-01
Philipp Hanschke, Thom Frühwirth: Terminological Reasoning with Constraint Handling Rules
12 pages

D-93-02
Gabriele Schmidt, Frank Peters, Gernod Laufkötter: User Manual of COKAM+
23 pages

D-93-03
Stephan Busemann, Karin Harbusch(Eds.): DFKI Workshop on Natural Language Systems: Reusability and Modularity - Proceedings
74 pages

D-93-04
DFKI Wissenschaftlich-Technischer Jahresbericht 1992
194 Seiten

D-93-05
Elisabeth André, Winfried Graf, Jochen Heinsohn, Bernhard Nebel, Hans-Jürgen Profitlich, Thomas Rist, Wolfgang Wahlster: PPP: Personalized Plan-Based Presenter
70 pages

D-93-06
Jürgen Müller (Hrsg.): Beiträge zum Gründungsworkshop der Fachgruppe Verteilte Künstliche Intelligenz Saarbrücken 29.-30. April 1993
235 Seiten
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

D-93-07
Klaus-Peter Gores, Rainer Bleisinger: Ein erwartungsgesteuerter Koordinator zur partiellen Textanalyse
53 Seiten

D-93-08
Thomas Kieninger, Rainer Hoch: Ein Generator mit Anfragesystem für strukturierte Wörterbücher zur Unterstützung von Texterkennung und Textanalyse
125 Seiten

D-93-09
Hans-Ulrich Krieger, Ulrich Schäfer: TDL ExtraLight User's Guide
35 pages

D-93-10
Elizabeth Hinkelman, Markus Vonerden,Christoph Jung: Natural Language Software Registry (Second Edition)
174 pages

D-93-11
Knut Hinkelmann, Armin Laux (Eds.): DFKI Workshop on Knowledge Representation Techniques — Proceedings
88 pages

D-93-12
Harold Boley, Klaus Elsbernd, Michael Herfert, Michael Sintek, Werner Stein: RELFUN Guide: Programming with Relations and Functions Made Easy
86 pages

D-93-14
Manfred Meyer (Ed.): Constraint Processing – Proceedings of the International Workshop at CSAM'93, July 20-21, 1993
264 pages
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

The First-Order Theory of Lexicographic Path Orderings is Undecidable

Hubert Comon, Ralf Treinen

RR-93-42

Research Report