Unique Normalization for Shallow TRS

Guillem Godoy[1] and Florent Jacquemard[2]

[1] Technical University of Catalonia, Jordi Girona 1, Barcelona, Spain
ggodoy@lsi.upc.edu*
[2] INRIA Saclay & LSV (UMR CNRS/ENS Cachan), France
florent.jacquemard@inria.fr

Abstract. Computation with a term rewrite system (TRS) consists in the application of its rules from a given starting term until a normal form is reached, which is considered the result of the computation. The unique normalization (UN) property for a TRS $\mathcal{R}$ states that any starting term can reach at most one normal form when $\mathcal{R}$ is used, i.e. that the computation with $\mathcal{R}$ is unique.
We study the decidability of this property for classes of TRS defined by syntactic restrictions such as linearity (variables can occur only once in each side of the rules), flatness (sides of the rules have height at most one) and shallowness (variables occur at depth at most one in the rules). We prove that UN is decidable in polynomial time for shallow and linear TRS, using tree automata techniques. This result is very near to the limits of decidability, since this property is known undecidable even for very restricted classes like right-ground TRS, flat TRS and also right-flat and linear TRS. We also show that UN is even undecidable for flat and right-linear TRS. The latter result is in contrast with the fact that many other natural properties like reachability, termination, confluence, weak normalization, etc. are decidable for this class of TRS.

Introduction

Term rewriting is a Turing-complete model of computation. Computation with a TRS consists in the application of its rules from a given starting term until a normal form is reached, i.e. a term that cannot be rewritten any more, which is usually considered as the result of the computation.

The unique normalization (UN) property for a TRS $\mathcal{R}$ states that any starting term can reach at most one normal form when $\mathcal{R}$ is used, i.e. that the computation with $\mathcal{R}$ is unique. This property is hence very desirable when dealing with TRS as computation models, and therefore it is important to establish the borders of its decidability.

Other interesting and much studied properties of TRS are reachability (whether a given term can be derived with a given TRS from another given term),

* The first author was supported by Spanish Min. of Educ. and Science by the FORMALISM project (TIN2007-66523) and by the LOGICTOOLS-2 project (TIN2007-68093-C02-01)

joinability (whether two given terms can be rewritten into one common term), termination (whether there are no infinite derivations from any starting term), confluence (whether every two terms derived from a common one can also be rewritten into another common term), weak normalization (whether every term can be rewritten into a normal form), uniqueness of normal forms (whether every term has at most one normal form equivalent to it modulo the equational theory induced by the TRS), etc. Note that a TRS may satisfy the unique normalization property and simultaneously unsatisfy the uniqueness of normal forms property. This is the case of $\mathcal{R} = \{a \to b,\ a \to c,\ c \to c,\ d \to c,\ d \to e\}$, where all terms can reach at most one normal form, but it holds that the normal forms b and c are equivalent.

In the recent years there has been much progress on determining decidability of these fundamental properties for several classes of TRS, which are defined by imposing certain syntactic restrictions on the rules. Some of the restrictions usually taken into consideration are groundness (no variable appears in the rules), linearity (variables can occur only once in each side of the rules), flatness (sides of the rules have height at most one) and shallowness (variables occur at depth at most one in the rules). When these restrictions refer only to one side of the rules, then we talk about left-linearity, right-linearity, left-flatness, etc.

Some of the strongest known decidability results are the following. Reachability and joinability are decidable for right-shallow right-linear TRS [11], and even for weaker restrictions [15]. Termination is decidable for right-shallow right-linear TRS [5] and other variants of syntactic restrictions based on the form of the dependency pairs obtained from a TRS [20]. Confluence is decidable for shallow right-linear TRS [8], and for right-(ground or variable) TRS [7]. The weak normalization problem is decidable for left-shallow left-linear TRS [11], right-shallow linear TRS and shallow right-linear TRS [6]. Uniqueness of normal forms is decidable in polynomial time for linear shallow TRS [19].

On the negative side, all of these properties have been proved undecidable for flat TRS [9, 10, 5, 4].

The case of the UN property seems to be more difficult. In [18], a polynomial time algorithm is given for UN and TRS with ground rules. On the negative side, UN is undecidable for right-flat and linear TRS [6], for right-ground TRS [16] and for flat TRS [4]. UN has also been shown undecidable for TRS whose rules have at most height two and, moreover, they are left-flat, right-linear, noncollapsing, or linear and noncollapsing in [17]. In [6] the decidability of this problem is left open for flat right-linear TRS.

In this paper, we provide a polynomial time algorithm for deciding UN for shallow and linear TRS (Section 2). We also prove (Section 3) undecidability of UN for flat and right-linear TRS. Our approach for decidability in polynomial time consists in giving a certain characterization of UN. We essentially show that UN is equivalent to the fact that certain regular sets of terms can reach at most one normal form. This characterization can be checked using tree automata techniques. The proof of undecidability is an adequate adaptation of the reductions appearing in [5, 4].

1 Preliminaries

Terms Algebra. We use standard notation from the term rewriting literature [1]. A *signature* is a finite set $\Sigma = \bigcup_{i=0}^{max\,\Sigma} \Sigma_i$ of function symbols, with i being the arity of symbols in Σ_i. Function symbols of arity 0 are called *constants*. Sometimes we denote a signature as $\{f_1 : a_1, \ldots, f_n : a_n\}$ where each f_i is a function symbol, and each a_i is its corresponding arity. Let $\mathcal{V}$ be a set disjoint from Σ whose elements are called *variables*. The set $\mathcal{T}(\Sigma, \mathcal{V})$ of terms over Σ and $\mathcal{V}$ is defined to be the smallest set containing $\mathcal{V}$ and such that $f(t_1, \ldots, t_m) \in \mathcal{T}(\Sigma, \mathcal{V})$ whenever $f \in \Sigma_m$ and $t_1, \ldots, t_m \in \mathcal{T}(\Sigma, \mathcal{V})$. $\mathcal{V}ar(t)$ denotes the set of variables occurring in the term t.

The *size* $\|t\|$ of a term t is the number of occurrences of variables and function symbols in t. The *height* of a term t, denoted as $height(t)$, is 0 if t is a constant or a variable, and $1 + max\{height(t_1), \ldots, height(t_m)\}$ if $t = f(t_1, \ldots, t_m)$. The positions of a term t, denoted p, q, are sequences of natural numbers that are used to identify the location of subterms of t. The set $\mathcal{P}os(t)$ of *positions* of t is defined by $\mathcal{P}os(t) = \{\epsilon\}$ if t is a constant or a variable, and $\mathcal{P}os(t) = \{\epsilon\} \cup \{1.p \mid p \in \mathcal{P}os(t_1)\} \cup \ldots \cup \{m.p \mid p \in \mathcal{P}os(t_m)\}$ if $t = f(t_1, \ldots, t_m)$, where ϵ denotes the empty sequence and $p.q$ denotes the concatenation of p and q. If t is a term and $p \in \mathcal{P}os(t)$ a position, then $t|_p$ is the subterm of t at position p. More formally, $t|_\epsilon = t$ and $f(t_1, \ldots, t_m)|_{i.p} = t_i|_p$. We denote by $t[s]_p$ ($p \in \mathcal{P}os(t)$) the term that is like t except that the subterm $t|_p$ is replaced by s. More formally, $t[s]_\epsilon = s$ and $f(t_1, \ldots, t_m)[s]_{i.p} = f(t_1, \ldots, t_{i-1}, t_i[s]_p, t_{i+1}, \ldots, t_m)$. We can define a partial order $\leq$ on $\mathcal{P}os(t)$ by $p \leq q$ if and only if p is a prefix of q, i.e there is a sequence p' such that $q = p.p'$. We say that two positions p and q are *parallel*, denoted $p \parallel q$, if they are incomparable with respect to $\leq$. Given a position p in a term s, the *depth* of the occurrence $s|_p$ in s is $|p|$. A *substitution* is a mapping $\mathcal{V} \to \mathcal{T}(\Sigma, \mathcal{V})$. Substitutions can also be applied to arbitrary terms by homomorphically extending its application to variables. The application of a substitution σ to a term t, denoted as $\sigma(t)$, is defined as follows for non-variable terms: $\sigma(f(t_1, \ldots, t_m)) = f(\sigma(t_1), \ldots, \sigma(t_m))$. A *variable renaming* is a substitution from variables to variables.

Term Rewriting. An *extended rewrite rule*, denoted $\ell \to r$, is a pair of terms $\ell \in \mathcal{T}(\Sigma, \mathcal{V})$ (the left-hand side) and $r \in \mathcal{T}(\Sigma, \mathcal{V})$ (the right-hand side). When ℓ is not a variable, and every variable occurring in r occurs also in ℓ, $\ell \to r$ is called a *rewrite rule*. An *extended term rewrite system* (extended TRS) $\mathcal{R}$ is a finite set of extended rewrite rules. A *term rewrite system* (TRS) $\mathcal{R}$ is a finite set of rewrite rules. A term s rewrites to t in one step at position p (by an extended TRS $\mathcal{R}$), denoted by $s \xrightarrow[\mathcal{R},p]{} t$, if $s|_p = \sigma(\ell)$ and $t = s[\sigma(r)]_p$, for some $\ell \to r \in R$ and substitution σ. In this case, s is said to be *$\mathcal{R}$-reducible*. Otherwise s is called an *$\mathcal{R}$-normal form*. The set of $\mathcal{R}$-normal forms is denoted by $\mathsf{NF}_\mathcal{R}$. Sometimes we write $s \xrightarrow[\mathcal{R}]{} t$ when p is not important, or $s \xrightarrow[\mathcal{R},\sigma,p]{} t$, for making the used substitution explicit. The transitive closure, and symmetric and transitive closure of $\xrightarrow[\mathcal{R}]{}$ are denoted as $\xrightarrow[\mathcal{R}]{*}$ and $\xleftrightarrow[\mathcal{R}]{*}$. When $s \xrightarrow[\mathcal{R}]{*} t$ we say that t is *reachable* from s, or that s reaches t. When $s \xleftrightarrow[\mathcal{R}]{*} t$ we say that

s and t are *equivalent*. When there exists a term u reachable from s and t, we say that s and t are *joinable*. When there exists a term u reachable from every term in a set S we say that S is *joinable*. Given $L \subseteq \mathcal{T}(\Sigma)$, we denote $\mathcal{R}^*(L) = \{t \mid \exists s \in L, s \xrightarrow[\mathcal{R}]{*} t\}$. The size of $\mathcal{R}$ is $\|\mathcal{R}\| = \sum_{\ell \to r \in \mathcal{R}}(\|\ell\| + \|r\|)$.

A term is *linear* if no variable occurs more than once in it. A term is *shallow* if all variables occur at depth at most one. A term is *flat* if its height is at most one. A rewrite rule $\ell \to r$ is *flat* (*linear*, *shallow*) if ℓ and r are. A rewrite rule $\ell \to r$ is *right-flat* (*right-linear*, *right-shallow*) if r is. A rewrite rule $\ell \to r$ is *left-flat* (*left-linear*, *left-shallow*) if ℓ is. A TRS is *flat* (*linear*, *shallow*) if all its rules are. A TRS is *right-flat* (*right-linear*, *right-shallow*, *left-flat*, *left-linear*, *left-shallow*) if all its rules are. A TRS is *uniquely normalizing*, or satisfies the *unique normalization* (UN) property, if for each term s and each two normal forms t_1 and t_2 reachable from s, $t_1 = t_2$ holds.

Tree Automata. A *tree automaton* (TA) $\mathcal{A}$ over a signature Σ is a tuple $(Q, Q^{\mathsf{f}}, \Delta)$ where Q is a finite set of nullary state symbols, disjoint from Σ, $Q^{\mathsf{f}} \subseteq Q$ is the subset of final states and Δ is a set of ground rewrite rules of the form: $f(q_1, \ldots, q_m) \to q$, or $q_1 \to q$ (ε-transition) where $f \in \Sigma_m$, and $q_1, \ldots, q_m, q \in Q$ (q is called the *target* state of the rule). The size of $\mathcal{A}$ is $\|\mathcal{A}\| = \sum_{f(q_1,\ldots,q_m)\to q \in \Delta}(m+2) + \sum_{q_1 \to q \in \Delta}(2)$.

The language of ground terms *accepted* by a TA $\mathcal{A}$ on Σ in a state q is the set $L(\mathcal{A}, q) := \{t \in \mathcal{T}(\Sigma) \mid t \xrightarrow[\Delta]{*} q\}$. The language of $\mathcal{A}$ is $L(\mathcal{A}) := \bigcup_{q \in Q^{\mathsf{f}}} L(\mathcal{A}, q)$ and a subset of $\mathcal{T}(\Sigma)$ is called *regular* if it is the language of a TA.

We shall use the following classical properties and problems of TA, see [2] for details.

Proposition 1. *Given two TA $\mathcal{A}_1$ and $\mathcal{A}_2$ over the same signature Σ, one can construct in polynomial time two TA recognizing respectively $L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$ and $L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$, whose sizes are respectively linear and quadratic in $\|\mathcal{A}_1\| + \|\mathcal{A}_2\|$.*

We will consider the two following decision problems for TA:

Problem: Emptiness. **Instance:** a TA $\mathcal{A}$; **Question:** $L(\mathcal{A}) = \emptyset$?
Problem: Singleton. **Instance:** a TA $\mathcal{A}$; **Question:** $|L(\mathcal{A})| = 1$?

Proposition 2. *[2] The emptiness problem is decidable in linear time. The singleton problem is decidable in polynomial time.*

2 Decidability of UN for Shallow and Linear TRS

In this section we prove that UN is decidable in polynomial time for flat and linear TRS (Theorem 1), and this result is immediately extended to shallow and linear TRS (Theorem 2). ¿From now on, we assume a fixed signature Σ and consider the maximal arity max_Σ of its function symbols as a constant. Hence, when analysing complexity of deciding the UN problem, this value is not considered as part of the input. This is a common approach when evaluating complexity for TRS problems (see e.g. the computation of a congruence closure

for shallow equations [12], or the time complexity given for certain problems using tree automata techniques [2]) . This is because, in practice, non-constant function symbols are usually fixed and represent relations with small arity or functions with few parameters, while constants are not fixed and represent the input data of the problem.

After proving some technical lemmas about rewrite reduction with flat and linear TRS in Subsection 2.1, we identify in Subsection 2.2 some necessary and sufficient conditions for UN of such TRS. Finally, in Subsection 2.3 we show how the conditions can be decided by reduction to the above tree automata decision problems.

2.1 Preliminary Results

In this subsection, we first present some technical lemmas concerning the rewriting sequences with flat and linear TRS which will be useful in the proof of Theorem 1. They are based on the notion of the *use* of a position and a derivation which is roughly a form of the descendant of the position.

Definition 1. *Let $\mathcal{R}$ be a flat and linear TRS over Σ. Given a derivation $s \xrightarrow[\mathcal{R}]{*} t$ and a position $p \in \mathcal{P}os(s)$, we define $use(p, s \xrightarrow[\mathcal{R}]{*} t)$ recursively on the length of $s \xrightarrow[\mathcal{R}]{*} t$ as follows:*
– If $s|_p \in \Sigma_0$, then $use(p, s \xrightarrow[\mathcal{R}]{*} t) := s|_p$.
– If $s|_p \notin \Sigma_0$ and $s \xrightarrow[\mathcal{R}]{*} t$ has length 0, then $use(p, s \xrightarrow[\mathcal{R}]{*} t)$ is undefined.
– If $s|_p \notin \Sigma_0$ and $s \xrightarrow[\mathcal{R}]{*} t$ is of the form $s \xrightarrow[p_1,\mathcal{R}]{} s' \xrightarrow[\mathcal{R}]{*} t$ for a position p_1 such that $p_1 \geq p$ or $p_1 \parallel p$ then $use(p, s \xrightarrow[\mathcal{R}]{*} t) := use(p, s' \xrightarrow[\mathcal{R}]{*} t)$.
– If $s|_p \notin \Sigma_0$ and $s \xrightarrow[\mathcal{R}]{*} t$ is of the form $s \xrightarrow[p_1,\ell\to r]{} s' \xrightarrow[\mathcal{R}]{*} t$, where $p = p_1.i.p_2$, and $\ell|_i \in \mathcal{V}$, we consider two cases. If $\ell|_i$ does not occur in r, then $use(p, s \xrightarrow[\mathcal{R}]{*} t)$ is undefined. Otherwise, if $\ell|_i = r|_q$ for some $q \in \mathcal{P}os(r)$, then $use(p, s \xrightarrow[\mathcal{R}]{*} t) := use(p_1.q.p_2, s' \xrightarrow[\mathcal{R}]{*} t)$.

Note that all possible cases are considered since $\mathcal{R}$ is flat. Moreover, the last one is well (uniquely) defined since $\mathcal{R}$ is linear.

Example 1. Let us consider the following flat and linear TRS
$\mathcal{R}_1 = \{x + 0 \to x, s(0) \to c_1, x + c_1 \to s(x), x + y \to y + x, s(c_1) \to 0\}$, and the derivation: $\rho_1 := 0 + s(0) \xrightarrow[\mathcal{R}_1]{} 0 + c_1 \xrightarrow[\mathcal{R}_1]{} s(0) \xrightarrow[\mathcal{R}_1]{} c_1$, and let ρ_1' be its subderivation starting with $0 + c_1$. We have $use(1, \rho_1) = 0$ and $use(2, \rho_1) = use(2, \rho_1') = c_1$. ◇

The following three lemmas can be easily proved by induction on the length of the rewrite sequences.

Lemma 1. *For any flat and linear TRS $\mathcal{R}$, derivation $s \xrightarrow[\mathcal{R}]{*} t$, position $p \in \mathcal{P}os(s)$ and all constant c, if $use(p, s \xrightarrow[\mathcal{R}]{*} t) = c$, then $s[c]_p \xrightarrow[\mathcal{R}]{*} t$ and $s|_p \xrightarrow[\mathcal{R}]{*} c$.*

Lemma 2. *For any flat and linear TRS $\mathcal{R}$, derivation $s \xrightarrow[\mathcal{R}]{*} t$, position $p \in \mathcal{P}os(s)$ such that $use(p, s \xrightarrow[\mathcal{R}]{*} t)$ is undefined, and all variable x, either $s[x]_p \xrightarrow[\mathcal{R}]{*} t$, or there exists a position $q \in \mathcal{P}os(t)$ such that $s[x]_p \xrightarrow[\mathcal{R}]{*} t[x]_q$ and $s|_p \xrightarrow[\mathcal{R}]{*} t|_q$.*

Lemma 3. *For any flat and linear TRS $\mathcal{R}$, derivation $s \xrightarrow[\mathcal{R}]{*} t$, position $p \in \mathcal{P}os(s)$ such that $s|_p$ is a certain variable x, and all (new) variable w not occurring in s, either $s[w]_p \xrightarrow[\mathcal{R}]{*} t$ or there exists a position $q \in \mathcal{P}os(t)$ such that $t|_q = x$ and $s[w]_p \xrightarrow[\mathcal{R}]{*} t[w]_q$.*

2.2 Necessary and sufficient conditions for UN

Definition 2. *Let $\mathcal{R}$ be a flat and linear TRS over Σ. A* fork *of $\mathcal{R}$ is a pair of terms $\langle f(s_1, \ldots, s_n), f(t_1, \ldots, t_n)\rangle$, for a function symbol $f \in \Sigma_n$ $(n \geq 0)$, such that for all $i \in \{1, \ldots, n\}$, either s_i and t_i are the same variable of $\mathcal{V}$, or s_i is a constant of Σ_0 and $t_i \xrightarrow[\mathcal{R}]{*} s_i$, or t_i is a constant of Σ_0 and $s_i \xrightarrow[\mathcal{R}]{*} t_i$.*

Example 2. $\langle x + (0 + s(0)), x + c_1\rangle$ and $\langle c_1 + x, s(0) + x\rangle$ are forks of the TRS $\mathcal{R}_1$ given in Example 1. ◇

Proposition 3. *A flat and linear TRS $\mathcal{R}$ over Σ is UN if and only if for each fork $\langle s, t\rangle$ of $\mathcal{R}$ and every $\mathcal{R}$-normal forms s', t' such that $s \xrightarrow[\mathcal{R}]{*} s'$ and $t \xrightarrow[\mathcal{R}]{*} t'$, it holds $s' = t'$.*

Proof. First, we show that the condition is necessary for unique normalization proceeding by contradiction. Assume that there exists a fork $\langle s, t\rangle$ and two different normal forms s' and t' such that $s \xrightarrow[\mathcal{R}]{*} s'$ and $t \xrightarrow[\mathcal{R}]{*} t'$. It suffices to construct a term u reaching both s and t in order to prove that $\mathcal{R}$ is not uniquely normalizing. Let s and t be of the form $f(s_1, \ldots, s_n)$ and $f(t_1, \ldots, t_n)$, respectively. We construct $u = f(u_1, \ldots, u_n)$ as follows. For every i in $\{1, \ldots, n\}$, if s_i and t_i are the same variable, then we define $u_i := s_i$. Otherwise, if $s_i \xrightarrow[\mathcal{R}]{*} t_i$ then we define $u_i := s_i$. Otherwise, $t_i \xrightarrow[\mathcal{R}]{*} s_i$ holds, and we define $u_i := t_i$. It is clear from this construction that $u \xrightarrow[\mathcal{R}]{*} s$ and $u \xrightarrow[\mathcal{R}]{*} t$ hold, and this concludes the proof for the only if direction.

We prove that the condition is sufficient again by contradiction: we assume that $\mathcal{R}$ is not uniquely normalizing and then prove the existence of a fork not as in Proposition 3. We choose a term u minimal in size with two distinct normal forms v and w reachable from u. The term u cannot be a variable, since variables cannot be rewritten by $\mathcal{R}$. Hence, $u = f(u_1, \ldots, u_n)$ for some $f \in \Sigma_n$ with $n \geq 0$. In order to conclude, it suffices to construct a fork $\langle s, t\rangle$ and two different normal forms s' and t' reachable from s and t, respectively.

For the construction of s, t, s' and t' we proceed iteratively as follows, by initializing them, and modifying them along n steps. The invariant is that s' and t' are always different normal forms reachable from s, and t, respectively, and that every $s|_i$ and $t|_i$ are either both the same variable, or both are $u|_i$, or one of them is a constant reachable from $u|_i$ and the other is $u|_i$, for $i \in \{1, \ldots, n\}$. At the end of the process, $\langle s, t\rangle$ will be a fork.

First, we set $s := u$, $t := u$, $s' := v$ and $t' := w$. After that, for each i in $\{1, \ldots, n\}$, we modify the values of s, t, s' and t' depending on the (un-)definition of $use(i, s \xrightarrow[\mathcal{R}]{*} s')$ and $use(i, t \xrightarrow[\mathcal{R}]{*} t')$.

If $use(i, s \xrightarrow[\mathcal{R}]{*} s')$ is defined to be a constant c, then, by Lemma 1, $s[c]_i \xrightarrow[\mathcal{R}]{*} s'$ and $s|_i \xrightarrow[\mathcal{R}]{*} c$. In this case we just set $s := s[c]_i$ and leave s', t and t' unchanged.

The case where $use(i, s \xrightarrow[\mathcal{R}]{*} s')$ is undefined but $use(i, t \xrightarrow[\mathcal{R}]{*} t')$ is defined to a constant is solved analogously to the previous one.

If both $use(i, s \xrightarrow[\mathcal{R}]{*} s')$ and $use(i, t \xrightarrow[\mathcal{R}]{*} t')$ are undefined, let x be a new variable not occurring in s nor t. By Lemma 2, either $s[x]_i \xrightarrow[\mathcal{R}]{*} s'$, or there exists a position q in $\mathcal{P}os(s')$ such that $s[x]_i \xrightarrow[\mathcal{R}]{*} s'[x]_q$ and $s|_i \xrightarrow[\mathcal{R}]{*} s'|_q$. (This is also analogously true for t and t'.) In any case we set $s := s[x]_i$. In the first case we leave s' unchanged and in the second case we let $s' := s'[x]_q$. We proceed analogously with t and t'. Note that both $s|_i$ and $t|_i$ are now the variable x, and that the new s' and t' are also normal forms reachable from s and t, respectively. But the preservation of the invariant stating that s' and t' are still different requires an explanation. They could only be equal if the same position q has been replaced to x in both terms, and in such a case, the old values $s'|_q$ and $t'|_q$ must be different. This would imply that the old $s|_i$ and $t|_i$ reach different normal forms, and hence, $u|_i$ can reach two different normal forms. But this is in contradiction with the fact that u is a minimal term in size reaching two different normal forms. □

We will use tree automata techniques for checking the condition provided by Proposition 3. But there is a difficulty: with tree automata, we can recognize just ground terms, or terms with variables chosen over a finite set of variables (seen as constants). Fortunately, the condition of Proposition 3 can be simplified by forcing the fork to contain at most two different variables.

Proposition 4. *Let x, y be two distinct variables. A flat and linear TRS $\mathcal{R}$ over Σ is not UN if and only if there exists a fork $\langle s, t\rangle$ of $\mathcal{R}$ with $s, t \in \mathcal{T}(\Sigma, \{x, y\})$ and two different $\mathcal{R}$-normal forms s', t' such that $s \xrightarrow[\mathcal{R}]{*} s'$ and $t \xrightarrow[\mathcal{R}]{*} t'$.*

For proving Proposition 4 it suffices to apply some variable renaming to the given fork by making an adequate use of Lemma 3.

Checking the hypotheses of Proposition 4 requires to test an infinite number of forks. In order to obtain a decision procedure for UN based on the notion of forks, we need a finite representation of such infinite sets of forks. For this purpose, we generalize Definition 2 of forks with regular sets of terms (Definition 3 below), and generalize Proposition 4 into Proposition 5 accordingly. In the next definition, we write $f(L_1, \ldots, L_n)$, where $f \in \Sigma_n$ and $L_1, \ldots, L_n \subseteq \mathcal{T}(\Sigma, X)$ for the set $\{f(t_1, \ldots, t_n) \mid t_1 \in L_1, \ldots, t_n \in L_n\}$.

Definition 3. *Let x, y be two fixed different variables. A* fork of languages *with respect to a flat and linear TRS $\mathcal{R}$ over Σ is a pair $\langle L, L'\rangle$ of sets of terms of $\mathcal{T}(\Sigma, \{x, y\})$ where L and L' have the form $f(L_1, \ldots, L_n)$ and $f(L'_1, \ldots, L'_n)$, respectively, and for every $i \in \{1, \ldots, n\}$, either $L_i = L'_i = \{x\}$, or $L_i = L'_i = \{y\}$, or $L_i = \{c\}$ and $L'_i = (\mathcal{R}^{-1})^*(\{c\}) \cap \mathcal{T}(\Sigma, \{x, y\})$, or $L'_i = \{c\}$ and $L_i = (\mathcal{R}^{-1})^*(\{c\}) \cap \mathcal{T}(\Sigma, \{x, y\})$, for some constant c.*

Recall that $\mathcal{R}^{-1}$ is not necessarily a TRS, but it is an extended TRS. The following lemma is an immediate consequence of Definition 3 (following the assumption that the maximal arity of a function symbol in Σ is fixed).

Lemma 4. *The number of forks of languages with respect to a flat and linear TRS $\mathcal{R}$ is polynomial in the size of $\mathcal{R}$.*

Proof. A fork of languages is determined by choosing a function symbol in Σ_n, and by iterating n times the election of either a constant in Σ_0, placed in one of two component of the fork of languages, or the variable x or the variable y. Thus, there are at most $|\Sigma| \cdot (2|\Sigma|+2)^{max_\Sigma}$ elections. If Σ (and not only max_Σ) is fixed then this is a constant. Otherwise, $|\Sigma|$ can be assumed bounded by $\|\mathcal{R}\|$, and hence, this is a polynomial on $\|\mathcal{R}\|$. □

Proposition 5. *A flat and linear TRS $\mathcal{R}$ over Σ is UN if and only if for all fork of languages $\langle L, L'\rangle$ with respect to $\mathcal{R}$, if $\mathcal{R}^*(L) \cap \mathsf{NF}_\mathcal{R} \neq \emptyset$ and $\mathcal{R}^*(L') \cap \mathsf{NF}_\mathcal{R} \neq \emptyset$, then $\mathcal{R}^*(L) \cap \mathsf{NF}_\mathcal{R} = \mathcal{R}^*(L') \cap \mathsf{NF}_\mathcal{R} = \{t\}$, for some term t.*

Proposition 5 is a direct consequence of Proposition 4.

2.3 Decision of UN

We show now how to decide the condition of Proposition 5, and thus, how to decide UN for flat and linear TRS, using tree automata techniques.

Lemma 5. *Given a flat and linear TRS $\mathcal{R}$ over $\Sigma \cup \{x, y\}$, there exists a TA $\mathcal{A}_\mathcal{R}$ on Σ, of size polynomial in the size of $\mathcal{R}$, recognizing $\mathsf{NF}_\mathcal{R} \cap \mathcal{T}(\Sigma, \{x, y\})$. Moreover, $\mathcal{A}$ can be computed in polynomial time.*

Proof. We construct a TA $\mathcal{A}_\mathcal{R} = (Q, Q^\mathsf{f}, \Delta)$ where $Q = Q^\mathsf{f} = \{q_\alpha \mid \alpha \in ((\Sigma_0 \cup \{x, y\}) \cap \mathsf{NF}_\mathcal{R})\} \cup \{q\}$, and Δ contains one rule $\alpha \to q_\alpha$ for every $q_\alpha \in Q$, and one rule $f(q_1, \ldots, q_n) \to q$ for all $q_1, \ldots, q_n \in Q$ such that the linear term associated to $f(q_1, \ldots, q_n)$ by replacing every occurrence of q_c by c, for each $c \in \Sigma_0$, and all occurrences of q_x, q_y and q by distinct variables, is not reducible by $\mathcal{R}$. The identity $L(\mathcal{A}_\mathcal{R}) = \mathsf{NF}_\mathcal{R} \cap \mathcal{T}(\Sigma, \{x, y\})$ follows by induction on terms for both inclusions. The construction of $\mathcal{A}$ takes time proportional to its size, which is $\mathcal{O}(|\Sigma| \cdot (|\Sigma_0| + 3)^{max_\Sigma})$. □

Note that linearity and flatness are both crucial for the above construction in the given complexity bounds. First, it is known that when $\mathcal{R}$ is not left-linear, then $\mathsf{NF}_\mathcal{R}$ is not necessarily a TA language. Second, $\mathsf{NF}_\mathcal{R} \cap \mathcal{T}(\Sigma, \{x, y\})$ is a TA language as soon as $\mathcal{R}$ is left-linear, but when $\mathcal{R}$ is not flat, there is no polynomial time construction of a TA for $\mathsf{NF}_\mathcal{R}$ with a polynomial number of states. This is a consequence of the EXPTIME lower bound for the ground reducibility of a linear term wrt a linear TRS [3].
The following lemma can be proved using a construction in [13] (Lemma 5.11).

Lemma 6. *Given a TA $\mathcal{A}$ over $\Sigma \cup \{x, y\}$ and an extended flat and linear TRS $\mathcal{R}$ over Σ, there exists a TA of size polynomial in $\|\mathcal{R}\| + \|\mathcal{A}\|$, recognizing $\mathcal{R}^*\big(L(\mathcal{A})\big) \cap \mathcal{T}(\Sigma, \{x, y\})$, which can be constructed in polynomial time.*

There exists a TA construction for larger classes of extended TRS than flat and linear, like right-shallow and right-linear TRS [11]. We focus on the flat and linear case here for complexity reasons. Nevertheless, as we shall see later, UN is undecidable for these larger classes of TRS.

Example 3. The TA $\mathcal{A}'_0 = \left(\{q_0, q_{c_1}\}, \{q_0\}, \Delta\right)$ recognizes $\left(\mathcal{R}_1^{-1}\right)^*(\{0\}) \cap \mathcal{T}(\Sigma, \{x, y\})$, for the TRS $\mathcal{R}_1$ of Example 1, where Δ contains $0 \to q_0$, $c_1 \to q_{c_1}$, $s(q_0) \to q_{c_1}$, $s(q_{c_1}) \to q_0$, $q_0 + q_0 \to q_0$, $q_{c_1} + q_{c_1} \to q_0$, $q_0 + q_{c_1} \to q_{c_1}$ and $q_{c_1} + q_0 \to q_{c_1}$. ◇

Now we shall use the above results for the decision of the sufficient condition for UN given in Proposition 5. The following lemma is a direct consequence of Lemma 6.

Lemma 7. *Let $\mathcal{R}$ be a flat and linear TRS over Σ. For each fork of languages $\langle L, L'\rangle$ with respect to $\mathcal{R}$, L and L' are recognized by two TA over $\Sigma \cup \{x, y\}$ whose respective sizes are polynomial in the size of $\mathcal{R}$, and which can be constructed in polynomial time.*

Now, we have all the ingredients to prove the main result of the paper.

Theorem 1. *UN is decidable in polynomial time for flat and linear TRS.*

Proof. Let $\mathcal{R}$ be a flat and linear TRS over Σ. We construct first a TA $\mathcal{A}_\mathcal{R}$ over $\Sigma \cup \{x, y\}$ recognizing $\mathsf{NF}_\mathcal{R} \cap \mathcal{T}(\Sigma, \{x, y\})$, and whose size $\|\mathcal{A}_\mathcal{R}\|$ is polynomial in the size of $\mathcal{R}$, following Lemma 5.

Now, for each fork of languages $\langle L, L'\rangle$ with respect to $\mathcal{R}$, we perform the following test. Let $\mathcal{A}$ and $\mathcal{A}'$ be two TA of size polynomial in the size of $\mathcal{R}$, recognizing respectively L and L' (constructed according to Lemma 7).

1. Construct two TA recognizing respectively $\mathcal{R}^*(L)$ and $\mathcal{R}^*(L')$. According to Lemma 6, their sizes are polynomial in the size of $\mathcal{R}$.
2. Construct two TA recognizing respectively $\mathcal{R}^*(L) \cap \mathsf{NF}_\mathcal{R}$ and $\mathcal{R}^*(L') \cap \mathsf{NF}_\mathcal{R}$, using the TA constructed at the above step and the above $\mathcal{A}_\mathcal{R}$. According to Proposition 1, the sizes of these two TA are still polynomial in the size of $\mathcal{R}$.
3. If one of the languages $\mathcal{R}^*(L) \cap \mathsf{NF}_\mathcal{R}$ or $\mathcal{R}^*(L') \cap \mathsf{NF}_\mathcal{R}$ is empty (this emptiness test is performed in polynomial time, according to Proposition 2) then the test passes successfully.
4. Otherwise, check whether both languages $\mathcal{R}^*(L) \cap \mathsf{NF}_\mathcal{R}$ and $\mathcal{R}^*(L') \cap \mathsf{NF}_\mathcal{R}$ are singleton sets (this test is performed in polynomial time according to Proposition 2). If it is not the case, the test fails. Otherwise, we construct a TA recognizing their intersection and the test succeeds iff its language is not empty (the construction and emptiness test can still be performed in polynomial time).

According to Proposition 5, $\mathcal{R}$ is UN iff all fork of languages with respect to $\mathcal{R}$ pass the test. According to Lemma 4, there will be a polynomial number of such tests, and following the above evaluation, each test is performed in polynomial time. Altogether, the upper bound on the complexity of deciding UN is polynomial. □

This result can be immediately extended to shallow TRS thanks to the following proposition given in [19].

Proposition 6. *Given a shallow and linear TRS $\mathcal{R}$ over Σ, there exists a flat and linear TRS $\mathcal{R}'$ on an extended signature $\Sigma' \supseteq \Sigma$ such that $\mathcal{R}'$ is UN iff $\mathcal{R}$ is UN. Moreover, the size of $\mathcal{R}'$ is polynomial in the size of $\mathcal{R}$, and can be computed in polynomial time.*

As a consequence of Theorem 1 and Proposition 6 we conclude decidability of UN in polynomial time for shallow and linear TRS.

Theorem 2. *UN is decidable in polynomial time for shallow and linear TRS.*

3 Undecidability of UN for Flat and Right-Linear TRS

Some undecidability results for UN have been recalled in the introduction. In particular, UN is undecidable for right-flat and linear TRS [6]. Thus, the result of Theorem 1 is no longer valid if we relax the assumption on flatness for the left-hand sides of rules. In this section, we show that one can neither relax the assumption on linearity for left-hand sides of rules in Theorem 1. More precisely, we prove (Theorem 3 below) undecidability of UN for flat and right-linear TRS. This result is in contrast with other properties like reachability, joinability, confluence, termination and weak normalization which are all decidable for flat and right-linear TRS [11, 8, 5, 6] (and in some cases for weaker restrictions). The proof involves a reduction from the Post correspondence problem (PCP) restricted to nonempty strings over a fixed finite alphabet Γ.

Problem: restricted-PCP (rPCP).
Instance: a sequence of pairs of words $\langle u_1, v_1\rangle \dots \langle u_n, v_n\rangle$, with $\forall i \leq n$, $u_i, v_i \in \Gamma^* \setminus \epsilon$.
Question: is there a non-empty sequence of indexes $1 \leq i_1, \dots, i_k \leq n$ such that $u_{i_1} \dots u_{i_k} = v_{i_1} \dots v_{i_k}$?

The rPCP is known to be undecidable [14]. We call *solution* of a rPCP instance a non-empty sequence of indexes that gives a positive answer to the above question.

Example 4. The instance of rPCP $\langle ab, a\rangle, \langle c, bc\rangle$ has a solution obtained by choosing the pairs 1 and 2 consecutively, obtaining abc at both sides. $\diamond$

In the proof of the undecidability theorem below, we assume a given instance $\langle u_1, v_1\rangle \dots \langle u_n, v_n\rangle$ of rPCP that is, u_i, v_i are nonempty strings over the alphabet Γ. The j'th symbol of u_i and v_i, whenever they exist, are denoted by u_{ij} and v_{ij} respectively. For the sake of readability, we shall sometimes write terms made of symbols of arity 0 and 1 as words over the same symbols.

Theorem 3. *UN is undecidable for flat and right-linear TRS.*

We shall define a TRS $\mathcal{R}$ such that the given rPCP instance has a solution iff $\mathcal{R}$ is not uniquely normalizing. The TRS $\mathcal{R}$ is defined as the union of several flat and right-linear TRS. The role of these sub-TRS is described below, as well as the signature they are based on.

We will represent the words of Γ^* by string terms of $\mathcal{T}\big(\Sigma_A \setminus \{A\}\big)$ where $\Sigma_A := \{\gamma : 1 \mid \gamma \in \Gamma\} \cup \{A : 0, \bot : 0\}$. These terms are generated by a regular tree grammar with a single non-terminal symbol (the constant A), and whose production rules are the rules of the TRS $\mathcal{R}_A$ below.

$$\mathcal{R}_A := \{A \to \gamma(A) \mid \gamma \in \Gamma\} \cup \{A \to \bot\}$$

Let $L = max(|u_1|, \ldots, |u_n|, |v_1|, \ldots, |v_n|)$, and let $\Sigma_U := \{U_{ij} : 1 \mid i \in [1..n], j \in [1..L]\big\} \cup \{\bot : 0\}$ and $\Sigma_V := \{V_{ij} : 1 \mid i \in [1..n], j \in [1..L]\big\} \cup \{\bot : 0\}$. Intuitively, we associate to a sequence $i_1, \ldots, i_k$ the pair made of the terms $U_{i_1 1} \ldots U_{i_1 L} \ldots U_{i_k 1} \ldots U_{i_k L} \bot$ and $V_{i_1 1} \ldots V_{i_1 L} \ldots V_{i_k 1} \ldots V_{i_k L} \bot$, which we call a *carrier* of the sequence. The following TRS permits to recover a solution (or more precisely the terms $u_{i_1} \ldots u_{i_k}$ and $v_{i_1} \ldots v_{i_k}$ that must be equal) from a carrier.

$$\begin{aligned}\mathcal{R}_s := \big\{U_{ij}x \to u_{ij}x \;\big|\; j \in \big[1..|u_i|\big]\big\} \cup \big\{U_{ij}x \to x \;\big|\; j \in \big[|u_i|+1..L\big]\big\} \\ \cup\; \big\{V_{ij}x \to v_{ij}x \;\big|\; j \in \big[1..|v_i|\big]\big\} \cup \big\{V_{ij}x \to x \;\big|\; j \in \big[|v_i|+1..L\big]\big\}\end{aligned}$$

Let us now consider several copies of the terms in carriers of solutions, built over the following signatures:

$$\begin{aligned}\Sigma''_U &:= \big\{U''_{ij} : 1, U'_{ij} : 0 \mid i \in [1..n], j \in [1..L]\big\} \cup \{U : 0, \bot : 0\}, \\ \Sigma''_V &:= \big\{V''_{ij} : 1, V'_{ij} : 0 \mid i \in [1..n], j \in [1..L]\big\} \cup \{V : 0, \bot : 0\}, \\ \Sigma_P &:= \big\{P_{ij} : 1, P'_{ij} : 0 \mid i \in [1..n], j \in [1..L]\big\} \cup \{P : 0, \bot : 0\}.\end{aligned}$$

The regular grammars generating copies of carriers, using the above (nullary) non-terminal symbols U, V and P, are called respectively R''_U, $\mathcal{R}''_V$ and $\mathcal{R}_P$.

$$\begin{aligned}\mathcal{R}''_U &:= \big\{U \to U'_{i1},\ U'_{ij} \to U''_{ij}U'_{i(j+1)},\ U'_{iL} \to U''_{iL}U,\ U'_{iL} \to U''_{iL}\bot \\ &\qquad\qquad \big|\ i \in [1..n], j \in [1..L-1]\big\} \\ \mathcal{R}''_V &:= \big\{V \to V'_{i1},\ V'_{ij} \to V''_{ij}V'_{i(j+1)},\ V'_{iL} \to V''_{iL}V,\ V'_{iL} \to V''_{iL}\bot \\ &\qquad\qquad \big|\ i \in [1..n], j \in [1..L-1]\big\} \\ \mathcal{R}_P &:= \big\{P \to P'_{i1},\ P'_{ij} \to P_{ij}P'_{i(j+1)},\ P'_{iL} \to P_{iL}P,\ P'_{iL} \to P_{iL}\bot \\ &\qquad\qquad \big|\ i \in [1..n], j \in [1..L-1]\big\}\end{aligned}$$

The following TRS casts carrier of solutions into copies.

$$\mathcal{R}_c := \left\{ \begin{matrix} U_{ij}x \to U''_{ij}x, U_{ij}x \to P_{ij}x \\ V_{ij}x \to V''_{ij}x,\ V_{ij}x \to P_{ij}x \end{matrix} \;\middle|\; i \in [1..n], j \in [1..L] \right\}$$

Let us now define the signature

$$\Sigma := \Sigma_A \cup \Sigma_U \cup \Sigma_V \cup \Sigma''_U \cup \Sigma''_V \cup \Sigma_P \cup \{f : 8, 0 : 0, 1 : 0\}$$

The additional symbols f, 0 and 1 are used in the next crux rewrite rules, which act as *checkers* for solutions, in order to ensure the correctness of the reduction.

$$\mathcal{R}_f := \{f(U,V,x,y,x,y,x,y) \to 0,\ f(x,y,A,P,x,x,y,y) \to 1\}$$

Finally, the last TRS $\mathcal{R}_n$ will ensure that 0 and 1 are the only non variable terms in normal form.

$$\begin{aligned}\mathcal{R}_n := &\left\{c \to c \mid c \in \Sigma_0 \setminus \{0,1\}\right\} \cup \{h(x) \to h(x) \mid h \in \Sigma_1\} \\ &\cup \{f(x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8) \to f(x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8)\}\end{aligned}$$

To summarize, the TRS $\mathcal{R}$ on Σ is defined by:

$$\mathcal{R} := \mathcal{R}_A \cup \mathcal{R}_s \cup \mathcal{R}''_U \cup \mathcal{R}''_V \cup \mathcal{R}_P \cup \mathcal{R}_c \cup \mathcal{R}_f \cup \mathcal{R}_n.$$

Note that all the rules of $\mathcal{R}$ are flat, all the rules of $\mathcal{R} \setminus \mathcal{R}_f$ are linear, and the rules in $\mathcal{R}_f$ are right-ground. Theorem 3 follows immediately from Lemmas 15 and 8 below, which ensure respectively the correctness and completeness of the reduction of the instance rPCP into the non-UN of $\mathcal{R}$.

Lemma 8. *If the given rPCP instance has a solution then $\mathcal{R}$ is not UN.*

Proof. We assume a solution $i_1, \ldots, i_k$ of the rPCP instance, and show that there exists a term $s \in \mathcal{T}(\Sigma)$ such that $0 \xleftarrow[\mathcal{R}]{*} s \xrightarrow[\mathcal{R}]{*} 1$. It permits us to conclude since 0 and 1 are $\mathcal{R}$-normal forms.

Let $w = u_{i_1} \ldots u_{i_k} \bot = v_{i_1} \ldots v_{i_k} \bot$, and let (s_U, s_V) be a carrier of the solution, and s''_U, s''_V, s_P be copies defined as follows.

$$\begin{array}{ll} s_U := U_{i_1 1} \ldots U_{i_1 L} \ldots U_{i_k 1} \ldots U_{i_k L} \bot & s_V := V_{i_1 1} \ldots V_{i_1 L} \ldots V_{i_k 1} \ldots V_{i_k L} \bot \\ s''_U := U''_{i_1 1} \ldots U''_{i_1 L} \ldots U''_{i_k 1} \ldots U''_{i_k L} \bot & s''_V := V''_{i_1 1} \ldots V''_{i_1 L} \ldots V''_{i_k 1} \ldots V''_{i_k L} \bot \\ s_P := P_{i_1 1} \ldots P_{i_1 L} \ldots P_{i_k 1} \ldots P_{i_k L} \bot & s := f(U,V,A,P,s_U,s_U,s_V,s_V) \end{array}$$

It is easy to verify $U \xrightarrow[\mathcal{R}''_U]{*} s''_U \xleftarrow[\mathcal{R}_c]{*} s_U$ and $V \xrightarrow[\mathcal{R}''_V]{*} s''_V \xleftarrow[\mathcal{R}_c]{*} s_V$. Moreover, A, s_U and s_V rewrite to w using $\mathcal{R}_A$ and $\mathcal{R}_s$. Also, P, s_U and s_V rewrite to s_P using $\mathcal{R}_P$ and $\mathcal{R}_c$. Therefore, there exist derivations $s \xrightarrow[\mathcal{R}]{*} f(U,V,w,s_P,w,s_P,w,s_P) \xrightarrow[\mathcal{R}_f]{} 0$ and $s \xrightarrow[\mathcal{R}]{*} f(s''_U,s''_V,A,P,s''_U,s''_U,s''_V,s''_V) \xrightarrow[\mathcal{R}_f]{} 1$ and this concludes the proof. □

Lemma 15 is slightly more difficult and requires some additional definitions and intermediate lemmas for its proof. For space reasons, they are given below withour proof.

Given a word w, we define *indexes*(w) to be the word obtained by applying to w the morphism φ defined as $\varphi(U_{i1}) = \varphi(U''_{i1}) = \varphi(V_{i1}) = \varphi(V''_{i1}) = \varphi(P_{i1}) = i$ and $\varphi(h) = \epsilon$ for any other symbol h. Note that two copies of the same carrier will have the same *indexes*. We say that a word $w\alpha$ is a *generator* if α is in $\{U,V,A,P,U'_{ij},V'_{ij},P'_{ij} | i \in [1..n], j \in [1..L]\}$. Otherwise, if α is $\bot$, we say that $w\alpha$ is a *non-generator*. We define Σ_c as the subset of Σ_1 of the unary symbols h for which there exists a collapsing rule $h(x) \to x$ in $\mathcal{R}$, i.e. Σ_c contains all U_{ij}

such that $j > |u_i|$, and all V_{ij} such that $j > |v_i|$. For any term t define $clean(t)$ recursively as follows. If the top symbol h of t is in Σ_c then we define $clean(t) = clean(t|_1)$. Otherwise, we define $clean(t) = t$. In other words, $clean(w\alpha)$ removes from t the longest prefix of unary symbols with all them in Σ_c.

Lemma 9. *Let s and t be terms satisfying $s \xrightarrow[\mathcal{R}]{*} t$. Then $clean(s) \xrightarrow[\mathcal{R}]{*} clean(t)$.*

Lemma 10. *If $s \xrightarrow[\mathcal{R}]{*} U$ then $clean(s) = U$; if $s \xrightarrow[\mathcal{R}]{*} V$ then $clean(s) = V$; if $s \xrightarrow[\mathcal{R}]{*} A$ then $clean(s) = A$; if $s \xrightarrow[\mathcal{R}]{*} P$ then $clean(s) = P$.*

Lemma 11. *A word $w\alpha$ is necessarily a non-generator if both $\{w\alpha, U\}$ and $\{w\alpha, A\}$, or both $\{w\alpha, V\}$ and $\{w\alpha, A\}$, or both $\{w\alpha, U\}$ and $\{w\alpha, P\}$, or both $\{w\alpha, V\}$ and $\{w\alpha, P\}$ are joinable.*

Lemma 12. *Let $w_1\bot$ and $w_2\bot$ be two non-generator words. Let α be either U or V or P. If $\{w_1\bot, w_2\bot, \alpha\}$ is $\mathcal{R}$-joinable, then $indexes(w_1) = indexes(w_2)$.*

Lemma 13. *Let $w_1\bot$ be a non-generator word joinable with U. Let $w_2\bot$ be a word reachable from $w_1\bot$ and such that $w_2 \in \Gamma^*$. If $i_1, \ldots, i_k = indexes(w_1)$ then $w_2 = u_{i_1} \ldots u_{i_k}$.*

The following lemma is the analogous to the previous lemma, but with V's and v's instead of U's and u's.

Lemma 14. *Let $w_1\bot$ be a non-generator word joinable with V. Let $w_2\bot$ be a word reachable from $w_1\bot$ and such that $w_2 \in \Gamma^*$. If $i_1, \ldots, i_k = indexes(w_1)$ then $w_2 = v_{i_1} \ldots v_{i_k}$.*

Now, we have all the necessary ingredients for proving Lemma 15.

Lemma 15. *If $\mathcal{R}$ is not UN then the given rPCP instance has a solution.*

Proof. Let s be a term in $\mathcal{T}(\Sigma, \mathcal{V})$, minimal in size, such that $s' \xleftarrow[\mathcal{R}]{*} s \xrightarrow[\mathcal{R}]{*} s''$ where s' and s'' are two distinct $\mathcal{R}$-normal forms. Note that only 0, 1 and variables are $\mathcal{R}$-normal forms.

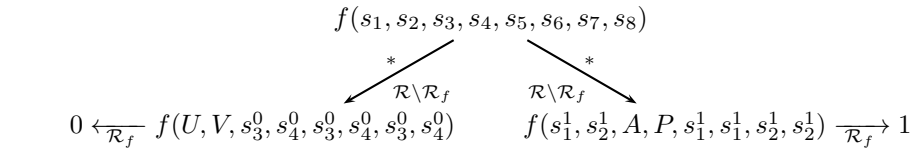
The term s is not rooted by a symbol in $\Sigma \setminus (\Sigma_c \cup \{f, 0, 1\})$: otherwise it could not reach a $\mathcal{R}$-normal form, because of the rules of $\mathcal{R}_n$ and because such symbols cannot be removed.

Assume that s is rooted by a symbol h in Σ_c. Thus, s is of the form $h(s_1)$ for some term s_1. Then in both derivations $s \xrightarrow[\mathcal{R}]{*} s'$ and $s \xrightarrow[\mathcal{R}]{*} s''$, the rule $h(x) \to x$ is applied at the root position. Hence, the term s_1 also reaches s' and s'' with $\mathcal{R}$, contradicting the minimality of s.

Assume that s is rooted by 0 or 1 or a variable. It means that s is directly 0 or 1 or a variable. Any of these terms is a normal form, and this is in contradiction with the fact that s reaches two different $\mathcal{R}$-normal forms.

From the above observations it follows that s is of the form $f(s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8)$ for some terms $s_1, \ldots, s_8$. Since no variable is reachable from a term of this form (there is no collapsing rule in $\mathcal{R}$ with a f at the top of its left-hand side), we conclude that $s' = 0$ and $s'' = 1$, or vice-versa. Thus,

the two rules of $\mathcal{R}_f$ are applied at the root position in the derivations from s and we have the following rewrite sequences:



Note that only rewrite steps with $f(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8) \to f(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8)$ occur at the root position in the two above diagonal rewrite sequences. This implies that each subterm at depth 1 in $f(s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8)$ reaches the subterm at depth 1 in $f(U, V, s_3^0, s_4^0, s_3^0, s_4^0, s_3^0, s_4^0)$ located at the same position, and also the subterm at depth 1 in $f(s_1^1, s_2^1, A, P, s_1^1, s_1^1, s_2^1, s_2^1)$ located at the same position.

By Lemma 9, we can assume without loss of generality that any subterm t at depth 1 in any of those three terms satisfies $\mathit{clean}(t) = t$. Moreover, by Lemma 10, it follows $s_1 = U$, $s_2 = V$, $s_3 = A$ and $s_4 = P$. This implies the following facts.

- U and s_5 are joinable, and s_5 and A are joinable, and hence, by Lemma 11, s_5 is a non-generator. Similarly, s_6, s_7 and s_8 are non-generators.
- $\{U, s_5, s_6\}$ is joinable, and hence, by Lemma 12, $\mathit{indexes}(s_5) = \mathit{indexes}(s_6)$. Similarly, $\mathit{indexes}(s_7) = \mathit{indexes}(s_8)$, and $\mathit{indexes}(s_6) = \mathit{indexes}(s_8)$. Let $i_1, \ldots, i_k$ be *indexes* of any of them.
- $\{U, s_5\}$ is joinable, and $\{A, s_5\}$ is joinable to a word s_3^0 of the form $w\bot$ such that $w \in \Gamma^*$. Hence, by Lemma 13, $w = u_{i_1} \ldots u_{i_k}$. Similarly, $\{V, s_7\}$ is joinable, and $\{A, s_7\}$ is joinable to the same $s_3^0 = w\bot$. Hence, by Lemma 14, $w = v_{i_1} \ldots v_{i_k}$. Thus, $u_{i_1} \ldots u_{i_k} = v_{i_1} \ldots v_{i_k}$.

From the above facts, it follows that $i_1, \ldots, i_k$ is a solution of the given rPCP instance. □

Conclusion

We have shown that UN is decidable in polynomial time for shallow and linear TRS (Theorem 2), and is undecidable for flat and right-linear TRS (Theorem 3). With these results, the problem of decidability of UN for classes of TRS defined by syntactic restrictions like linearity, flatness or shallowness is essentially closed. Perhaps, one could still consider this problem for other variants of syntactic restrictions based on the form of the dependency pairs obtained from a TRS, like the ones given in [20].

References

1. F. Baader and T. Nipkow. *Term rewriting and all that.* Cambridge University Press, New York, NY, USA, 1998.
2. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available on http://tata.gforge.inria.fr, 2007.
3. H. Comon and F. Jacquemard. Ground reducibility is EXPTIME-complete. *Information and Computation*, 187(1):123–153, 2003.
4. G. Godoy and H. Hernández. Undecidable properties for flat term rewrite systems. *Submitted*, 2008.
5. G. Godoy, E. Huntingford, and A.Tiwari. Termination of rewriting with right-flat rules. In *18th Int. Conf. Rewriting Techniques and Applications, (RTA)*, volume 4533 of *LNCS*, pages 200–213. Springer, 2007.
6. G. Godoy and S. Tison. On the normalization and unique normalization properties of term rewrite systems. In *21st Int. Conf. on Automated Deduction (CADE)*, volume 4603 of *LNCS*, pages 247–262. Springer, 2007.
7. G. Godoy and A. Tiwari. Deciding fundamental properties of right-(ground or variable) rewrite systems by rewrite closure. In *Intl. Joint Conf. on Automated Deduction (IJCAR)*, volume 3097 of *LNAI*, pages 91–106. Springer, 2004.
8. G. Godoy and A. Tiwari. Confluence of shallow right-linear rewrite systems. In *19th Int. Workshop of Computer Science Logic (CSL)*, volume 3634 of *LNCS*, pages 541–556. Springer, 2005.
9. F. Jacquemard. Reachability and confluence are undecidable for flat term rewriting systems. *Inf. Process. Lett.*, 87(5):265–270, 2003.
10. I. Mitsuhashi, M. Oyamaguchi, and F. Jacquemard. The confluence problem for flat TRSs. In *Proc. 8th Intl. Conf. on Artificial Intelligence and Symbolic Computation (AISC'06)*, volume 4120 of *LNAI*, pages 68–81. Springer, 2006.
11. T. Nagaya and Y. Toyama. Decidability for left-linear growing term rewriting systems. *Inf. Comput.*, 178(2):499–514, 2002.
12. R. Nieuwenhuis. Basic paramodulation and decidable theories (extended abstract). In *LICS*, pages 473–482, 1996.
13. K. Salomaa. Deterministic tree pushdown automata and monadic tree rewriting systems. *J. Comput. Syst. Sci.*, 37:367–394, 1998.
14. M. Sipser. *Introduction to the Theory of Computation.* Course Technology, 2006.
15. T. Takai, Y. Kaji, and H. Seki. Right-linear finite path overlapping term rewriting systems effectively preserve recognizability. In *11th Int. Conf. RTA*, volume 1833 of *LNCS*, pages 246–260. Springer, 2000.
16. R. Verma. Complexity of normal form properties and reductions for rewriting problems. *Fundamenta Informaticae*, 2008. Accepted for publication.
17. R. Verma. New undecidability results for properies of term rewrite systems. In *9th International workshop, RULE*, pages 1–16, 2008.
18. R. Verma and A. Hayrapetyan. A new decidability technique for ground term rewriting systems. *ACM Trans. Comput. Log.*, 6(1):102–123, Dec 2005.
19. R. Verma and J. Zinn. A polynomial-time algorithm for uniqueness of normal forms of linear shallow term rewrite systems. In *Symposium on Logic in Computer Science LICS (short presentation)*, 2006.
20. Y. Wang and M. Sakai. Decidability of termination for semi-constructor trss, left-linear shallow TRSs and related systems. In *17th Int. Conf. RTA*, volume 4098 of *LNCS*, pages 343–356. Springer, 2006.