Classes of Term Rewrite Systems with Polynomial Confluence Problems

Guillem Godoy[1], Robert Nieuwenhuis[2], and Ashish Tiwari[3]

[1] Technical University of Catalonia, Barcelona
ggodoy@lsi.upc.es
[2] Technical University of Catalonia, Barcelona
roberto@lsi.upc.es
[3] SRI Internaltional, Menlo Park, CA 94025
tiwari@csl.sri.com

Abstract. The confluence property of ground (i.e., variable-free) term rewrite systems (TRS) is well-known to be decidable. This was proved independently in [DTHL87,DHLT90] and in [Oya87] using tree automata techniques and ground tree transducer techniques (originated from this problem), yielding EXPTIME decision procedures (PSPACE for strings). Since then, and until last year, the optimality of this bound had been a well-known longstanding open question (see, e.g., [RTA01]).
In [CGN01]) we gave the first polynomial-time algorithm for deciding the confluence of ground TRS. Later in [Tiw02] this result was extended, using abstract congruent closure techniques, to linear shallow TRS, i.e., TRS where no variable occurs twice in the same rule nor at depth greater than one. Here, we give a new and much simpler proof of the latter result.

1 Introduction

The fields of rewriting theory, formal language theory (and in particular automata theory), and algebra have provided many useful tools to each other [BO93,Ott99]. The theory of rewriting also contributes with fundamental results to programming languages (semantics, implementation, static analysis) and automated deduction (symbolic computation, decision procedures, combination procedures, constraint solving, parallel deduction), see, e.g., [DJ90,DP01] and their references.

Rewriting essentially consists of the application of rules to expressions, replacing subexpressions by other ones (usually equivalent ones, in some sense).

Example 1. A (bottom up) *tree automaton* essentially consists of a ground term rewrite system, like

$$\begin{array}{rr} a \to q_a & g(q_g) \to q_g \\ g(q_a) \to q_g & f(q_a, q_a) \to q_f \\ f(q_g, q_f) \to q_{accept} & \end{array}$$

for an automaton recognizing the regular tree language $f(g^+(a), f(a, a))$.

String rewrite systems or Thue systems are a particular case of term rewrite systems, restricted to the language of unary function symbols and constants. A fundamental property of term rewrite systems is the *confluence* or *Church-Rosser* property: it essentially says that if some irreducible expression is reached by successive rewrite (or *reduction*) steps, then this is also the case independently of which subexpression is rewritten first and by which rule, i.e., confluence converts nondeterminism from "don't know" into "don't care", thus avoiding the need for backtracking. Therefore, confluence ensures the uniqueness of expressions in normal (i.e., irreducible) form. For instance, the tree automaton of Example 1 can be regarded as deterministic since the rewrite system is confluent.

Confluence is, of course, a well-studied property for different classes of rewrite systems. It is undecidable in general (see e.g., [KNO90] and its references), it is decidable for terminating systems [KB70], and, interestingly, it is also decidable for arbitrary ground systems. This latter result was proved independently in [DTHL87,DHLT90] and in [Oya87] using tree automata techniques and ground tree transducer techniques (originated from this problem), yielding in both cases EXPTIME decision procedures. The basic idea is that one can build a polynomial-size tree automaton A_1 accepting the triples $\langle t, u, v\rangle$ such that u and v are reachable from t, and another one A_2 accepting the triples $\langle t, u, v\rangle$ such that u and v are joinable. Then, roughly, deciding confluence amounts to checking the inclusion $A_1 \subseteq A_2$, which is in EXPTIME.

Since these ideas were developed, it has been a well-known open question whether this bound is optimal (see, e.g., [RTA01]), but until last year no algorithms better than PSPACE and EXPTIME had been found for ground string and term rewrite systems, respectively, and no hardness results for these complexity classes were found either.

In [CGN01] we gave the first polynomial-time algorithm for deciding the confluence of ground term rewrite systems. Later, in [Tiw02], this result was extended using abstract congruence closure techniques to linear shallow term rewrite systems, i.e., TRS where no variable occurs twice in the same rule nor at depth greater than one. Here, we present a new and much simpler proof of the latter result, by combining the concept of top-stabilizability from [CGN01] with other ideas from [Tiw02].

2 Basic Notions

Let $\mathcal{F}$ be a (finite) set of function symbols with an arity function $arity\colon \mathcal{F} \to I\!N$. Function symbols f with $arity(f) = n$ are called n-*ary* symbols (when $n = 1$, one says *unary* and when $n = 2$, *binary*). If $arity(f) = 0$, then f is a *constant symbol*. Let $\mathcal{X}$ be a set of variable symbols. The set of terms over $\mathcal{F}$ and $\mathcal{X}$, denoted by $\mathcal{T}(\mathcal{F}, \mathcal{X})$, is the smallest set containing all constant and variable symbols such that $f(t_1, \ldots, t_n)$ is in $\mathcal{T}(\mathcal{F}, \mathcal{X})$ whenever $f \in \mathcal{F}$, $arity(f) = n$, and $t_1, \ldots, t_n \in \mathcal{T}(\mathcal{F}, \mathcal{X})$. A *position* is a sequence of positive integers. If p is a position and t is a term, then by $t|_p$ we denote the *subterm of* t *at position* p: we have $t|_\lambda = t$ (where λ denotes the empty sequence) and $f(t_1, \ldots, t_n)|_{i.p} = t_i|_p$

if $1 \leq i \leq n$ (and is undefined if $i > n$). We also write $t[s]_p$ to denote the term obtained by replacing in t the subterm at position p by the term s. For example, if t is $f(a, g(b, h(c)), d)$, then $t|_{2.2.1} = c$, and $t[d]_{2.2} = f(a, g(b, d), d)$. By $|s|$ we denote the *size* (number of symbols) of a term s: we have $|a| = 1$ if a is a constant symbol or a variable, and $|f(t_1, \ldots, t_n)| = 1 + |t_1| + \ldots + |t_n|$. The *height* (or *depth*) of a term s is denoted by $height(s)$ and is defined as: $height(a) = 1$ if a is a constant symbol or a variable and $height(f(t_1, \ldots, t_n)) = 1 + max(height(t_1), \ldots, height(t_n))$.

If $\rightarrow$ is a binary relation on a set S, then $\rightarrow^+$ is its transitive closure and $\rightarrow^*$ is its reflexive-transitive closure. Two elements s and t of S are called *joinable* by $\rightarrow$, denoted $s \downarrow t$, if there exists a u in S such that $s \rightarrow^* u$ and $t \rightarrow^* u$.

The relation $\rightarrow$ is called *confluent* or *Church-Rosser* if the relation $\leftarrow^* \circ \rightarrow^*$ is contained in $\rightarrow^* \circ \leftarrow^*$, that is, for all s, t_1 and t_2 in S, if $s \rightarrow^* t_1$ and $s \rightarrow^* t_2$, then $t_1 \downarrow t_2$. An equivalent definition of confluence of $\rightarrow$ is that $\leftrightarrow^*$ is contained in $\rightarrow^* \circ \leftarrow^*$, that is, all s and t in S such that $s \leftrightarrow^* t$ are joinable. (If $\rightarrow$ is confluent by the latter definition, then it is trivially also confluent by the former one; the reverse implication follows by a simple induction on the number of $\leftarrow^* \circ \rightarrow^*$ patterns in the proof $s \leftrightarrow^* t$.)

A *substitution* σ is a mapping from variables to terms. It can be extended to a function from terms to terms in the usual way: using a postfix notation, $t\sigma$ denotes the result of simultaneously replacing in t every $x \in Dom(\sigma)$ by $x\sigma$. Substitutions are sometimes written as finite sets of pairs $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$, where each x_i is a variable and each t_i is a term. For example, if σ is $\{x \mapsto f(b, y), y \mapsto a\}$, then $g(x, y)\sigma$ is $g(f(b, y), a)$.

A *rewrite rule* is pair of terms (l, r), denoted by $l \rightarrow r$, with left-hand side (lhs) l and right-hand side (rhs) r. A *term rewrite system* (TRS) R is a finite set of rewrite rules. We say that s rewrites to t in one step at position p (by R) if $s|_p = l\sigma$ and $t = s[r\sigma]_p$, for some $l \rightarrow r \in R$ and subsitution σ. If $p = \lambda$, then the rewrite step is said to be applied *at the topmost position*. The rewrite relation $\rightarrow_R$ induced by R on $\mathcal{T}(\mathcal{F}, \mathcal{X})$ is defined by $s \rightarrow_R t$ if $s \rightarrow_{R,p} t$ for some position p.

A *(rewrite) derivation* from s is a sequence of rewrite steps starting from s, that is, a sequence $s \rightarrow_R s_1 \rightarrow_R s_2 \rightarrow_R \ldots$. A TRS R is said to be confluent if the relation $\rightarrow_R$ is confluent. The *size* $|R|$ of a TRS R of the form $\{l_1 \rightarrow r_1, \ldots, l_n \rightarrow r_n\}$ is $|l_1| + |r_1| + \ldots + |l_n| + |r_n|$.

Definition 1. *A term t is called:*

- linear *if no variable occurs more than once in t.*
- shallow *if no variable occurs in t at depth greater than 1, i.e., if $t|_p$ is a variable, then p is a position of length zero or one.*
- flat *if t is a non-constant term of the form $f(s_1, \ldots, s_n)$ where all s_i are variables or constants.*

Definition 2. *A term rewrite system R is called* shallow *if all its sides are shallow and is called* flat *if all its sides are either flat or constants. It is called*

linear *if for all its rules $l \to r$, the term $f(l, r)$ is linear, i.e., no variable occurs more than once in each rule.*

Definition 3. *let R be a TRS.*
A term s is reachable *from t by R if $t \to_R^* s$.*
Two terms s and t are equivalent *by R if $s \leftrightarrow_R^* t$.*
Two terms s and t are joinable *by R if $s \downarrow_R t$.*

Theorem 1 (see, e.g., [Nie98], Theorem 4.4). *The following problem is decidable in polynomial time:*

Input: a shallow TRS R and two terms s and t
Question: are s and t equivalent by R?

Theorem 2 (see, e.g., [CDG$^+$01]). *The following two problems are decidable in polynomial time:*

Input: a linear TRS R and two terms s and t
Question: is t reachable from s by R?

Input: a linear TRS R and two terms s and t
Question: are s and t joinable by R?

3 Confluence of Linear Shallow TRS

In this section we first give a polynomial-time confluence-preserving procedure that transforms any linear shallow TRS into a linear flat TRS R such that each rule of R has at least one constant side. It is based on the transformation of Section 3.2 of [CGN01].

Lemma 1. *Any linear shallow term rewrite system R (over $\mathcal{T}(\mathcal{F}, \mathcal{X})$) can be transformed in polynomial time into a linear flat term rewrite system R' over $\mathcal{T}(\mathcal{F} \cup \mathcal{K}, \mathcal{X})$, where $\mathcal{K}$ is a finite set of constants disjoint from $\mathcal{F}$, such that*

- *if $l \to r \in R'$, then either l or r is a constant,*
- *R is confluent over $\mathcal{T}(\mathcal{F}, \mathcal{X})$ if and only if R' is confluent over $\mathcal{T}(\mathcal{F} \cup \mathcal{K}, \mathcal{X})$.*

Proof. First, one can flatten R by repeatedly applying the following transformation step:

$$R_1 \Rightarrow R_2 \cup \{c \to t, t \to c\} \qquad (\textit{Constant introduction and replacement})$$

where t is a non-constant term occurring in R_1 at depth 1 or more, c is a new constant symbol not occurring anywhere in R_1, and R_2 is obtained by replacing all occurrences of t in R_1 by c. Similar transformations are used as well in fast algorithms for congruence closure [DST80,NO80,Sho84].

In order to see that such steps preserve confluence, first consider only introducing the rules $c \to t$ and $t \to c$, and note that if c is a new constrant, for any R we have that $\mathcal{T}(\mathcal{F}, \mathcal{X})$ with the relation $\to_R^*$ is isomorphic to the

congruence classes generated by $\{s \to t, t \to c\}$ on $T(\mathcal{F} \cup \{c\}, X)$ with the relation $\to^*_{R \cup \{c \to t, t \to c\}}$. Second, for all terms s and t we have $\to^*_{R_1 \cup \{s \to t, t \to s\}} = \to^*_{R_2 \cup \{s \to t, t \to s\}}$ if R_2 is obtained from R_1 by replacing somewhere s by t, and hence confluence is preserved as well. By considering a polynomial sequence of such constant introduction and replacement steps we can obtain a flat TRS.

By a similar transformation process, one can enforce the condition that at least one side of each rule is a constant:

$$R \cup \{l \to r\} \Rightarrow R \cup \{l \to c, c \to r\} \qquad (\textit{Rule splitting})$$

where l and r are non-constant flat terms and c is a new constant symbol not occurring anywhere in R, l or r. By similar reasoning as before, again each such steps preserve confluence. □

As a consequence of the previous lemma, in the following we can restrict our attention to flat linear TRS where every rule has at least one constant side. We can also assume that no side of the TRS is a variable: a variable lhs makes it trivially confluent; and if there is some variable rhs, but no variable lhs, then it is trivially non-confluent.

Definition 4. *Let R be a TRS. A term t is* top-stable *w.r.t. R if there exists no derivation from t containing a rewrite step at the top, i.e., there exists no derivation of the form $t \to_R \ldots \to_R t' \to_{R,\lambda} t''$.*

Definition 5. *Let R be a TRS.*

1. *A non-constant term $f(s_1, \ldots, s_n)$ is* top-stabilizable *w.r.t. R if there exist terms $s'_1, \ldots, s'_n$ such that $s_i \leftrightarrow^*_R s'_i$ and $f(s'_1, \ldots, s'_n)$ is top-stable.*
2. *A constant c is* top-stabilizable *w.r.t. R if there exists a top-stable non-constant term s such that $c \leftrightarrow^*_R s$.*

Theorem 3. *Let R be a linear flat TRS where each rule of R has at least one constant side. The set of constants and flat terms that are top-stabilizable w.r.t. R is computable in polynomial time.*

Proof. In the following fixpoint construction, which is based on the polynomial tests for reachability and equivalence, c denotes a constant, and s, t, and l denote flat terms of the form $f(s_1, \ldots, s_n)$, $f(t_1, \ldots, t_n)$, and $f(l_1, \ldots, l_n)$, respectively:

$$\begin{aligned}
S_0 &= \emptyset \\
S_{j+1} &= S_j \\
&\quad \cup \{\, c \mid \exists\, s \in S_j \text{ s.t. } c \leftrightarrow^*_R s \,\} && (FP1)\\
&\quad \cup \{\, s \mid \exists\, t \in S_j \text{ s.t. } s_i \leftrightarrow^*_R t_i \text{ for all } i \text{ in } 1..n \,\} && (FP2)\\
&\quad \cup \{\, s \mid \forall\, l \to r \in R \quad \exists\, i \text{ s.t. } l_i \text{ constant, and } s_i \in S_j \text{ or } s_i \not\to^*_R l_i \,\} && (FP3)
\end{aligned}$$

For a given (fixed; see the next section) signature there are polynomially many constants and flat terms. Hence a fixpoint S_k is reached after polynomially many iterations.

We first show, by induction on the index j, that every term in S_k is top-stabilizable. Suppose S_j contains only top-stabilizable terms. Terms added to S_{j+1} by rules (FP1) and (FP2) are clearly top-stabilizable. Now, let s be added to S_{j+1} by (FP3). By induction hypothesis, for each s_i that is in S_j there is a top-stable, non-constant s'_i such that $s_i \leftrightarrow_R^* s'_i$. Consider the term s' of the form $f(s'_1, \ldots, s'_n)$ obtained from s by replacing all such s_i by its corresponding s'_i, and not replacing the remaining s_i. Now s is top-stabilizable since s' is top-stable: no rule is applicable at the top to any term reachable from s', because for each rule $f(l_1, \ldots, l_n) \to r$ in R there is an i in $1..n$ such that l_i is a constant, and either s'_i is a top-stable non-constant, or else $s'_i \not\to_R^* l_i$.

We next prove, by contradiction, that S_k contains all top-stabilizable constants and flat terms in $\mathcal{T}(\mathcal{F}, \mathcal{X})$. Assume that there exists some top-stable (not necessarily flat!) non-constant term u of the form $f(u_1, \ldots, u_n)$ such that (i) for some c not in S_k we have $c \leftrightarrow_R^* u$ or (ii) for some flat term $s \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ not in S_k, for all i we have $s_i \leftrightarrow_R^* u_i$. We choose such a u to be minimal in size, and, in case we have a choice between cases (i) and (ii) with the same minimal size of u, we choose case (ii).

We first deal with case (i). We can split $c \leftrightarrow_R^* u$ into two parts $c \leftrightarrow_R^* f(v_1, \ldots, v_n) \leftrightarrow_R^* u$, where the second part is the maximal suffix with no step at the top. Hence there exists a side of a rule $f(l_1, \ldots, l_n)$ of which $f(v_1, \ldots, v_n)$ is an instance. Now consider the term u' of the form $f(u'_1, \ldots, u'_n)$ where each u'_i is l_i if l_i is a variable and where u'_i is u_i if l_i is a constant. Then we have $c \leftrightarrow_R^* f(l_1, \ldots, l_n) \leftrightarrow_R^* u'$ where u' is top-stable and $|u| \geq |u'|$. By our assumption on the choice of u and c, it follows that $f(l_1, \ldots, l_n)$ is in S_k. But then by (FP1) $c \in S_k$.

For case (ii), we can assume that each constant or variable u_i is equal to the corresponding s_i (otherwise, after replacing all such s_i by u_i in s the resulting s is top-stabilizable with the same u and is not in S_k due to (FP2)). Since u is top-stable, for every rule $f(l_1, \ldots, l_n) \to r$ there is some constant l_i such that (a) u_i is a variable or a constant s.t. $u_i \not\to_R^* l_i$, or (b) u_i is top-stable (otherwise, since each rule has a constant side, it can be rewritten into a constant, producing a smaller top-stable counterexample u). In case (b), by our minimality assumptions, s_i is a constant in S_k. In both cases s_i is in S_k or $s_i \not\to_R^* l_i$. Then by (FP3), s is in S_k: contradiction. □

Together with Theorems 1 and 3 and Lemma 1, the following theorem shows that the confluence of linear shallow TRS is decidable in polynomial time. This result was proved for the first time in [Tiw02]. The proof we give here combines the ideas about top-stabilizability from [CGN01] with other ideas from [Tiw02].

Theorem 4. *Let R be a linear flat TRS such that each rule of R has at least one constant side. Then R is confluent if, and only if, the following four conditions hold:*

1. *All constants c and d with $c \leftrightarrow_R^* d$ are joinable.*
2. *If s is a top-stabilizable side of R, then s is ground.*

3. *If* $f(s_1, \ldots, s_n)$ *and* $g(t_1, \ldots, t_m)$ *are top-stabilizable sides of* R *such that* $f(s_1, \ldots, s_n) \leftrightarrow_R^* g(t_1, \ldots, t_m)$*, then* $f = g$ *(and hence* $n = m$*) and* $s_i \leftrightarrow_R^* t_i$ *for all* i *in* $\{1, \ldots, n\}$*.*
4. *If* c *is a top-stabilizable constant then* $c \to_R^* f(s_1, \ldots, s_n)$ *for some top-stabilizable side* $f(s_1, \ldots, s_n)$ *of* R*.*

Proof. The left-to-right implication is straightforward:

1. If R is confluent then all terms s and t with $s \leftrightarrow_R^* t$ are joinable.
2. Let s be a non-ground top-stabilizable side of R, c be the constant on the other side of s in the rule in R, and s' be the top-stable term corresponding to s. Clearly, s' is non-ground (otherwise, by definition of top-stabilizability, some variable in s would be congruent to a ground subterm of s'). Therefore, let x be a variable in s'. Then, the terms s' and $s'\sigma$, where σ renames x to a new variable y, are top-stable terms equivalent by R (both are equivalent to c). If R is confluent, then x and y are joinable by R, a contradiction.
3. The terms $f(s_1, \ldots, s_n)$ and $g(t_1, \ldots, t_m)$ are top-stabilizable, i.e., there exist top-stable terms $f(s'_1, \ldots, s'_n)$ and $g(t'_1, \ldots, t'_m)$ such that $s_i \leftrightarrow_R^* s'_i$ and $t_i \leftrightarrow_R^* t'_i$ for all i. Since R is confluent and $f(s'_1, \ldots, s'_n) \leftrightarrow_R^* g(t'_1, \ldots, t'_m)$, the top-stable terms $f(s'_1, \ldots, s'_n)$ and $g(t'_1, \ldots, t'_m)$ are joinable. This can only be the case if $f = g$ and if the s'_i and t'_i are pairwise joinable for all i. This implies that $s_i \leftrightarrow_R^* t_i$ for all i in $\{1, \ldots, n\}$.
4. The constant c is top-stabilizable, i.e., there exists a top-stable $f(t_1, \ldots, t_n)$ such that $c \leftrightarrow_R^* f(t_1, \ldots, t_n)$. By confluence, c and $f(t_1, \ldots, t_n)$ are joinable: they both rewrite into a term u. Since $f(t_1, \ldots, t_n)$ is top-stable, u must be of the form $f(u_1, \ldots, u_n)$. Then the last topmost step in the derivation $c \to_R^* f(u_1, \ldots, u_n)$ must be with a rule of the form $l \to f(s_1, \ldots, s_n)$, where $f(s_1, \ldots, s_n)$ is the top-stabilizable side of R we were looking for.

For the right-to-left implication, assume that the four conditions hold but R is not confluent. Let (s, t) be a counterexample to confluence, i.e. $s \leftrightarrow_R^* t$ but $s \not\downarrow_R t$, where $height(s) + height(t)$ is minimal. We have the following cases:

- Either s or t is a variable.
  This cannot happen since variables are only congruent to themselves by R.
- Both s and t are constants.
  Then case 1 leads to a contradiction.
- s is a constant c and t is of the form $f(t_1, \ldots, t_n)$.
  Then t must be top-stable, since otherwise (because all rules have a constant side) t could be rewritten into some constant, which would contradict the minimality assumption.
  Then $c \leftrightarrow_R^* t$ can be written $c \leftrightarrow_R^* f(t'_1, \ldots, t'_n) \leftrightarrow_R^* f(t_1, \ldots, t_n)$, where the second part is the longest possible suffix with no steps at the top. Hence $f(t'_1, \ldots, t'_n)$ is a top-stabilizable instance of a side $f(l_1, \ldots, l_n)$ of R. In fact, w.l.o.g. we can assume that $f(t'_1, \ldots, t'_n)$ is itself a side of R: otherwise, some l_i is a variable, and we can replace this t'_i by l_i in $f(t'_1, \ldots, t'_n)$ and also t_i by l_i in t and omit all steps of the subderivation $t'_i \leftrightarrow_R t_i$ in

$f(t'_1, \ldots, t'_n) \leftrightarrow^*_R f(t_1, \ldots, t_n)$. Note that this replacement in t does not make the counterexample any larger, and that it does not affect the non-joinability of c and t, nor the top-stability of t.
Hence we have $c \leftrightarrow^*_R f(t'_1, \ldots, t'_n) \leftrightarrow^*_R f(t_1, \ldots, t_n)$, and we can assume, using additionally condition 2, that $f(t'_1, \ldots, t'_n)$ is a ground top-stabilizable side of R such that $c \leftrightarrow^*_R f(t'_1, \ldots, t'_n)$ and $t'_i \leftrightarrow^*_R t_i$ for all i in $\{1, \ldots, n\}$. Since c is top-stabilizable, by conditions 2 and 4, we have $c \rightarrow^*_R g(s_1, \ldots, s_m)$ for some ground top-stabilizable side $g(s_1, \ldots, s_m)$.
By condition 3 we have $f = g$ and $s_i \leftrightarrow^*_R t'_i$, and hence $s_i \leftrightarrow^*_R t_i$, for all i in $\{1, \ldots, n\}$. This contradicts our minimality assumption: $f(s_1, \ldots, s_n) \not\downarrow_R f(t_1, \ldots, t_n)$ implies that $s_i \not\downarrow_R t_i$ for some i in $\{1, \ldots, n\}$ where $height(s_i) = 1$ and $height(t_i) < height(t)$.

– s and t are of the form $f(s_1, \ldots, s_n)$ and $g(t_1, \ldots, t_m)$.
Either $f \neq g$ or for some i we have $s_i \not\leftrightarrow^*_R t_i$, since otherwise we would have $s_j \leftrightarrow^*_R t_j$ for some non-joinable s_j and t_j, contradicting our minimality assumption. Therefore $f(s_1, \ldots, s_n) \leftrightarrow^*_R c \leftrightarrow^*_R g(t_1, \ldots, t_m)$ for some constant c. Moreover, both terms are top-stable (as in the previous case, because all rules have a constant side: otherwise they could be rewritten into some constant, which would contradict the minimality assumptions).
Considering the leftmost step at the top in $f(s_1, \ldots, s_n) \leftrightarrow^*_R c$, we have $f(s_1, \ldots, s_n) \leftrightarrow^*_R f(s'_1, \ldots, s'_n) \leftrightarrow_\lambda \leftrightarrow^*_R c$. Doing similarly with $g(t_1, \ldots, t_m)$, we know that there exist top-stabilizable $f(s'_1, \ldots, s'_n)$ and $g(t'_1, \ldots, t'_m)$ that are instances of sides, such that $f(s'_1, \ldots, s'_n) \leftrightarrow^*_R c \leftrightarrow^*_R g(t'_1, \ldots, t'_m)$ and $s_i \leftrightarrow^*_R s'_i$ and $t_i \leftrightarrow^*_R t'_i$ for all i in $\{1, \ldots, n\}$. Reasoning as in the previous case, w.l.o.g. we can assume that $f(s'_1, \ldots, s'_n)$ and $g(t'_1, \ldots, t'_m)$ are not only instances of sides, but that they are the sides themselves, which are also ground.
Then, by condition 3, $f = g$ and $s'_i \leftrightarrow^*_R t'_i$, which is a contradiction with the fact that either $f \neq g$ or $s_i \not\leftrightarrow^*_R t_i$ for some i. □

Theorem 5. *The confluence of linear shallow term rewrite systems is decidable in polynomial time.*

4 We need to consider a fixed signature

All along this article, we have assumed that all terms and the TRS R are built over a fixed signature $\mathcal{F}$, which is not part of the input of the confluence problem. Indeed, if the arities of the input symbols are not bounded, we have the following result.

Theorem 6. *The following problem is NP-hard:*
Input: a signature $\mathcal{F}$, a shallow TRS R over $\mathcal{F}$, and two terms s and t
Question: is s reachable from t by R?
Similarly, the joinability and confluence problems are NP-hard when $\mathcal{F}$ is part of the input.

Proof. By reducing the 3-SAT problem. Here we only give the proof for the reachability case. Given a 3-SAT instance P with variables $x_1, \ldots, x_n$ and clauses $c_1, \ldots, c_m$, we create a TRS R_P built over an m-ary symbol f, an n-ary symbol g, and constants $a, b, c, 0, 1, c_1, \ldots, c_m$.

The TRS R_P contains the rules $a \to f(c_1, \ldots, c_m)$, $c \to 0$, $c \to 1$, and $f(x, \ldots, x) \to b$, and, for each clause c_i of P, for each negative (positive) variable x_j in it there is a rule $c_i \to g(c, \ldots, c, 0, c, \ldots, c)$ (repectively $c_i \to g(c, \ldots, c, 1, c, \ldots, c)$) where the 0 or 1 is at position j in the rhs. It is not hard to see that P is satisfiable iff b is reachable from a by R_P. □

5 Conclusions and further work

We have given an –in our opinion surprisingly simple– polynomial-time decision procedure for the confluence of linear shallow TRS. Our result stricly subsumes the previous work of [CGN01] on variable-free TRS, which was already a longstanding open problem [RTA01].

We also believe that our simpler proof techniques will open the door to polynomial algorithms for more general classes of TRS. In particular, the restriction that both sides of the rules do not share any variables could perhaps be dropped, i.e., our techniques could perhaps deal with shallow TRS where in each rule both sides are linear terms. In the other direction, one could consider arbitrary shallow TRS where both sides do not share any variables.

Finally, using the techniques introduced here, it might be possible to prove the decidability of confluence for arbitrary TRS where both sides do not share any variables; this includes the so-called right ground TRS, another well-known open problem.

References

[BO93] R. Book and F. Otto. *String Rewriting Systems.* Springer-Verlag Inc., New York, NY, USA, 1993.

[CDG+01] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree Automata Techniques and Applications. This electronic book is available at `http://www.grappa.univ-lille3.fr/tata`, 2001.

[CGN01] Hubert Comon, Guillem Godoy, and Robert Nieuwenhuis. The confluence of ground term rewrite systems is decidable in polynomial time. In *42nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2001.

[DHLT90] Max Dauchet, Thierry Heuillard, Pierre Lescanne, and Sophie Tison. Decidability of the confluence of finite ground term rewrite systems and of other related term rewrite systems. *Information and Computation*, 88(2):187–201, October 1990.

[DJ90] Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, chapter 6, pages 244–320. Elsevier Science Publishers B.V., Amsterdam, New York, Oxford, Tokyo, 1990.

[DP01] Nachum Dershowitz and David Plaisted. Rewriting. In J.A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*. Elsevier Science Publishers and MIT Press, 2001.

[DST80] Peter J. Downey, Ravi Sethi, and Robert E. Tarjan. Variations on the common subexpressions problem. *J. of the Association for Computing Machinery*, 27(4):758–771, 1980.

[DTHL87] Max Dauchet, Sophie Tison, Thierry Heuillard, and Pierre Lescanne. Decidability of the confluence of ground term rewriting systems. In *Proceedings, Symposium on Logic in Computer Science*, pages 353–359, Ithaca, New York, 22–25 June 1987. The Computer Society of the IEEE.

[KB70] D.E. Knuth and P.B. Bendix. Simple word problems in universal algebras. In *J. Leech, ed., Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, Oxford, 1970.

[KNO90] Deepak Kapur, Paliath Narendran, and Friedrich Otto. On ground confluence of term rewriting systems. *Inform. Comput.*, 86(1):14–31, 1990.

[Nie98] Robert Nieuwenhuis. Decidability and complexity analysis by basic paramodulation. *Information and Computation*, 147:1–21, 1998.

[NO80] Greg Nelson and Derek C. Oppen. Fast decision procedures bases on congruence closure. *Journal of the Association for Computing Machinery*, 27(2):356–364, April 1980.

[Ott99] F. Otto. On the connections between rewriting and formal language theory. In P. Narendran and M. Rusinowitch, editors, *Tenth International Conference on Rewriting Techniques and Applications (RTA)*, LNCS 1631, pages 332–355, Trento, Italy, July 2–4, 1999. Springer-Verlag.

[Oya87] M. Oyamaguchi. The Church-Rosser property for ground term-rewriting systems is decidable. *Theoretical Computer Science*, 49(1):43–79, 1987.

[RTA01] RTA-LOOP. Problem #12, posed by Wayne Snyder in 1991. Int. Conf. on Rewriting Techniques and Applications, The list of open problems, 2001. Maintained at `http://www.lri.fr/~rtaloop/` (by R. Treinen).

[Sho84] Robert E. Shostak. Deciding combinations of theories. *Journal of the ACM*, 31(1):1–12, January 1984.

[Tiw02] A. Tiwari. Deciding confluence of certain term rewriting systems in polynomial time. In Gordon Plotkin, editor, *IEEE Symposium on Logic in Computer Science, LICS 2002*, volume xxx of *xxx*, page xxx. IEEE Society, July 2002.