# Rigid Reachability[⋆]

Harald Ganzinger, Florent Jacquemard, and Margus Veanes

Max-Planck-Institut für Informatik
Im Stadtwald, 66123 Saarbrücken, Germany

**Abstract.** We show that rigid reachability, the non-symmetric form of rigid $E$-unification, is undecidable already in the case of a single constraint. From this we infer the undecidability of a new rather restricted kind of second-order unification. We also show that certain decidable subclasses of the problem which are $\mathcal{P}$-complete in the equational case become EXPTIME-complete when symmetry is absent. By applying automata-theoretic methods, simultaneous monadic rigid reachability with ground rules is shown to be in EXPTIME.

## 1 Introduction

Rigid reachability is the problem, given a rewrite system $R$ and two terms $s$ and $t$, whether there exists a ground substitution $\sigma$ such that $s\sigma$ rewrites in some number of steps via $R\sigma$ into $t\sigma$. The term "rigid" stems from the fact that for no rule more than one instance can be used in the rewriting process. Simultaneous rigid reachability is the problem in which a substitution is sought which simultaneously solves each member of a system of reachability constraints $(R_i, s_i, t_i)$. A special case of [simultaneous] rigid reachability arises when the $R_i$ are symmetric, containing for each rule $l \to r$ also its converse $r \to l$. The latter problem was introduced in [14] as "simultaneous rigid $E$-unification". (Symmetric systems $R$ arise, for instance, from orienting a given set of equations $E$ in both directions.) It has been shown in [8] that simultaneous rigid $E$-unification is undecidable, whereas the non-simultaneous case with just one rigid equation to solve is NP-complete [13]. The main result in this paper is that for non-symmetric rigid reachability already the case of a single reachability constraint is undecidable, even when the rule set is ground. From this we infer undecidability of a rather restricted form of second-order unification for problems which contain just a single second-order variable which, in addition, occurs at most twice in the unification problem. The latter result contrasts a statement in [19] to the opposite.

The absence of symmetry makes the problem much more difficult. This phenomenon is also observed in decidable cases which we investigate in the second part of the paper. For instance we prove that a certain class of rigid problems which is $\mathcal{P}$-complete in the equational case becomes EXPTIME-complete when symmetry is absent. Our results demonstrate a very thin borderline between the decidable and the undecidable fragments of rigid reachability with respect to several syntactical criteria. In particular, for ground $R$ and variable-disjoint $s$ and

[⋆] A full version of this paper is available as MPI-I Research Report MPI-I-98-2-013.

$t$, the problem is undecidable, whereas it becomes *decidable* when, in addition, either $s$ or $t$ is linear.

In the Section 6 we will apply automata-theoretic methods to the monadic case and establish an EXPTIME upper bound for monadic simultaneous rigid reachability for ground rewrite systems. This generalizes the analogous result of [17] for simultaneous rigid $E$-unification. Also, our proof is more direct and provides a better upper bound, closer to the PSPACE lower bound given in [17]. A PSPACE upper bound for this problem has been proved more recently in a joint work with Cortier [4].

## 2 Preliminaries

A *signature* $\Sigma$ is a collection of *function symbols* with fixed arities $\geq 0$ and, unless otherwise stated, $\Sigma$ is assumed to contain at least one *constant*, that is, one function symbol with arity 0. The set of all constants in $\Sigma$ is denoted by $Con(\Sigma)$. We use $a, b, c, d, a_1, \ldots$ for constants and $f, g, f_1, \ldots$ for function symbols in general. A designated constant in $\Sigma$ is denoted by $c_\Sigma$.

A *term language* or simply *language* is a triple $L = (\Sigma_L, \mathcal{X}_L, \mathcal{F}_L)$ where (i) $\Sigma_L$ is a signature, (ii) $\mathcal{X}_L$ $(x, y, x_1, y_1, \ldots)$ is a collection of first-order variables, and (iii) $\mathcal{F}_L$ $(F, G, F_1, F', \ldots)$ is a collection of symbols with fixed arities $\geq 1$, called *second-order variables*. The various sets of symbols are assumed to be pairwise disjoint. Let $L$ be a language. $L$ is *first-order*, if $\mathcal{F}_L$ is empty; $L$ is *second-order*, otherwise. $L$ is *monadic* if all function symbols in $\Sigma_L$ have arity $\leq 1$. The set of all terms in a language $L$, or *$L$-terms*, is denoted by $\mathcal{T}_L$. We use $s, t, l, r, s_1, \ldots$ for terms. We usually omit mentioning $L$ when it is clear from the context. The set of first-order variables of a term $t$ is denoted by $Var(t)$. A *ground* term is one that contains no variables. The set of all ground terms in a language $L$ is denoted by $\mathcal{T}_{\Sigma_L}$. A term is called *shallow* if all variables in it occur at depth $\leq 1$. The size $\|t\|$ of a term $t$ is defined recursively by: $\|t\| = 1$ if $t \in \mathcal{X}_L \cup Con(\Sigma_L)$ and $\|f(t_1, \ldots, t_n)\| = \|t_1\| + \ldots + \|t_n\| + 1$ when $f \in \Sigma_L \cup \mathcal{F}_L$.

We assume that the reader is familiar with the basic concepts in term rewriting [9, 1]. We write $u[s]$ when $s$ occurs as a subterm of $u$. In that case $u[t]$ denotes the replacement of the indicated occurrence of $s$ by $t$. An *equation in $L$* is an unordered pair of $L$-terms, denoted by $s \approx t$. A *rule in $L$* is an ordered pair of $L$-terms, denoted by $s \rightarrow t$. An equation or a rule is *ground* if the terms in it are ground. A *system* is a finite set. Let $R$ be a system of ground rules, and $s$ and $t$ two ground terms. Then $s$ *rewrites* in $R$ to $t$, denoted by $s \xrightarrow[R]{} t$, if $t$ is obtained from $s$ by replacing an occurrence of a term $l$ in $s$ by a term $r$ for some rule $l \rightarrow r$ in $R$. The term $s$ *reduces* in $R$ to $t$, denoted by $s \xrightarrow[R]{*} t$, if either $s = t$ or $s$ rewrites to a term that reduces to $t$. $R$ is called *symmetric* if, with any rule $l \rightarrow r$ in $R$, $R$ also contains its converse $r \rightarrow l$. Below we shall not distinguish between systems of equations and symmetric systems of rewrite rules. The *size* of a system $R$ is the sum of the sizes of its components: $\|R\| = \sum_{l \rightarrow r \in R}(\|l\| + \|r\|)$.

*Rigid Reachability.* Let $L$ be a first-order language. A *reachability constraint*, or simply a constraint in $L$ is a triple $(R, s, t)$ where $R$ is a set of rules in $L$, and $s$ and

$t$ are terms in $L$. We refer to $R$, $s$ and $t$ as the *rule set*, the *source term* and the *target term*, respectively, of the constraint. A substitution $\theta$ in $L$ *solves* $(R, s, t)$ (in $L$) if $\theta$ is grounding for $R$, $s$ and $t$, and $s\theta \xrightarrow[R\theta]{*} t\theta$. The problem of solving constraints (in $L$) is called *rigid reachability* (for $L$). A system of constraints is *solvable* if there exists a substitution that solves all constraints in that system. *Simultaneous rigid reachability* or *SRR* is the problem of solving systems of constraints. *Monadic* (simultaneous) rigid reachability is (simultaneous) rigid reachability for monadic languages.

*Rigid E-unification* is rigid reachability for constraints $(E, s, t)$ with sets of equations $E$. *Simultaneous Rigid E-unification* or *SREU* is defined accordingly.

*Finite Tree Automata.* Finite bottom-up tree automata, or simply, tree automata, from here on, are a generalization of classical automata [10, 22]. Using a rewrite rule based definition [3, 5], a *tree automaton* (or *TA*) $A$ is a quadruple $(Q_A, \Sigma_A, R_A, F_A)$, where (i) $Q_A$ is a finite set of constants called *states*, (ii) $\Sigma_A$ is a *signature* that is disjoint from $Q_A$, (iii) $R_A$ is a system of *rules* of the form $f(q_1, \ldots, q_n) \to q$, where $f \in \Sigma_A$ has arity $n \geq 0$ and $q, q_1, \ldots, q_n \in Q_A$, and (iv) $F_A \subseteq Q_A$ is the set of *final states*. The *size* of a TA $A$ is $\|A\| = |Q_A| + \|R_A\|$.

We denote by $T(A, q)$ the set $\{t \in \mathcal{T}_{\Sigma_A} \mid t \xrightarrow[R_A]{*} q\}$ of ground terms *accepted* by $A$ in state $q$. The set of terms *recognized* by the TA $A$ is the set $\bigcup_{q \in F_A} T(A, q)$. A set of terms is called *recognizable* or *regular* if it is recognized by some TA.

*Word automata.* In monadic signatures, every function symbol has an arity at most 1, thus terms are words. For monadic signatures, we thus use the traditional, equivalent concepts of alphabets, words, finite automata, and regular expressions. A word with a variable $a_1 \ldots a_n x$ corresponds to the term $a_1(a_2(\ldots a_n(x))) \in \mathcal{T}_\Sigma$. The substitution of $x$ by a term $u$ is the same as the concatenation of the respective words. A finite (word) automaton $A$ is a tuple $(Q_A, \Sigma_A, R_A, q^{\mathrm{i}}_A, F_A)$ where the components $Q_A$, $\Sigma_A$, $R_A$, $F_A$ have the same form and meaning as the corresponding components of a tree automaton over a monadic signature, and where, additionally, $q^{\mathrm{i}}_A$ is the initial state. A transition $a(q) \to q'$ of $R_A$ ($a \in \Sigma_A$, $q, q' \in Q_A$) is denoted $q \xrightarrow{a} q'$.

*Second-Order Unification.* Second-order unification is unification for second-order terms. For representing unifiers, we need expressions representing functions which, when applied, produce instances of a term in the given language $L$. Following Goldfarb [15] and Farmer [11], we, therefore, introduce the concept of an *expansion* $L^*$ of $L$. Let $\{z_i\}_{i \geq 1}$ be an infinite collection of new symbols not in $L$. The language $L^*$ differs from $L$ by having $\{z_i\}_{i \geq 1}$ as additional first-order variables, called *bound variables*. The *rank* of a term $t$ in $L^*$, is either 0 if $t$ contains no bound variables (*i.e.*, $t \in \mathcal{T}_L$), or the largest $n$ such that $z_n$ occurs in $t$. Given terms $t$ and $t_1, t_2, \ldots, t_n$ in $L^*$, we write $t[t_1, t_2, \ldots, t_n]$ for the term that results from $t$ by simultaneously replacing $z_i$ in $t$ by $t_i$ for $1 \leq i \leq n$. An $L^*$-term is called *closed* if it contains no variables other than bound variables. Note that closed $L^*$-terms of rank 0 are ground $L$-terms.

A *substitution in $L$* is a function $\theta$ with finite domain $\mathrm{dom}(\theta) \subseteq \mathcal{X}_L \cup \mathcal{F}_L$ that maps first-order variables to $L$-terms, and $n$-ary second-order variables to $L^*$-terms of rank $\leq n$. The result of applying a substitution $\theta$ to an $L$-term $s$, denoted by $s\theta$, is defined by induction on $s$:

1. If $s = x$ and $x \in \mathrm{dom}(\theta)$ then $s\theta = \theta(x)$.
2. If $s = x$ and $x \notin \mathrm{dom}(\theta)$ then $s\theta = x$.
3. If $s = F(t_1, \dots, t_n)$ and $F \in \mathrm{dom}(\theta)$ then $s\theta = \theta(F)[t_1\theta, \dots, t_n\theta]$.
4. If $s = F(t_1, \dots, t_n)$ and $F \notin \mathrm{dom}(\theta)$ then $s\theta = F(t_1\theta, \dots, t_n\theta)$.
5. If $s = f(t_1, \dots, t_n)$ then $s\theta = f(t_1\theta, \dots, t_n\theta)$.

We also write $F\theta$ for $\theta(F)$, where $F$ is a second-order variable. A substitution is called *closed*, if its range is a set of closed terms. Given a term $t$, a substitution $\theta$ is said to be *grounding for $t$* if $t\theta$ is ground, similarly for other $L$-expressions. Given a sequence $\boldsymbol{t} = t_1, \dots, t_n$ of terms, we write $\boldsymbol{t}\theta$ for $t_1\theta, \dots, t_n\theta$.

Let $E$ be a system of equations in $L$. A *unifier* of $E$ is a substitution $\theta$ (in $L$) such that $s\theta = t\theta$ for all equations $s \approx t$ in $E$. $E$ is *unifiable* if there exists a unifier of $E$. Note that if $E$ is unifiable then it has a closed unifier that is grounding for $E$, since $\mathcal{T}_{\Sigma_L}$ is nonempty. The *unification problem for $L$* is the problem of deciding whether a given equation system in $L$ is unifiable. In general, the *second-order unification* problem or *SOU* is the unification problem for arbitrary second-order languages. *Monadic* SOU is SOU for monadic second-order languages. By SOU *with one second-order variable* we mean the unification problem for second-order languages $L$ such that $|\mathcal{F}_L| = 1$.

Following common practice, by an *exponential* function we mean an integer function of the form $f(n) = 2^{P(n)}$ where $P$ is a polynomial. The complexity class EXPTIME is defined accordingly.

## 3 Rigid Reachability is Undecidable

We prove that rigid reachability is undecidable. The undecidability holds already for constraints with some fixed ground rule set which is, moreover, terminating. Our main tool in proving the undecidability result is the following statement.

**Lemma 1 ([16]).** *One can effectively construct two tree automata $A_{\mathrm{mv}} = (Q_{\mathrm{mv}}, \Sigma_{\mathrm{mv}}, R_{\mathrm{mv}}, \{q_{\mathrm{mv}}\})$, $A_{\mathrm{id}} = (Q_{\mathrm{id}}, \Sigma_{\mathrm{id}}, R_{\mathrm{id}}, \{q_{\mathrm{id}}\})$, and two canonical systems of ground rules $\Pi_1, \Pi_2 \subseteq \mathcal{T}_{\Sigma_{\mathrm{mv}}} \times \mathcal{T}_{\Sigma_{\mathrm{id}}}$, where the only common symbol in $A_{\mathrm{mv}}$ and $A_{\mathrm{id}}$ is a binary function symbol $\centerdot$,[1] such that, it is undecidable whether, given $t_{\mathrm{id}} \in \mathcal{T}_{\Sigma_{\mathrm{id}}}$, there exists $s \in T(A_{\mathrm{mv}})$ and $t \in T(A_{\mathrm{id}})$ such that $s \xrightarrow[\Pi_1]{*} t$ and $t_{\mathrm{id}} \centerdot s \xrightarrow[\Pi_2]{*} t$.*

The main idea behind the proof of Lemma 1 is illustrated in Figure 1. In the rest of this section, we consider fixed $A_{\mathrm{mv}}$, $A_{\mathrm{id}}$, $\Pi_1$ and $\Pi_2$ given by Lemma 1.

Undecidability of simultaneous rigid $E$-unification follows from this lemma by viewing the rules $R_{\mathrm{mv}}$ and $R_{\mathrm{id}}$ of the automata $A_{\mathrm{mv}}$ and $A_{\mathrm{id}}$, respectively, as

[1] We write $\centerdot$ ("dot") as an infix operator.

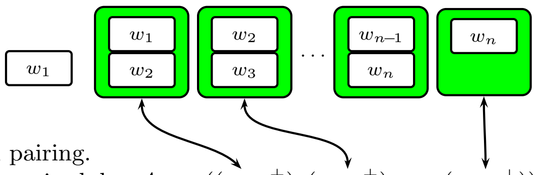

**Figure 1:** Shifted pairing.

The terms recognized by $A_{\mathrm{mv}}$, $((v_1, v_1^+), (v_2, v_2^+), \dots, (v_n, v_n^+))$, represent a sequence of moves of a given Turing machine, where $v_i^+$ is the successor of $v_i$ according to the transition function of the TM.

Each term $t$ recognized by $A_{\mathrm{id}}$ represents a sequence of IDs of the TM $(w_1, w_2, \dots, w_n)$.

The two rewrite systems $\Pi_1$ and $\Pi_2$ are such that $s$ reduces in $\Pi_1$ to $t$ if and only if $v_i = w_i$ for $1 \le i \le k = n$, and $t_{\mathrm{id}} \cdot s$ reduces in $\Pi_2$ to $t$ if and only if $t_{\mathrm{id}}$ represents $w_1$, $v_i^+ = w_{i+1}$ for $1 \le i < n$, and $w_n$ is the final ID of the TM. It follows that such $s$ and $t$ exist if and only if the TM accepts the input string represented by $t_{\mathrm{id}}$.

---

well as the rewrite systems $\Pi_1$ and $\Pi_2$, as sets of equations, and by formulating the reachability constraints between $s$ and $t$ as a system of rigid equations. It is *not* possible, though, to achieve the same effect by a *single* rigid $E$-unification constraint for a combined system of equations. The interference between the component systems cannot be controlled due to the symmetry of equality. This is different for reachability where rewrite rules are only applied from left to right. In fact, our main idea in the undecidability proof is to combine the four rewrite systems $R_{\mathrm{mv}}$, $R_{\mathrm{id}}$, $\Pi_1$, and $\Pi_2$ into a single system and achieve mutual non-overlapping of rewrite rules by renaming the constants in the respective signatures.

## 3.1 Renaming of Constants

For any integer $m$ and a signature $\Sigma$ we write $\Sigma^{(m)}$ for the constant-disjoint copy of $\Sigma$ where each constant $c$ has been replaced with a new constant $c^{(m)}$, we say that $c^{(m)}$ *has label* $m$. Note that non-constant symbols are not renamed. For a ground term $t$ and a set of ground rules $R$ over $\Sigma$, we define $t^{(m)}$ and $R^{(m)}$ over $\Sigma^{(m)}$ accordingly.

Given a signature $\Sigma$ and two different integers $m$ and $n$, we write $\Sigma^{(m,n)}$ for the following set of rules that simply replaces each label $m$ with label $n$:

$$\Sigma^{(m,n)} = \{\, c^{(m)} \to c^{(n)} \mid c \in \mathit{Con}(\Sigma) \,\}.$$

We write $\Pi^{(m,n)}$, where $\Pi$ is either $\Pi_1$ or $\Pi_2$, for the following set of rules:

$$\Pi^{(m,n)} = \{\, l^{(m)} \to r^{(n)} \mid l \to r \in \Pi \,\}.$$

**Lemma 2.** *Let $m$, $n$, $k$ and $l$ be distinct integers. The statements (i) and (ii) are equivalent for all all $s \in \mathcal{T}_{\Sigma_{\mathrm{mv}}}$ and $t_{\mathrm{id}}, t \in \mathcal{T}_{\Sigma_{\mathrm{id}}}$.*

*(i)* $s \xrightarrow[\Pi_1]{*} t$ *and* $t_{\mathrm{id}} \cdot s \xrightarrow[\Pi_2]{*} t$.

*(ii)* $s^{(m)} \xrightarrow[{\Pi_1}^{(m,n)}]{*} t^{(n)}$ *and* $t_{\mathrm{id}}^{(l)} \centerdot s^{(k)} \xrightarrow[{\Pi_2}^{(k,l)}]{*} t^{(l)}$.

*Proof.* The left-hand sides of the rules in $\Pi_1$ and $\Pi_2$ are terms in $\mathcal{T}_{\Sigma_{\mathrm{mv}}}$ and the right-hand sides of the rules in $\Pi_1$ and $\Pi_2$ are terms in $\mathcal{T}_{\Sigma_{\mathrm{id}}}$. But $\Sigma_{\mathrm{mv}}$ and $\Sigma_{\mathrm{id}}$ are constant-disjoint. ⊠

### 3.2 The Main Construction

Let $R_{\mathrm{u}}$ be the following system of ground rules:

$$\begin{aligned} R_{\mathrm{u}} = {} & R_{\mathrm{mv}}^{(0)} \cup R_{\mathrm{mv}}^{(2)} \cup {\Sigma_{\mathrm{mv}}}^{(0,1)} \cup {\Sigma_{\mathrm{mv}}}^{(2,1)} \cup R_{\mathrm{id}}^{(4)} \cup R_{\mathrm{id}}^{(6)} \cup {\Sigma_{\mathrm{id}}}^{(4,3)} \cup {\Sigma_{\mathrm{id}}}^{(6,3)} \cup \\ & {\Sigma_{\mathrm{id}}}^{(4,5)} \cup {\Pi_1}^{(0,5)} \cup {\Sigma_{\mathrm{id}}}^{(6,7)} \cup {\Pi_2}^{(2,7)} \end{aligned}$$

Note that constants with odd labels occur only in the right-hand sides of rules and can, once introduced, subsequently not be removed by $R_{\mathrm{u}}$. Let $f_{\mathrm{u}}$ be a new function symbol with arity 12. We consider the following constraint:

$$\left( \begin{array}{l} R_{\mathrm{u}}, f_{\mathrm{u}}(\ x_0,\ \ x_2,\ x_0, x_2,\ y_4,\ \ y_6,\ y_4, y_6, y_4, x_0, y_6, t_{\mathrm{id}}^{(7)} \centerdot x_2\ ), \\ \qquad f_{\mathrm{u}}(\ q_{\mathrm{mv}}^{(0)}, q_{\mathrm{mv}}^{(2)}, x_1, x_1, q_{\mathrm{id}}^{(4)}, q_{\mathrm{id}}^{(6)}, y_3, y_3, y_5, y_5, y_7,\ \ y_7\ ) \end{array} \right) \tag{1}$$

Our goal is to show that solvability of (1), for a given $t_{\mathrm{id}} \in \Sigma_{\mathrm{id}}$, is equivalent to the existence of $s$ and $t$ satisfying the condition in Lemma 1. Note that, for all ground terms $t_i$ and $s_i$, for $1 \leq i \leq 12$,

$$f_{\mathrm{u}}(t_1, \ldots, t_{12}) \xrightarrow[R_{\mathrm{u}}]{*} f_{\mathrm{u}}(s_1, \ldots, s_{12}) \quad \Leftrightarrow \quad t_i \xrightarrow[R_{\mathrm{u}}]{*} s_i \text{ (for } 1 \leq i \leq 12).$$

As a first step, we prove a lemma that allows us to separate the different subsystems of $R_{\mathrm{u}}$ that are relevant for the reductions between the corresponding arguments of $f_{\mathrm{u}}$ in the source term and the target term of (1).

**Lemma 3.** *For every substitution $\theta$, $\theta$ solves the constraint (1) if and only if $\theta$ solves the system (2)–(5) of constraints.*

$$\left.\begin{array}{llll} ( & R_{\mathrm{mv}}^{(0)}, & x_0, & q_{\mathrm{mv}}^{(0)} \ ) \\ ( & R_{\mathrm{mv}}^{(2)}, & x_2, & q_{\mathrm{mv}}^{(2)} \ ) \\ ( & {\Sigma_{\mathrm{mv}}}^{(0,1)}, & x_0, & x_1 \ ) \\ ( & {\Sigma_{\mathrm{mv}}}^{(2,1)}, & x_2, & x_1 \ ) \end{array}\right\} \tag{2}$$

$$\left.\begin{array}{llll} ( & R_{\mathrm{id}}^{(4)}, & y_4, & q_{\mathrm{id}}^{(4)} \ ) \\ ( & R_{\mathrm{id}}^{(6)}, & y_6, & q_{\mathrm{id}}^{(6)} \ ) \\ ( & {\Sigma_{\mathrm{id}}}^{(4,3)}, & y_4, & y_3 \ ) \\ ( & {\Sigma_{\mathrm{id}}}^{(6,3)}, & y_6, & y_3 \ ) \end{array}\right\} \tag{3}$$

$$\left.\begin{array}{llll} ( & {\Sigma_{\mathrm{id}}}^{(4,5)}, & y_4, & y_5 \ ) \\ ( & {\Pi_1}^{(0,5)}, & x_0, & y_5 \ ) \end{array}\right\} \tag{4}$$

$$\left.\begin{array}{llll} ( & {\Sigma_{\mathrm{id}}}^{(6,7)}, & y_6, & y_7 \ ) \\ ( & {\Pi_2}^{(2,7)}, & t_{\mathrm{id}}^{(7)} \centerdot x_2, & y_7 \ ) \end{array}\right\} \tag{5}$$

*Proof.* The direction '$\Leftarrow$' is immediate, since if $\theta$ solves a constraint $(R, s, t)$ then obviously it solves any constraint $(R', s, t)$ where $R \subseteq R'$.
The direction '$\Rightarrow$' can be proved by case analysis, many cases being symmetrical. For instance, we show that if $\theta$ solves (1), then $\theta$ solves the two first constraints of (2), namely $x_i\theta \xrightarrow[R^{(i)}_{\mathrm{mv}}]{*} q^{(i)}_{\mathrm{mv}}$ for $i = 0, 2$. We give the proof for $i = 0$, which, by symmetry, also proves the case $i = 2$. We know that $x_0\theta \xrightarrow[R_{\mathrm{u}}]{*} q^{(0)}_{\mathrm{mv}}$. We prove by induction on the length of reductions that, for all $t$, if $t \xrightarrow[R_{\mathrm{u}}]{*} q^{(0)}_{\mathrm{mv}}$ then $t \xrightarrow[R^{(0)}_{\mathrm{mv}}]{*} q^{(0)}_{\mathrm{mv}}$. The base case (reduction is empty) holds trivially. If the reduction is nonempty, then we have for some $l \to r \in R_{\mathrm{u}}$, and by using the induction hypothesis, that $t \xrightarrow[l \to r]{} s \xrightarrow[R^{(0)}_{\mathrm{mv}}]{*} q^{(0)}_{\mathrm{mv}}$. Therefore all constants in $r$ have label 0, since $r$ is a subterm of $s$ and $s \in \mathcal{T}_{\Sigma^{(0)}_{\mathrm{mv}} \cup Q^{(0)}_{\mathrm{mv}}}$. Hence $l \to r \in R^{(0)}_{\mathrm{mv}}$, and consequently $t \xrightarrow[R^{(0)}_{\mathrm{mv}}]{*} q^{(0)}_{\mathrm{mv}}$. $\boxtimes$

The following lemma relates the solvability of (1) to Lemma 1.

**Lemma 4.** *For $t_{\mathrm{id}} \in \mathcal{T}_{\Sigma_{\mathrm{id}}}$, the constraint (1) is solvable if and only if there exists $s \in T(A_{\mathrm{mv}})$ and $t \in T(A_{\mathrm{id}})$ such that $s \xrightarrow[\Pi_1]{*} t$ and $t_{\mathrm{id}} \centerdot s \xrightarrow[\Pi_2]{*} t$.*

*Proof.* ($\Leftarrow$) Assuming that given $s$ and $t$ exist, define $x_i\theta = s^{(i)}$ for $i \in \{0, 1, 2\}$ and $y_i\theta = t^{(i)}$ for $i \in \{3, 4, 5, 6, 7\}$. It follows easily from Lemma 2 and Lemma 3 that $\theta$ solves (1).

($\Rightarrow$) Assume that $\theta$ solves (1). By Lemma 3, $\theta$ solves (2)–(5). First we observe the following facts.
(i) From $\theta$ solving (2), it follows that there exists $s \in T(A_{\mathrm{mv}})$ such that $x_0\theta = s^{(0)}$ and $x_2\theta = s^{(2)}$.
(ii) From $\theta$ solving (3), it follows that there exists $t \in T(A_{\mathrm{id}})$ such that $y_4\theta = t^{(4)}$ and $y_6\theta = t^{(6)}$.
From $\theta$ solving (4) and by using (ii), it follows that $y_5\theta = t^{(5)}$. Now, due to the second component of (4) and by using (i), we get that: $s^{(0)} \xrightarrow[{\Pi_1}^{(0,5)}]{*} t^{(5)}$.
From $\theta$ solving (5) and by using (ii), it follows that $y_7\theta = t^{(7)}$. Now, due to the second component of (5) and by using (i), we get that: $t^{(7)}_{\mathrm{id}} \centerdot s^{(2)} \xrightarrow[{\Pi_2}^{(2,7)}]{*} t^{(7)}$.
Finally, use Lemma 2. $\boxtimes$

We conclude with the following result.

**Theorem 1.** *Rigid reachability is undecidable. Specifically, it is undecidable already under the following restrictions:*

- *the rule set is some fixed ground terminating rewrite system;*
- *there are at most eight variables;*
- *each variable occurs at most three times;*
- *the source term and the target term do not share variables.*

*Proof.* The undecidability follows from Lemma 1 and Lemma 4. The system $R_{\mathrm{u}}$ is easily seen to be terminating, by simply examining the subsystems. The other restrictions follow immediately as properties of (1). $\boxtimes$

We have not attempted to minimize the number of variables in (1). Observe also that all but one of the occurrences of variables are shallow (the target term is shallow).

## 4 A New Undecidability Result for SOU

We prove that SOU is undecidable already when unification problems contain just a single second-order variable which, in addition, occurs twice. This result contrasts a claim to the opposite in [19]. Let $\Sigma_{\mathrm{u}}$ be the signature consisting of the symbols in $R_{\mathrm{u}}$ and the symbol $f_{\mathrm{u}}$. Let $R_{\mathrm{u}} = \{\, l_i \to r_i \mid 1 \leq i \leq m \,\}$. Let $\boldsymbol{l}_{\mathrm{u}}$ denote the sequence $l_1, l_2, \ldots, l_m$ and $\boldsymbol{r}_{\mathrm{u}}$ the sequence $r_1, r_2, \ldots, r_m$. Let $L_{\mathrm{u}}$ be the following language:

$$L_{\mathrm{u}} = (\Sigma_{\mathrm{u}}, \{x_0, x_1, x_2, y_3, y_4, y_5, y_6, y_7\})$$

Let $F_{\mathrm{u}}$ be a new second-order variable with arity $m+1$. Let $\underline{\mathrm{cons}}$ be a new binary function symbol and $\underline{\mathrm{nil}}$ a new constant. The language $L_1$ is defined as the following expansion of $L_{\mathrm{u}}$:

$$L_1 = (\Sigma_{\mathrm{u}} \cup \{\underline{\mathrm{cons}}, \underline{\mathrm{nil}}\}, \mathcal{X}_{L_{\mathrm{u}}}, \{F_{\mathrm{u}}\}).$$

We can show that, given $t_{\mathrm{id}} \in \mathcal{T}_{\Sigma_{\mathrm{id}}}$, the following second-order equation in $L_1$ is solvable if and only if the constraint (1) is solvable:

$$\begin{aligned} &F_{\mathrm{u}}(\boldsymbol{l}_{\mathrm{u}}, \underline{\mathrm{cons}}(f_{\mathrm{u}}(q^{(0)}_{\mathrm{mv}}, q^{(2)}_{\mathrm{mv}}, x_1, x_1, q^{(4)}_{\mathrm{id}}, q^{(6)}_{\mathrm{id}}, y_3, y_3, y_5, y_5, y_7, y_7), \underline{\mathrm{nil}})) \approx \\ &\quad \underline{\mathrm{cons}}(f_{\mathrm{u}}(x_0, x_2, x_0, x_2, y_4, y_6, y_4, y_6, y_4, x_0, y_6, t^{(7)}_{\mathrm{id}} \centerdot x_2), F_{\mathrm{u}}(\boldsymbol{r}_{\mathrm{u}}, \underline{\mathrm{nil}})) \end{aligned} \tag{6}$$

**Lemma 5.** *Given $t_{\mathrm{id}} \in \mathcal{T}_{\Sigma_{\mathrm{id}}}$, (1) is solvable if and only if (6) is solvable.*

*Proof.* The direction '$\Rightarrow$' follows from [23, Lemma 2] and the observation that if $\theta$ solves (1) then $x\theta \in \mathcal{T}_{\Sigma_{\mathrm{u}}}$ for all $x \in \mathcal{X}_{L_{\mathrm{u}}}$. In particular, it is not possible that $\underline{\mathrm{cons}}$ or $\underline{\mathrm{nil}}$ appear in the terms that are substituted for $\mathcal{X}_{L_{\mathrm{u}}}$.

We prove the other direction. Assume that $\theta$ solves (6). We show that $\theta$ solves (1). A straightforward inductive argument shows that $F_{\mathrm{u}}\theta$ is an $L_1^*$-term of rank $m+1$ of the following form: (recall that $z_i$ is the $i$'th bound variable)

$$F_{\mathrm{u}}\theta = \underline{\mathrm{cons}}(s_1, \underline{\mathrm{cons}}(s_2, \ldots, \underline{\mathrm{cons}}(s_k, z_{m+1}) \cdots)),$$

for some $k \geq 1$, by using that $R_{\mathrm{u}}$ is ground and that $\underline{\mathrm{cons}} \notin \Sigma_{\mathrm{u}}$ (see [23, Lemma 1]). Hence, since $\theta$ solves (6), it follows that

$$\begin{aligned} &\underline{\mathrm{cons}}(s_1[\boldsymbol{l}_{\mathrm{u}}, t'], \ldots \underline{\mathrm{cons}}(s_{i+1}[\boldsymbol{l}_{\mathrm{u}}, t'], \ldots \underline{\mathrm{cons}}(t\theta, \qquad \underline{\mathrm{nil}}) \cdots) \cdots) = \\ &\underline{\mathrm{cons}}(s\theta, \qquad \ldots \underline{\mathrm{cons}}(s_i[\boldsymbol{r}_{\mathrm{u}}, \underline{\mathrm{nil}}], \ \ldots \underline{\mathrm{cons}}(s_k[\boldsymbol{r}_{\mathrm{u}}, \underline{\mathrm{nil}}], \underline{\mathrm{nil}}) \cdots) \cdots), \end{aligned} \tag{7}$$

where $s$ is the source term of (1), $t$ is the target term of (1), and $t' = \underline{\mathrm{cons}}(t\theta, \underline{\mathrm{nil}})$. So there exists a reduction in $R_{\mathrm{u}} \cup \{t' \to \underline{\mathrm{nil}}\}$ of the following form:

$$\begin{tikzcd} s_1[\boldsymbol{l}_{\mathrm{u}}, t'] \arrow[d, equal] \arrow[rrd, "*"] & & s_2[\boldsymbol{l}_{\mathrm{u}}, t'] \arrow[d, equal] & \cdots & s_k[\boldsymbol{l}_{\mathrm{u}}, t'] \arrow[d, equal] \arrow[rrd, "*"] & & t\theta \arrow[d, equal] \\ s\theta & & s_1[\boldsymbol{r}_{\mathrm{u}}, \underline{\mathrm{nil}}] & & s_{k-1}[\boldsymbol{r}_{\mathrm{u}}, \underline{\mathrm{nil}}] & & s_k[\boldsymbol{r}_{\mathrm{u}}, \underline{\mathrm{nil}}] \end{tikzcd}$$

This means that $s\theta \xrightarrow[R_u \cup \{t' \to \underline{\mathrm{nil}}\}]{*} t\theta$, *i.e.*,

$$f_u(\ x_0,\ \ x_2,\ x_0, x_2,\ y_4,\ \ y_6,\ y_4, y_6, y_4, x_0, y_6, t_{\mathrm{id}}^{(7)} \cdot x_2\ )\theta$$
$$\xrightarrow[R_u \cup \{t' \to \underline{\mathrm{nil}}\}]{*}$$
$$f_u(\ q_{\mathrm{mv}}^{(0)}, q_{\mathrm{mv}}^{(2)}, x_1, x_1, q_{\mathrm{id}}^{(4)}, q_{\mathrm{id}}^{(6)}, y_3, y_3, y_5, y_5, y_7,\quad y_7\quad )\theta$$

It is sufficient to show that $x_0\theta, x_2\theta, y_4\theta, y_6\theta \in \mathcal{T}_{\Sigma_u}$. Because then $s\theta \in \mathcal{T}_{\Sigma_u}$ and the rule $t' \to \underline{\mathrm{nil}}$ can not be used in the reduction of $s\theta$ to $t\theta$, since $\underline{\mathrm{nil}}$ does not occur in $\Sigma_u$. To begin with, we observe that

$$x_i\theta \xrightarrow[R_u \cup \{t' \to \underline{\mathrm{nil}}\}]{*} q_{\mathrm{mv}}^{(i)} \quad (i = 0, 2) \quad \text{and that} \quad y_i\theta \xrightarrow[R_u \cup \{t' \to \underline{\mathrm{nil}}\}]{*} q_{\mathrm{id}}^{(i)} \quad (i = 4, 6).$$

It follows by easy induction on the length of reductions that $t' \to \underline{\mathrm{nil}}$ can not be used in these reductions, since $\underline{\mathrm{nil}}$ does not not occur in $R_u$. Hence, $x_0\theta, x_2\theta, y_4\theta, y_6\theta \in \mathcal{T}_{\Sigma_u}$, as needed. ⊠

We conclude with the following result, that follows from Lemma 1, Lemma 4, and Lemma 5.

**Theorem 2.** *Second-order unification is undecidable with one second-order variable that occurs at most twice.*

The role of first-order variables in the above undecidability result is important. Without first-order variables, and if there is only one second-order variable that occurs at most twice, second-order unification reduces to *ground reachability*, [20], and is thus decidable.

## 5 Decidable Cases

We show that rigid reachability and simultaneous rigid reachability is decidable when the rules are all ground, either the source $s$ or the target $t$ of any constraint is linear, and the source $s$ and the target $t$ are *variable-disjoint*, that is, $Var(s) \cap Var(t) = \emptyset$. The non-simultaneous case then turns out to be EXPTIME-complete. EXPTIME-hardness holds already with just a single variable. This contrasts with the fact that rigid $E$-unification with one variable is $\mathcal{P}$-complete [7]. When additionally both the source and target terms are linear, then rigid reachability and simultaneous rigid reachability are both $\mathcal{P}$-complete.

Note that the only difference between the conditions for undecidability of rigid reachability in Theorem 1 and the condition for decidability in Theorems 4, 5, and 6 is the linearity of source and (or) target terms. In the rest of the section, we assume fixed a signature $\Sigma$.

### 5.1 Decidable Cases of Rigid Reachability

We begin with defining a reduction from rigid reachability to the emptiness problem of the intersection of $n$ regular languages recognized by tree automata

$A_1$,...,$A_n$. This intersection emptiness problem is known to be EXPTIME-complete, see [12], [21] and [24]. We may assume the states sets of the $A_1$,...,$A_n$ to be disjoint and that each of these tree automata has only one final state. We call these final states, respectively, $q^{\mathrm{f}}_{A_1}, \ldots, q^{\mathrm{f}}_{A_n}$. For stating the following lemma, we extend the signature $\Sigma$ by a new symbol $f$ of arity $n$, and assume that $n > 1$.

**Lemma 6.** *$T(A_1) \cap \ldots \cap T(A_n) \neq \emptyset$ iff the following constraint has a solution:*

$$\big(R_{A_1} \cup \ldots \cup R_{A_n},\ f(x, \ldots, x),\ f(q^{\mathrm{f}}_{A_1}, \ldots, q^{\mathrm{f}}_{A_n})\big)$$

*Proof.* ($\Rightarrow$) is obvious. For ($\Leftarrow$) we use the fact that the new symbol does not occur in any transition rule of the $A_1$, ..., $A_n$. Therefore, and since the state sets are disjoint, any reduction in $f(x, \ldots, x)\theta \xrightarrow[R_{A_1} \cup \ldots \cup R_{A_n}]{*} f(q^{\mathrm{f}}_{A_1}, \ldots, q^{\mathrm{f}}_{A_n})$ (where $\theta$ is a solution) takes place in one of the arguments of $f(x, \ldots, x)\theta$. Moreover, if the reduction is in the $i$-th subterm, it corresponds to the application of a rule in $R_{A_i}$. (It is possible, though, to apply a start rule in $R_{A_j}$ within the $i$-th subterm, with $i \neq j$.) But any reduction of this form blocks in that the final state $q^{\mathrm{f}}_{A_i}$ can not be reached from the reduct.) The fact than $n > 1$ prohibits states of the automata to appear in $x\theta$. ⊠

**Theorem 3.** *Rigid reachability is* EXPTIME*-hard even when the rules and the target are ground and the source contains only a single variable.*

For obtaining also an EXPTIME upper bound for a somewhat less restrictive case of rigid reachability we will apply tree automata techniques. In particular, we will exploit the following fact of preservation of recognizability under rewriting, which is a direct consequence of results in [6].

**Proposition 1 ([2]).** *Let $R$ be a ground rewrite system and $t$ a linear term. The set $\{u \in \mathcal{T}_\Sigma \mid u \xrightarrow[R]{*} t\sigma, t\sigma \text{ ground}\}$ is recognizable by a tree automaton $A$ the size of which is in $O(\|t\| * \|R\|^2)$.*

**Proposition 2.** *The subset of $\mathcal{T}_\Sigma$ of ground instances of a given linear term $s$ is recognizable by a tree automaton $A_s$ the size of which is linear in the size of $s$.*

**Theorem 4.** *Rigid reachability, when rules are ground, the target is linear and the source and the target are variable-disjoint, can be decided in times $O(n^{3k+4})$, where $n$ is the size of the constraint, and $k$ is the total number of occurrences of non-linear variables in the source term.*

Observe that the time upper bound becomes $O(n^4)$ when $s$ is linear, since $k = 0$ in this case.

*Proof.* Assume to be given a reachability constraint $(R, s, t)$ of the required form. We first construct a tree automaton $A$ from $t$ and $R$ with the properties as provided by Proposition 1, that is, recognizing the predecessors with respect to $R$ of the ground instances of $t$. The size $\|A\|$ of $A$ is in $O(n^3)$.

If the source $s$ is linear, then there is a solution for $(R, s, t)$ iff $T(A) \cap T(A_s) \neq \emptyset$, where $A_s$ is the tree automaton of Proposition 2. Since the intersection of recognizable languages is recognizable by a tree automaton the size of which is the product of the original tree automata, the solvability of the given constraint can be decided in time $O(\|s\| * n^3) \subseteq O(n^4)$.

If the source $s$ is not linear, we reduce our rigid reachability problem to $|Q_A|^k$ problems of the above type. We assume wlog that $A$ has only one final state $q^{\mathrm{f}}$. Let $(s_i)$ be the finite sequence of terms which can be obtained from the source $s$ by the following replacements: for every variable $x$ which occurs $j \geq 2$ times in $s$, we choose a tuple $(q_1, \ldots, q_j)$ of states of $A$ such that[2] $\cap_{i \leq j} T(A, q_i) \neq \emptyset$, and we replace the $i$th occurrence of $x$ with $q_i$ for $i \leq j$ in $s$.

Then the two following statements are equivalent:

(i) the constraint $(R, s, t)$ has a solution.
(ii) one of the constraints $(R_A, s_i, q^{\mathrm{f}})$ has a solution.

*(i) ⇒ (ii):* Assume that $\sigma$ is a solution of the constraint $(R, s, t)$. This means in particular that $s\sigma \in T(A)$ *i.e.* $s\sigma \xrightarrow[A]{*} q^{\mathrm{f}}$. Let $\tau$ be the restriction of $\sigma$ to the set of linear variables of $s$ and $\theta$ be its restriction to the set of non-linear variables of $s$. We have $s\theta \xrightarrow[R_A]{*} s_i$ by construction and $\tau$ is a solution of the constraint $(R_A, s_i, q^{\mathrm{f}})$.

*(ii) ⇒ (i):* Assume $s_i\tau \xrightarrow[R_A]{*} q^{\mathrm{f}}$ for some $i$ and some grounding substitution $\tau$. To each non-linear variable $x$ of $s$, we associate a term $s_x \in \cap_{i \leq j} T(A, q_i)$ (it exists by construction) where $q_1, \ldots, q_j$ are the states occurring in $s_i$ at positions corresponding to $x$ in $s$. This defines a substitution $\theta$ on the non-linear variables of $s$ (by $x\theta = s_x$) such that $s\tau\theta \in T(A)$. Hence $s\tau\theta \xrightarrow[R]{*} t\sigma$ for some grounding substitution $\sigma$ which is only defined on the variables of $t$. Since $Var(s) \cap Var(t) = \emptyset$, the domains of $\theta$, $\tau$ and $\sigma$ are pairwise disjoint and $\tau \cup \theta \cup \sigma$ is indeed a solution to the constraint $(R, s, t)$.

*Complexity:* The number of possible $s_i$ is smaller than $|Q_A|^k$ *i.e.* it is in $O(n^{3k})$. Rigid reachability for one constraint $(A, s_i, q^{\mathrm{f}})$ can be decided in time $O(n^4)$, according to the first part of this proof. Altogether, this gives a decision time in $O(n^{3k+4})$. ⊠

By symmetry, rigid reachability is also decidable when rules are ground, the source is linear and the source and the target are variable-disjoint, with the same complexities as in Theorem 4 according to the (non)-linearity of the target.

As a consequence we obtain these two theorems:

**Theorem 5.** *Rigid reachability is* EXPTIME*-complete when rules are ground, the source and the target are variable-disjoint, and either the source or the target is linear.*

**Theorem 6.** *Rigid reachability is $\mathcal{P}$-complete when the rules are ground, the source and the target are variable-disjoint, one of the source or the target is linear, and the number of occurrences of non-linear variables in the other is bounded by some fixed constant $k$ independent from the problem.*

[2] One can decide this property in time $\|A\|^k \in O(n^{3k})$.

Note that the linear case corresponds to $k = 0$.

*Proof.* For obtaining the lower bound, one may reduce the $\mathcal{P}$-complete uniform ground word problem (see [18]) to rigid reachability where rules, source and target are ground. The upper bound has been proved in Theorem 4. ⊠

## 5.2 Decidable Cases of Simultaneous Rigid Reachability

We now generalize Theorem 6 to the simultaneous case of rigid reachability.

**Theorem 7.** *Simultaneous rigid reachability is $\mathcal{P}$-complete for systems of pairwise variable-disjoint constraints with ground rules, and sources and targets that are variable-disjoint and linear.*

*Proof.* Apply Theorem 6 separately to each constraint of the system. ⊠

Similarly, we can prove:

**Theorem 8.** *Simultaneous rigid reachability is* EXPTIME-*complete for systems of pairwise variable-disjoint constraints with ground rules, and sources and targets that are variable-disjoint and such that at least one of them is linear for each constraint.*

The problem remains in $\mathcal{P}$ (see Theorem 6) if there is a constant $k$ independent from the problem and for each $s_i$ (resp. $t_i$) which is non-linear, the total number of occurrences of non-linear variable in $s_i$ (resp. $t_i$) is smaller than $k$.

We can relax the conditions in the above Theorem 8 by allowing some common variables between the $s_i$.

**Theorem 9.** *Simultaneous rigid reachability is in* EXPTIME *when all the rules of a system of constraints $\big((R_1, s_1, t_1), \ldots, (R_m, s_m, t_m)\big)$ are ground, either every $t_i$ is linear or every $s_i$ is linear, and for all $i, j \leq n$, the terms $s_i$ and $t_j$ and respectively the terms $t_i$ and $t_j$ (when $i \neq j$) are variable-disjoint.*

*Proof.* We prove for the case where every $t_i$ is linear, the other case follows by symmetry. We reduce this problem to an exponential number of problems of the type of Theorem 8.

We associate a TA $A_i$ to each pair $(t_i, R_i)$ which recognizes the language $\{u \in \mathcal{T}_\Sigma \mid u \xrightarrow[R_i]{*} t_i\sigma, t_i\sigma \text{ ground}\}$ (see Proposition 1). The size of each $A_i$ is in $O(\|t_i\| * \|R_i\|^2)$. We assume wlog that the states sets of the $A_i$ are pairwise disjoint and that the final states sets of the $A_i$ are singletons, namely $F_{A_i} = \{q_i^{\mathrm{f}}\}$. We construct for each $i \leq m$ a sequence of terms $(s_{i,j})$ obtained by replacement of variables occurrences in $s_i$ (regardless of linearity) by states of $A_i$. To each $m$-tuple $(s_{1,j_1}, \ldots, s_{m,j_m})$, we associate a system which contains the constraints:

1. $(R_{A_1}, s_{1,j_1}, q_1^{\mathrm{f}}), \ldots, (R_{A_m}, s_{1,j_m}, q_m^{\mathrm{f}})$
2. for every variable $x$ which occurs $k$ times in $\{s_1, \ldots, s_n\}$, with $k \geq 2$, $\big(R_{A_1} \uplus \ldots \uplus R_{A_m}, f_{\mathrm{u}}^k(x, \ldots, x), f_{\mathrm{u}}^k(q_1, \ldots, q_k)\big)$, where $f_{\mathrm{u}}^k$ is a new function symbol of arity $k$ and $q_1,\ldots,q_k$ are the states occurring in $s_{1,j_1},\ldots,s_{1,j_m}$ at the positions corresponding to $x$ in $s_1,\ldots,s_m$.

Then the system $\big((R_1, s_1, t_1), \ldots, (R_n, s_n, t_n)\big)$ has a solution iff one of the above systems has a solution. Each of these systems has a size which is polynomial in the size of the original system and moreover, each fulfills the hypothesis of Theorem 8 and can thus be decided in EXPTIME. Since the number of the above systems is exponential (in the size of the initial problem), we have an EXPTIME upper bound for the decision problem. ⊠

## 6 Monadic Simultaneous Rigid Reachability

Our second main decidability result generalizes the decidability proof of Monadic SREU for ground rules [17]. Moreover, our proof gives an EXPTIME upper bound to monadic SREU for ground rules. Although, the lower bound is known to be PSPACE [17], no interesting upper bound has been known before for this problem. We shall use basic automata theory for obtaining our result. More recently, and using different techniques, monadic rigid reachability with ground rules was found to be decidable also in PSPACE [4]. The presentation in this section will be in terms of words rather than monadic terms.

*Recognizing Substitution Instances.* We will first show that $n$-tuples of substitution instances of monadic terms are recognizable. For this purpose we let automata compute on $((\Sigma \cup \{\perp\})^n)^*$, where $\perp$ is a new symbol. The representation of a pair of words of $\Sigma^*$ as a word $((\Sigma \cup \{\perp\})^2)^*$ is given by the product $\otimes$ defined as follows:

$$\begin{array}{ll} a_1 \ldots a_n \otimes b_1 \ldots b_m = \langle a_1, b_1\rangle, \ldots, \langle a_n, b_n\rangle, \langle \perp, b_{n+1}\rangle, \ldots, \langle \perp, b_m\rangle & \text{if } n < m \\ a_1 \ldots a_n \otimes b_1 \ldots b_m = \langle a_1, b_1\rangle, \ldots, \langle a_m, b_m\rangle, \langle a_{m+1}, \perp\rangle, \ldots, \langle a_n, \perp\rangle & \text{if } n \geq m \end{array}$$

We extend this definition of $\otimes$ associatively to tuples of arbitrary length in the obvious way.

**Lemma 7.** *Let $L_1$, …, $L_n$ be recognizable subsets of $\Sigma^*$. Then $L_1 \otimes \ldots \otimes L_n$ is recognizable (in $((\Sigma \cup \{\perp\})^n)^*$). The size of the product automaton can be bounded by the product of the sizes of its factor automata.*

When constructing an automaton for $\Sigma^* \otimes L$ from an automaton $A$ for $L$, the size of the product automaton can be bounded by $c * \|A\|$, for some (small) constant $c$ if the alphabet $\Sigma$ is assumed to be fixed.

**Theorem 10.** *Given $p$ monadic $\Sigma$-terms $s_i$, the set of tuples of their ground instances $\{s_1\theta \otimes \ldots \otimes s_p\theta \mid \theta \text{ ground }\}$ is recognizable by an automaton with size exponential in $\sum_{j=1}^{p} |s_j|$.*

The proof of Theorem 10 will be based on three technical lemmas.

**Lemma 8.** *Let $L$ be a language of tuples of the form $a_1 \otimes \ldots \otimes a_n$ that is recognized by $A$. Then, for any permutation $\pi$, the language*

$$L_\pi = \{a_{\pi 1} \otimes \ldots \otimes a_{\pi n} \mid a_1 \otimes \ldots \otimes a_n \in L\}$$

*can be recognized by an automaton of the same size as $A$.*

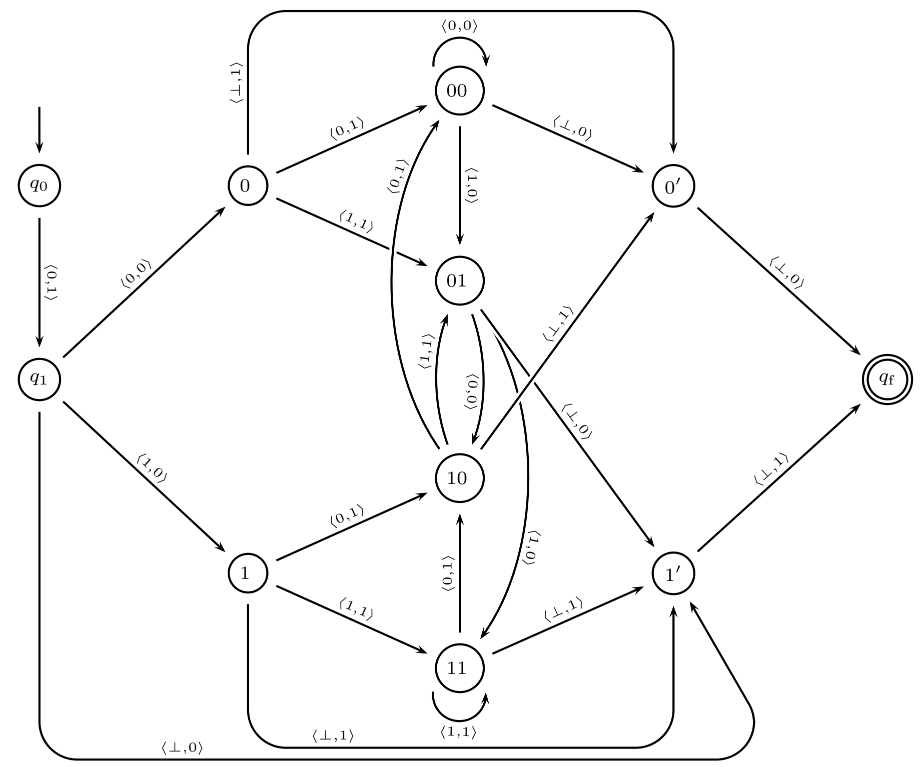

**Figure 2:** An example illustrating the proof of Lemma 9.

*Proof.* We apply the permutation $\pi$ to the tuples of symbols appearing in the rules of $R_A$. ☒

**Lemma 9.** *Given $s, t \in \Sigma^*$, the set $\{su \otimes tu \bigm| u \in \Sigma^*\}$ is recognizable by an automaton with size exponential in $|s| + |t|$.*

*Proof.* The automaton reads words $s'u \otimes t'v$, checking that $s' = s$ and $t' = t$ and that $u = v$. For the latter test, the automaton has to memorize (in states) the last $\bigl||t| - |s|\bigr|$ symbols of $\Sigma$ read in $s'u$. This is the reason of the exponential number of states. An example of the construction for $\Sigma = \{0, 1\}$, $s = 0$ and $t = 101$, is given in figure 2. ☒

The construction underlying the proof of Lemma 9 cannot be generalized to the case of non-monadic signatures. However, one can generalize it (in the monadic case) to the product of an arbitrary number $p$ of words.

**Lemma 10.** *Given $p \geq 1$, the set $\{s_1u \otimes \ldots \otimes s_pu \bigm| u \in \Sigma^*\}$ is recognizable by an automaton with size exponential in $\sum_{j=1}^{p} |s_j|$.*

*Proof.* By induction on $p$. The base case $p = 1$ is trivial, and the case $p = 2$ is proved in Lemma 9. Assume that $p \geq 2$ and that we have an automaton $A$ with $T(A) = \{s_1u \otimes \ldots \otimes s_pu \bigm| u \in \Sigma^*\}$ and one more word $s_{p+1}$. Let $A''$ be an automaton such that $T(A'') = \{s_p u \otimes s_{p+1} u \mid u \in \Sigma^*\}$. $A''$ may be obtained by applying Lemma 9 again. Clearly,

$$\big(T(A) \otimes \Sigma^*\big) \cap \big(\underbrace{\Sigma^* \otimes \ldots \otimes \Sigma^*}_{p-1} \otimes T(A'')\big) = \{s_1 u \otimes \ldots \otimes s_{p+1} u \mid u \in \Sigma^*\}.$$

According to Lemma 7, this language is recognizable by an automaton $A'$, and, by Lemmas 7 and 9, $\|A'\|$ is of the order $2^{\sum_{j=1}^{p} |s_j|} * 2^{|s_p|+|s_{p+1}|}$. $\boxtimes$

Now we are ready to prove Theorem 10.

*Proof.* The terms $s_i$ are either ground or have the form $s_i(x_i)$ with one occurrence of a variable $x_i$ "at the end". Let, for any variable $x$ occurring in any of the terms, $s_{i_1}, \ldots, s_{i_n}$ be those terms among the $s_i$ which contain $x$. According to Lemma 10, the language

$$L_1^x = \{s_{i_1}\theta \otimes \ldots \otimes s_{i_n}\theta \mid \theta \text{ ground }\}$$

is recognizable by an automaton of size exponential in $\sum_j |s_{i_j}|$. From the Lem -ma 7 we infer that $L_2^x = L_1^x \otimes \Sigma^* \otimes \ldots \otimes \Sigma^*$, with $p - n$ factors of $\Sigma^*$, is recognizable by an automaton with size exponential in $\sum_j |s_{i_j}|$. Finally, $L^x = (L_2^x)_\pi$, with $\pi$ a permutation which maps the first $n$ indices $j$ to $i_j$, that is, puts the $s_{i_j}$ into their right place in the sequence $1 \ldots p$, is also recognizable by an automaton of the same size, see Lemma 8. Moreover, it is not difficult to see that

$$L_g = \{t_1 \otimes \ldots \otimes t_p \mid t_i = s_i \text{ if } s_i \text{ ground, and } t_i \in \Sigma^*, \text{ otherwise}\}$$

is recognizable by an automaton with size polynomial in $\max |s_i|$. The desired language arises as the intersection of the languages $L^x$ and $L_g$ so that recognizability with the stated complexity bound follows. $\boxtimes$

For solving reachability constraints, we also need to recognize rewriting relations.

**Theorem 11 ([6]).** *Given a ground rewrite system $R$ on $\Sigma^*$, the set $\{u \otimes v \mid u \xrightarrow[R]{*} v\}$ is recognizable by an automaton the size of which is polynomial in the size of $R$.*

**Theorem 12.** *Rigid reachability in monadic signatures is in* EXPTIME *when the rules are ground.*

*Proof.* Let $(R, s, t)$ be a constraint over the monadic signature $\Sigma$. We show that the set of "solutions" $\mathcal{S} = \{s\theta \otimes t\theta \mid s\theta \xrightarrow[R]{*} t\theta\}$ is recognizable. $s$ and $t$ may contain at most one variable which we denote by $x$ and $y$, respectively. These two variables may or may not be identical. Applying Theorem 10, we may infer that $\{s\theta \otimes t\theta \mid x\theta \text{ and } y\theta \text{ ground }\}$ is recognizable by an automaton $A$ with size exponential in $|s| + |t|$. By Theorem 11, the set $\{u \otimes v \mid u, v \in \Sigma^*, u \xrightarrow[R]{*} v\}$ is recognizable by an automaton $A'$ with size polynomial in the size of $R$. Clearly $\mathcal{S} = T(A) \cap T(A')$, and emptiness is decidable in time linear in the size of the corresponding intersection automaton (which is exponential in $|s| + |t|$ and polynomial in the size of $R$). $\boxtimes$

The extension to the simultaneous case of Theorem 12 generalizes and improves a result of [17].

**Theorem 13.** *Simultaneous rigid reachability in monadic signatures is decidable in* EXPTIME *when the rules are ground.*

*Proof.* The construction is a generalization of the one for Theorem 12. Suppose we are given the system of constraints $(R_i, s_i, t_i)$, $1 \leq i \leq n$. We first construct an automaton $A_i$ for each $i \leq n$ such that $T(A_i) = \{u \otimes v \mid u, v \in \Sigma^*, u \xrightarrow[R_i]{*} v\}$. Then $A = \bigotimes_{i=1}^n A_i$ (see Lemma 7) recognizes the language:

$$T(A) = \{u_1 \otimes v_1 \otimes u_2 \otimes \ldots \otimes u_n \otimes v_n \mid \text{ for all } i \leq n, u_i, v_i \in \Sigma^*, u_i \xrightarrow[R_i]{*} v_i\}.$$

The size of $A$ is the product of the sizes of the $A_i$, hence of order $M^n$ where $M$ is the maximum of the sizes of the $A_i$. In Theorem 10 we have shown that the language

$$L^G = \{s_1\theta \otimes t_1\theta \otimes \ldots \otimes s_n\theta \otimes t_n\theta \mid \theta \text{ ground}\}$$

is recognizable by an automaton $A^G$ of size exponential in $\sum_i |s_i| + |t_i|$. The simultaneous reachability constraint is solvable if and only the intersection $L^G \cap T(A)$ is non-empty. According to the respective sizes of the automata in the above intersection, this gives an EXPTIME upper-bound for deciding simultaneous rigid reachability. ⊠

## 7 Conclusion

We have shown that absence of symmetry makes the solving of rigid reachability constraints in general much harder. In the non-simultaneous case one jumps from decidability to undecidability. In the case of ground rewrite rules, source terms with just a single variable, and ground target terms, the complexity increases from $\mathcal{P}$-completeness to EXPTIME-completeness. The undecidability of rigid reachability implies a new undecidability result for second-order unification problems with just a single second-order variable that occurs twice. We have also seen that automata-theoretic methods provide us with rather simple proofs of upper bounds in the monadic case.

## Acknowledgments

The authors wish to thank the referees for their useful remarks and suggestions, which led to significant improvements of the results presented in this paper.

## References

1. F. Baader and T. Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998.

2. J.-L. Coquidé and R. Gilleron. Proofs and reachability problem for ground rewrite systems. In *Proc. of the 6th International Meeting of Young Computer Scientists*, volume 464 of *LNCS*, pages 120–129, 1990.
3. J.L. Coquidé, M. Dauchet, R. Gilleron, and S. Vágvölgyi. Bottom-up tree pushdown automata: classification and connection with rewrite systems. *Theoretical Computer Science*, 127:69–98, 1994.
4. V. Cortier. PSPACE-completeness of monadic simultaneous rigid reachability. Unpublished manuscript, july 1998.
5. M. Dauchet. Rewriting and tree automata. In H. Comon and J.P. Jouannaud, editors, *Term Rewriting (French Spring School of Theoretical Computer Science)*, volume 909 of *Lecture Notes in Computer Science*, pages 95–113. Springer Verlag, Font Romeux, France, 1993.
6. Max Dauchet, Thierry Heuillard, Pierre Lescanne, and Sophie Tison. The confluence of ground term rewriting systems is decidable. In *Proc. 3rd IEEE Symp. Logic in Computer Science, Edinburgh*, 1988.
7. A. Degtyarev, Yu. Gurevich, P. Narendran, M. Veanes, and A. Voronkov. The decidability of simultaneous rigid $E$-unification with one variable. In T. Nipkow, editor, *Rewriting Techniques and Applications*, volume 1379 of *Lecture Notes in Computer Science*, pages 181–195. Springer Verlag, 1998.
8. A. Degtyarev and A. Voronkov. Simultaneous rigid $E$-unification is undecidable. UPMAIL Technical Report 105, Uppsala University, Computing Science Department, May 1995.
9. N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In J. Van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Methods and Semantics, chapter 6, pages 243–309. North Holland, Amsterdam, 1990.
10. J. Doner. Tree acceptors and some of their applications. *Journal of Computer and System Sciences*, 4:406–451, 1970.
11. W.M. Farmer. Simple second-order languages for which unification is undecidable. *Theoretical Computer Science*, 87:25–41, 1991.
12. Thom Frühwirth, Ehud Shapiro, Moshe Y. Vardi, and Eyal Yardemi. Logic programs as types for logic programs. In *6th IEEE Symp. Logic In Computer Science*, pages 300–309, 1991.
13. J.H. Gallier, P. Narendran, D. Plaisted, and W. Snyder. Rigid $E$-unification is NP-complete. In *Proc. IEEE Conference on Logic in Computer Science (LICS)*, pages 338–346. IEEE Computer Society Press, July 1988.
14. J.H. Gallier, S. Raatz, and W. Snyder. Theorem proving using rigid $E$-unification: Equational matings. In *Proc. IEEE Conference on Logic in Computer Science (LICS)*, pages 338–346. IEEE Computer Society Press, 1987.
15. W.D. Goldfarb. The undecidability of the second-order unification problem. *Theoretical Computer Science*, 13:225–230, 1981.
16. Y. Gurevich and M. Veanes. Partisan corroboration, and shifted pairing. Research Report MPI-I-98-2-014, Max-Planck-Institut für Informatik, September 1998.
17. Y. Gurevich and A. Voronkov. Monadic simultaneous rigid $E$-unification and related problems. In P. Degano, R. Corrieri, and A. Marchetti-Spaccamella, editors, *Automata, Languages and Programming, 24th International Colloquium, ICALP'97*, volume 1256 of *Lecture Notes in Computer Science*, pages 154–165. Springer Verlag, 1997.
18. D. Kozen. Complexity of finitely presented algebras. In *Proc. 9th STOC*, pages 164–177, 1977.

19. J. Levy. Decidable and undecidable second-order unification problems. In T. Nipkow, editor, *Rewriting Techniques and Applications, 9th International Conference, RTA-98, Tsukuba, Japan, March/April 1998, Proceedings*, volume 1379 of *Lecture Notes in Computer Science*, pages 47–60. Springer Verlag, 1998.
20. J. Levy and M. Veanes. On the undecidability of second-order unification. Submitted to *Information and Computation*, 1998.
21. Helmut Seidl. Haskell overloading is dexptime-complete. *Inf. Process. Letters*, 52:57–60, 1994.
22. J.W. Thatcher and J.B. Wright. Generalized finite automata theory with an application to a decision problem of second-order logic. *Mathematical Systems Theory*, 2(1):57–81, 1968.
23. M. Veanes. The relation between second-order unification and simultaneous rigid $E$-unification. In *Proc. Thirteenth Annual IEEE Symposium on Logic in Computer Science, June 21–24, 1998, Indianapolis, Indiana (LICS'98)*, pages 264–275. IEEE Computer Society Press, 1998.
24. Margus Veanes. On computational complexity of basic decision problems of finite tree automata. Technical Report 133, Computing Science Department, Uppsala University, 1997.