ELSEVIER

Information Processing Letters 74 (2000) 47–53

Information Processing Letters

www.elsevier.com/locate/ipl

Farmer's Theorem revisited

Margus Veanes [1]

Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA

Received 15 April 1999; received in revised form 6 December 1999
Communicated by H. Ganzinger

Abstract

The main result is that there is an integer n, such that second-order unification is undecidable in *all non-monadic* second-order term languages with at least n *first-order* variables, and even if the arguments of all second-order variables are ground terms of size bounded by n and the total number of variable occurrences is at most n. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Second-order unification; Ground term rewriting; Programming languages

1. Introduction

Farmer [3] studied the second-order unification problem or SOU for very restricted second-order term languages and, improving upon Goldfarb's [6] undecidability result of SOU in general, showed that the undecidability occurs already with a bounded number of *unary second-order variables* (if the bound is large enough) as soon as there is a single constant and a single nonunary function symbol in the language. The unarity of the second-order variables is relevant to certain fundamental decision problems of finite first-order axiomatic systems studied by Parikh [10].

Another important special case of SOU, the so-called *simple* case, arises by restricting all occurrences of second-order variables to have (first-order) ground terms as arguments. The undecidability of this case was established by Schubert [11], and had important implications for certain type inference problems. The result was sharpened by Veanes [12] to hold already in the presence of two second-order variables (with sufficiently large arities) and no first-order variables, even if the arguments of second-order variables have bounded size. Detyarev and Voronkov [2], Levy [7], and Veanes [12] have shown that SOU is intimately related to the *simultaneous rigid E-unification* problem or SREU [4], and thus to several fundamental decision problems in automated theorem proving in logic with equality [1]. This close connection has enabled the translation of several results and techniques from the context of SREU to the context of SOU [7,12] and vice versa [2]. Moreover, SOU was recently shown to be undecidable already with a *single unary second-order variable* [8] at the cost of having an unbounded number of first-order variables in the language.

Here it is shown that the above undecidability results are instances of an even stronger undecidability result, by applying Farmer's [3] encoding techniques to the key results in [9,12]. Namely, undecidability of SOU is shown to hold already in the simple case with a single unary second-order variable. Moreover, even

[1] Email: margus@microsoft.com. Part of this work was done during the author's postgraduate studies at the Max-Planck-Institut für Informatik, Im Stadtwald, 66123 Saarbrücken, Germany. An extended version of the paper with full proofs appears as Microsoft Research Technical Report.

0020-0190/00/$ – see front matter
PII: S0020-0190(00)00029-6

putting a fixed bound on the size of the arguments of the second-order variable occurrences, and on the total number of occurrences of all variables, does not lead to decidability, if the bound is too large. It should be interesting to investigate what implications this result has in the context of the above mentioned areas.

When the simplicity requirement is dropped then the undecidability of SOU holds already if there is a single second-order variable (with large arity) that occurs twice. This case is intimately related to the *rigid reachability* problem [5].

2. Preliminaries

We assume that the reader is familiar with first-order unification problems, ground term rewriting, and basic notions in λ-calculus. In this section, we fix the notation and briefly go through the necessary definitions.

A *term language* L, or simply a *language*, consists of

(1) a *signature* Σ that is a finite set of *function symbols* $(f, g, h, \ldots)$ with fixed arities $\geqslant 0$ containing at least one *constant* $(a, b, c, \ldots)$ or function symbol with arity 0.[2] Symbols of arity 0 are also called *nullary* symbols,
(2) a set $\boldsymbol{X}$ of *first-order variables* $(x, y, z, \ldots)$ defined to be nullary, and
(3) a set $\boldsymbol{F}$ of *second-order variables* $(F, G, H, \ldots)$ with fixed arities $\geqslant 1$.

L is *second-order*, if $\boldsymbol{F}$ is nonempty; L is *first-order*, otherwise. We will use ν as a generic symbol denoting either a function symbol or a variable. The set of *terms of arity* 0 or *nullary terms* is defined as the least set that contains all nullary symbols, and contains $\nu(t_1, \ldots, t_n)$ where ν has arity $n > 0$ and $t_1, \ldots, t_n$ are nullary terms. A *term of arity* n for $n > 0$ is a λ-expression $\lambda x_1 \cdots x_n.t$ where t is a term of arity 0 and $x_1, \ldots, x_n$ are distinct first-order variables. The variables $x_1 \cdots x_n$ in $\lambda x_1 \cdots x_n.t$ are called *bound*. All other variables in $\lambda x_1 \cdots x_n.t$ are said to be *free*. By *terms* we understand terms of arity n for $n \geqslant 0$. A *first-order* term is a nullary term that contains no second-order variables. A term is *closed* if it contains no free variables, and *ground* if it contains neither free nor bound variables, i.e., a ground term is a closed first-order term.

A *substitution* $(\sigma, \theta, \rho \ldots)$ is a finite mapping $\{\nu_i \mapsto t_i \mid 1 \leqslant i \leqslant m\}$, say θ, of variables to terms, with each ν_i and t_i having the same arity. The *domain* of θ is $\{\nu_1, \ldots, \nu_m\}$ and the *range* of θ is $\{t_1, \ldots, t_m\}$. A substitution is *closed* (ground) if its range is a set of closed (ground) terms. Application $\theta(s)$ of a substitution θ to a term s of arity 0 is defined as usual by induction on s, say $s = \nu(s_1, \ldots, s_n)$ for some $n \geqslant 0$: if ν is not in the domain of θ then $\theta(s) = \nu(\theta(s_1), \ldots, \theta(s_n))$, else θ maps ν to an n-ary term $\lambda x_1 \cdots x_n.t$ and $\theta(s) = \rho(t)$, where $\rho = \{x_i \mapsto \theta(s_i) \mid 1 \leqslant i \leqslant n\}$. We write also $t\theta$ for $\theta(t)$. Given an m-ary term $t = \lambda z_1 \cdots z_m.u$ and terms $t_1, \ldots, t_m$, we will write $t(t_1, \ldots, t_m)$ for its β-reduced form $u\theta$ where $\theta = \{z_i \mapsto t_i \mid 1 \leqslant i \leqslant m\}$.

An *equation* is an unordered pair $s \approx t$ of terms. A *rule* is an ordered pair $s \to t$ of terms.[3] A *ground rewrite system* is a finite set of ground rules. Given ground terms s and t, and a ground rewrite system R, $s \xrightarrow{*}_R t$ denotes that there is a (possibly empty) reduction in R from s to t, and $s \xrightarrow{\shortparallel}_R t$ denotes that s reduces to t by a simultaneous application of 0 or more non-overlapping rewrite steps. Notice that both $\xrightarrow{*}_R$ and $\xrightarrow{\shortparallel}_R$ are, by definition, reflexive.

A *unifier* of a finite set $\{s_i \approx t_i \mid 1 \leqslant i \leqslant n\}$ of equations, is a substitution θ, such that $s_i\theta = t_i\theta$ for $1 \leqslant i \leqslant n$. The *unification problem* is, given a finite set $\boldsymbol{S}$ of equations, to decide if there exists a unifier of $\boldsymbol{S}$. The *second-order unification* problem or SOU is the unification problem in second-order languages.

Given an expression or a set of expressions X, we write $Sig(X)$ for the *signature* of X, or the set of all function symbols that occur in X. We write $Sig(X_1, \ldots, X_n)$ for the union of the signatures of all the X_i's.

3. Overview

The main result of this paper (as stated in the abstract) is more precisely formulated in Theorem 13. The following is a roadmap of how we get to that result.

[2] The restriction that Σ contains at least one constant can be relaxed at the expense of having to treat tedious special cases (cf. [3]).

[3] By rules we understand thus *directed equations*. Only ground instantiations of rules are considered as *rewrite rules*.

The undecidable problem that we start from is directly based on the undecidability result of the second-order unification problem when restricted to systems of two simple second-order equations of the form

$$F(l_1, \ldots, l_m) \approx f\big(s, F(r_1, \ldots, r_m)\big)$$

[9]. See Section 5, Lemma 9.

(1) We apply Farmer's [3] encoding technique to replace each such equation with a collection of equations involving only unary second-order variables. The details of the encoding are presented in Section 4. The precise relationship between the original set of equations and the translated ones is formulated in Lemma 10.

As an example of this technique, a simple equation

$$F(l_1, l_2) \approx f\big(s, F(r_1, r_2)\big)$$

would be translated into a system

$$\big\{F_0(\mathbf{l}) \approx F_1(f_1(\mathbf{l})),\ F_1(l_1) \approx F_2(f_2(\mathbf{l})),\ F_2(l_2) \approx f(s, F_3(r_2)),\ F_0(\mathbf{r}) \approx F_4(f_1(\mathbf{r})),\ F_4(r_1) \approx F_3(f_2(\mathbf{r}))\big\},$$

where $\mathbf{l}$ and $\mathbf{r}$ stand for new constants and f_1 and f_2 are new unary function symbols. Notice that simplicity is preserved. Now assume a unifier θ of this set of equations exists. Then

$$\begin{aligned}
&F_0\theta(\mathbf{l}) = F_1\theta\big(f_1(\mathbf{l})\big)\\
&\xrightarrow{\ \cup\ }_{f_1(\mathbf{l})\to l_1} F_1\theta(l_1) = F_2\theta\big(f_2(\mathbf{l})\big)\\
&\xrightarrow{\ \cup\ }_{f_2(\mathbf{l})\to l_2} F_2\theta(l_2) = f\big(s\theta, F_3\theta(r_2)\big)\\
&\xrightarrow{\ \cup\ }_{r_2\to f_2(\mathbf{r})} f\big(s\theta, F_3\theta(f_2(\mathbf{r}))\big) = f\big(s\theta, F_4\theta(r_1)\big)\\
&\xrightarrow{\ \cup\ }_{r_1\to f_1(\mathbf{r})} f\big(s\theta, F_4\theta(f_1(\mathbf{r}))\big) = f\big(s\theta, F_0\theta(\mathbf{r})\big).
\end{aligned}$$

The constants $\mathbf{l}$ and $\mathbf{r}$ encode the (ground term) sequences l_1, l_2 and r_1, r_2, respectively, and each f_i (conceptually) projects the ith element of either sequence (i.e., $f_i(\mathbf{l}) = l_i$ and $f_i(\mathbf{r}) = r_i$). With this understanding, the parallel reduction steps above are trivial and hence $F_0\theta(\mathbf{l})$ coincides with $f(s\theta, F_0\theta(\mathbf{r}))$. One can now extract a unifier of the original equation out of θ.

(2) The obtained set of equations can be translated into a set of equations with a single unary second-order variable using a coding technique due to Levy and Veanes [9] that also preserves simplicity. See Lemma 11.

(3) Finally, well-known encoding techniques at the signature level allow us to reduce the signature to just one binary function symbol and one constant. Again, simplicity is preserved. See Lemma 12.

4. Reduction to unary second-order variables

In this section we show how to encode unifiability of equations of the following form by using only unary second-order variables:

$$S[x, F]: \quad F\big(l_1, \ldots, l_{m-1}, f(t, c)\big) \approx f\big(x, F(r_1, \ldots, r_{m-1}, c)\big),$$

where $m \geqslant 1$ and $l_1, \ldots, l_{m-1}, r_1, \ldots, r_{m-1}, t$ are ground terms not containing the constant c and the function symbol f. We will apply Farmer's [3] encoding techniques to such second-order equations. Let $S[x, F]$ be fixed throughout the rest of this section. Let $l_m = f(t, c)$ and $r_m = c$. We will write $\boldsymbol{l}$ for the sequence $l_1, \ldots, l_m$ and $\boldsymbol{r}$ for the sequence $r_1, \ldots, r_m$ and we will consider the rewrite system

$$R = \{l_i \to r_i \mid 1 \leqslant i < m\}.$$

Note that the rule $f(t, c) \to c$ is not in R, and thus $c, f \notin Sig(R)$. We will use the following connection between unifiability of $S[x, F]$ and ground rewriting:

Lemma 1. *For all ground terms s, $s \xrightarrow{*}_R t$ if and only if $S[s, F]$ is unifiable. Moreover, if θ unifies $S[s, F]$, then*

$$Sig(F\theta) \subseteq Sig\big(S[x, F]\big).$$

Proof. See [9, Lemma 2]. To illustrate the main idea we consider a simple case with R consisting of a single rule $l \to r$. Consider a reduction $s \xrightarrow{*}_R t$. We can view the reduction as follows, where each s_i is a closed unary term with exactly one occurrence of the bound variable corresponding to the position in the term $s_i(l)$ where the ith reduction step is applied:

$$\begin{aligned}
s &= s_1(l) \to s_1(r) = s_2(l) \to s_2(r)\\
&= s_3(l) \to \cdots \to s_{k-1}(r)\\
&= s_k(l) \to s_k(r) = t.
\end{aligned}$$

The key observation is that an equivalent way of viewing this reduction is as the following identity between two "lists":

$$f(s_1(l),\ f(s_2(l),\ f(s_3(l),\ldots,\ f(s_k(l),\ f(t,\ c))))) =$$
$$f(s,\ f(s_1(r),\ f(s_2(r),\ldots,\ f(s_{k-1}(r),\ f(s_k(r),\ c))))).$$

Moreover, we can extract the skeleton $\lambda xy.f(s_1(x), f(s_2(x),\ldots,f(s_k(x),y)))$, say u, from both sides of the identity and write identity as $u(l, f(t,c)) = f(s, u(r,c))$. In other words, $\theta = \{F \mapsto u\}$ solves the equation:

$$F\big(l, f(t,c)\big) \approx f\big(s, F(r,c)\big).$$

Conversely, if we consider unifiers of this equation then they must have the above form and thus correspond to the given ground rewriting problem.

Let f_i, for $0 \leqslant i \leqslant m$, be distinct unary function symbols not in $Sig(S[x,F])$, and let $\mathbf{l}$ and $\mathbf{r}$ be distinct constants not in $Sig(S[x,F])$. Intuitively, $\mathbf{l}$ ($\mathbf{r}$) is a constant that encodes the sequence $\boldsymbol{l}$ ($\boldsymbol{r}$), and f_i is the "projection function" that projects the ith component from $\boldsymbol{l}$ (or $\boldsymbol{r}$). The equation $S[x,F]$ is encoded by the following set of equations, with fresh unary second-order variables:

$$\boldsymbol{S}^0[x,F_0]\colon \quad \begin{cases} F_0(\mathbf{l}) \approx F_1(f_1(\mathbf{l})), \\ F_0(\mathbf{r}) \approx F_1'(f_1(\mathbf{r})), \\ F_i(l_i) \approx f_{i+1}(f_{i+1}(\mathbf{l})) \\ F_i'(r_i) \approx F_{i+1}'(f_{i+1}(\mathbf{r})) \\ F_m(l_m) \approx f(x, F_m'(r_m)). \end{cases} \quad (1 \leqslant i \leqslant m),$$

The main idea behind this construction was illustrated in Section 3. Unifiability of $S[s,F]$ implies unifiability of $\boldsymbol{S}^0[s,F_0]$, and it is straightforward to construct a unifier of the latter set of equations from a unifier of $S[s,F]$. □

Lemma 2. *For all ground terms s, if θ unifies $S[s,F]$ then σ unifies $\boldsymbol{S}^0[s,F_0]$, where $F_0\sigma = \lambda x.(F\theta)(f_1(x),\ldots,f_m(x))$, and, for $1 \leqslant i \leqslant m$,*

$$F_i\sigma = \lambda x.(F\theta)\big(l_1,\ldots,l_{i-1},x,f_{i+1}(\mathbf{l}),\ldots,f_m(\mathbf{l})\big)$$

and

$$F_i'\sigma = \lambda x.(F\theta)\big(r_1,\ldots,r_{i-1},x,f_{i+1}(\mathbf{r}),\ldots,f_m(\mathbf{r})\big).$$

The converse direction requires a little more work. The following lemmas will be used to show that unifiability of $\boldsymbol{S}^0[s,F_0]$ implies unifiability of $S[s,F]$. Let $\Pi_{\boldsymbol{l}}$ and $\Pi_{\boldsymbol{r}}$ be the following ground rewrite systems:

$$\Pi_{\boldsymbol{l}} = \big\{f_i(\mathbf{l}) \to l_i \mid i \leqslant m\big\},$$
$$\Pi_{\boldsymbol{r}} = \big\{f_i(\mathbf{r}) \to r_i \mid i \leqslant m\big\}.$$

Lemma 3. *For all ground terms s, if σ unifies $\boldsymbol{S}^0[s,F_0]$ then there exists a ground term u such that $F_0\sigma(\mathbf{l}) \xrightarrow{\shortparallel}_{\Pi_{\boldsymbol{l}}} f(s,u)$ and $F_0\sigma(\mathbf{r}) \xrightarrow{\shortparallel}_{\Pi_{\boldsymbol{r}}} u$.*

In the next two lemmas, we will use the following ground rewrite system:

$$E = \big\{f_i(\mathbf{l}) \to l_i, l_i \to f_i(\mathbf{l}),$$
$$f_i(\mathbf{r}) \to r_i, r_i \to f_i(\mathbf{r}) \mid 1 \leqslant i < m\big\}.$$

Note that $c, f, f_m \notin Sig(E)$. Using this fact, the structure of $\Pi_{\boldsymbol{l}}$ and $\Pi_{\boldsymbol{r}}$, and the fact that the f_i's, f, c, and f_m are not in $Sig(R)$, the proof of Lemma 4 is straightforward by induction on v.

Lemma 4. *For all closed terms u of arity 1, and ground terms s and v,*

if $u(\mathbf{l}) \xrightarrow{\shortparallel}_{\Pi_{\boldsymbol{l}}} f(s,v)$ and $u(\mathbf{r}) \xrightarrow{\shortparallel}_{\Pi_{\boldsymbol{r}}} v$,
then $s \xrightarrow{*}_{E\cup R} t$.

The following lemma follows from basic properties of E and R, in particular from the fact that $f_1,\ldots,f_m,\mathbf{l},\mathbf{r} \notin Sig(R,t)$.

Lemma 5. *For all ground terms s such that $\mathbf{l},\mathbf{r} \notin Sig(s)$, if $s \xrightarrow{*}_{E\cup R} t$ then $s \xrightarrow{*}_R t$.*

We can now prove the following lemma.

Lemma 6. *For all ground terms s such that $\mathbf{l},\mathbf{r} \notin Sig(s)$, if $S^0[s,F_0]$ is unifiable then $S[s,F]$ is unifiable.*

Proof. Let s and σ unifying $S_0[s,F_0]$ be given. By Lemma 3, there is a term u such that

$$F_0\sigma(\mathbf{l}) \xrightarrow{\shortparallel}_{\Pi_{\boldsymbol{l}}} f(s,u) \quad \text{and} \quad F_0\sigma(\mathbf{r}) \xrightarrow{\shortparallel}_{\Pi_{\boldsymbol{r}}} u.$$

It follows from Lemmas 4 and 5 (by using that $\mathbf{l},\mathbf{r} \notin Sig(s)$) that $s \xrightarrow{*}_R t$. Finally, it follows from Lemma 1, that $S[s,F]$ is unifiable. □

We will also need the following version of Lemma 6, stating how to construct a unifier of $S[s, F]$ from certain unifiers of $\boldsymbol{S}^0[s, F_0]$.

Lemma 7. *For all ground terms s such that $\mathbf{l}, \mathbf{r}, \notin Sig(s)$, if σ unifies $\boldsymbol{S}^0[s, F_0]$ and $\mathbf{l}, \mathbf{r} \notin Sig(F_0\sigma)$ then there is a closed m-ary term $F\theta$ such that $F_0\sigma = \lambda z.(F\theta)(f_1(z), f_2(z), \ldots, f_m(z))$, and θ unifies $S[s, F]$.*

Proof. Let s and σ be given. It follows from Lemma 3 that there exists a term v such that $F_0\sigma(\mathbf{l}) \xrightarrow{\shortparallel}_{\Pi_l} f(s, v)$ and $F_0\sigma(\mathbf{r}) \xrightarrow{\shortparallel}_{\Pi_r} v$. Since $\mathbf{l} \notin Sig(F_0\sigma, \Pi_r)$ it follows that $\mathbf{l} \notin Sig(v)$. Similarly, $\mathbf{r} \notin Sig(f(s, v))$. So each occurrence of the bound variable in $F_0\sigma$ is as the argument of some f_i, or else the above reductions are not possible. Hence, there is a closed m-ary term u, such that

$$F_0\sigma = \lambda z.u\big(f_1(z), f_2(z), \ldots, f_m(z)\big),$$

and

$$\begin{aligned} &u\big(f_1(\mathbf{l}), f_2(\mathbf{l}), \ldots, f_m(\mathbf{l})\big) \\ &\quad \xrightarrow{\shortparallel}_{\Pi_l} u(l_1, l_2, \ldots, l_m) = f(s, v), \\ &u\big(f_1(\mathbf{r}), f_2(\mathbf{r}), \ldots, f_m(\mathbf{r})\big) \\ &\quad \xrightarrow{\shortparallel}_{\Pi_r} u(r_1, r_2, \ldots, r_m) = v. \end{aligned}$$

Hence, $\theta = \{F \mapsto u\}$ unifies $S[s, F]$, as needed. □

We will use also the following fact.

Lemma 8. *For all terms s, if σ unifies $\boldsymbol{S}^0[s, F_0]$ then $Sig(F_0\sigma, s\sigma)$ is a subset of $Sig(\boldsymbol{S}^0[x, F_0])$.*

Proof. Let s and σ unifying $\boldsymbol{S}^0[s, F_0]$ be given. (Note that σ unifies $\boldsymbol{S}^0[s\sigma, F_0]$.) It follows from Lemma 3 that there exists a ground term u such that $F_0\sigma(\mathbf{l})$ reduces in Π_l to $f(s\sigma, u)$, and $F_0\sigma(\mathbf{r})$ reduces in Π_r to u. Let ν be a function symbol not in $Sig(\boldsymbol{S}^0[x, F_0])$, and thus, $\nu \notin Sig(\Pi_l, \Pi_r)$. Suppose $\nu \in Sig(F_0\sigma)$. Then ν occurs the same positive number of times in $f(s\sigma, u)$ and in u, and thus not in $s\sigma$. Moreover, ν must occur at the same positions in $f(s\sigma, u)$ and in u, which is impossible. If $\nu \in Sig(s\sigma)$ then $\nu \in Sig(F_0\sigma)$, which we have just proved to be impossible. □

5. Main construction

Our main tool is the following lemma.

Lemma 9. *One can effectively construct two simple second-order equations*:

$$\begin{aligned} &S_1[x, F]: \quad F(l_1, \ldots, l_m) \approx f\big(x, F(r_1, \ldots, r_m)\big), \\ &S_2[y, G]: \quad G(u_1, \ldots, u_n) \approx g\big(y, G(v_1, \ldots, v_m)\big) \end{aligned}$$

such that the problem of unifiability of $\{S_1[s, F], S_2[F(f_1(a), \ldots, f_m(a)), G]\}$ for a given ground term s, where $a, f_1, \ldots, f_n$ are symbols in $Sig(S_2[y, G])$ but not in $Sig(S_1[x, F])$, is undecidable.

Proof. A corollary of [12, Theorem 4] (or [9, Theorem 14]) and Lemma 1. □

Let the equations given by Lemma 9 be fixed. We let $\boldsymbol{l}, \boldsymbol{r}$, $\mathbf{l}$, and $\mathbf{r}$ be as above, and we write $\boldsymbol{u}$ for the sequence $u_1, \ldots, u_n$, and $\boldsymbol{v}$ for the sequence $v_1, \ldots, v_n$. Let g_i, for $1 \leqslant i \leqslant n$, be fresh unary function symbols, and let $\mathbf{u}$ and $\mathbf{v}$ be new constants. Intuitively, the g_i's are new "projection functions" and $\mathbf{u}(\mathbf{v})$ encodes the sequence $\boldsymbol{u}(\boldsymbol{v})$. We define the following sets of equations:

$$\boldsymbol{S}_1^0[x, F_0]: \quad \begin{cases} F_0(\mathbf{l}) \approx F_1(f_1(\mathbf{l})), \\ F_0(\mathbf{r}) \approx F_1'(f_1(\mathbf{r})), \\ F_i(l_i) \approx F_{i+1}(f_{i+1}(\mathbf{l})) \\ F_i'(r_i) \approx F_{i+1}'(f_{i+1}(\mathbf{r})) \\ F_m(l_m) \approx f(x, F_m'(r_m)); \end{cases} \quad (1 \leqslant i < m),$$

$$\boldsymbol{S}_2^0[y, G_0]: \quad \begin{cases} G_0(\mathbf{u}) \approx G_1(g_1(\mathbf{u})), \\ G_0(\mathbf{v}) \approx G_1'(g_1(\mathbf{v})), \\ G_i(u_1) \approx G_{i+1}(g_{i+1}(\mathbf{u})) \\ G_i'(v_i) \approx G_{i+1}'(g_{i+1}(\mathbf{v})) \\ G_n(u_n) \approx g(y, G_n'(v_n)). \end{cases} \quad (1 \leqslant i < n),$$

The main result of this paper is due to the following lemma.

Lemma 10. *For any ground term s such that $\mathbf{l}, \mathbf{r} \notin Sig(s)$, the equation system $\{S_1[s, F], S_2[F(f_1(a), \ldots, f_m(a)), G]\}$ is unifiable if and only if the system $\boldsymbol{S}_1^0[s, F_0] \cup \boldsymbol{S}_2^0[F_0(a), G_0]$ is unifiable.*

Proof. Let s be given.

($\Rightarrow$) Assume that θ unifies $\{S_1[s, F], S_2[F(f_1(a), \ldots, f_m(a)), G]\}$ and let $\sigma = \{F_0 \mapsto \lambda x.(F\theta)(f_1(x), \ldots, f_m(x))\}$. It follows from Lemma 2 applied twice, that $\boldsymbol{S}_1^0[s, F_0\sigma] \cup \boldsymbol{S}_2^0[F_0\sigma(a), G_0]$ is unifiable.

($\Leftarrow$) Assume that σ unifies $\boldsymbol{S}_1^0[s, F_0] \cup \boldsymbol{S}_2^0[F_0(a), G_0]$. Since σ unifies $\boldsymbol{S}_2^0[F_0(a), G_0]$, it follows from Lemma 8 that $\mathbf{l}, \mathbf{r} \notin \mathit{Sig}(F_0\sigma)$, and since σ unifies $\boldsymbol{S}_1^0[s, F_0]$, it follows from Lemma 8 that $\mathbf{u}, \mathbf{v} \notin \mathit{Sig}(F_0\sigma)$. Hence, it follows from σ unifying $\boldsymbol{S}_1^0[s, F_0]$ and Lemma 7, that there is a closed m-ary term $F\theta$ such that

$$F_0\sigma = \lambda x.F\theta\big(f_1(x), \ldots, f_m(x)\big)$$

and θ unifies $S_1[s, F]$. It follows from Lemma 6 together with the facts that $\boldsymbol{S}_2^0[F_0\sigma(a), G_0]$ is unifiable and that $\mathbf{u}, \mathbf{v} \notin \mathit{Sig}(F_0\sigma(a))$, that $S_2[F_0\sigma(a), G]$ is unifiable. Consequently, $\{S_1[s, F], S_2[F(f_1(a), \ldots, f_m(a)), G]\}$ is unifiable. □

6. Final reduction

It was shown by Levy and Veanes [8,9] that any system of second-order equations can be transformed in a straightforward manner into an equivalent system of second-order equations that uses just a *single second-order variable*, where the transformation preserves *simplicity* and *maximal arity of the second-order variables*.

Lemma 11. *Let $\boldsymbol{S}[x]$ be a system of second-order equations. One can effectively construct a system $\boldsymbol{S}'[x]$ from $\boldsymbol{S}[x]$ such that $\boldsymbol{S}'[x]$ includes a single second-order variable of arity equal to the maximal arity of second-order variables in $\boldsymbol{S}[x]$, and for all ground terms t, $\boldsymbol{S}'[t]$ is unifiable if and only if $\boldsymbol{S}[t]$ is unifiable. Moreover, if $\boldsymbol{S}[x]$ is simple then $\boldsymbol{S}'[x]$ is simple.*

Proof. Immediate corollary of [9, Theorem 9]. □

Furthermore, a single binary function symbol and a single constant are sufficient to encode arbitrary nonmonadic signatures. Every function symbol f of arity n can be encoded by a closed n-ary term $\bar{f} = \lambda x_1 \cdots x_n.t$, where each x_i occurs exactly once in t, by using a single binary function symbol and a single constant [3]. This encoding is extended to terms, equations, and systems of equations in the usual way, e.g., $\overline{F(f(c, g(x)))}$ is the term $F(\bar{f}(\bar{c}, \bar{g}(x)))$. Note that the variables are not affected by the encoding. Obviously, all the important properties (such as simplicity) are preserved.

Lemma 12. *For all systems $\boldsymbol{S}$ of simple equations, $\boldsymbol{S}$ is unifiable if and only if $\overline{\boldsymbol{S}}$ is unifiable.*

Proof. A special case of [3, Lemma 6.5]. □

The main result of this paper follows from Lemmas 9–12. Let $L^{(n)}$ denote a second-order language consisting of one constant, one binary function symbol, one unary second-order variable, and n first-order variables.

Theorem 13. *There is an integer n and a system $\boldsymbol{S}[x]$ of simple second-order equations in $L^{(n)}$, such that, the problem of unifiability of $\boldsymbol{S}[t]$ for a given ground term t in $L^{(n)}$ is undecidable.*

Proof. Construct $\boldsymbol{S}'[x]$ from $\boldsymbol{S}_1^0[x, F_0] \cup \boldsymbol{S}_2^0[F_0(a), G_0]$ by using Lemma 11 and let $\boldsymbol{S}[x] = \overline{\boldsymbol{S}'[x]}$. It follows from Lemmas 10–12 that, for all ground terms s containing neither $\mathbf{l}$ nor $\mathbf{r}$, $\{S_1[s, F], S_2[F(f_1(a), \ldots, f_m(a)), G]\}$ is unifiable if and only if $\boldsymbol{S}[\bar{s}]$ is unifiable. Hence, the claim follows from Lemma 9. □

An immediate consequence of Theorem 13 is that second-order unification is undecidable for simple equations in all nonmonadic second-order term languages if the language has at least n first-order variables for some large enough n. Note that all occurrences of the second-order variable in the system $\boldsymbol{S}[x]$ constructed in Theorem 13 have fixed ground arguments. Hence, it is not possible to obtain decidability even if we fix the maximum number of occurrences of variables and the maximum size of the arguments of the second-order variables.

Acknowledgements

I thank Harald Ganzinger for valuable comments.

References

[1] A. Degtyarev, Yu. Gurevich, A. Voronkov, Herbrand's theorem and equational reasoning: Problems and solutions, in: Bulletin of the European Association for Theoretical Computer Science, Vol. 60, October 1996, The "Logic in Computer Science" column.

[2] A. Degtyarev, A. Voronkov, Reduction of second-order unification to simultaneous rigid E-unification, UPMAIL Technical Report 109, Uppsala University, Computing Science Department, June 1995.

[3] W.M. Farmer, Simple second-order languages for which unification is undecidable, Theoret. Comput. Sci. 87 (1991) 25–41.

[4] J.H. Galler, S. Raatz, W. Snyder, Theorem proving using rigid E-unification: Equational matings, in: Proc. IEEE Conference on Logic in Computer Science (LICS), IEEE Computer Society Press, 1987, pp. 338–346.

[5] H. Ganzinger, F. Jacquemard, M. Veanes, Rigid reachability, in: J. Hsiang, A. Ohori (Eds.), Advances in Computing Science—ASIAN'98, 4th Asian Computing Science Conference, Manila, The Philippines, December 1998, Proceedings, Lecture Notes in Comput. Sci., Vol. 1538, Springer, Berlin, 1998, pp. 4–21.

[6] W.D. Goldfarb, The undecidability of the second-order unification problem, Theoret. Comput. Sci. 13 (1981) 225–230.

[7] J. Levy, Decidable and undecidable second-order unification problems, in: T. Nipkow (Ed.), Rewriting Techniques and Applications, 9th International Conference, RTA-98, Tsukuba, Japan, March/April 1998, Proceedings, Lecture Notes in Comput. Sci., Vol. 1379, Springer, Berlin, 1998, pp. 47–60.

[8] J. Levy, M. Veanes, On unification problems in restricted second-order languages, in: Annual Conference of the European Association for Computer Science Logic (CSL'98), Brno, Czech Republic, 1998.

[9] J. Levy, M. Veanes, On the undecidability of second-order unification, Inform. and Comput., to appear.

[10] R. Parikh, Some results on the lengths of proofs, Trans. Amer. Math. Soc. 177 (1973) 29–36.

[11] A. Schubert, Second-order unification and type inference for Church-style polymorphism, Technical Report TR 97-02 (239), Institute of Informatics Warsaw University, January 1997, Short version appears in: Proc. POPL'98.

[12] M. Veanes, The relation between second-order unification and simultaneous rigid E-unification, in: Proc. 13th Annual IEEE Symposium on Logic in Computer Science, June 21–24, 1998, Indianapolis, Indiana (LICS'98), IEEE Computer Society Press, 1998, pp. 264–275.