The Undecidability of Simultaneous Rigid *E*-Unification with Two Variables

Margus Veanes

Uppsala University Computing Science Department
P.O. Box 311, S-751 05 Uppsala, Sweden

Abstract. Recently it was proved that the problem of simultaneous rigid E-unification, or SREU, is undecidable. Here we show that 4 rigid equations with ground left-hand sides and 2 variables already imply undecidability. As a corollary we improve the undecidability result of the $\exists^*$-fragment of intuitionistic logic with equality. Our proof shows undecidability of a very restricted subset of the $\exists\exists$-fragment. Together with other results, it contributes to a complete characterization of decidability of the prenex fragment of intuitionistic logic with equality, in terms of the quantifier prefix.

1 Introduction

Recently it was proved that the problem of simultaneous rigid E-unification (SREU) is undecidable [11]. This (quite unexpected) undecidability result has lead to other new undecidability results, in particular that the $\exists^*$-fragment of intuitionistic logic with equality is undecidable [13,15]. Here we show that 4 rigid equations[1] with ground left-hand sides and 2 variables already imply undecidability. As a corollary we improve the undecidability result of the $\exists^*$-fragment of intuitionistic logic with equality. Namely that the $\exists\exists$-fragment is undecidable. In fact, our proof shows undecidability of a very restricted subset of the $\exists\exists$-fragment. Together with the result that the $\exists$-fragment is decidable [6], it contributes to a complete characterization of decidability of the prenex fragment of intuitionistic logic with equality, in terms of the quantifier prefix.

1.1 Background of SREU

Simultaneous rigid E-unification was proposed by Gallier, Raatz and Snyder [21] as a method for automated theorem proving in classical logics with equality. It can be used in automatic proof methods, like semantic tableaux [18], the connection method [3] or the mating method [1], model elimination [32], and others that are based on the Herbrand theorem, and use the property that a formula φ is valid (i.e., $\neg\varphi$ is unsatisfiable) iff all paths through a matrix of φ are inconsistent. This property was first recognized by Prawitz [38] (for first-order logic without equality) and later by Kanger [28] (for first-order logic with equality).

[1] It has been noted by Gurevich and Veanes that 3 rigid equations suffices [25].

In first-order logic with equality, the problem of checking the inconsistency of the paths results in SREU. Before SREU was proved to be undecidable, there were several faulty statements of its decidability, e.g. [19,24].

1.2 Outline of the Paper

In Section 2 we introduce the notations used in this paper and briefly explain the background material. In Section 3 we prove the main result of this paper (Theorem 8), that implies immediately the undecidability result of a very restricted case of SREU. In Section 4 we use this result to obtain undecidability of a restricted subset of the ∃∃-fragment of intuitionistic logic with equality. Finally, the current status about SREU is summarized and some open problems are listed in Section 5.

2 Preliminaries

We introduce here the main notions and definitions used in this paper. Given a signature Σ, i.e., a set of function symbols with fixed arities, the set of all ground (or closed) terms over Σ is denoted by $\mathcal{T}_\Sigma$. Unless otherwise stated it is always assumed that Σ is nonempty, finite and includes at least one constant (function symbol of arity 0). We also assume certain familiarity with basic notions from term rewriting [16], regarding ground rewriting systems. By a *substitution* we understand a function from variables to *ground* terms and a substitution is always denoted by θ. An application of θ on a variable x is written as $x\theta$ instead of $\theta(x)$.

2.1 Finite Tree Automata

Finite tree automata [17,39] is a natural generalization of classical finite automata to automata that accept or recognize trees of symbols, not just strings. Here we adopt a definition of tree automata based on rewrite rules. This definition is used for example by Dauchet [4].

▶ A *tree automaton* or *TA* is a quadruple $A = (Q, \Sigma, R, F)$ where
- Q is a finite set of constants called *states*,
- Σ is a *signature* or an *input alphabet* disjoint from Q,
- R is a set of *rules* of the form $\sigma(q_1, \ldots, q_n) \rightarrow q$, where $\sigma \in \Sigma$ has arity $n \geq 0$ and $q, q_1, \ldots, q_n \in Q$,
- $F \subseteq Q$ is the set of *final states.*

A is called a *deterministic* TA or DTA if there are no two different rules in R with the same left-hand side.

Note that if A is deterministic then R is a reduced set of ground rewrite rules, i.e., for any rule $s \rightarrow t$ in R t is irreducible and s is irreducible with respect to $R \setminus \{s \rightarrow t\}$. So R is a ground canonical rewrite system. In this context terms are also called trees. A set of terms (or trees) is called a *forest.*

▶ The forest *recognized* by a TA $A = (Q, \Sigma, R, F)$ is the set

$$T(A) = \{ t \in \mathcal{T}_\Sigma \mid (\exists q \in F)\, t \xrightarrow{*}_R q \}.$$

A forest is called *recognizable* if it is recognized by some TA.

We assume that the reader is familiar with classical automata theory and we follow Hopcroft and Ullman [27] in that respect.

2.2 Simultaneous Rigid E-Unification

A *rigid equation* is an expression of the form $E \models_\forall s \approx t$ where E is a finite set of equations, called the *left-hand side* of the rigid equation, and s and t are arbitrary terms. A *system* of rigid equations is a finite set of rigid equations. A substitution θ is a *solution of* or *solves* a rigid equation $E \models_\forall s \approx t$ if

$$\vdash \quad (\bigwedge_{e \in E} e\theta) \Rightarrow s\theta \approx t\theta,$$

and θ is a *solution of* or *solves* a system of rigid equations if it solves each member of that system. Here $\vdash$ is classical or intuitionistic provability (for this class of formulas both provabilities coincide). The problem of solvability of systems of rigid equations is called *simultaneous rigid E-unification* or SREU for short. Solvability of a single rigid equation is called *rigid E-unification.* Rigid E-unification is known to be decidable, in fact NP-complete [20]. The following simple lemma is useful.

Lemma 1. *Let $A = (Q, \Sigma, R, F)$ be a DTA, f a binary function symbol, and c_1 and c_2 constants not in Q or Σ. There is a set of ground equations E such that for all θ such that $x\theta \in \mathcal{T}_\Sigma$, θ solves $E \models_\forall f(c_1, x) \approx c_2$ iff $x\theta \in T(A)$.*

Proof. Let $E = R \cup \{ f(c_1, q) \to c_2 \mid q \in F \}$. It follows easily that E is a canonical rewrite system, and since c_2 is irreducible with respect to E we have in particular for all $t \in \mathcal{T}_\Sigma$, that (cf [16, Section 2.4])

$$E \vdash f(c_1, t) \approx c_2 \quad \Leftrightarrow \quad f(c_1, t) \xrightarrow{*}_E c_2.$$

But

$$f(c_1, t) \xrightarrow{*}_E c_2 \quad \Leftrightarrow \quad (\exists q \in F)\, t \xrightarrow{*}_R q.$$

The rest is obvious. □

3 Minimal Undecidable Case of SREU

We present yet another proof of the undecidability of SREU. At the end of this section we give a brief summary of the other proofs. The main idea behind this proof is based on a technique that we call *shifted pairing* after Plaisted [37].

The idea is to express repetition explicitly by a sequence of strings (like IDs of a TM). The first string of the sequence fulfills some initial conditions, the last string some final conditions and another sequence is used to check that the consequtive strings of the first sequence satisfy some relationship (like validity of a computation step).

A similar technique was used already by Goldfarb in the proof of the undecidability of second-order unification [23] (which was by reduction of Hilbert's tenth problem) and later, adopted from that proof, also in the third proof of the undecidability of SREU by Degtyarev and Voronkov [13] (which was also by reduction of Hilbert's tenth problem). There the key point was to explicitly represent the history of a multiplication process.

Shifted pairing bears also certain similarities to the technique that is used to prove that any recursively enumerable set of strings is given by the intersection of two (deterministic) context free languages [27, Lemma 8.6].

3.1 Overview of the Construction

We consider a fixed Turing machine M,

$$M = (Q_M, \varSigma_{\mathrm{in}}, \varSigma_{\mathrm{tape}}, \delta, q_0, \bar{b}, \{q_{\mathrm{acc}}\}).$$

We can assume, without loss of generality, that the final ID of M is simply q_{acc} (and that $q_0 \neq q_{\mathrm{acc}}$), i.e., the tape is always empty when M enters the final state. We construct a system $S_M(x, y)$ of four rigid equations:

$$\begin{aligned} S_M(x,y) = \{\; & E_{\mathrm{id}} \models_{\forall} c'_{\mathrm{id}} \centerdot x \approx c_{\mathrm{id}}, && (1)\\ & E_{\mathrm{mv}} \models_{\forall} c'_{\mathrm{mv}} \centerdot y \approx c_{\mathrm{mv}}, && (2)\\ & \varPi_1 \models_{\forall} x \approx y, && (3)\\ & \varPi_2 \models_{\forall} x \approx (q_0 \centerdot e_0) \centerdot y \,\} && (4) \end{aligned}$$

where all the left-hand sides are ground, c'_{id}, c_{id}, c'_{mv} and c_{mv} are constants, and $q_0 \centerdot e_0$ is a word that represents the initial ID of M with empty input string (ϵ). We prove that M accepts ϵ iff S_M is solvable. This establishes the undecidability result because all the steps in the construction are effective.

The main idea behind the rigid equations is roughly as follows. Assume that there is a substitution θ that solves the system.

- From θ being a solution of (1), it follows that $x\theta$ represents a sequence
$$(v_0, v_1, \ldots, v_m)$$
of IDs of M, where v_m is the final ID of M.
- From θ being a solution of (2), it follows that $y\theta$ represents a sequence
$$((w_0, w_0^+), (w_1, w_1^+), \ldots, (w_n, w_n^+))$$
of *moves* of M, i.e., $w_i \vdash_M w_i^+$ for $0 \leq i \leq n$.

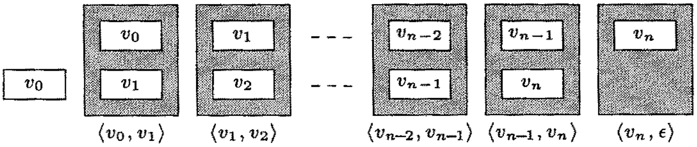
v_0
v_0
v_1
v_1
v_2
v_{n-2}
v_{n-1}
v_{n-1}
v_n
v_n
$\langle v_0, v_1\rangle$
$\langle v_1, v_2\rangle$
$\langle v_{n-2}, v_{n-1}\rangle$
$\langle v_{n-1}, v_n\rangle$
$\langle v_n, \epsilon\rangle$

Fig. 1. $(\langle v_0, v_1\rangle, \langle v_1, v_2\rangle, \ldots, \langle v_n, \epsilon\rangle)$ is a shifted pairing of $(v_0, v_1, \ldots, v_n)$.

- From θ being a solution of (3) it follows that $n = m$ and $v_i = w_i$ for $0 \leq i \leq m$.
- And finally, from θ being a solution of (4) it follows that v_0 is the initial ID and $v_i = w_{i-1}^{+}$ for $1 \leq i \leq m$.

The combination of the last two points is the so-called "shifted pairing" technique. This is illustrated by Figure 1. The outcome of this shifted pairing is that $x\theta$ is a valid computation of M with input ϵ, and thus M accepts ϵ. Conversely, if M accepts ϵ then it is easy to construct a solution of the system. We now give a formal construction of the above idea.

3.2 Words and Trains

Words are certain terms that we choose to represent strings with, and trains are certain terms that we choose to represent sequences of strings with. We use the letters v and w to stand for strings of constants. Let $\centerdot$ be a binary function symbol. We write it in infix notation and assume that it associates to the right. For example $t_1 \centerdot t_2 \centerdot t_3$ stands for the term $\centerdot(t_1, \centerdot(t_2, t_3))$.

▶ We say that a (ground) term t is a *c-word* if it has the form

$$a_1 \centerdot a_2 \centerdot \cdots \centerdot a_n \centerdot c$$

for some $n \geq 0$ where each a_i and c is a constant. A *word* is a c-word for some constant c.

We use the following convenient shorthand notation for words. Let t be the word $a_1 \centerdot a_2 \centerdot \cdots \centerdot a_n \centerdot c$ and v the string $a_1 a_2 \cdots a_n$. We write $v \centerdot c$ for t and say that t *represents* v.

▶ *A* term t is called a *c-train* if it has the form

$$t_1 \centerdot t_2 \centerdot \cdots \centerdot t_n \centerdot c$$

for some $n \geq 0$ where each t_i is a word and c is a constant. If $n = 0$ then t is said to be *empty*. The t_i's are called the *words of t*. A *train* is a c-train for some constant c.

By the *pattern* of a train

$$(v_1 \centerdot c_1) \centerdot (v_2 \centerdot c_2) \centerdot \cdots \centerdot (v_n \centerdot c_n) \centerdot c$$

we mean the string $c_1 c_2 \cdots c_n$. Let $\mathcal{V} = \{V_i\}_{i \in I}$ be a finite family of regular sets of strings over a finite set Σ of constants, where I is a set of constants disjoint from Σ. Let U be a regular set of strings over I and let c be a constant not in Σ or I.

- ▶ We let $\mathrm{Tn}(\mathcal{V}, U, c)$ denote the set of all c-trains t such that the pattern of t is in U and, for $i \in I$, each i-word of t represents a string in V_i.

Example 2. Consider the set $\mathrm{Tn}(\{V_a, V_b, V_c\}, ab^*c, \Lambda)$. This is the set of all Λ-trains t such that the first word of t is an a-word representing a string in V_a, the last word of t is a c-word representing a string in V_c and the middle ones (if any) are b-words representing strings in V_b.

We say that a set of trains *has a regular pattern* if it is equal to some set $\mathrm{Tn}(\mathcal{V}, U, c)$ with $\mathcal{V}$, U and c as above. The following theorem is proved in Veanes [40].

Theorem 3 (Train Theorem). *Any set of trains with a regular pattern is recognizable and a DTA that recognizes this set can be obtained effectively.*

3.3 Representing IDs and Moves

Recall that an *ID* of M is any string in $\Sigma^*_{\mathrm{tape}} Q_M \Sigma^*_{\mathrm{tape}}$ that doesn't end with a blank ($\mathfrak{b}$). Let us assign arity 0 to all the tape symbols (Σ_{tape}) and all the states (Q_M) of M, and let Σ denote the signature consisting of all those constants, the binary function symbol $\centerdot$ and four new constants e_0, e_1, e_{acc} and Λ.

ID-trains IDs are represented by e-words, where e is one of e_0, e_1 or e_{acc}. In particular, the initial ID is represented by the word $q_0 \centerdot e_0$. The final ID is represented by the word $q_{\mathrm{acc}} \centerdot e_{\mathrm{acc}}$ and all the other IDs are represented by corresponding e_1-words. The term

$$(q_0 \centerdot e_0) \centerdot (v_1 \centerdot e_1) \centerdot (v_2 \centerdot e_1) \centerdot \cdots \centerdot (v_n \centerdot e_1) \centerdot (q_{\mathrm{acc}} \centerdot e_{\mathrm{acc}}) \centerdot \Lambda$$

is called an *ID-train.* By using Theorem 3 let

$$A_{\mathrm{id}} = (Q_{\mathrm{id}}, \Sigma, R_{\mathrm{id}}, F_{\mathrm{id}})$$

be a DTA that recognizes the set of all ID-trains. Let c'_{id} and c_{id} be new constants and (1) the rigid equation provided by Lemma 1, i.e., for all θ such that $x\theta \in \mathcal{T}_\Sigma$,

$$\theta \text{ solves (1)} \quad \Leftrightarrow \quad x\theta \in T(A_{\mathrm{id}}).$$

Move-trains Let c_{ab} be a new constant for each pair of constants a and b in the set $\Sigma_{\mathrm{tape}} \cup Q_M$. Let also e_2 and Λ' be new constants. Let now Γ be a signature that consists of $\bullet$, all those c_{ab}'s, e_2 and Λ'.

For and ID w of M we let w^+ denote the successor of w with respect to the transition function of M. For technical reasons it is convenient to let $q_{\mathrm{acc}}^+ = \epsilon$, i.e., the successor of the final ID is the empty string. The pair (w, w^+) is called a *move*. Let $w = a_1 a_2 \cdots a_m$ and $w^+ = b_1 b_2 \cdots b_n$ for some $m \geq 1$ and $n \geq 0$. Note that $n \in \{m-1, m, m+1\}$. Let $k = \max(m, n)$. If $m < n$ let $a_k = \bar{b}$ and if $n < m$ let $b_k = \bar{b}$, i.e., pad the shorter of the two strings with a blank at the end.

▶ We write $\langle w, w^+ \rangle$ for the string $c_{a_1 b_1} c_{a_2 b_2} \cdots c_{a_k b_k}$ and say that the e_2-word $\langle w, w^+ \rangle \bullet e_2$ *represents* the move (w, w^+). By a *move-train* we mean any term

$$t = t_1 \bullet t_2 \bullet \cdots \bullet t_n \bullet \Lambda'$$

such that each t_i represents a move.

Example 4. Take $\Sigma_{\mathrm{in}} = \{0, 1\}$, and let $\mathrm{q}, \mathrm{p} \in Q_M$. Assume that the transition function δ is such that, when the tape head points to a blank and the state is q then a 1 is written to the tape, the tape head moves left and M enters state p, i.e., $\delta(\mathrm{q}, \bar{b}) = (\mathrm{p}, 1, \mathrm{L})$. Let the current ID be 00q, i.e., the tape contains the string 00 and the tape head points to the bank following the last 0. So $(00\mathrm{q}, 0\mathrm{p}01)$ is a move. This move is represented by the term $c_{00} \bullet c_{0\mathrm{p}} \bullet c_{\mathrm{q}0} \bullet c_{\bar{b}1} \bullet e_2$, or $\langle 00\mathrm{q}, 0\mathrm{p}01 \rangle \bullet e_2$ if we use the above notation.

It is easy to see that the set of all strings $\langle w, w^+ \rangle$ where w is an ID, is a regular set. By using Theorem 3 let

$$A_{\mathrm{mv}} = (Q_{\mathrm{mv}}, \Gamma, R_{\mathrm{mv}}, F_{\mathrm{mv}})$$

be a DTA that recognizes the set of all move-trains. Let c'_{mv} and c_{mv} be new constants and (2) the rigid equation provided by Lemma 1, i.e., for all θ such that $y\theta \in \mathcal{T}_\Gamma$,

$$\theta \text{ solves (2)} \quad \Leftrightarrow \quad y\theta \in T(A_{\mathrm{mv}}).$$

3.4 Shifted Pairing

We continue with the contruction of S_M. What has remained to define is Π_1 and Π_2. These are defined as sets of equations corresponding to the following canonical rewrite systems.

$$\begin{aligned}
\Pi_1 = {} & \{\, \boldsymbol{c_{ab}} \to \boldsymbol{a} \mid a, b \in \Sigma_{\mathrm{tape}} \cup Q_M \,\} \cup \\
& \{\, e_1 \to e_0,\ e_2 \to e_0,\ e_{\mathrm{acc}} \to e_0,\ \Lambda' \to \Lambda,\ \bar{b} \bullet e_0 \to e_0 \,\} \\
\Pi_2 = {} & \{\, \boldsymbol{c_{ab}} \to \boldsymbol{b} \mid a, b \in \Sigma_{\mathrm{tape}} \cup Q_M \,\} \cup \\
& \{\, e_1 \to e_0,\ e_2 \to e_0,\ e_{\mathrm{acc}} \to e_0,\ \Lambda' \to \Lambda,\ \bar{b} \bullet e_0 \to e_0,\ \boldsymbol{e_0 \bullet \Lambda \to \Lambda} \,\}
\end{aligned}$$

The differences between Π_1 and Π_2 are indicated in boldface.

Lemma 5. *If θ solves (3) and (4) then $x\theta, y\theta \in \mathcal{T}_{\Sigma \cup \Gamma}$.*

Proof. By induction on the size of $x\theta$ [40]. □

Lemma 6. *If θ solves $S_M(x,y)$ then $x\theta \in \mathcal{T}_\Sigma$ and $y\theta \in \mathcal{T}_\Gamma$.*

Proof. Assume that θ solves $S_M(x,y)$. Obviously $x\theta \in \mathcal{T}_{\Sigma \cup Q_{\mathrm{id}}}$ since θ solves (1). By Lemma 5 we know also that $x\theta \in \mathcal{T}_{\Sigma \cup \Gamma}$. But Σ, Γ and Q_{id} don't share any constants. So $x\theta \in \mathcal{T}_\Sigma$. A similar argument shows that $y\theta \in \mathcal{T}_\Gamma$. □

Lemma 7. *If θ solves $S_M(x,y)$ then $x\theta$ is an ID-train and $y\theta$ is a move-train.*

Proof. By Lemma 6, the definition of A_{id} and A_{mv} and Lemma 1. □

We have now reached the main theorem of this paper.

Theorem 8. *$S_M(x,y)$ is solvable iff M accepts ϵ.*

Proof. ($\Rightarrow$) Let θ be a substitution that solves $S_M(x,y)$. By using Lemma 7 we get that $x\theta$ and $y\theta$ have the following form:

$$\begin{aligned} x\theta &= (v_0 \centerdot e_0) \centerdot (v_1 \centerdot e_1) \centerdot \cdots \centerdot (v_{m-1} \centerdot e_1) \centerdot (v_m \centerdot e_{\mathrm{acc}}) \centerdot \Lambda \\ y\theta &= (\langle w_0, w_0^+ \rangle \centerdot e_2) \centerdot (\langle w_1, w_1^+ \rangle \centerdot e_2) \centerdot \cdots \centerdot (\langle w_n, w_n^+ \rangle \centerdot e_2) \centerdot \Lambda' \end{aligned}$$

where all the v_i's and w_i's are IDs of M, $v_0 = q_0$ and $v_m = q_{\mathrm{acc}}$.

Since θ solves (3) it follows that the normal forms of $x\theta$ and $y\theta$ under Π_1 must coincide. The normal form of $x\theta$ under Π_1 is

$$(v_0 \centerdot e_0) \centerdot (v_1 \centerdot e_0) \centerdot \cdots \centerdot (v_{m-1} \centerdot e_0) \centerdot (v_m \centerdot e_0) \centerdot \Lambda.$$

The normal form of $y\theta$ under Π_1 is

$$(w_0 \centerdot e_0) \centerdot (w_1 \centerdot e_0) \centerdot \cdots \centerdot (w_{n-1} \centerdot e_0) \centerdot (w_n \centerdot e_0) \centerdot \Lambda.$$

Note that each term $\langle w_i, w_i^+ \rangle \centerdot e_2$ reduces first to $w_i' \centerdot e_0$ where $w_i' = w_i$ or $w_i' = w_i\mathfrak{b}$. The extra blank at the end is removed with the rule $\mathfrak{b} \centerdot e_0 \to e_0$. So

$$v_0 = q_0, \quad v_n = q_{\mathrm{acc}}, \quad v_i = w_i \ (0 \le i \le n = m). \tag{5}$$

Since θ solves (4) it follows that the normal forms of $x\theta$ and $(q_0 \centerdot e_0) \centerdot y\theta$ under Π_2 must coincide. The normal form of $x\theta$ under Π_2 is the same as under Π_1 because $x\theta$ doesn't contain any constants from Γ and the rule $e_0 \centerdot \Lambda \to \Lambda$ is not applicable. From $w_n = q_{\mathrm{acc}}$ follows that $w_n^+ = \epsilon$ and thus $\langle w_n, w_n^+ \rangle \centerdot e_0 = c_{q_{\mathrm{acc}}\mathfrak{b}} \centerdot e_0$. But

$$(c_{q_{\mathrm{acc}}\mathfrak{b}} \centerdot e_0) \centerdot \Lambda \longrightarrow_{\Pi_2} (\mathfrak{b} \centerdot e_0) \centerdot \Lambda \longrightarrow_{\Pi_2} e_0 \centerdot \Lambda \longrightarrow_{\Pi_2} \Lambda.$$

The normal form of $(q_0 \centerdot e_0) \centerdot y\theta$ under Π_2 is thus

$$(q_0 \centerdot e_0) \centerdot (w_0^+ \centerdot e_0) \centerdot (w_1^+ \centerdot e_0) \centerdot \cdots \centerdot (w_{n-1}^+ \centerdot e_0) \centerdot \Lambda.$$

It follows that

$$w_i^+ = v_{i+1} \ (0 \leq i < n). \tag{6}$$

From (5) and (6) follows that $(v_0, v_1, \ldots, v_n)$ is a valid computation of M, and thus M accepts ϵ.

($\Leftarrow$) Assume that M accepts ϵ. So there exists a valid computation

$$(v_0, v_1, \ldots, v_n)$$

of M where $v_0 = q_0$, $v_n = q_{\mathrm{acc}}$ and $v_i^+ = v_{i+1}$ for $0 \leq i < n$. Let θ be such that $x\theta$ is the corresponding ID-train and $y\theta$ the corresponding move-train. It follows easily that θ solves $S_M(x, y)$. □

The "shifted pairing" technique that is used in Theorem 8 is illustrated in Figure 1.

The following result is an immediate consequence of Theorem 8 because all the constructions involved with it are effective.

Corollary 9. *SREU is undecidable if the left-hand sides are ground, there are two variables and four rigid equations.*

It was observed by Gurevich and Veanes that the two DTAs A_{id} and A_{mv} can be combined into one DTA (by using elementary techniques of finite tree automata theory [22]), and by this way reducing the number of rigid equtions in S_M into *three* [25]. It is still an open question if SREU with *two* rigid equations is decidable.

3.5 Previous Undecidability Proofs of SREU

The first proof of the udecidability of SREU [11] was by reduction of the monadic semi-unification [2] to SREU. This proof was followed by two alternative (more transparent) proofs by the same authors, first by reducing second-order unification to SREU [10,15], and then by reducing Hilbert's tenth problem to SREU [14]. The undecidability of second-order unification was proved by Goldfarb [23]. Reduction of second-order unification to SREU is very simple, showing how close these problem are to each other. Plaisted took the Post's Correspondence Problem and reduced it to SREU [37]. From his proof follows that SREU is undecidable already with ground left-hand sides and three variables. He uses several function symbols of arity 1 and 2.

3.6 Herbrand Skeleton Problem

The *Herbrand skeleton problem of multiplicity* n is a fundamental problem in automated theorem proving [7], e.g., by the method of matings [1], the tableaux method [18], and others. It can can be formulated as follows:

> *Given a quantifier free formula $\varphi(\boldsymbol{x})$, does there exist a sequence of ground terms $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_n$ such that the disjunction $\varphi(\boldsymbol{t}_1) \vee \cdots \vee \varphi(\boldsymbol{t}_n)$ is valid?*

The undecidability of this problem was established recently by Voda and Komara [41] by a technique similar to the one used in the reduction of Hilbert's tenth problem to SREU [14]. Their proof is very complicated and (contrary to their claim) it is shown in Gurevich and Veanes [25] by using a novel logical lemma that the Herbrand skeleton problem of any fixed multiplicity reduces easily to SREU. As a corollary (by using Theorem 8) improving the result in [41], by proving the undecidability of this problem for a restricted Horn fragment of classical logic (where variables occur only positively).

4 Undecidability of the $\exists\exists$-fragment of Intuitionistic Logic with Equality

Undecidability of the $\exists^*$-fragment of intuitionistic logic with equality was established recently by Degtyarev and Voronkov [13,15]. We obtain the following improvement of this result. Let $\vdash_{\mathrm{i}}$ stand for provability in intuitionistic predicate calculus with equality and let $\vdash_{\mathrm{c}}$ stand for provability in classical predicate calculus (with equality).

Theorem 10. *The class of formulas in intuitionistic logic with equality of the form $\exists x \exists y\, \varphi(x, y)$ where φ is quantifier free, and*

- *the language contains (besides constants) a function symbol of arity ≥ 2,*
- *the only connectives in φ are '$\wedge$' and '$\Rightarrow$' and*
- *the antecedents of all implications in φ are closed,*

is undecidable.

Proof. Let $S_M(x, y)$ be the system of rigid equations given by Theorem 8. So

$$S_M(x,y) \quad = \quad \{\, E_i \vDash_{\forall} s_i \approx t_i \mid 1 \leq i \leq 4 \,\},$$

where each E_i is a set of (ground) equations. Let $\psi_i = \bigwedge_{e \in E_i} e$ for $1 \leq i \leq 4$. Note that each ψ_i is closed. Let

$$\varphi(x,y) \quad = \quad \bigwedge_{1 \leq i \leq 4} (\psi_i \Rightarrow s_i \approx t_i).$$

The construction of φ from S_M and thus from M is clearly effective. To prove the theorem it is enough to prove the following statement:

$$\epsilon \in L(M) \quad \Leftrightarrow \quad \vdash_{\mathrm{i}} \exists x \exists y \varphi(x,y).$$

($\Rightarrow$) Assume $\epsilon \in L(M)$. By Theorem 8 there is a substitution θ that solves $S_M(x, y)$. By definition, this means that $\vdash_{\mathrm{c}} \varphi(x\theta, y\theta)$. But

$$\vdash_{\mathrm{c}} \varphi(x\theta, y\theta) \Rightarrow \vdash_{\mathrm{i}} \varphi(x\theta, y\theta)$$

for this particular class of formulas. The rest is obvious.

($\Leftarrow$) Assume that $\vdash_{\mathrm{i}} \exists x \exists y \varphi(x, y)$. By the explicit definabilty property of intuitionistic logic there are ground terms t and s such that $\vdash_{\mathrm{i}} \varphi(t, s)$ and thus $\vdash_{\mathrm{c}} \varphi(t, s)$. Let θ be such that $x\theta = t$ and $y\theta = s$. It follows that θ solves the system $S_M(x, y)$, and thus $\epsilon \in L(M)$ by Theorem 8. □

A closely related problem is the *skeleton instantiation problem* (the problem of existence of a derivation with a given skeleton in a given proof system). Voronkov shows that SREU is polynomially reducible to this problem [42, Theorem 3.12] (where the actual proof system under consideration is a sequent calculus $LJ^{\approx}$ for intuitionistic logic with equality). Moreover, the basic structure of the skeleton is determined by the number of variables in the SREU problem and the number of rigid equations in it. The above corollary implies that this problem is undecidable already for a very restricted class of skeletons.

In Degtyarev, Gurevich, Narendran, Veanes and Voronkov [6] it is proved that the $\exists$-fragment of intuitionistic logic with equality is decidable. For further results about the prenex fragment see Degtyarev, Matiyasevich and Voronkov [9], Degtyarev and Voronkov [12] and Voronkov [43,42]. Decidabilty problems for some other fragments of intuitionistic logic with and without equality were studied by Orevkov [35,36], Mints [34] and Lifschitz [31].

5 Current Status and Open problems

The first decidability proof of rigid E-unification is given in Gallier, Narendran, Plaisted and Snyder [20]. Recently a simpler proof, without computational complexity considerations, has been given by de Kogel [5]. We start with the *solved cases*:

- Rigid E-unification with ground left-hand side is NP-complete [30]. Rigid E-unification in general is NP-complete and there exist finite complete sets of unifiers [19,20].
- SREU with one variable and a fixed number of rigid equations is P-complete [6].
- If all function symbols have arity ≤ 1 (the *monadic* case) then SREU is PSPACE-hard [24]. If only one unary function symbol is allowed then the problem is decidable [8,9]. If only constants are allowed then the problem is NP-complete [9] if there are at least two constants.
- About the monadic case it is known that if there are more than 1 unary function symbols then SREU is decidable iff it is decidable with just 2 unary function symbols [9].
- If the left-hand sides are ground then the monadic case is decidable [26]. Monadic SREU with one variable is PSPACE-complete [26].
- The word equation solving [33] (i.e., unification under associativity), which is an extremely hard problem with no interesting known computational complexity bounds, can be reduced to monadic SREU [8].
- Monadic SREU is equivalent to a non-trivial extension of word equations [26].
- Monadic SREU is equivalent to the decidability problem of the prenex fragment of intuitionistic logic with equality with function symbols of arity ≤ 1 [12].
- In general SREU is undecidable [11]. Moreover, SREU is undecidable under the following restrictions:
  - The left-hand sides of the rigid equations are ground [37].

  - Furthermore, there are only two variables and three rigid equations with fixed ground left-hand sides [25].

- SREU with one variable is decidable, in fact EXPTIME-complete [6].

Note also that SREU is decidable when there are no variables, since each rigid equation can be decided for example by using any congruence closure algorithm or ground term rewriting technique. Actually, the problem is then P-complete because the uniform word problem for ground equations is P-complete [29]. The *unsolved cases* are:

? Decidability of monadic SREU [26].
? Decidability of SREU with *two* rigid equations.

Both problems are highly non-trivial.

References

1. P.B. Andrews. Theorem proving via general matings. *Journal of the Association for Computing Machinery*, 28(2):193–214, 1981.
2. M. Baaz. Note on the existence of most general semi-unifiers. In *Arithmetic, Proof Theory and Computation Complexity*, volume 23 of *Oxford Logic Guides*, pages 20–29. Oxford University Press, 1993.
3. W. Bibel. *Deduction. Automated Logic.* Academic Press, 1993.
4. 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.
5. E. De Kogel. Rigid E-unification simplified. In P. Baumgartner, R. Hähnle, and J. Posegga, editors, *Theorem Proving with Analytic Tableaux and Related Methods*, number 918 in Lecture Notes in Artificial Intelligence, pages 17–30, Schloß Rheinfels, St. Goar, Germany, May 1995.
6. A. Degtyarev, Yu. Gurevich, P. Narendran, M. Veanes, and A. Voronkov. The decidability of simultaneous rigid E-unification with one variable. UPMAIL Technical Report 139, Uppsala University, Computing Science Department, March 1997.
7. A. Degtyarev, Yu. Gurevich, and A. Voronkov. Herbrand's theorem and equational reasoning: Problems and solutions. In *Bulletin of the European Association for Theoretical Computer Science*, volume 60. October 1996. The "Logic in Computer Science" column.
8. A. Degtyarev, Yu. Matiyasevich, and A. Voronkov. Simultaneous rigid E-unification is not so simple. UPMAIL Technical Report 104, Uppsala University, Computing Science Department, April 1995.
9. A. Degtyarev, Yu. Matiyasevich, and A. Voronkov. Simultaneous rigid E-unification and related algorithmic problems. In *Eleventh Annual IEEE Symposium on Logic in Computer Science (LICS'96)*, pages 494–502, New Brunswick, NJ, July 1996. IEEE Computer Society Press.
10. A. Degtyarev and A. Voronkov. Reduction of second-order unification to simultaneous rigid E-unification. UPMAIL Technical Report 109, Uppsala University, Computing Science Department, June 1995.

11. A. Degtyarev and A. Voronkov. Simultaneous rigid E-unification is undecidable. UPMAIL Technical Report 105, Uppsala University, Computing Science Department, May 1995.
12. A. Degtyarev and A. Voronkov. Decidability problems for the prenex fragment of intuitionistic logic. In *Eleventh Annual IEEE Symposium on Logic in Computer Science (LICS'96)*, pages 503–512, New Brunswick, NJ, July 1996. IEEE Computer Society Press.
13. A. Degtyarev and A. Voronkov. Simultaneous rigid E-unification is undecidable. In H. Kleine Büning, editor, *Computer Science Logic. 9th International Workshop, CSL'95*, volume 1092 of *Lecture Notes in Computer Science*, pages 178–190, Paderborn, Germany, September 1995, 1996.
14. A. Degtyarev and A. Voronkov. Simultaneous rigid E-unification is undecidable. In H. Kleine Büning, editor, *Computer Science Logic. 9th International Workshop, CSL'95*, volume 1092 of *Lecture Notes in Computer Science*, pages 178–190, Paderborn, Germany, September 1995, 1996.
15. A. Degtyarev and A. Voronkov. The undecidability of simultaneous rigid E-unification. *Theoretical Computer Science*, 166(1–2):291–300, 1996.
16. 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.
17. J. Doner. Tree acceptors and some of their applications. *Journal of Computer and System Sciences*, 4:406–451, 1970.
18. M. Fitting. First-order modal tableaux. *Journal of Automated Reasoning*, 4:191–213, 1988.
19. J. Gallier, P. Narendran, D. Plaisted, and W. Snyder. Rigid E-unification: NP-completeness and applications to equational matings. *Information and Computation*, 87(1/2):129–195, 1990.
20. 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.
21. 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.
22. F. Gécseg and M. Steinby. *Tree Automata.* Akadémiai Kiodó, Budapest, 1984.
23. W.D. Goldfarb. The undecidability of the second-order unification problem. *Theoretical Computer Science*, 13:225–230, 1981.
24. J. Goubault. Rigid E-unifiability is DEXPTIME-complete. In *Proc. IEEE Conference on Logic in Computer Science (LICS).* IEEE Computer Society Press, 1994.
25. Y. Gurevich and M. Veanes. Some undecidable problems related to the Herbrand theorem. UPMAIL Technical Report 138, Uppsala University, Computing Science Department, March 1997.
26. Y. Gurevich and A. Voronkov. The monadic case of simultaneous rigid E-unification. UPMAIL Technical Report 137, Uppsala University, Computing Science Department, 1997. To appear in Proc. of *ICALP'97.*
27. J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley Publishing Co., 1979.
28. S. Kanger. A simplified proof method for elementary logic. In J. Siekmann and G. Wrightson, editors, *Automation of Reasoning. Classical Papers on Computational Logic*, volume 1, pages 364–371. Springer Verlag, 1983. Originally appeared in 1963.

29. D. Kozen. Complexity of finitely presented algebras. In *Proc. of the 9th Annual Symposium on Theory of Computing*, pages 164–177, New York, 1977. ACM.
30. D. Kozen. Positive first-order logic is NP-complete. *IBM J. of Research and Development*, 25(4):327–332, 1981.
31. V. Lifschitz. Problem of decidability for some constructive theories of equalities (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 4:78–85, 1967. English Translation in: Seminars in Mathematics: Steklov Math. Inst. 4, Consultants Bureau, NY-London, 1969, p.29–31.
32. D.W. Loveland. Mechanical theorem proving by model elimination. *Journal of the Association for Computing Machinery*, 15:236–251, 1968.
33. G.S. Makanin. The problem of solvability of equations in free semigroups. *Mat. Sbornik (in Russian)*, 103(2):147–236, 1977. English Translation in American Mathematical Soc. Translations (2), vol. 117, 1981.
34. G.E. Mints. Choice of terms in quantifier rules of constructive predicate calculus (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 4:78–85, 1967. English Translation in: Seminars in Mathematics: Steklov Math. Inst. 4, Consultants Bureau, NY-London, 1969, p.43–46.
35. V.P. Orevkov. Unsolvability in the constructive predicate calculus of the class of the formulas of the type $\neg\neg\forall\exists$ (in Russian). *Soviet Mathematical Doklady*, 163(3):581–583, 1965.
36. V.P. Orevkov. Solvable classes of pseudo-prenex formulas (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 60:109–170, 1976. English translation in: Journal of Soviet Mathematics.
37. D.A. Plaisted. Special cases and substitutes for rigid E-unification. Technical Report MPI-I-95-2-010, Max-Planck-Institut für Informatik, November 1995.
38. D. Prawitz. An improved proof procedure. In J. Siekmann and G. Wrightson, editors, *Automation of Reasoning. Classical Papers on Computational Logic*, volume 1, pages 162–201. Springer Verlag, 1983. Originally appeared in 1960.
39. 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.
40. Margus Veanes. *On Simultaneous Rigid E-Unification*. PhD thesis, Computing Science Department, Uppsala University, 1997.
41. P.J. Voda and J. Komara. On Herbrand skeletons. Technical report, Institute of Informatics, Comenius University Bratislava, July 1995. Revised January 1996.
42. A. Voronkov. On proof-search in intuitionistic logic with equality, or back to simultaneous rigid E-Unification. UPMAIL Technical Report 121, Uppsala University, Computing Science Department, January 1996.
43. A. Voronkov. Proof-search in intuitionistic logic based on the constraint satisfaction. UPMAIL Technical Report 120, Uppsala University, Computing Science Department, January 1996. Updated March 11, 1996.