The Undecidability of Simultaneous Rigid E-Unification

Anatoli Degtyarev*   Andrei Voronkov†

Abstract

Simultaneous rigid E-unification was introduced in 1987 by Gallier, Raatz and Snyder. It is used in the area of automated reasoning with equality in *extension procedures,* like the tableau method or the connection method. Many articles in this area assumed the existence of an algorithm for the simultaneous rigid E-unification problem. There were several faulty proofs of the decidability of this problem. In this paper we prove that simultaneous rigid E-unification is undecidable. As a consequence, we obtain the undecidability of the $\exists^*$-fragment of intuitionistic logic with equality.

1 Introduction

Simultaneous rigid E-unification plays a crucial role in automatic proof methods for first order logic with equality based on sequent calculi, such as semantic tableaux [13], the connection method [7] (also known as the mating method [1]), model elimination [25] and a dozen other procedures. All these methods are based on the Herbrand theorem and express the idea that the proof-search can be considered as the problem of checking that every path through a matrix of the goal formula is inconsistent (for this reason these methods are sometimes characterized as *matrix methods*). This idea was originally justified by Prawitz [30]. Instead of generating instances of a quantifier-free formula $M(\bar{x})$, he proposed to search for a substitution σ for $\bar{x}_1, \ldots, \bar{x}_n$ such that every path in $(M(\bar{x}_1) \wedge \ldots \wedge M(\bar{x}_n))\sigma$ becomes complementary (or inconsistent). In the case without equality, the search for such a substitution can be carried out using ordinary (simultaneous) unification. Thus, for a matrix $M(\bar{x}_1) \wedge \ldots \wedge M(\bar{x}_n)$, the problem of the existence of an appropriate substitution is decidable and can be performed by any known unification algorithm. The idea of reducing the provability (or, dually, unsatisfiability) problem to checking paths in matrices for inconsistency can also be used for logic with equality [22, 15], but for proving inconsistency of (a set of) paths one needs (simultaneous) rigid E-unification.

Simultaneous rigid E-unification can be formulated as follows. Let us call a *rigid equation* any expression of the form $E \vdash_\forall s = t$, where E is a finite set of equations. Given a finite set of rigid equations $E_i \vdash_\forall s_i = t_i$, $i \in \{1, \ldots, n\}$, find a substitution σ such that for all i we have $\vdash \forall((\bigwedge_{e \in E_i} e\sigma) \supset s_i\sigma = t_i\sigma)$ (here $\vdash$ means provability in first-order logic with equality).

---

*Supported by a grant from the Swedish Institute.

†Supported by a TFR grant. Address for correspondence: Andrei Voronkov, Computing Science Department, Uppsala University, Box 311, S 751 05 Uppsala, Sweden.

Consider an example. Assume that we want to establish the unsatisfiability of the formula $\forall y \forall x (h(x) = a \wedge [(h(a) = y \wedge y \neq g(h(y))) \vee (h(b) = f(y) \wedge y \neq g(f(y)))])$. To this end, we can extend the matrix of this formula by duplicating the quantifier $\forall x$ which creates two copies of the subformula $h(x) = a$. Following [1], we can display the resulting matrix in a two-dimensional format, with disjunctions being displayed horizontally and conjunctions being displayed vertically:

$$\left[\begin{array}{c} h(x_1) = a \\ h(x_2) = a \\ \left[\begin{array}{c} h(a) = y \\ y \neq g(h(y)) \end{array}\right] \bigvee \left[\begin{array}{c} h(b) = f(y) \\ y \neq g(f(y)) \end{array}\right] \end{array}\right]$$

Collecting formulas lying on the two vertical paths in this matrix we obtain the following two rigid equations expressing inconsistency of this set of paths

$$\begin{array}{l} h(x_1) = a, h(x_2) = a, h(a) = y \vdash_\forall y = g(h(y)) \\ h(x_1) = a, h(x_2) = a, h(b) = f(y) \vdash_\forall y = g(f(y)) \end{array}$$

Some solutions to this set of rigid equations are the substitutions $\{a/x_1, b/x_2, g(h^n(a))/y\}$, for every $n \geq 0$.

Since simultaneous rigid E-unification was introduced by Gallier, Raatz and Snyder [17], there have been a number of publications on simultaneous rigid E-unification itself and its use in theorem proving [15, 18, 16, 3, 6, 14, 7, 4, 5, 21, 29]. Some of these articles were based on the conjecture that simultaneous rigid E-unification is decidable. There were several faulty proofs of the decidability of this problem (e.g. [15, 16, 21]).

Results on simultaneous rigid E-unification known so far are the following:

1. Non-simultaneous (i.e. the case $n = 1$) rigid E-unification with E_1 ground (i.e. without variables) is NP-complete [23].

2. Non-simultaneous rigid E-unification is NP-complete [15, 16].

3. For non-simultaneous rigid E-unification there exist finite complete sets of unifiers [15, 16].

4. The monadic case (the arity of function symbols is ≤ 1) is PSPACE-hard [21].

5. The word equation problem is polynomially reducible to monadic simultaneous rigid E-unification [8].

6. Simultaneous rigid E-unification in the signature with one unary function symbol and a countable number of constants is decidable [8]. (The decidability proof for this case uses deep number-theoretic facts.)

In [9] we proved the undecidability of simultaneous rigid E-unification. The proof was by reduction of so called monadic semi-unification whose undecidability was proven in [2]. Here we give a more intuitive proof following [10], using reduction of second-order unification whose undecidability was proven in [20]. Article [20] used reduction of Hilbert's tenth problem. In principle, we could combine the two proofs obtaining a shorter undecidability proof by direct reduction of Hilbert's tenth problem. However, we use reduction of second-order unification since we find it more transparent.

2 Second-order unification

We define second-order unification mainly following [20].

Definition 1 *A* second-order language $\mathcal{L}^2$ *is a triple* $\langle \Sigma, \mathcal{X}, \mathcal{W} \rangle$ *of disjoint sets, where* Σ *is finite and* $\mathcal{X}, \mathcal{W}$ *are countable. For any such second-order language* Σ *is the set of* function symbols *of* $\mathcal{L}^2$, $\mathcal{X}$ *is the set of* free variables *of* $\mathcal{L}^2$ *and* $\mathcal{W}$ *is the set of* bound variables *of* $\mathcal{L}^2$. *A* variable *of a second-order language* $\mathcal{L}^2$ *is either a free variable or a bound variable of* $\mathcal{L}^2$. *For any such second-order language we assume that the sets* Σ *and* $\mathcal{X}$ *are the unions of disjoint sets* Σ_n *and* $\mathcal{X}_n$ *of function symbols and variables of* arity n, $n \in \{0, 1, \ldots\}$. *Elements of* $\mathcal{W}$ *have arity* 0. *Elements of* Σ_0 *are called* constants.

Unlike [20] we do not introduce individual variables; their role is played by free variables of arity 0.

For the rest of this section we assume that $\mathcal{L}^2$ stands for a fixed second-order language $\langle \Sigma, \mathcal{X}, \mathcal{W} \rangle$.

Definition 2 *The set* $T(\mathcal{L}^2)$ *of* second order terms *of the language* $\mathcal{L}^2$ *is defined inductively as follows: for every* $f \in \Sigma \cup \mathcal{X} \cup \mathcal{W}$ *of arity* m *and any* $t_1, \ldots, t_m \in T(\mathcal{L}^2)$, $m \geq 0$, *we have* $f(t_1, \ldots, t_m) \in T(\mathcal{L}^2)$. *A second-order term in* $T(\mathcal{L}^2)$ *is called* ground *iff it uses no variables.*

As usual, for any symbol f of arity 0 we write f instead of $f()$. The equality predicate will be denoted by $=$, whereas $\approx$ means the literal identity of two expressions. We assume $=$ be symmetric, i.e. we do not distinguish $s = t$ from $t = s$. The notation $\rightleftharpoons$ stands for "equal by definition". The set of all variables occurring in a second-order term t is denoted by $var(t)$. Instead of $T(\langle \Sigma, \mathcal{X}, \mathcal{W} \rangle)$ we shall simply write $T(\Sigma, \mathcal{X}, \mathcal{W})$.

Definition 3 *A* second-order substitution *in the language* $\mathcal{L}^2$ *is any expression of the form* $\{t_1/x_1, \ldots, t_n/x_n\}$ *where* $x_i \in \mathcal{X} \cup \mathcal{W}$ *are pairwise different variables and if* x_i *has arity* m, *then* $t_i \in T(\Sigma, \mathcal{X}, \{w_1, \ldots, w_m\})$. *The* domain $dom(\theta)$ *of such a substitution is the set* $\{x_1, \ldots, x_n\}$; *the* range $ran(\theta)$ *of* θ *is the set* $\{t_1, \ldots, t_n\}$.

Let θ *be a second-order substitution* $\{t_1/x_1, \ldots, t_n/x_n\}$, t *be a term. The result* $t\theta$ *of* application *of* θ *to* t *is defined inductively as follows:*

1. *For every* $f \in \Sigma_m$ *we have* $f(s_1, \ldots, s_m)\theta \rightleftharpoons f(s_1\theta, \ldots, s_m\theta)$.
2. *For every* $x \in \mathcal{X} \cup \mathcal{W}$ *of arity* m

$$x(s_1, \ldots, s_m)\theta \rightleftharpoons \begin{cases} t_i\{s_1\theta/w_1, \ldots, s_m\theta/w_m\} & \text{if } x \approx x_i,\ 1 \leq i \leq n \\ x(s_1\theta, \ldots, s_m\theta) & \text{if } x \notin \{x_1, \ldots, x_n\} \end{cases}$$

For purely technical reasons we extend the notation $x\theta$ to variables x of arity > 0 as follows. Let θ be as in Definition 3. Then

$$x\theta \rightleftharpoons \begin{cases} t_i & \text{if } x \approx x_i,\ 1 \leq i \leq n \\ x & \text{if } x \notin \{x_1, \ldots, x_n\} \end{cases}$$

Definition 4 *A* second-order equation *in the language $\mathcal{L}^2$ is an expression of the form $s_1 = s_2$, where $s_1, s_2 \in T(\Sigma, \mathcal{X}, \emptyset)$. A* unifier *for $s_1 = s_2$ is a second-order substitution θ such that $s_1\theta \approx s_2\theta$. A substitution θ is a* unifier *for a set of equations S iff it is a unifier for every $E \in S$. A set of second-order equations possessing a unifier is called* unifiable.

For a set of second-order equations S, $var(S)$ is the set of all variables occurring in S. Note that second order equations do not contain bound variables. The *second order unification problem* is the problem of determining whether any finite set of second order equations is unifiable.

By a routine inspection of the definitions we obtain

Lemma 1 *Let S be a unifiable set of equations in the language $\mathcal{L}^2$. Then there is a unifier θ for S such that*

1. *$var(S) = dom(\theta)$;*
2. *Variables in $\mathcal{X}$ do not occur in $ran(\theta)$.*

Substitutions θ satisfying these two conditions will be called *ground for S.* Note that for every equation $(s = t) \in S$ and for every substitution θ ground for S the terms $s\theta, t\theta$ are ground.

The following definition is technical:

Definition 5 *A set S of second order equations is called* reduced *iff all equations in S have either the form $s = t$ where s, t are terms not using variables of arity > 0 or the form $x(t_1, \ldots, t_m) = t$, where x is a variable, $m > 0$ and $t, t_1, \ldots, t_m$ are terms not using variables of arity > 0.*

Lemma 2 *For any set S of second-order equations one can effectively find a reduced set S' of second order equations such that S is unifiable iff S' is unifiable.*

Proof. Assume that S contains an equation $s = t$ such that s has an occurrence of a term $x(t_1, \ldots, t_m)$. Let $y \in \mathcal{X}_0$ be a variable foreign to S. Then there is a term s' such that $s \approx s'\{x(t_1, \ldots, t_m)/y\}$. Let S_1 be obtained from S by the replacement of $s = t$ by two equations $s' = t, x(t_1, \ldots, t_m) = y$. By routine inspection of the definitions one can see that S is unifiable iff S_1 is unifiable.

With the help of such replacements one can obtain a system S' satisfying the claim. □

3 Simultaneous rigid E-unification

Definition 6 *A* first-order language *$\mathcal{L}^1$ is a pair $\langle \Sigma, \mathcal{X} \rangle$ of disjoint sets, where Σ is finite and $\mathcal{X}$ is countable. For any such language Σ is the set of* function symbols *of $\mathcal{L}^1$ and $\mathcal{X}$ is the set of* variables *of $\mathcal{L}^1$. We assume that the set Σ is the union of disjoint sets Σ_n of function symbols* of arity n, *$n \in \{0, 1, \ldots\}$. Elements of Σ_0 are called* constants.

For the rest of this section we assume that $\mathcal{L}^1$ stands for a fixed first-order language $\langle \Sigma, \mathcal{X} \rangle$.

Definition 7 *The set $T(\mathcal{L}^1)$ of* first order terms *of the language $\mathcal{L}^1$ is defined inductively as follows: every $x \in \mathcal{X}$ belongs to $T(\mathcal{L}^1)$; for every $f \in \Sigma_n$ and $t_1, \ldots, t_n \in T(\mathcal{L}^1)$, $n \geq 0$, we have $f(t_1, \ldots, t_n) \in T(\mathcal{L}^1)$. A first-order term in $T(\mathcal{L}^1)$ is called* ground *iff it uses no variables.*

Instead of $T(\langle \Sigma, \mathcal{X} \rangle)$ we shall simply write $T(\Sigma, \mathcal{X})$.

Definition 8 *A* first-order substitution *in the language $\mathcal{L}^1$ is any expression of the form $\{t_1/x_1, \ldots, t_n/x_n\}$ where $x_i \in \mathcal{X}$ are pairwise different variables and $t_i \in T(\mathcal{L}^1)$. The* domain *$dom(\theta)$ of such a substitution is the set $\{x_1, \ldots, x_n\}$.*

Let θ be a first-order substitution $\{t_1/x_1, \ldots, t_n/x_n\}$ and t be a term. The result $t\theta$ of application *of θ to t is defined inductively as follows:*

1. *For every $f \in \Sigma_m$ we have $f(s_1, \ldots, s_m)\theta \rightleftharpoons f(s_1\theta, \ldots, s_m\theta)$.*
2. *For every $x \in \mathcal{X}$*

$$x\theta \rightleftharpoons \begin{cases} t_i & \text{if } x \approx x_i,\ 1 \leq i \leq n \\ x & \text{if } x \notin \{x_1, \ldots, x_n\} \end{cases}$$

A first-order equation *in the language $\mathcal{L}^1$ is an expression of the form $s_1 = s_2$, where $s_1, s_2 \in T(\mathcal{L}^1)$.*

For a first-order equation $s = t$ and a set of equations E we write $E \vdash s = t$ to denote that the formula $\forall((\bigwedge_{e \in E} e) \supset s = t)$ is provable in first order logic with equality. For such formulas provability can be tested by the congruence closure algorithm (see [14]).

For a set E of first-order equations and a substitution θ we denote by $E\theta$ the set of first-order equations $\{s\theta = t\theta \mid (s = t) \in E\}$.

Definition 9 *A* rigid equation *is an expression of the form $E \vdash_\forall s = t$, where E is a finite set of first order equations and $s = t$ is a first-order equation. Its* solution *is any first-order substitution θ such that $E\theta \vdash s\theta = t\theta$. A substitution θ is a solution to a set S of rigid equations iff θ is a solution to every rigid equation in S*[1]*.*

The *simultaneous rigid E-unification problem* is the problem of determining whether any finite set of rigid equations possesses a solution.

We shall introduce one particular kind of a rigid equation that will be used as a technical tool for proofs in this paper. For any first-order language $\langle \Delta, \mathcal{Y} \rangle$ with at least one constant, any variable $x \in \mathcal{Y}$ and any constant $c \in \Delta_0$ introduce the following rigid equation:

$$Gr(\Delta, x) \rightleftharpoons \bigcup_{m=0}^{\infty} \{f(a_1, \ldots, a_m) = c \mid f \in \Delta_m,\ a_1, \ldots, a_m \in \Delta_0\} \vdash_\forall x = c$$

We shall use the following two obvious lemmas:

[1]The term "rigid equation" could be more adequately expressed as "instance of a (non-simultaneous) rigid E-unification problem", but this would be too lengthy.

Lemma 3 *A substitution θ is a solution to $Gr(\Delta, x)$ iff $x\theta$ is a ground term of the language $\langle \Delta, \mathcal{Y} \rangle$.*

We shall use the substitution notation $\{t_1/c_1, \ldots, t_n/c_n\}$ where c_i are constants. This will denote the operation of the simultaneous replacement of *all* occurrences of c_i by t_i.

Lemma 4 *Let $c_1, \ldots, c_n$ be pairwise different constants and $t_1, \ldots, t_n$ be first order terms such that c_i does not occur in t_j for all $i, j \in \{1, \ldots, n\}$. Then $c_1 = t_1, \ldots, c_n = t_n \vdash s_1 = s_2$ iff $s_1\{t_1/c_1, \ldots, t_n/c_n\} \approx s_2\{t_1/c_1, \ldots, t_n/c_n\}$.*

4 Undecidability proof

In this section we reduce second order unification to simultaneous rigid E-unification. For the rest of this section we assume that $\mathcal{L}^2$ stands for a fixed second-order language $\langle \Sigma, \mathcal{X}, \mathcal{W} \rangle$.

Theorem 1 *There is an effective method that reduces second order unification to simultaneous rigid E-unification.*

Proof. Let S be a set of second-order equations of the language $\mathcal{L}^2$. By Lemma 2 we can assume that S is reduced. By Lemma 1 we can restrict ourselves to substitutions ground for S. Let $V = var(S)$ be the set of all variables occurring in S and let k be the maximal arity of variables in V. Introduce new constants $c_1, \ldots, c_k$ foreign to Σ and define the first-order language $\mathcal{L}^1$ as $\langle \Sigma \cup \{c_1, \ldots, c_k\}, \mathcal{X} \cup \mathcal{W} \rangle$. Let σ be the first-order substitution $\{c_1/w_1, \ldots, c_k/w_k\}$ in this language.

Consider the set R consisting of the following rigid equations in the language $\mathcal{L}^1$:

1. For every $x \in V$ of arity $m \geq 0$ the rigid equation

$$G_x \rightleftharpoons Gr(\Sigma \cup \{c_1, \ldots, c_m\}, x)$$

2. For every equation $(s = t) \in S$ where s, t use only variables of arity 0 the rigid equation

$$E_{s=t} \rightleftharpoons \vdash_{\forall} s = t$$

3. For every equation $(x(t_1, \ldots, t_m) = t) \in S$ with $m > 0$ the rigid equation

$$C_{x(t_1,\ldots,t_m)=t} \rightleftharpoons c_1 = t_1, \ldots, c_m = t_m \vdash_{\forall} x = t$$

For any second-order substitution $\theta = \{t_1/x_1, \ldots, t_l/x_l\}$ ground for S denote by $\hat{\theta}$ the first-order substitution $\{t_1\sigma/x_1, \ldots, t_l\sigma/x_l\}$. Note that for every term t without variables of arity > 0 we have $t\theta \approx t\hat{\theta}$.

We prove the following technical

Lemma 5

1. *Let θ be a substitution ground for S. Then θ is a unifier for S iff $\hat{\theta}$ is a solution to R.*

2. *Every solution τ to R with $dom(\tau) = V$ has the form $\hat{\theta}$ for a suitable θ ground for S.*

Proof.

1. First we prove the "only if" part. Suppose θ is a unifier for S and consider any $x \in V$ of arity m. Since θ is ground for S, we have $x\theta \in T(\Sigma, \emptyset, \{w_1, \ldots, w_m\})$. Hence $x\hat{\theta} \in T(\Sigma \cup \{c_1, \ldots, c_m\}, \emptyset)$. By Lemma 3, $\hat{\theta}$ is a solution to G_x.

   Consider any equation $(s = t) \in S$, where s, t use only variables of arity 0. Since θ is a unifier for $(s = t)$, we have $s\hat{\theta} \approx s\theta \approx t\theta \approx t\hat{\theta}$, and hence $\hat{\theta}$ is a solution to $E_{s=t}$.

   Let $(x(t_1, \ldots, t_m) = t) \in S$ and $m > 0$. Then

   $$\begin{array}{l}
   x\hat{\theta}\{t_1\hat{\theta}/c_1, \ldots, t_m\hat{\theta}/c_m\} \approx \text{ (since } t_i\theta \approx t_i\hat{\theta}) \\
   x\hat{\theta}\{t_1\theta/c_1, \ldots, t_m\theta/c_m\} \approx \text{ (using the definition of } \theta \\
   \qquad\qquad\qquad\qquad\qquad\qquad \text{and } x\theta \in T(\Sigma, \emptyset, \{w_1, \ldots, w_m\}) \text{)} \\
   x\theta\{t_1\theta/w_1, \ldots, t_m\theta/w_m\} \approx \text{ (by the definition of substitution application)} \\
   x(t_1, \ldots, t_m)\theta \approx \text{ (since } \theta \text{ is a unifier for } x(t_1, \ldots, t_m) = t) \\
   t\theta \approx \text{ (since } t\theta \approx t\hat{\theta}) \\
   t\hat{\theta} \approx \text{ (since } c_i \text{ do not occur in } t\hat{\theta}) \\
   t\hat{\theta}\{t_1\hat{\theta}/c_1, \ldots, t_m\hat{\theta}/c_m\}
   \end{array}$$

   We obtained $x\hat{\theta}\{t_1\hat{\theta}/c_1, \ldots, t_m\hat{\theta}/c_m\} \approx t\hat{\theta}\{t_1\hat{\theta}/c_1, \ldots, t_m\hat{\theta}/c_m\}$. By Lemma 4 this yields $c_1 = t_1\hat{\theta}, \ldots, c_m = t_m\hat{\theta} \vdash x\hat{\theta} = t\hat{\theta}$. Hence, $\hat{\theta}$ is a solution to $C_{x(t_1, \ldots, t_m)=t}$.

   Thus, $\hat{\theta}$ is a solution to R. By the same considerations we can prove the "if" part.

2. Suppose τ is any solution to R with $dom(\tau) = V$. Define the second-order substitution θ with $dom(\theta) = V$ by $x\theta \rightleftharpoons (x\tau)\{w_1/c_1, \ldots, w_k/c_k\}$, for all $x \in V$. Since τ is a solution to all G_x, by Lemma 3 for any x of arity m we have $x\tau \in T(\Sigma \cup \{c_1, \ldots, c_m\}, \emptyset)$, which implies $x\theta \in T(\Sigma, \emptyset, \{w_1, \ldots, w_m\})$. Hence, θ is ground for S. It is easy to see that $\hat{\theta} = \tau$.

□

We continue the proof of Theorem 1. Note that if there is a solution to R then there is a solution τ to R with $dom(\tau) = V$. Applying Lemmas 1 and 5 we obtain that S is unifiable iff R has a solution.

□

In our construction we used a potentially infinite number of new constants c_i. One can note that the proof can be modified as well so that the language $\mathcal{L}^1$ of rigid equations will be fixed. The modification is as follows. Let f, g be unary function symbols and b be a constant all foreign to Σ. Replace all new constants c_i used in the proofs by terms of the form $f(g^i(b))$. The proof remains correct because $f(g^k(b))$ and $f(g^l(b))$ are not subterms of each other when $k \neq l$. This shows that second-order unification in the signature Σ is effectively reducible to simultaneous rigid E-unification in the signature $\Sigma \cup \{f, g, b\}$. By the undecidability of second order unification [20] we obtain

Theorem 2 *Simultaneous rigid E-unification is undecidable.*

Our Theorem 2 reveals an interesting phenomenon: the proposal of using simultaneous rigid E-unification in matrix methods in first-order logic with equality in fact leads to second-order logic. For matrix methods in classical logic there are alternative approaches, e.g. the one proposed in [11, 12]. However, the problem of handling simultaneous rigid E-unification seems to be unavoidable in proof methods for intuitionistic logic with equality [31].

5 The undecidability of the $\exists^*$-fragment of intuitionistic logic with equality

In [16] it was noted that "rigid E-unification and Girard's linear logic [19] share the same spirit". In this section we show that simultaneous rigid E-unification is representable already in a very "weak" fragment of intuitionistic logic with equality, thus obtaining the undecidability result for this fragment. The decidability problems for some other fragments of intuitionistic logic with and without equality were studied in [27, 26, 24, 28].

Theorem 3 *The class of formulas provable in intuitionistic logic with equality of the form*

$$\exists x_1 \ldots \exists x_k \Phi$$

where Φ is a quantifier-free formula built from atomic formulas using $\wedge, \supset$, is undecidable.

Proof. We use Theorem 2 on the undecidability of simultaneous rigid E-unification.

Let $E_i \vdash_\forall s_i = t_i$, where $i \in 1, \ldots, n$ be rigid equations all whose variables are among $x_1, \ldots, x_k$. Consider the formula

$$\exists x_1 \ldots \exists x_k \bigwedge_{i=1}^{n} ((\bigwedge_{e \in E_i} e) \supset s_i = t_i) \tag{1}$$

By the explicit definability property of intuitionistic predicate calculus with equality, formula (1) is provable iff there are terms $r_1, \ldots, r_k$ such that the formula

$$\bigwedge_{i=1}^{n} ((\bigwedge_{e \in E_i} e) \supset s_i = t_i)\{r_1/x_1, \ldots, r_k/x_k\} \tag{2}$$

is provable.

Denote the substitution $\{r_1/x_1, \ldots, r_k/x_k\}$ by σ. It is easy to see that formula (2) is provable in intuitionistic logic iff each of the formulas $(\bigwedge_{e \in E_i} e\sigma) \supset s_i\sigma = t_i\sigma$ is provable in intuitionistic logic, which (for this class of formulas) is the same as their provability in classical logic. Provability of all these formulas in classical logic means that σ is a solution to the rigid equations $E_i \vdash_\forall s_i = t_i$.

$\square$

Acknowledgments

We thank Yuri Matiyasevich for many helpful discussions and suggestions.

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] Peter Baumgartner. An ordered theory resolution calculus. In A. Voronkov, editor, *Logic Programming and Automated Reasoning (LPAR'92)*, volume 624 of *Lecture Notes in Computer Science*, pages 119–130, 1992.

[4] G. Becher and U. Petermann. Rigid unification by completion and rigid paramodulation. In B. Nebel and L. Dreschler-Fischer, editors, *KI-94: Advances in Artificial Intelligence. 18th German Annual Conference on Artificial Intelligence*, volume 861 of *Lecture Notes in Artificial Intelligence*, pages 319–330, Saarbrücken, Germany, September 1994. Springer Verlag.

[5] B. Beckert. A completion-based method for mixed universal and rigid E-unification. In A. Bundy, editor, *Automated Deduction — CADE-12. 12th International Conference on Automated Deduction.*, volume 814 of *Lecture Notes in Artificial Intelligence*, pages 678–692, Nancy, France, June/July 1994.

[6] B. Beckert and R. Hähnle. An improved method for adding equality to free variable semantic tableaux. In D. Kapur, editor, *11th International Conference on Automated Deduction (CADE)*, volume 607 of *Lecture Notes in Artificial Intelligence*, pages 678–692, Saratoga Springs, NY, USA, June 1992. Springer Verlag.

[7] W. Bibel. *Deduction. Automated Logic.* Academic Press, 1993.

[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. To appear in LICS'96.

[9] A. Degtyarev and A. Voronkov. Simultaneous rigid E-unification is undecidable. UPMAIL Technical Report 105, Uppsala University, Computing Science Department, May 1995.

[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. General connections via equality elimination. In M. De Glas and Z. Pawlak, editors, *Second World Conference on the Fundamentals of Artificial Intelligence (WOCFAI-95)*, pages 109–120, Paris, July 1995. Angkor.

[12] A. Degtyarev and A. Voronkov. Equality elimination for the inverse method and extension procedures. In C.S. Mellish, editor, *Proc. International Joint Conference on Artificial Intelligence (IJCAI)*, volume 1, pages 342–347, Montréal, August 1995.

[13] M. Fitting. First-order modal tableaux. *Journal of Automated Reasoning*, 4:191–213, 1988.

[14] J. Gallier, P. Narendran, S. Raatz, and W. Snyder. Theorem proving using equational matings and rigid E-unification. *Journal of the Association for Computing Machinery*, 39(2):377–429, 1992.

[15] J.H. Gallier, P. Narendran, D. Plaisted, and W. Snyder. Rigid E-unification is NP-complete. In *Logic in Computer Science (LICS'88) (Edinburgh, Scotland)*, pages 338–346. IEEE Computer Society Press, July 1988.

[16] 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.

[17] J.H. Gallier, S. Raatz, and W. Snyder. Theorem proving using rigid E-unification: Equational matings. In *Logic in Computer Science (LICS'87) (Ithaca, N.Y.)*, pages 338–346. IEEE Computer Society Press, 1987.

[18] J.H. Gallier, S. Raatz, and W. Snyder. Rigid E-unification and its applications to equational matings. In H. Aït Kaci and M. Nivat, editors, *Resolution of Equations in Algebraic Structures*, volume 1, pages 151–216. Academic Press, 1989.

[19] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50(1):1–101, 1987.

[20] Warren D. Goldfarb. The undecidability of the second-order unification problem. *Theoretical Computer Science*, 13:225–230, 1981.

[21] J. Goubault. Rigid $\vec{E}$-unifiability is DEXPTIME-complete. In *Logic in Computer Science (LICS'94)*. IEEE Computer Society Press, 1994.

[22] 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.

[23] D. Kozen. Positive first-order logic is NP-complete. *IBM J. of Research and Development*, 25(4):327–332, 1981.

[24] V. Lifschitz. The decidability problem for some constructive theories of equality (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 4:78–85, 1967.

[25] D.W. Loveland. Mechanical theorem proving by model elimination. *Journal of the Association for Computing Machinery*, 15:236–251, 1968.

[26] G.E. Mints. Collecting terms in the quantifier rules of the constructive predicate calculus (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 4:78–85, 1967.

[27] V.P. Orevkov. The undecidability in the constructive predicate calculus of the class of formulas of the form $\neg\neg\forall\exists$ (in Russian). *Soviet Mathematical Doklady*, 163(3):581–583, 1965. The journal is cover-to-cover translated to English.

[28] 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.

[29] U. Petermann. A complete connection calculus with rigid E-unification. In *JELIA'94*, volume 838 of *Lecture Notes in Computer Science*, pages 152–166, 1994.

[30] 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.

[31] 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. To appear in CADE'96.