BERNHARD BECKERT

RIGID *E*-UNIFICATION

1. INTRODUCTION

1.1. *Overview of this Chapter*

By replacing syntactical unification with rigid E-unification, equality handling can be added to *rigid variable* calculi for first-order logic, including free variable tableau (Fitting, 1996), the mating method (Andrews, 1981), the connection method (Bibel, 1982), and model elimination (Loveland, 1969); for an overview of these calculi, see Chapters I.1.1 and I.1.2.

Rigid E-unification and its significance for automated theorem proving was first described in (Gallier et al., 1987). An earlier attempt to formulate the generalized unification problem that has to be solved for handling equality in rigid variable calculi can be found in (Bibel, 1982).

Ground E-unification (i.e., E-unification with variable-free equalities) has long been known to be decidable (Sect. 2.3), and classical universal E-unification has long been known to be undecidable (Chap. I.2.7). Rigid E-unification is in between: It is decidable in the simple, non-simultaneous case (Sect. 2.4), but it is undecidable whether there is a simultaneous solution for several rigid E-unification problems (Sect. 3.2), which is unfortunate as simultaneous rigid E-unification is of great importance for handling equality in automated theorem proving (Sect. 5).

In the remainder of this section, we describe the basic idea of rigid E-unification and its importance for adding equality to rigid variable calculi and introduce syntax and semantics of first-order logic with equality. In Section 2, we formally define (non-simultaneous) rigid E-unification and the notion of (minimal) complete sets of unifiers; and we briefly sketch proofs for the decidability of ground E-unification and—based on this—for rigid E-unification; methods for solving rigid E-unification problems are compared. In Section 3.3, the problem of finding a simultaneous solution for several rigid E-unification problems is discussed; and in Section 4, *mixed* E-unification is introduced, that is a combination of classical and rigid E-unification. Using the example of free variable semantic tableaux, we show in Section 5 how rigid E-unification can be used to handle equality in a rigid variable calcu-

lus. Finally, in Section 6, we briefly summarize the properties of the different types of E-unification.

1.2. *The Idea of Rigid E-unification*

In classical E-unification, the equalities defining the theory E are implicitly universally quantified w.r.t. the variables they contain. To solve a classical E-unification problem, the question has to be answered whether the equality of two given terms (or of instances of these terms) follows from E or, equivalently, whether the terms are equal in the free algebra of E (Chap. I.2.7). However, for adding equality to rigid variable calculi, being able to answer that question is not sufficient.

Consider, for example, the problem of proving that the conjunction of the three formulae $\neg p(f(a))$, $p(c)$, and $f(x) \doteq c \vee \phi(x)$ is unsatisfiable. The equality $f(x) \doteq c$ can be applied to E-unify the atoms $p(f(a))$ and $p(c)$ and, thus, to show that the literals $\neg p(f(a))$ and $p(c)$ are inconsistent. But to derive this knowledge is not sufficient for a proof; the E-unification procedure has, in addition, to provide the information consisting of which instances of the equality have actually been used (in the example, only the instance where the variable x is instantiated with a). This information is needed to justify the equality applications by proving that the corresponding instances of $\phi(x)$ are inconsistent (in this case the instance $\phi(x)\{x \mapsto a\} = \phi(a)$); one has to show that applying the substitution $\{x \mapsto a\}$ makes both disjuncts inconsistent *simultaneously*. In general, substitutions have to be found that simultaneously solve several rigid E-unification problems corresponding to disjunctively connected (sub-)formulae.

The solution to a (non-simultaneous) rigid E-unification problem is a substitution representing the instantiations of free variables that have been necessary to show that the two given terms are equal. A single variable can only be instantiated once by a substitution and, accordingly, to solve a rigid E-unification problem, the equalities of the problem can only be used with (at most) one instantiation for each variable they contain; a variable is either instantiated or not, that is, uninstantiated variables have to be treated as constants.

Rigid E-unification does not provide an answer to the question of how many different instantiations of an equality are needed to solve a problem. If a single instance is not sufficient, then the answer is "not unifiable". If several different instances of an equality are needed, a sufficient number of copies of that equality (with different rigid variables) has to be provided for the rigid E-unification problem to be solvable.

1.3. *Syntax*

A *first-order signature* $\Sigma = \langle P_\Sigma, F_\Sigma, \alpha_\Sigma \rangle$ consists of a set P_Σ of *predicate symbols*, a set F_Σ of *function symbols*, and a function α_Σ assigning an *arity* $n \geq 0$ to the predicate and function symbols; function symbols of arity 0 are called *constants*. In addition, there is an infinite set V of *object variables*. We only consider signatures Σ where P_Σ contains the binary predicate symbol $\doteq$ denoting equality. The set $Term_\Sigma$ of terms over a signature Σ is built in the usual manner.

We use the logical connectives $\wedge$ (conjunction), $\vee$ (disjunction), and $\neg$ (negation), and the quantifier symbols $\forall$ and $\exists$. The set of well-formed first-order formulae over Σ is denoted by $Form_\Sigma$; well-formed formulae are defined as in Chapter I.1.1 (Def. 1), with the additional restriction that a formula $\phi \in Form_\Sigma$ must not contain a variable that is both bound and free in ϕ. A variable $x \in V$ is *bound* in ϕ if it occurs inside the scope of a quantification $(\forall x)$ or $(\exists x)$; x is *free* in ϕ if it occurs outside the scope of all quantifications $(\forall x)$ and $(\exists x)$. The set of all literals in $Form_\Sigma$ is denoted by Lit_Σ. Since substitutions play an important rôle in this chapter, they are formally defined:

DEFINITION 1. *A* substitution *assigns to each object variable in V a term in $Term_\Sigma$; the set of all substitutions is denoted by $Subst_\Sigma$. The* domain *of a substitution $\sigma \in Subst_\Sigma$ is the set of all $x \in V$ such that $\sigma(x) \neq x$. If σ has a finite domain $\{x_1, \ldots, x_n\}$, $n \geq 0$, it can be denoted by $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ where $t_i = \sigma(x_i)$, $1 \leq i \leq n$. The set of all idempotent substitutions with finite domain is denoted by $Subst^*_\Sigma$.*

The application of a substitution σ to a term t or a formula ϕ is denoted by $t\sigma$ resp. $\phi\sigma$. It may be applied to a quantified formula ϕ; however, to avoid undesired results, the bound variables in ϕ must neither occur in the domain nor in the range of σ.

DEFINITION 2. *Given a finite set $W \subset V$, a substitution $\sigma \in Subst_\Sigma$ is* more general than a substitution $\tau \in Subst_\Sigma$ *(on W), denoted by $\sigma \leq^W \tau$, iff there is a substitution $\rho \in Subst_\Sigma$ such that $\tau(x) = (\sigma(x))\rho$ for all $x \in W$.*

1.4. *Semantics*

A first-order *structure* $M = \langle D, I \rangle$ for a signature Σ consists of a non-empty domain D and an interpretation I which gives meaning to the function and predicate symbols of Σ.

The combination of an interpretation I and a variable assignment μ, that maps the set V of all variables to the domain D, associates (by structural recursion) with each term $t \in Term_\Sigma$ an element in D.

The *evaluation function* $val_{I,\mu}$ maps the formulae in $Form_\Sigma$ to the truth values *true* and *false* (in the usual way, e.g. Def. 10 in Chap. I.1.1).

If $val_{I,\mu}(\phi) = true$ for all variable assignments μ, then M *satisfies* ϕ (is a *model* of ϕ); M satisfies a set Φ of formulae if it satisfies all elements of Φ.

In this chapter, we only consider *normal* structures where the symbol $\doteq$ has the intended meaning; i.e., a structure is normal iff $I(\doteq)$ is the identity relation on D.

DEFINITION 3. *A formula $\psi \in Form_\Sigma$ is a* (weak) consequence *of a set $\Phi \subset Form_\Sigma$ of formulae, denoted by $\Phi \models \psi$, if all normal structures that are models of Φ are models of ψ as well.*

In addition to the normal (weak) consequence relation $\models$, we use the notion of strong consequence:

DEFINITION 4. *A formula $\psi \in Form_\Sigma$ is a* strong consequence *of a set $\Phi \subset Form_\Sigma$ of formulae, denoted by $\Phi \models^\circ \psi$, if for all normal structures $M = \langle D, I\rangle$ and for all variable assignments μ:*

$$\textit{If } val_{I,\mu}(\phi) = true \textit{ for all } \phi \in \Phi, \textit{ then } val_{I,\mu}(\psi) = true \ .$$

A difference between the strong consequence relation $\models^\circ$ and the weak consequence relation $\models$ is that the following holds for $\models^\circ$ (but not for $\models$): If $\Phi \models^\circ \psi$, then $\Phi\sigma \models^\circ \psi\sigma$ for all substitutions $\sigma \in Subst^*_\Sigma$.

2. Non-simultaneous Rigid E-unification

2.1. *Definition and Basic Properties*

The problem of simple (i.e. non-simultaneous) rigid E-unification is defined as follows:

DEFINITION 5. *A* (rigid) equality *is a formula of the form $l \doteq r$. A* rigid E-unification problem *$\langle E, s, t\rangle$ consists of a finite set E of (rigid) equalities in $Form_\Sigma$ and terms $s, t \in Term_\Sigma$. If there are no variables in $\langle E, s, t\rangle$, then it is a* ground *E-unification problem. A substitution $\sigma \in Subst^*_\Sigma$ is a* solution *to (or* unifier *of) $\langle E, s, t\rangle$ iff $E\sigma \models^\circ (s\sigma \doteq t\sigma)$.*

The major differences between this definition and that of classical (universal) E-unification are that (a) the substitution σ is applied not only to the terms s and t but also to the set E of equalities and that (b) the strong consequence relation $\models^{\circ}$ is used in the definition instead of $\models$ (this is equivalent to treating the variables in $E\sigma$ as constants).

The following theorem clarifies the basic properties of rigid E-unification by listing different characterizations of the set of solutions of a given problem:

THEOREM 1. *Given a rigid E-unification problem $\langle E,s,t\rangle$ and a substitution $\sigma = \{x_1 \mapsto t_1,\ldots,x_n \mapsto t_n\} \in Subst^*_\Sigma$, the following are equivalent conditions for σ being a solution to $\langle E,s,t\rangle$:*

1. *$E\sigma \models^{\circ} s\sigma \doteq t\sigma$, i.e., σ is by definition a solution to $\langle E,s,t\rangle$;*
2. *$E\sigma \models s\sigma \doteq t\sigma$ over a set V^0 of variables and a signature Σ^0 such that the variables occurring in $\langle E,s,t\rangle$ are constants, i.e., $V^0 = V \setminus W$ and $\Sigma^0 = \langle P_\Sigma, F_\Sigma \cup W, \alpha_\Sigma \cup \{x \mapsto 0 \mid x \in W\}\rangle$ where W is the set of variables occurring in $\langle E,s,t\rangle$.*
3. *$(E\sigma)\tau \models (s\sigma)\tau \doteq (t\sigma)\tau$ for all substitutions $\tau \in Subst^*_\Sigma$;*
4. *$E \cup \{x_1 \doteq t_1,\ldots,x_n \doteq t_n\} \models^{\circ} s \doteq t$; provided that none of the variables x_i occurs in any of the terms t_j ($1 \le i,j \le n$);*
5. *σ is the restriction to the variables occurring in $\langle E,s,t\rangle$ of a substitution which is a solution to the rigid E-unification problem $\langle E',yes,no\rangle$ where $E' = E \cup \{eq(x,x) \doteq yes, eq(s,t) \doteq no\}$, and (a) the predicate eq, the constants yes, no, and the variable x do not occur in $\langle E,s,t\rangle$ and (b) the constants yes, no do not occur in the terms $t_1,\ldots,t_n$.*

The last characterization of solutions in the above theorem shows that it is always possible to solve a rigid E-unification problem by transforming it into a problem in which the terms to be unified are constants.

Syntactical unification is a special case of rigid E-unification, namely the case where the set E of equalities is empty.

2.2. *Complete Sets of Unifiers*

It is possible to represent the set of all solutions to (unifiers of) a *syntactical* unification problem by a single most general unifier (MGU), that is more general than all other unifiers w.r.t. the subsumption relation $\leq^W$ (Def. 2). For rigid E-unification problems, however, a single MGU is not sufficient to represent all solutions. Instead, a *set* $\mathcal{U}$ of (most general) unifiers has to be used; $\mathcal{U}$ is *complete* if every solution to the given problem is subsumed by one of the unifiers in $\mathcal{U}$. The number of substitutions in a complete set of unifiers can be reduced by using, instead of $\leq^W$, a subsumption relation that takes equality into account:

DEFINITION 6. *Let $E \subset Form_\Sigma$ be a set of equalities; and let $W \subset V$ be a finite set of variables. Then the relations $\sqsubseteq_E^W, \leq_E^W \in Subst_\Sigma^* \times Subst_\Sigma^*$ are defined by: $\sigma \sqsubseteq_E^W \tau$ iff $E\sigma \models^\circ \sigma(x) \doteq \tau(x)$ for all $x \in W$; $\sigma \leq_E^W \tau$ iff there is a substitution $\sigma' \in Subst_\Sigma^*$ such that $\sigma \leq^W \sigma'$ and $\sigma' \sqsubseteq_E^W \tau$.*

The intuitive meaning of $\sigma \leq_E^W \tau$ is that the effects of applying τ to the set E of equalities can be simulated by first applying σ, then some other substitution ρ, and then equalities form $(E\sigma)\rho$.

LEMMA 1. *Let $E \subset Form_\Sigma$ be a set of equalities, and let $\sigma, \tau \in Subst_\Sigma^*$ be substitutions such that $\sigma \leq_E^W \tau$ where the set W contains all variables occurring in E. Then there is a substitution $\rho \in Subst_\Sigma^*$ such that $(E\sigma)\rho \models^\circ E\tau$.*

The set W contains the "relevant" variables, including *at least* those occurring in the E-unification problem. If, for example, rigid E-unification is used to close a tableau branch, then W contains all free variables occurring in the tableau. It is of advantage to keep the set W as small as possible because, for example, $\sigma = \{x \mapsto f(y)\}$ subsumes $\tau = \{x \mapsto f(c)\}$ if $y \notin W$; otherwise, if $y \in W$, σ subsumes the substitution $\tau' = \{x \mapsto f(c), y \mapsto c\}$ but not τ.

In automated deduction, the cardinality of a complete set of unifiers is closely related to the number of choice points when rigid E-unification is used for a deduction step. Therefore, it is desirable to compute a *minimal* complete set of unifiers. Nevertheless, it is often not useful to ensure minimality since there is a trade-off between the gain of computing a minimal set and the extra cost for checking minimality and removing subsumed substitutions (it is not as easy to decide whether $\sigma \leq_E^W \tau$ as it is to decide whether $\sigma \leq^W \tau$).

DEFINITION 7. *A set $\mathcal{U} \subset Subst_\Sigma^*$ is a* complete set of unifiers *for a rigid E-unification problem $\langle E, s, t\rangle$ w.r.t. the relation $\leq^W$ (resp. $\leq_E^W$) if*

1. *each $\sigma \in \mathcal{U}$ is a solution to $\langle E, s, t\rangle$* (soundness),
2. *for each solution τ of $\langle E, s, t\rangle$ there is a solution $\sigma \in \mathcal{U}$ such that $\sigma \leq^W \tau$ (resp. $\sigma \leq_E^W \tau$) where W is the set of variables occurring in $\langle E, s, t\rangle$* (completeness).

The set $\mathcal{U}$ is called a minimal *complete set of unifiers if, in addition,*

3. *there are* no *$\sigma_1, \sigma_2 \in \mathcal{U}$, $\sigma_1 \neq \sigma_2$, such that $\sigma_1 \leq^W \sigma_2$ (resp. $\sigma_1 \leq_E^W \sigma_2$)* (minimality).

EXAMPLE 1. *Consider the rigid E-unification problem $\langle E, s, t\rangle$ consisting of the set $E = \{a \doteq x, b \doteq c\}$ of equalities and the terms $s = a$ and $t = c$. The substitutions $\sigma_1 = \{x \mapsto b\}$ and $\sigma_2 = \{x \mapsto c\}$ are both solutions to $\langle E, s, t\rangle$, whereas the substitution $\tau = \{x \mapsto a\}$ is not a solution.*

Since both $\sigma_1 \sqsubseteq_E^W \sigma_2$ *and* $\sigma_2 \sqsubseteq_E^W \sigma_1$*, the unifiers* σ_1 *and* σ_2 *subsume each other; and* $\{\sigma_1\}$ *and* $\{\sigma_2\}$ *are both minimal complete sets of unifiers.*

Note that, although σ_1 *is a solution and* $\sigma_1 \sqsubseteq_E^W \tau$*, the substitution* τ *is* not *a solution to* $\langle E, s, t\rangle$.

2.3. *Deciding Ground E-unification Problems*

In (Shostak, 1978), it is proven that *ground* E-unification is decidable; consequently, by considering all variables to be constants, it is decidable whether the empty substitution *id* is a solution to a given *rigid* E-unification problem $\langle E, s, t\rangle$, i.e., whether $E \models^{\circ} s \doteq t$. In this section, we briefly describe Shostak's decision procedure, because the decidability of ground E-unification is the basis for proving rigid E-unification to be decidable.

Whether a ground E-unification problem $\langle E, s, t\rangle$ is solvable or not, can be decided by computing a congruence closure, namely the equivalence classes of the terms (and subterms) occurring in $\langle E, s, t\rangle$ w.r.t. the equalities in E.

DEFINITION 8. *Let* $\langle E, s, t\rangle$ *be a ground (or rigid) E-unification problem; and let* $T_{\langle E,s,t\rangle} \subset Term_\Sigma$ *be the set of all (sub-)terms occurring in* $\langle E, s, t\rangle$*. The equivalence class* $[t]_{\langle E,s,t\rangle}$ *of a term* $t \in T_{\langle E,s,t\rangle}$ *is defined by:*

$$[t]_{\langle E,s,t\rangle} = \{s \in T_{\langle E,s,t\rangle} \mid E \models^{\circ} s \doteq t\} \ .$$

Since a ground E-unification problem $\langle E, s, t\rangle$ is solvable (and *id* a solution) if and only if $[s]_{\langle E,s,t\rangle} = [t]_{\langle E,s,t\rangle}$, one can decide whether $\langle E, s, t\rangle$ is solvable by computing these equivalence classes. Shostak proved that for computing the equivalence classes of all terms in $T_{\langle E,s,t\rangle}$, no terms that are not in $T_{\langle E,s,t\rangle}$ have to be considered: If s can be derived from t using the equalities in E, then this can be done without using an intermediate term that does not occur in the original problem, i.e., there is a sequence of terms $s = r_0, r_1, \ldots, r_k = t$, $k \geq 0$, all occurring in $\langle E, s, t\rangle$ such that r_i is derivable in one step from r_{i-1} using the equalities in E.

Since the number of subterms in a given problem is polynomial in its size, and the congruence closure can be computed in time polynomial in the number of subterms and the number of equalities, the solvability of a ground E-unification problem can be decided in polynomial time.

There are very efficient and sophisticated methods for computing the congruence closure, for example the algorithm described in (Nelson and Oppen, 1980) which is based on techniques from graph theory.

Here, we present Shostak's straightforward and easy to understand algorithm: The idea is to start with a separate class for each term in the problem

and then to stepwise join classes containing terms r and r' which can be derived from each other in a single step using equalities in E.

DEFINITION 9. *Let $\langle E,s,t\rangle$ be a rigid E-unification problem; and let $T_{\langle E,s,t\rangle}$ be the set of all (sub-)terms occurring in $\langle E,s,t\rangle$. Then, for all $r \in T_{\langle E,s,t\rangle}$, the classes $[r]^n_{\langle E,s,t\rangle}$ (which are all subsets of $T_{\langle E,s,t\rangle}$) are inductively defined as follows: $[r]^0_{\langle E,s,t\rangle} = \{r\}$ and, for $n \geq 1$, $[r]^n_{\langle E,s,t\rangle}$ is the union of all classes $[r']^{n-1}_{\langle E,s,t\rangle}$ of terms $r' \in T_{\langle E,s,t\rangle}$ such that $[r]^{n-1}_{\langle E,s,t\rangle}$ and $[r']^{n-1}_{\langle E,s,t\rangle}$ are connected, i.e., such that they contain terms r_0 and r'_0, respectively, with (a) $r_0 = r'_0$, (b) $r_0 \doteq r'_0$ is an equality in E, or (c) $r_0 = f(r_1,\ldots,r_{\alpha_\Sigma(f)})$, $r'_0 = f(r'_1,\ldots,r'_{\alpha_\Sigma(f)})$ where $f \in F_\Sigma$ and $[r_i]^{n-1}_{\langle E,s,t\rangle} = [r'_i]^{n-1}_{\langle E,s,t\rangle}$ for all $1 \leq i \leq \alpha_\Sigma(f)$.*

There are only finitely many classes at the beginning, and at each step connected classes are joined and, thus, the number of *different* classes is reduced. Therefore, the process has to terminate after finitely many steps. In (Shostak, 1978), the following theorem is proved which implies soundness and completeness of the congruence closure method.

THEOREM 2. *Given a ground E-unification problem $\langle E,s,t\rangle$, there is an $n \geq 0$ such that, for all terms occurring in $\langle E,s,t\rangle$ and all $m \geq n$:*

$$[t]^m_{\langle E,s,t\rangle} = [t]^n_{\langle E,s,t\rangle} = [t]_{\langle E,s,t\rangle} \quad .$$

2.4. *Deciding and Solving Rigid E-unification Problems*

If a rigid E-unification problem is solvable, then it has infinitely many solutions. But there are, for each problem, *finite* minimal complete sets of solutions w.r.t. the subsumption relation $\leq^W_E$; and such a finite set can be effectively computed.[1] This immediately implies the decidability of the question whether a given unification problem $\langle E,s,t\rangle$ is solvable or not. On first sight this might be somewhat surprising since classical E-unification is undecidable; however, the additional restriction of rigid E-unification, that variables in E may only be instantiated once, is strong enough to turn an undecidable problem into a decidable one.

The problem of deciding whether a rigid E-unification problem has a solution is, in fact, NP-complete. This was first proven in (Gallier et al., 1988)

[1] However, it is not known whether (non-simultaneous) rigid E-unification can be used for handling equality in a rigid variable calculus such that the resulting calculus is complete if only solutions are used that are minimal w.r.t. $\leq^W_E$ (Sect. 5.2).

and then, more detailed, in (Gallier et al., 1990; Gallier et al., 1992). The NP-hardness of the problem was already shown in (Kozen, 1981).

An alternative proof for the decidability of rigid E-unification, that is sketched below, was presented in (de Kogel, 1995). It is easier to understand than the previous proofs, because the complexity of the decision procedure is not taken into consideration. The main lemma on which de Kogel's proof is based is the following:

LEMMA 2. *A rigid E-unification problem $\langle E, s, t\rangle$ is solvable if and only if there are a variable x and a term r, both occurring in $\langle E, s, t\rangle$, and an x-free*[2] *term $r' \in Term_\Sigma$ (which does not have to occur in $\langle E, s, t\rangle$) such that*

1. *$E \models^\circ r \doteq r'$, and*
2. *the rigid E-unification problem $\langle E, s, t\rangle\{x \mapsto r'\}$ is solvable.*

In that case, $\langle E, s, t\rangle\{x \mapsto r'\}$ is solvable for all *x-free terms r' such that $E \models^\circ r \doteq r'$.*

Obviously, if σ' is a solution to the derived problem $\langle E, s, t\rangle\{x \mapsto r'\}$, then $\sigma = \sigma' \circ \{x \mapsto r'\}$ is a solution to the original problem $\langle E, s, t\rangle$. Thus, the problem of solving a rigid E-unification problem with n variables can be reduced to the problem of solving problems with $n-1$ variables (and problems with no variables are decidable using the algorithm for *ground* E-unification described in the previous section). One has to consider all variables x and all terms r occurring in $\langle E, s, t\rangle$; for each of these combinations, an x-free term r' such that $\langle E, s, t\rangle \models^\circ r \doteq r'$ has to be found. Since r' does not have to occur in $\langle E, s, t\rangle$ there are infinitely many candidates. Nevertheless, whether such a term r' exists is decidable because, in fact, only those r' have to be considered that can be deduced in a single step from a term occurring in $\langle E, s, t\rangle$.[3]

Thus, the non-deterministic algorithm shown in Table I can be used to compute solutions to a given rigid E-unification problem $\langle E, s, t\rangle$—provided a solution exists. Since all non-deterministic choices are made from finitely many possibilities, a deterministic decision procedure can be constructed using this algorithm and backtracking.

[2] A term is x-free if it does not contain the variable x.

[3] The reason for this is the following: Supposed there is an x-free term r'' such that $\langle E, s, t\rangle \models^\circ r \doteq r''$. Then, according to Theorem 2, it is possible to derive r'' stepwise from r in such a way that all the intermediate terms are subterms of $\langle E, r, r''\rangle$. Let $r' = r_i$ be the first x-free term in that sequence; the term r_{i-1} that r_i is derived from is not x-free (since r_i is the first) and, therefore, cannot be a subterm of r'' (which is x-free); so, r_{i-1} has to be a subterm of E or r and, thus, of $\langle E, s, t\rangle$.

Table I. Non-deterministic algorithm for computing a solution to a rigid E-unification problem $\langle E, s, t\rangle$.

$\sigma := id$;
while not $E \models^{\circ} s \doteq t$ do
    choose a variable x occurring in $\langle E, s, t\rangle$;
    choose a (sub-)term r occurring in $\langle E, s, t\rangle$ such that
        there is an x-free term $r' \in Term_\Sigma$ such that $E \models^{\circ} r \doteq r'$;
    $\langle E, s, t\rangle := \langle E, s, t\rangle\{x \mapsto r'\}$;
    $\sigma := \{x \mapsto r'\} \circ \sigma$
od;
output "σ is a solution"

The substitution that is applied at each step connects two equivalence classes of terms occurring in the problem; since it is only useful to join different classes, the choice of a variable x and a term r can be restricted to the case that *not* already $E \models^{\circ} x \doteq r$.

EXAMPLE 2. *As an example for the application of the algorithm from Table I, consider the rigid E-unification problem*[4]

$$\langle E, s, t\rangle = \langle \{ fa \doteq a, g^2x \doteq fa\},\ g^3x,\ x\rangle \ .$$

The empty substitution id is not a solution to $\langle E, s, t\rangle$, so $E \models^{\circ} s \doteq t$ does not (yet) hold. The choice of a variable is deterministic, because x is the only one occurring in the problem. None of the terms in $\langle E, s, t\rangle$—except x itself—is equivalent to x, so they are all suitable choices for r; most of these possible choices (including a and fa), however, do not lead to a solution to the problem. The only useful choice is $r = g^3x$. This term is not x-free; it is, however, possible to derive the x-free term $r' = gfa$ from r in one step using the equality $g^2x \doteq fa$. Thus, after applying the substitution $\{x \mapsto gfa\}$, we get the new problem

$$\langle E', s', t'\rangle = \langle \{ fa \doteq a, g^3fa \doteq fa\},\ g^4fa,\ gfa\rangle \ .$$

Now the algorithm terminates because $E' \models^{\circ} s' \doteq t'$, and we conclude that $\{x \mapsto gfa\}$ is a solution to the original problem.

The non-deterministic choices make the algorithm described above too inefficient for an actual implementation. More efficient methods, such as the

[4] In this example, we use g^2x as an abbreviation for $g(g(x))$, etc.

procedures presented in (Gallier et al., 1992; Becher and Petermann, 1994; Plaisted, 1995), use term rewriting and search for *critical pairs* to choose the variable x and the term r. In addition, these completion-based methods allow to compute a minimal complete set of unifiers.

One of these efficient procedures, the one described in (Becher and Petermann, 1994), has been implemented and integrated into a prover for first-order logic with equality (Grieser, 1996).

3. Simultaneous Rigid E-unification

3.1. *Definition and Basic Properties*

For handling equality in rigid variable calculi for first-order logic, it is, actually, not sufficient to solve simple (non-simultaneous) rigid E-unification problems. For example, in semantic tableaux, solving a rigid E-unification problem corresponds to closing a single branch of a tableau. However, to close the whole tableau, a single substitution has to be found that closes all branches *simultaneously*. Thus, *simultaneous* rigid E-unification, the problem of finding a substitution that is a simultaneous solution to several (independent) rigid E-unification problems, plays an even more important rôle in automated deduction than the simple problem. If equality is not considered, i.e., if syntactical unification problems have to be solved, a simultaneous unifier can be computed from the most general unifiers of the simple problems (if one exists). Unfortunately, computing a simultaneous solution for rigid E-unification problems is a much more complex problem. In fact, it is undecidable whether a simultaneous solution exists.

As (non-simultaneous) rigid E-unification, the problem of simultaneous rigid E-unification was first formulated in (Gallier et al., 1987).

DEFINITION 10. *A finite set* $\{\langle E_1, s_1, t_1\rangle, \ldots, \langle E_n, s_n, t_n\rangle\}$ *($n \geq 1$) of rigid E-unification problems is called* simultaneous *rigid E-unification problem. A substitution σ is a solution to the simultaneous problem iff it is a solution to every component* $\langle E_k, s_k, t_k\rangle$ $(1 \leq k \leq n)$.

3.2. *Undecidability of Simultaneous Rigid E-unification*

Whether simultaneously rigid E-unification is decidable or not has long been an open problem. The literature contains several faulty proofs for its decidability. The debate came to an end after a reduction of the problem of *monadic semi-unification* to simultaneous rigid E-unification was presented

in (Degtyarev and Voronkov, 1995), that had been proven to be undecidable in (Baaz, 1993). Shortly afterwards, a simpler and more straightforward reduction of the well known undecidable problem of second-order unification (Chap. I.2.12) was presented in (Degtyarev and Voronkov, 1996a).

In the same way as it may be surprising on first sight that simple rigid E-unification is decidable, it may be surprising that moving from simple to simultaneous problems destroys decidability—even more so considering that the simultaneous versions of other decidable types of unification (including syntactical unification and ground E-unification) are decidable. However, simultaneous rigid E-unification turns out to have a much higher expressiveness than simple rigid E-unification; it is even possible to encode Turing Machines into simultaneous rigid E-unification problems (Veanes, 1997a). The following undecidable problems have been reduced to simultaneous rigid E-unification:

- monadic semi-unification (Degtyarev and Voronkov, 1995);
- second-order unification (Degtyarev and Voronkov, 1996a);
- Hilbert's Tenth Problem (Degtyarev and Voronkov, 1996c), this reduction is based on encoding addition and multiplication of natural numbers into a simultaneous rigid E-unification problem;
- Post's Correspondence Problem (Plaisted, 1995);
- the halting problem for Turing Machines (Veanes, 1997a).

In addition, it has been shown that simultaneous rigid E-unification is already undecidable if (a) all equalities are ground and only the terms to be unified contain variables (Plaisted, 1995) and (b) if the problems contain only two variables; even if both restrictions are combined, the problem remains undecidable (Veanes, 1997b).

The following *decidable* sub-cases of simultaneous rigid E-unification are known (for some of these, decidability is not obvious and difficult to prove): It is decidable whether a simultaneous rigid E-unification problem $\{\langle E_1, s_1, t_1\rangle, \ldots, \langle E_n, s_n, t_n\rangle\}$ ($n \geq 1$) has a solution in case

- $n = 1$; in that case the problem is non-simultaneous;
- the problem is ground; then the simultaneous problem is solvable iff all its components $\langle E_i, s_i, t_i\rangle$ ($1 \leq i \leq n$) are solvable;
- the sets E_i of equalities are identical, i.e., $E = E_1 = \cdots = E_n$; in that case, any substitution that (a) is a solution to the non-simultaneous problem $\langle E, f(s_1, \ldots, s_n), f(t_1, \ldots, t_n)\rangle$ and (b) does not instantiate variables with terms containing f is a solution to the original problem (the function symbol f must not occur in the original problem);
- the problem contains only one variable; then the decision problem is EXPTIME-complete (Degtyarev et al., 1997);

- the problem is monadic (i.e., all occurring function symbols are of arity 0 or 1) and all equalities are ground (Gurevich and Voronkov, 1997);
- the problem is monadic and only one function symbol occurs (Degtyarev et al., 1996b).

Whether the sub-class of *monadic* simultaneous rigid E-unification problems is decidable is an open problem.

The undecidability of simultaneous rigid E-unification implies the undecidability of the following (equivalent) problems, that had not been known to be undecidable before the undecidability of simultaneous rigid E-unification was proven:

- Deciding whether there is an unsatisfiable ground instance of a given quantifier-free formula ϕ of first-order logic with equality; this problem is equivalent to the question whether a fully expanded free variable tableau for ϕ can be closed (Gallier et al., 1987; Degtyarev et al., 1996a), see Section 5.
- Deciding whether a prenex-normal form formula is provable in intuitionistic logic (Degtyarev et al., 1996a; Degtyarev and Voronkov, 1996d).

3.3. *Solving Simultaneous Rigid E-unification Problems*

Since simultaneous rigid E-unification is undecidable, sets of unifiers can only be enumerated; in general they are not finite. Solutions to a simultaneous problem can be computed combining solutions to its constituents $\langle E_i, s_i, t_i \rangle$; however, it is not possible to compute a complete set of unifiers of a simultaneous problem by combining solutions from complete sets of unifiers of its constituents that are minimal w.r.t. the subsumption relation $\leq^W_E$, because they are minimal w.r.t. different relations $\leq^W_{E_i}$. Thus, the subsumption relation $\leq^W$ has to be used, which is the same for all i (but does not allow to construct *finite* complete sets of unifiers).

In (Degtyarev and Voronkov, 1998), a method is described for computing a *finite* (incomplete) set of solutions for rigid E-unification problems, which is shown to be sufficient for building a complete rigid variable calculus for first-order logic with equality (it is described in detail in Sect. 5.3).

4. MIXED RIGID AND CLASSICAL E-UNIFICATION

In automated deduction, it is useful to consider a combination of classical E-unification—where the variables in the equalities are implicitly universally quantified (Chap. I.2.7)—and rigid E-unification. The reason is that even if a

Table II. Examples for the different versions of E-unification (Example 3).

E	s	t	Unifier	Type
$\{f(x) \doteq x\}$	$f(x)$	a	$\{x \mapsto a\}$	rigid
$\{f(a) \doteq a\}$	$f(a)$	a	id	ground
$\{(\forall x)(f(x) \doteq x)\}$	$g(f(a), f(b))$	$g(a,b)$	id	classical
$\{f(x) \doteq x\}$	$g(f(a), f(b))$	$g(a,b)$	—	rigid
$\{(\forall x)(f(x,y) \doteq f(y,x))\}$	$f(a,b)$	$f(b,a)$	$\{y \mapsto b\}$	mixed

rigid variable calculus is used, equalities are often explicitly universally quantified or it is known that an arbitrary number of copies of a certain equality could be added such that it is implicitly universal.

If rigid and classical E-unification are mixed, equalities contain two types of variables, namely universal and rigid ones. To distinguish them syntactically, equalities $(\forall x_1) \cdots (\forall x_n)(l \doteq r)$ are used that can be explicitly quantified w.r.t. variables they contain; free variables are considered to be rigid.

DEFINITION 11. *A* mixed E-unification problem $\langle E, s, t \rangle$ *consists of a finite set E of equalities of the form $(\forall x_1) \cdots (\forall x_n)(l \doteq r)$ and terms s and t. A substitution σ is a* solution *to the problem iff $E\sigma \models^{\circ} s\sigma \doteq t\sigma$.*

If there are no free variables in E, then $\langle E, s, t \rangle$ is a classical E-unification problem, and, if all variables in $\langle E, s, t \rangle$ are free (and thus rigid), then $\langle E, s, t \rangle$ is a rigid E-unification problem.

A simultaneous version of mixed E-unification can be defined analogously to simultaneous rigid E-unification.

EXAMPLE 3. *Table II shows some simple examples for the different versions of E-unification. The unifiers in the table are most general w.r.t. the subsumptions relation $\leq_E^W$. The fourth problem has no solution, since the free variable x would have to instantiated with both a and b. Contrary to that, the empty substitution id is a solution to the third problem, where the variable x is universally quantified.*

Since classical E-unification, which is undecidable, is a special case of mixed E-unification, the latter is undecidable as well.

In (Beckert, 1994), a completion-based method for solving mixed E-unification problems is described (that is, for enumerating unifiers); it is an extension of the Unfailing Knuth-Bendix algorithm. This method has been implemented as part of the tableau-based theorem prover ${}_3T^AP$ and is used in ${}_3T^AP$ for handling equality (Beckert et al., 1996).

5. Using Rigid E-unification for Automated Theorem Proving

5.1. *Extending a Rigid Variable Calculus*

Rigid E-unification can be used to handle equality in any of the rigid variable calculi by replacing syntactical unification with rigid E-unification; examples for this technique have been described in (Beckert, 1997; Degtyarev and Voronkov, 1998) for free variable tableaux, in (Gallier et al., 1992) for the method of matings, and in (Petermann, 1994) and briefly in Section 3.3.2 of Chapter I.2.6 for the connection method.

In this section, we use free variable semantic tableaux as an example (as they are defined in Chap. I.1.1) to demonstrate the basic idea of using rigid E-unification for handling equality. A substitution σ is a closing substitution for a tableau branch B if all instances of $B\sigma$ are unsatisfiable—which is the case if and only if $B\sigma \models^\circ \mathit{false}$. Thus, if E is the set of equalities on B, the inequality $\neg(s \doteq t)$ is on B, and σ is a solution to $\langle E, s, t\rangle$, then σ is a closing substitution for B because $E\sigma \models^\circ s\sigma \doteq t\sigma$ and, therefore, $E\sigma \cup \{\neg(s\sigma \doteq t\sigma)\} \models^\circ \mathit{false}$. It is not sufficient if σ is only known to be a classical E-unifier of $\langle E, s, t\rangle$, because $E \models s\sigma \doteq t\sigma$ does *not* imply $E\sigma \cup \{\neg(s\sigma \doteq t\sigma)\} \models^\circ \mathit{false}$.

Which type of rigid E-unification problems has to be solved to decide whether a tableau branch is closed depends on the version of semantic tableaux that equality is to be added to. If the ground version of tableaux is used (Sect. 2.2 of Chap. I.1.1), equality can be added by solving ground E-unification problems. For handling equality in free variable tableaux, rigid E-unification problems have to be considered; and for tableaux with the universal formula method (described in Sect. 6.4 of Chap. I.1.1) mixed E-unification has to be used.

The relevant E-unification problems for closing a tableau branch B are constructed from the equalities on B and pairs of potentially closing atoms resp. inequalities on B (problems are only extracted from literals, which is sufficient for completeness of the calculus):

DEFINITION 12. *Let T be a free variable tableau for a set $\Phi \subset Form_\Sigma$ of sentences (formulae without free variables), and let B be a branch of T. The*

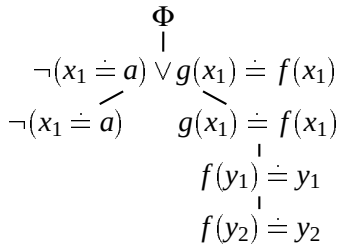

Figure 1. A tableau for the set Φ of formulae from Example 4.

set $E(B)$ of equalities consists of all atomic formulae in $B \cup \Phi$ of the form $l \doteq r$. The set of E-unification problems *of B, denoted by $\mathcal{P}(B)$, contains:*

1. *the E-unification problem $\langle E(B), \langle s_1, \ldots, s_{\alpha_\Sigma(p)} \rangle, \langle t_1, \ldots, t_{\alpha_\Sigma(p)} \rangle \rangle$ for each pair $p(s_1, \ldots, s_{\alpha_\Sigma(p)})$, $\neg p(t_1, \ldots, t_{\alpha_\Sigma(p)})$ of potentially closing literals in $B \cup \Phi$ where $p \neq \doteq$.*
2. *the E-unification problem $\langle E(B), s, t \rangle$ for each (potentially closing) inequality $\neg(s \doteq t)$ in $B \cup \Phi$.*

The problems in $\mathcal{P}(B)$ that are of the form $\langle E(B), \langle s_1, \ldots, s_k \rangle, \langle t_1, \ldots, t_k \rangle \rangle$ are actually simultaneous rigid E-unification problems since the non-simultaneous problems $\langle E(B), s_i, t_i \rangle$ $(1 \leq i \leq k)$ have to be solved simultaneously. Nevertheless, they are decidable and have finite complete sets of unifiers, because the non-simultaneous problems share the same set $E(B)$ of equalities (see Sect. 3.2).

If one of the problems in $\mathcal{P}(B)$ has a solution σ, then all instances of $B\sigma$ are unsatisfiable in normal structures; therefore the branch B is closed under the substitution σ. The pair of literals corresponding to the solved unification problem has been proven to actually be complementary; or the corresponding inequality has been proven to be inconsistent (when the unifier is applied to the tableau).

A sound and complete calculus for first-order logic *with equality* can be constructed by extending the closure rule of free variable tableaux (Def. 6 in Chap. I.1.1) in such a way that each solution to one of the unification problems in $\mathcal{P}(B)$ can be used to close the branch B (and the calculus remains unchanged otherwise); a soundness and completeness proof can be found in (Beckert, 1997)).

EXAMPLE 4. *As an example, we use free variable tableaux with a rigid E-unification closure rule to show that the set $\Phi \subset Form_\Sigma$ consisting of the formula $(\forall x)(\neg(x \doteq a) \vee g(x) \doteq f(x))$, the universally quantified equality $(\forall y)(f(y) \doteq y)$, and the literals $p(g(a), f(b))$ and $\neg p(a, b)$ is unsatisfiable in first-order logic with equality.*

Figure 1 shows a tableau T for Φ, that has been constructed using the free variable tableau expansion rules. Let B_1 be the left and B_2 be the right branch of T.

Since the left branch B_1 does not contain equalities, the set $E(B_1)$ is empty. The set $\mathcal{P}(B_1)$ of E-unification problems has two elements: the problem $\langle \emptyset, x_1, a \rangle$ (which is constructed from the inequality $\neg(x_1 \doteq a)$) and the problem $\langle \emptyset, \langle g(a), f(b) \rangle, \langle a, b \rangle \rangle$ (which is constructed from the potentially closing literals $p(g(a), f(b))$ and $\neg p(a,b)$ in Φ). The first problem has a most general unifier $\sigma_1 = \{x_1 \mapsto a\}$; the second problem has no solution.

The right branch B_2 contains three equalities; two of them are instances of the same γ-formula $(\forall y)(f(y) \doteq y)$:

$$E(B_2) = \{f(y_1) \doteq y_1,\ f(y_2) \doteq y_2,\ g(x_1) \doteq f(x_1)\}$$

The only unification problem of B_2 is $\langle E(B_2), \langle g(a), f(b) \rangle, \langle a, b \rangle \rangle$. The substitution $\sigma_2 = \{x_1 \mapsto a, y_1 \mapsto a, y_2 \mapsto b\}$ is a solution for this problem.

If the left branch B_1 is closed first, by applying the substitution σ_1 to the tableau, the right branch B_2 is then closed under $\{y_1 \mapsto a, y_2 \mapsto b\}$, which is a closing substitution for $B_2\sigma_1$. If B_2 is closed first and σ_2 is applied to the tableau, the left branch B_1 is then closed under the empty substitution.

Note that, if only one instance of $(\forall y)(f(y) \doteq y)$ is generated, B_2 cannot be closed because then $\langle \{f(y_1) \doteq y_1\}, g(x_1) \doteq f(x_1)\}, \langle g(a), f(b) \rangle, \langle a, b \rangle \rangle$ is its single unification problem, which has no solution. If the universal formula method is used, however, then one instance $f(y_1) \doteq y_1$ is sufficient (provided that it is recognized as being universal w.r.t. y_1). Then, the mixed E-unification problem $\langle \{(\forall y_1)(f(y_1) \doteq y_1), g(x_1) \doteq f(x_1)\}, \langle g(a), f(b) \rangle, \langle a, b \rangle \rangle$ has to be solved, for which $\{y_1 \mapsto a\}$ is a solution.

If the universal formula technique is used, mixed problems are constructed instead of purely rigid problems by explicitly universally quantifying equalities w.r.t. to variables w.r.t. which they are universal. In the terms to be unified, variables w.r.t. which the corresponding potentially complementary literals and inequalities are universal can be renamed by new variables not occurring in the tableau.

Since rigid E-unification is decidable, it is decidable whether a given free variable tableau branch (without considering universal formulae) can be closed. However, if a branch cannot be closed it may, nevertheless, be unsatisfiable (and, thus, be expandable to a closed branch).

To close a whole tableau consisting of branches $B_1, \ldots, B_k$, a solution has to be found to one of the simultaneous rigid E-unification problems in $\{\langle P_1, \ldots, P_k \rangle \mid P_i \in \mathcal{P}(B_i), 1 \leq i \leq k\}$. Whether one of these problems has a solution is undecidable (as simultaneous rigid E-unification is undecidable).

It is as well undecidable whether there is a substitution closing all branches of a given tableau simultaneously after it has been expanded by a *fixed* number of copies of the universally quantified formulae it contains (Voda and Komara, 1995; Gurevich and Veanes, 1997). The problem of determining the number of copies of universally quantified formulae necessary to find a proof in rigid variable calculi is discussed in (Voronkov, 1997).

Other approaches to equality handling in rigid variable calculi, which are not based on rigid E-unification, include: the extension of the calculus by special rules resembling *paramodulation* (Fitting, 1996; Beckert, 1998); the method of *equality elimination* (Degtyarev and Voronkov, 1996b), which allows to use classical E-unification instead of rigid E-unification; transformations from first-order logic with equality into first-order logic without equality (Brand, 1975; Bachmair et al., 1997).

Rigid E-unification (and equality reasoning in general) is an instance of *theory reasoning* (Chap. I.2.7), which can be used to improve the efficiency of automated deduction systems. Problems from a certain domain (or theory) that is defined by a set of axioms, are handled separately by a *background reasoner*. The background reasoner applies special purpose techniques and makes use of knowledge about the theory. In the case of equality theory, rigid E-unification is such a technique (it is a *total* theory reasoning method). Using the notions of theory reasoning, the set of formulae from which the rigid E-unification problems are constructed (equalities, inequalities, and potentially complementary literals) form the *key*; and a substitution that is a solution to one of the problems is a *refuter* for that key.

5.2. *Restricting the Search Space*

Completion-based Methods

Efficient methods for solving classical E-unification problems are mostly based on term rewriting and computing a completion of the set of equalities; the same holds for *rigid* E-unification and *mixed* E-unification (Gallier et al., 1992; Becher and Petermann, 1994; Beckert, 1994; Plaisted, 1995; Degtyarev and Voronkov, 1998).

For handling equality in rigid variable calculi, it is often necessary to solve several rigid E-unification problems that share the same set E of equalities. If a completion-based method is used to find solutions, it is sufficient to compute a single completion of E. This positive effect can be strengthened by interlacing the completion process and the proof search (Beckert and Pape, 1996). Then, if there is a case distinction in the proof (for example, when a tableau branches), the (partial) completion that has been computed up to that point can be shared by the sub-cases and has only to be computed once.

Using Finite Sets of Solutions
For completeness of the calculus, it is sufficient to only apply substitutions that are most general solutions w.r.t. the subsumption relation $\leq^W$, where W contains all variables occurring in the partial proof (e.g. the tableau). However, complete sets of solutions w.r.t. $\leq^W$ are infinite; this is unfortunate because it is of great advantage if only a finite number of solutions to each unification problem has to be considered. In that case, it is not necessary to interlace different enumeration processes, which is difficult to implement and to combine with techniques for restricting the search space.

The undecidability of simultaneous rigid E-unificiation implies that any procedure producing—in finite time—a finite number of solutions for a (non-simultaneous) rigid E-unification problem must be incomplete in the following sense: If the procedure is used to compute solutions to rigid E-unification problems that are, for example, extracted from tableau branches, then closing a tableau T may require to extend T by additional instances of equalities and terms although there is a substitution that closes all branches of T simultaneously and there is, thus, a solution to a simultaneous rigid E-unification problem extracted from T. That notwithstanding, the combined calculus *may* be complete for first-order logic with equality; and in that case the advantages of finite sets of solutions prevail. A procedure of this type, which can be used to build a complete calculus, has been presented in (Degtyarev and Voronkov, 1998); it is described in the following section. It is not known whether a *complete* calculus can be built as well using (finite) sets of unifiers that are minimal w.r.t. the subsumption relation $\leq^W_E$.

5.3. *Rigid Basic Superposition*

In (Degtyarev and Voronkov, 1998), a method called *rigid basic superposition* has been presented for computing a *finite* (incomplete) set of solutions for rigid E-unification problems that is "sufficient" for handling equality in rigid variable calculi. A calculus that is complete for first-order logic with equality can, for example, be constructed by extending the closure rule of free variable tableaux (Def. 6 in Chap. I.1.1) such that each solution that can be computed by rigid basic superposition for one of the unification problems in $\mathcal{P}(B)$ (Def. 12) may be used to close the branch B. The procedure is an adaptation of basic superposition (in the formulation presented in (Nieuwenhuis and Rubio, 1995)) to rigid variables. It uses the concept of ordering constraints:

DEFINITION 13. *An* (ordering) constraint *is a (finite) set of expressions of the form $s \simeq t$ or $s \succ t$ where s and t are terms. A substitution σ is a solution*

to a constraint C iff (a) $s\sigma = t\sigma$ for all $s \simeq t \in C$, i.e., σ is a unifier of s and t, (b) $s\sigma > t\sigma$ for all $s \succ t \in C$, where $>$ is an arbitrary but fixed term reduction ordering, and (c) σ instantiates all variables occurring in C with ground terms.

There are efficient methods for deciding the satisfiability of an ordering constraint C and for computing most general substitutions satisfying C in case the reduction ordering $>$ is a lexicographic path ordering (LPO) (Nieuwenhuis and Rubio, 1995).

The rigid basic superposition calculus consists of the two transformation rules shown below. They are applied to a rigid E-unification problem $\langle E, s, t\rangle \cdot C$ that has an ordering constraint C attached to it. The computation starts initially with the unification problem that is to be solved and the empty constraint. A transformation rule may be applied to $\langle E, s, t\rangle \cdot C$ only if the constraint is satisfiable before and after the application.

Left rigid basic superposition. If there are an equality $l \doteq r$ or $r \doteq l$ and an equality $u \doteq v$ or $v \doteq u$ in E and p is a subterm of u, then replace the latter equality by $u[r] \doteq v$ (where $u[r]$ is the result of replacing one occurrence of p in u by r) and add $l \succ r$, $u \succ v$, and $l \simeq p$ to C.

Right rigid basic superposition. If there is an equality $l \doteq r$ or $r \doteq l$ in E and p is a subterm of s or of t, then replace s (resp. t) with $s[r]$ (resp. $t[r]$) and add $l \succ r$, $s \succ t$ (resp. $t \succ s$) and $l \simeq p$ to C.

As the constraint expressions that are added by a rule application have to be satisfiable, they can be seen as a pre-condition for that application; for example, since $l \simeq p$ is added to C, the terms l and p have to be unifiable.

The two transformation rules are repeatedly applied, forming a non-deterministic procedure for transforming rigid E-unification problems. The process terminates when (a) the terms s and t become identical or (b) no further rule application is possible without making C inconsistent. Provided that no transformation is allowed that merely replaces an equality by itself, all transformation sequences are finite.

It is possible to only allow transformations where the term p is *not* a variable, thus improving the efficiency of the procedure and reducing the number of solutions that are computed.

Let $\langle E, s, t\rangle \cdot C$ be any of the unification problems that are reachable by applying rigid basic superposition transformations to the original problem. Then, any solution to $C \cup \{s \simeq t\}$ is a solution to the original problem. Let $\mathcal{U}$ be the set of all such solutions that are most general w.r.t. $\leq^W$. The set $\mathcal{U}$ is finite because the application of rigid basic superposition rules always terminates.

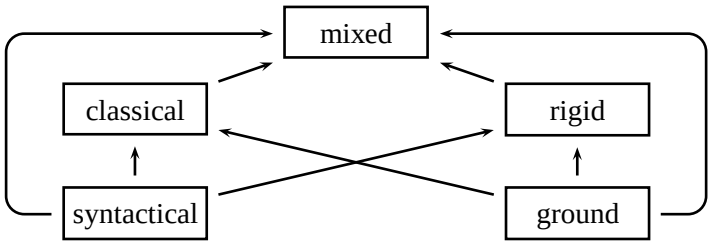

Figure 2. The sub-type relation between the different versions of E-unification.

EXAMPLE 5. *Consider again the rigid E-unification problem from Example 2; and let $>$ be the LPO induced by the ordering $g > f > a$ on the function symbols.*

The computation starts with

$$\langle E, s, t\rangle \cdot C = \langle \{ fa \doteq a, g^2x \doteq fa\},\, g^3x,\, x\rangle \cdot \emptyset \ .$$

The only possible transformation is to use the right rigid basic superposition rule, applying the equality $(l \doteq r) = (g^2x \doteq fa)$ to reduce the term g^3x (all other transformations would lead to an inconsistent constraint). The result is the unification problem $\langle E, gfa, x\rangle \cdot \{g^2x \succ fa, g^3x \succ x, g^2x \simeq g^2x\}$; its constraint can be reduced to $C_1 = \{g^2x \succ fa\}$. A most general substitution satisfying $C_1 \cup \{gfa \simeq x\}$ is $\sigma_1 = \{x \mapsto gfa\}$.

A second application of the right rigid basic superposition rule leads to the unification problem $\langle E, ga, x\rangle \cdot \{g^2x \succ fa,\ fa \succ a, gfa \succ x,\ fa \simeq fa\}$; its constraint can be reduced to $C_2 = \{g^2x \succ fa, gfa \succ x\}$. A most general substitution satisfying $C_2 \cup \{ga \simeq x\}$ is $\sigma_2 = \{x \mapsto ga\}$.

At that point the process terminates because no further rule application is possible. Thus, σ_1 and σ_2 are the only solutions that are computed by rigid basic superposition for this example.

6. Overview of the Different Types of E-unification

In this final section, we briefly summarize the properties of the different types of E-unification. Figure 2 shows the sub-type relation between them.

Syntactical unification, i.e., E-unification with an empty set of equalities, is decidable. Besides the standard method of Robinson, there are more efficient methods for syntactical unification, in particular to solve simultaneous

Table III. Decidability and undecidability of the different types of E-unification.

Type	Simple	Simultaneous
syntactical	decidable	decidable
ground	decidable	decidable
rigid	decidable	undecidable
universal	undecidable	undecidable
mixed	undecidable	undecidable

problems (for example (Martelli and Montanari, 1982)). Classical E-unification is undecidable in general, but there are many interesting special cases of equality sets where it is decidable (Chap. I.2.7). Rigid E-unification is in between: It is decidable (NP-complete) in the non-simultaneous case (Sect. 2.4) and undecidable in the simultaneous case (Sect. 3.2). Since mixed E-unification is more general than classical E-unification, it is undecidable in both the simultaneous and non-simultaneous case. Table III gives an overview of the decidability of different types of E-unification.

Minimal complete sets of unifiers for syntactical unification and ground E-unification problems are either empty (if there is no unifier) or contain a single MGU. For non-simultaneous rigid E-unification they are finite w.r.t. the subsumption relation $\leq_E^W$ and infinite w.r.t. $\leq^W$. For the undecidable types of E-unification they are (in general) infinite.

ACKNOWLEDGEMENTS

I thank Uwe Petermann, Mark Stickel, and Andrei Voronkov for useful comments on earlier versions of this chapter.

REFERENCES

Andrews, P. B.: 1981, 'Theorem Proving through General Matings'. *Journal of the ACM* **28**, 193–214.

Baaz, M.: 1993, 'Note on the Existence of Most General Semi-unifiers'. In: *Arithmetic, Proof Theory and Computational Complexity*, Vol. 23 of *Oxford Logic Guides*. Oxford University Press, pp. 20–29.

Bachmair, L., H. Ganzinger, and A. Voronkov: 1997, 'Elimination of Equality via Transformation with Ordering Constraints'. Technical Report MPI-I-97-2-012, MPI für Informatik, Saarbrücken.

Becher, G. and U. Petermann: 1994, 'Rigid Unification by Completion and Rigid Paramodulation'. In: B. Nebel and L. Dreschler-Fischer (eds.): *Proceedings, 18th German Annual Conference on Artificial Intelligence (KI-94), Saarbrücken, Germany*. pp. 319–330.

Beckert, B.: 1994, 'A Completion-Based Method for Mixed Universal and Rigid *E*-Unification'. In: A. Bundy (ed.): *Proceedings, 12th International Conference on Automated Deduction (CADE), Nancy, France*. pp. 678–692.

Beckert, B.: 1997, 'Semantic Tableaux with Equality'. *Journal of Logic and Computation* **7**(1), 39–58.

Beckert, B.: 1998, 'Equality and Other Theories'. In: M. D'Agostino, D. Gabbay, R. Hähnle, and J. Posegga (eds.): *Handbook of Tableau Methods*. Kluwer. To appear.

Beckert, B., R. Hähnle, P. Oel, and M. Sulzmann: 1996, 'The Tableau-based Theorem Prover $_3TAP$, Version 4.0'. In: *Proceedings, 13th International Conference on Automated Deduction (CADE), New Brunswick, NJ, USA*. pp. 303–307.

Beckert, B. and C. Pape: 1996, 'Incremental Theory Reasoning Methods for Semantic Tableaux'. In: P. Miglioli, U. Moscato, D. Mundici, and M. Ornaghi (eds.): *Proceedings, 5th Workshop on Theorem Proving with Analytic Tableaux and Related Methods, Palermo, Italy*. pp. 93–109.

Bibel, W.: 1982, *Automated Theorem Proving*. Vieweg, Braunschweig, first edition. Second edition published in 1987.

Brand, D.: 1975, 'Proving Theorems with the Modification Method'. *SIAM Journal on Computing* **4**(4), 412–430.

de Kogel, E.: 1995, 'Rigid *E*-Unification Simplified'. In: *Proceedings, 4th Workshop on Theorem Proving with Analytic Tableaux and Related Methods, St. Goar, Germany*. pp. 17–30.

Degtyarev, A., Y. Gurevich, P. Narendran, M. Veanes, and A. Voronkov: 1997, 'The Decidability of Simultaneous Rigid *E*-Unification with one Variable'. UPMAIL Technical Report 139, Uppsala University. To appear in *Theoretical Computer Science*.

Degtyarev, A., Y. Gurevich, and A. Voronkov: 1996a, 'Herbrand's Theorem and Equational Reasoning: Problems and Solutions'. *Bulletin of the EATCS* **60**, 78–95.

Degtyarev, A., Y. Matiyasevich, and A. Voronkov: 1996b, 'Simultaneous Rigid *E*-Unification and Related Algorithmic Problems'. In: *Proceedings, Symposium on Logic in Computer Science (LICS), New Brunswick, USA*. pp. 494–502.

Degtyarev, A. and A. Voronkov: 1995, 'Reduction of Second-Order Unification to Simultaneous Rigid *E*-Unification'. UPMAIL Technical Report 109, Uppsala University.

Degtyarev, A. and A. Voronkov: 1996a, 'Decidability Problems for the Prenex Fragment of Intuitionistic Logic'. In: *Proceedings, Symposium on Logic in Computer Science (LICS), New Brunswick, USA*. pp. 503–512.

Degtyarev, A. and A. Voronkov: 1996b, 'Equality Elimination for the Tableau Method'. In: J. Calmet and C. Limongelli (eds.): *Proceedings, International Symposium on Design and Implementation of Symbolic Computation Systems (DISCO), Karlsruhe, Germany*. pp. 46–60.

Degtyarev, A. and A. Voronkov: 1996c, 'Simultaneous Rigid E-Unification is Undecidable'. In: H. Kleine Büning (ed.): *Proceedings, Annual Conference of the European Association for Computer Science Logic (CSL'95)*. pp. 178–190.

Degtyarev, A. and A. Voronkov: 1996d, 'The Undecidability of Simultaneous Rigid E-Unification'. *Theoretical Computer Science* **166**(1–2), 291–300.

Degtyarev, A. and A. Voronkov: 1998, 'What You Always Wanted to Know About Rigid E-Unification'. *Journal of Automated Reasoning* **20**(1), 47–80.

Fitting, M. C.: 1996, *First-Order Logic and Automated Theorem Proving*. Springer, second edition.

Gallier, J. H., P. Narendran, D. Plaisted, and W. Snyder: 1988, 'Rigid E-Unification is NP-complete'. In: *Procceedings, Symposium on Logic in Computer Science (LICS)*.

Gallier, J. H., P. Narendran, D. Plaisted, and W. Snyder: 1990, 'Rigid E-Unification: NP-Completeness and Application to Equational Matings'. *Information and Computation* pp. 129–195.

Gallier, J. H., P. Narendran, S. Raatz, and W. Snyder: 1992, 'Theorem Proving Using Equational Matings and Rigid E-Unification'. *Journal of the ACM* **39**(2), 377–429.

Gallier, J. H., S. Raatz, and W. Snyder: 1987, 'Theorem Proving Using Rigid E-Unification, Equational Matings'. In: *Proceedings, Symposium on Logic in Computer Science (LICS), Ithaka, NY, USA*.

Grieser, G.: 1996, 'An Implementation of Rigid E-Unification Using Completion and Rigid Paramodulation'. Forschungsbericht FITL-96-4, FIT Leipzig e.V.

Gurevich, Y. and M. Veanes: 1997, 'Some Undecidable Problems Related to the Herbrand Theorem'. UPMAIL Technical Report 138, Uppsala University.

Gurevich, Y. and A. Voronkov: 1997, 'Monadic Simultaneous Rigid E-Unification and Related Problems'. In: P. Degano, R. Corrieri, and A. Marchetti-Spaccamella (eds.): *Proceedings, International Colloquium on Automata, Languages and Programming (ICALP), Bologna, Italy*. pp. 154–165. To appear in *Theoretical Computer Science*.

Kozen, D.: 1981, 'Positive First-Order Logic is NP-Complete'. *IBM Journal of Research and Development* **25**(4), 327–332.

Loveland, D. W.: 1969, 'A Simplified Format for the Model Elimination Procedure'. *Journal of the ACM* **16**(3), 233–248.

Martelli, A. and U. Montanari: 1982, 'An Efficient Unification Algorithm'. *ACM Transitions on Programming Languages and Systems* **4**(2), 258–282.

Nelson, G. and D. C. Oppen: 1980, 'Fast Decision Procedures Based on Congruence Closure'. *Journal of the ACM* **27**(2), 356–364.

Nieuwenhuis, R. and A. Rubio: 1995, 'Theorem Proving with Ordering and Equality Constrained Clauses'. *Journal of Symbolic Computation* **19**, 321–351.

Petermann, U.: 1994, 'A Complete Connection Calculus with Rigid E-Unification'.

In: C. MacNish, D. Pearce, and L. M. Pereira (eds.): *Proceedings, European Workshop on Logics in Artificial Intelligence (JELIA), York, UK*. pp. 152–166.

Plaisted, D. A.: 1995, 'Special Cases and Substitutes for Rigid E-Unification'. Technical Report MPI-I-95-2-010, Max-Planck-Institut für Informatik, Saarbrücken.

Shostak, R. E.: 1978, 'An Algorithm for Reasoning About Equality'. *Communications of the ACM* **21**(7), 583–585.

Veanes, M.: 1997a, 'On Simultaneous Rigid E-Unification'. PhD Thesis, Uppsala University, Sweden.

Veanes, M.: 1997b, 'The Undecidability of Simultaneous Rigid E-Unification with Two Variables'. In: G. Gottlob, A. Leitsch, and D. Mundici (eds.): *Proceedings, Kurt-Gödel-Colloquium (KGC), Vienna, Austria*. pp. 305–318.

Voda, P. and J. Komara: 1995, 'On Herbrand Skeletons'. Technical Report mff-ii-02-1995, Institute of Informatics, Comenius University, Bratislava, Slovakia.

Voronkov, A.: 1997, 'Strategies in Rigid-variable Methods'. In: M. Pollack (ed.): *Proceedings, International Joint Conference on Artificial Intelligence (IJCAI), Nagoya, Japan*, Vol. 1. pp. 114–119.