Unification in the Union of Disjoint Equational Theories: Combining Decision Procedures

Franz Baader
DFKI
Postfach 2080
6750 Kaiserslautern, Germany
baader@dfki.uni-kl.de

Klaus Schulz
CIS
University Munich, Leopoldstr. 139
8000 München 40, Germany
schulz@cis.uni-muenchen.dbp.de

Abstract

Most of the work on the combination of unification algorithms for the union of disjoint equational theories has been restricted to algorithms which compute finite complete sets of unifiers. Thus the developed combination methods usually cannot be used to combine decision procedures, i.e., algorithms which just decide solvability of unification problems without computing unifiers. In this paper we describe a combination algorithm for decision procedures which works for arbitrary equational theories, provided that solvability of so-called unification problems with constant restrictions—a slight generalization of unification problems with constants—is decidable for these theories. As a consequence of this new method, we can for example show that general A-unifiability, i.e., solvability of A-unification problems with free function symbols, is decidable. Here A stands for the equational theory of one associative function symbol.

Our method can also be used to combine algorithms which compute finite complete sets of unifiers. Manfred Schmidt-Schauß' combination result, the until now most general result in this direction, can be obtained as a consequence of this fact. We also get the new result that unification in the union of disjoint equational theories is finitary, if general unification—i.e., unification of terms with additional free function symbols—is finitary in the single theories.

Contents

1 Introduction

E-unification is concerned with solving term equations modulo an equational theory E. The theory is called “unitary” (“finitary”) if the solutions of a system of equations can always be represented by one (finitely many) solution(s). Otherwise the theory is of type “infinitary” or “zero” (see e.g., [Si89, JK90, Ba91] for an introduction to unification theory). Equational theories which are of unification type unitary or finitary play an important rôle in automated theorem provers with “built in” theories (see e.g., [Pl72, St85]), in generalizations of the Knuth-Bendix algorithm (see e.g., [JK86, Bc87]), and in logic programming with equality (see e.g., [JL84]). The reason is that these applications usually require algorithms which compute finite complete sets of unifiers, i.e., finite sets of unifiers from which all unifiers can be generated by instantiation. However, with the recent development of constraint approaches to theorem proving (see e.g., [Bü90]), term rewriting (see e.g., [KK89]), and logic programming (see e.g., [JL87, Co90]), the computation of finite complete sets of unifiers is no longer indespensable for these applications. It is enough to decide satisfiability of the constraints, that means e.g., solvability of the unification problems. In the present paper, the design of decision procedures for unification problems will be a major issue.

The signature matters

When considering unification in equational theories one has to be careful with regard to the signature over which the terms of the unification problems can be built. This leads to the distinction between *elementary unification* (where the terms to be unified are built over the signature of the equational theory, i.e., the function symbols occurring in the axioms of the theory), *unification with constants* (where additional free constant symbols may occur), and *general unification* (where additional free function symbols of arbitrary arity may occur).

The following facts show that there really is a difference between the three types of E-unification:

- There exist theories which are unitary with respect to elementary unification, but finitary with respect to unification with constants. An example for such a theory is the theory of abelian monoids, i.e., the theory of an associative-commutative (AC) function symbol with a unit element (see e.g., [He87]).

- There exists an equational theory for which elementary unification is decidable, but unification with constants is undecidable (see [Bü86]).
- From the development of the first algorithm for AC-unification with constants [St75, LS75] it took almost a decade until the termination of an algorithm for general AC-unification was shown by Fages [Fa84].

The applications of theory unification mentioned above require algorithms for general unification. This fact is illustrated by the following example.

Example 1.1 The theory $A = \{f(f(x,y),z) = f(x, f(y,z))\}$ only contains the binary symbol f. When talking about A-unification, one first thinks of unifying modulo A terms built by using just the symbol f and variables, or equivalently, of unifying words over the alphabet V of all variables.

However, suppose that a resolution theorem prover—which has built in the theory A—gets the formula

$$\exists x : (\forall y : f(x,y) = y \ \wedge \ \forall y \exists z : f(z,y) = x)$$

as axiom. In a first step, this formula has to be Skolemized, i.e., the existential quantifiers have to be replaced by new function symbols. In our example, we need a nullary symbol e and a unary symbol i in the Skolemized form

$$\forall y : f(e,y) = y \ \wedge \ \forall y : f(i(y),y) = e$$

of the axiom. This shows that, even if we start with formulae containing only terms built over f, our theorem prover has to handle terms containing additional free symbols.

The combination problem

We have seen that the question of how algorithms for elementary unification (or for unification with constants) can be used to get algorithms for general unification is nontrivial and important for applications. Even more general, one often would like to derive algorithms for *unification in the union of disjoint equational theories*, i.e., in the union of several equational theories over disjoint signatures, from unification algorithms in the single theories. The importance for applications of this so-called "combination problem" is illustrated by the following example.

Example 1.2 Assume that we want to compute a canonical term rewriting system for the theory of Boolean rings. Thus we have a signature consisting of two binary symbols "+" and "$*$", a unary symbol "$-$", and two nullary symbols "0" and "1". Since the addition and multiplication in Boolean rings is associative and commutative, and since commutativity cannot be oriented into a terminating rewrite rule, we have to use rewriting modulo associativity and commutativity of "+" and "$*$".

But then critical pairs also must be computed modulo associativity and commutativity of these two symbols. To be more precise, we consider the theories $AC_+ := \{(x+y)+z = x+(y+z), x+y = y+x\}$, and $AC_* := \{(x*y)*z = x*(y*z), x*y = y*x\}$. Critical pairs are computed with the help of general unification modulo $AC_+ \cup AC_*$, i.e., modulo the union of the two disjoint equational theories AC_+ and AC_*.

This example can also be used to demonstrate that going from elementary unification to general unification is in fact an instance of the combination problem. If we define the free theory for "$-$", "0" and "1" to be $F_{0,1,-} = \{-x = -x, 0 = 0, 1 = 1\}$, then one can use elementary unification modulo $AC_+ \cup AC_* \cup F_{0,1,-}$ instead of general unification modulo $AC_+ \cup AC_*$ for computing critical pairs.

When considering the combination problem, until now the attention was mostly restricted to finitary unifying theories, and by unification algorithm one meant a procedure which computes a finite complete set of unifiers. The problem was first considered in [St75, St81, Fa84, HS87] for the case where several AC-symbols and free symbols may occur in the terms to be unified. More general combination problems were, for example, treated in [Ki85, Ti86, He86, Ye87, BJ89], but the theories considered in these papers always had to satisfy certain restrictions (such as collapse-freeness or regularity[1]) on the syntactic form of their defining identities.

The problem was finally solved in its until now most general form by Schmidt-Schauß [Sc89]. His combination algorithm imposes no restriction on the syntactic form of the identities. The only requirements for a combination of disjoint theories E, F are:

- All unification problems with constants must be finitary solvable in E and F.

[1] A theory E is called collapse-free if it does not contain an identity of the form $x = t$ where x is a variable and t is a non-variable term, and it is called regular if the left and right hand sides of the identities contain the same variables.

- All constant elimination problems must be finitary solvable in E and F.

A more efficient version of this combination algorithm has been described by Boudet [Bo90].

The method of Schmidt-Schauß can also handle theories which are not finitary. In this case, procedures which enumerate complete sets of unifiers for the single theories can be combined to a procedure enumerating a complete set of unifiers for their union. However, even if unification in the single theories is decidable, this does not show how to get a decision algorithm for unifiability in the combined theory.

The infinitary theory $A = \{f(f(x,y),z) = f(x,f(y,z))\}$ is an example for this case. In 1972, Plotkin [Pl72] has described a procedure which enumerates minimal complete sets of A-unifiers for general A-unification problems, and in 1977 Makanin [Ma77] has shown that A-unification with constants is decidable. But in 1991, decidability of general A-unification was still mentioned as an open problem by Kapur and Narendran [KN91] in their table of known decidability and complexity results for unification. Such a decision procedure could, for example, be useful when building associativity into a theorem prover via constraint resolution; and it could be used to make Plotkin's enumeration procedure terminating for equations having finite complete sets of A-unifiers.

In his paper on unification in the combination of arbitrary disjoint equational theories [Sc89], Schmidt-Schauß also treats the problem of how to combine decision procedures. But in this case he needs decision procedures for general unification in the single theories as prerequisites for his combination algorithm. Thus his result cannot be used to solve the above mentioned open problem of decidability of general A-unification.

The research which will be presented in this paper builds up on the ideas of Schmidt-Schauß and Boudet. It was motivated by the question of how to get a decision procedure for general A-unification. However, the results we have obtained are more general. We shall present a method which allows one to decide unifiability in the union of arbitrary disjoint equational theories, provided that solvability of so-called unification problems with constant restrictions—a slight generalization of unification problems with constants—is decidable for the single theories. In addition, our method can also be used to combine algorithms which compute finite complete sets of unifiers.

These main results and some of the interesting consequences will be described in the next section. Among these consequences are the new results that general A-unification is in fact decidable, and that the union of disjoint equational theories is finitary if the single theories are finitary with respect to general unification.

In Section 3 we shall present the combination algorithm for the decision problem, and describe how it can also be used to generate complete sets of unifiers. Section 4 proves the correctness of the method. In the fifth section we shall describe conditions under which algorithms for solving unification problems with constant restrictions exist. Some of the consequences mentioned in Section 2 depend on these results.

2 Main results and consequences

As mentioned in the introduction, we have to consider a slight generalization of E-unification problems with constants, so-called E-unification problems with constant restriction, which will be introduced below. Having an algorithm which solves these kind of problems is the only prerequisite necessary for our combination method.

Recall that an E-unification problem with constants is a finite set of equations $\Gamma = \{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$, where the terms $s_1, \ldots, t_n$ are built from variables, the function symbols occurring in the axioms of E, and additional free constant symbols. Now, an *E-unification problem with constant restriction* is an ordinary E-unification problem with constants, Γ, where each free constant c occurring in the problem Γ is equipped with a set V_c of variables, namely, the variables in whose image c must not occur. A solution of the problem is an E-unifier σ of Γ such that for all c, x with $x \in V_c$, the constant c does not occur in $x\sigma$. Complete sets of solutions of unification problems with constant restriction are defined as in the case of ordinary unification problems.

It turns out that our combination method does not really need an algorithm which can handle E-unification problems with arbitrary constant restrictions; it is enough to deal with problems with a so-called *linear constant restriction.* Such a restriction is induced by a linear ordering on the variables and free constants as follows: Let X be the set of all variables and C be the set of all free constants occurring in Γ. For a given linear ordering $<$ on $X \cup C$, the sets V_c are defined as $\{x \mid x \text{ is a variable with } x < c\}$.

We are now ready to formulate our first main result, which is concerned with combining decision algorithms. The combination algorithm which is used to establish this result will be described in the next section.

Theorem 2.1 *Let $E_1, \ldots, E_n$ be equational theories over disjoint signatures such that solvability of E_i-unification problems with linear constant restriction is decidable for $i = 1, \ldots, n$. Then unifiability is decidable for the combined theory $E_1 \cup \ldots \cup E_n$.*

By "unifiability" we mean here solvability of elementary unification problems. However, we shall see below that the result can be lifted to general unification, and to solvability of unification problems with linear constant restriction. The theorem also has several other interesting consequences, which are listed below.

1. *Let E be an equational theory such that solvability of E-unification problems with linear constant restriction is decidable. Then solvability of general E-unification problems is decidable.*
   In fact, for a given set Ω of function symbols we can always build the free theory F_Ω as exemplified in Example 1.2. It is easy to see that F_Ω satisfies the assumption of the theorem; and obviously, any general unification problem modulo E can be seen as an elementary unification modulo $E \cup F_\Omega$ (if Ω contains all the additional free function symbols occurring in the problem).

2. This argument also shows why the *result of the theorem can be lifted to general unification*: in order to get decidability of general unification modulo $E_1 \cup \ldots \cup E_n$, apply the theorem to $E_1, \ldots, E_n, F_\Omega$.

3. *General A-unifiability is decidable.*
   For A, decidability of unification problems with constant restriction is an easy consequence of a result by Schulz [Sh91] on a generalization of Makanin's procedure. This result shows that it is still decidable whether a given A-unification problem with constants has a solution for which the words substituted for the variables in the problem are elements of given regular languages over the constants. It is easy to see that problems with constant restriction are a special case of these more generally restricted problems.

4. *General AI-unifiability, where $AI := A \cup \{f(x, x) = x\}$, is decidable.*
   This was also stated as an open problem in [KN91]. For AI, decidability of unification problems with constant restriction easily follows from

the well-known fact (see e.g., [Ho76]) that finitely generated idempotent semigroups are finite.

5. *If solvability of the E_i-unification problems with linear constant restriction can be decided by an NP-algorithm, then unifiability in the combined theory is also NP-decidable.*
   This fact will become obvious once we have described our combination algorithm. As a consequence one gets easy proofs of Kapur and Narendran's results [KN91] that solvability of general AC- and ACI-unification problems can be decided by NP-algorithms. For these theories, NP-decidability of unification problems with constant restriction can be shown very similarly as in the case of ordinary unification problems with constants.

6. *Let $E_1, \ldots, E_n$ be equational theories over disjoint signatures such that solvability of general E_i-unification problems is decidable for $i = 1, \ldots, n$. Then unifiability is decidable for the combined theory $E_1 \cup \ldots \cup E_n$.* This result, which was first proved by Schmidt-Schauß (see [Sc89], Theorem 10.6), can also be obtained as a corollary to our theorem. In fact, we can show that solvability of E-unification problems with linear constant restriction can be reduced to solvability of general E-unification problems (see Section 5).

7. Together with the second consequence mentioned above, this reduction also shows that *the result of Theorem 2.1 can be lifted to unification problems with linear constant restriction.*

The algorithm which will be introduced for proving Theorem 2.1 can also be used to compute complete sets of unifiers.

Theorem 2.2 *Let $E_1, \ldots, E_n$ be equational theories over disjoint signatures such that all E_i-unification problems with linear constant restriction have finite complete set of solutions ($i = 1, \ldots, n$). Then the combined theory $E_1 \cup \ldots \cup E_n$ is finitary.*

Again, we are talking about elementary unification for the combined theory; but as for the case of the decision problem, the result can easily be lifted to general unification, and to unification problems with linear constant restriction. It should be noted that this result is effective in the sense that we really get an algorithm computing finite complete set of unifiers for the combined theory, provided that for the single theories there exist algorithms computing finite complete sets of solutions of unification problems with linear constant restriction. In the following, we mention two other interesting consequences of the theorem.

8. *Let $E_1, \ldots, E_n$ be equational theories over disjoint signatures which are finitary with respect to general unification. Then the combined theory $E_1 \cup \ldots \cup E_n$ is finitary.*
   In fact, we can show how finite complete sets of unifiers for general E_i-unification problems can be used to construct finite complete sets of solutions for unification problems with linear constant restriction (see Section 5).

9. Algorithms which compute finite complete sets of unifiers for unification with constants, and finite complete sets of constant eliminators can be used to get an algorithm which computes finite complete sets of solutions for unification problems with constant restriction (see Section 5). As a consequence, *the combination result of Schmidt-Schauß* ([Sc89], Corollary 7.14) mentioned in the introduction *can also be obtained as a corollary to Theorem 2.2.*

3 The combination algorithm

For the sake of convenience we shall restrict the presentation to the combination of two theories. The combination of more than two theories can be treated analogously. Before we can start with the description of the algorithm we have to introduce some notation.

Let E_1, E_2 be two equational theories built over the disjoint signatures Ω_1, Ω_2, and let $E = E_1 \cup E_2$ denote their union. Since we are only interested in elementary E-unification, we can restrict our attention to terms built from variables and symbols of $\Omega_1 \cup \Omega_2$. The elements of Ω_1 will be called *1-symbols* and the elements of Ω_2 *2-symbols.* A term t is called *i-term* iff it is of the form $t = f(t_1, ..., t_n)$ for an i-symbol f $(i = 1, 2)$. A subterm s of a 1-term t is called *alien subterm* of t iff it is a 2-term such that every proper superterm of s in t is a 1-term. Alien subterms of 2-terms are defined analogously. An i-term s is *pure* iff it contains only i-symbols and variables. An equation $s \doteq t$ is pure iff there exists an $i, 1 \leq i \leq 2$, such that s and t are pure i-terms or variables; this equation is then called an i-equation. Please note that according to this definition equations of the form $x \doteq y$ where x and

y are variables are both 1- and 2-equations. In the following, the symbols x, y, z, with or without indices, will always stand for variables.

Example 3.1 Let Ω_1 consist of the binary (infix) symbol "$\circ$" and Ω_2 of the unary symbol "h", let $E_1 := \{x \circ (y \circ z) = (x \circ y) \circ z\}$ be the theory which says that "$\circ$" is associative, and let $E_2 := \{h(x) = h(x)\}$ be the free theory for "h".

The term $y \circ h(z \circ h(x))$ is a 1-term which has $h(z \circ h(x))$ as its only alien subterm. The equation $h(x_1) \circ x_2 \doteq y$ is not pure, but it can be replaced by two pure equations as follows. We replace the alien subterm $h(x_1)$ of $h(x_1) \circ x_2$ by a new variable z. This yields the pure equation $z \circ x_2 \doteq y$. In addition, we consider the new equation $z \doteq h(x_1)$. This process of replacing alien subterms by new variables is called *variable abstraction*. It will be the first of the five steps of our combination algorithm.

The main procedure

The input for the *combination algorithm* is an elementary E-unification problem, i.e., a system $\Gamma_0 = \{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$, where the terms $s_1, \ldots, t_n$ are built from variables and the function symbols occurring in $\Omega_1 \cup \Omega_2$, the signature of $E = E_1 \cup E_2$. The first two steps of the algorithm are deterministic, i.e., they transform the given system of equations into one new system.

Step 1: variable abstraction. Alien subterms are successively replaced by new variables until all terms occurring in the system are pure. To be more precise, assume that $s \doteq t$ or $t \doteq s$ is an equation in the current system, and that s contains the alien subterm s_1. Let x be a variable not occurring in the current system, and let s' be the term obtained from s by replacing s_1 by x. Then the original equation is replaced by the two equations $s' \doteq t$ and $x \doteq s_1$. This process has to be iterated until all terms occurring in the system are pure. It is easy to see that this can be achieved after finitely many iterations. Now all the terms in the system are pure, but there may still exist non-pure equations, consisting of a 1-term on one side and a 2-term on the other side.

Step 2: split non-pure equations. Each non-pure equations of the form $s \doteq t$ is replaced by two equations $x \doteq s, x \doteq t$ where the x are always new variables.

It is quite obvious that these two steps do not change solvability of the system. The result is a system which consists of pure equations. The third and the fourth step are nondeterministic, i.e., a given system is transformed into finitely many new systems. Here the idea is that the original system is solvable iff at least one of the new systems is solvable.

Step 3: variable identification. Consider all possible partitions of the set of all variables occurring in the system. Each of these partitions yields one of the new systems as follows. The variables in each class of the partition are "identified" with each other by choosing an element of the class as representative, and replacing in the system all occurrences of variables of the class by this representative.

Step 4: choose ordering and theory indices. This step doesn' t change a given system, it just adds some information which will be important in the next step. For a given system, consider all possible strict linear orderings $<$ on the variables of the system, and all mappings *ind* from the set of variables into the set of theory indices $\{1, 2\}$. Each pair $(<, \mathit{ind})$ yields one of the new systems obtained from the given one.

The last step is again deterministic. It splits each of the systems already obtained into a pair of pure systems.

Step 5: split systems. A given system Γ is split into two systems Γ_1 and Γ_2 such that Γ_1 contains only 1-equations and Γ_2 only 2-equations. These systems can now be considered as unification problems with linear constant restriction. In the system Γ_i, the variables with index i are still treated as variables, but the variables with alien index $j \neq i$ are treated as free constants. The linear constant restriction for Γ_i is induced by the linear ordering chosen in the previous step.

The output of the algorithm is thus a finite set of pairs (Γ_1, Γ_2) where the first component Γ_1 is an E_1-unification problem with linear constant restriction, and the second component Γ_2 is an E_2-unification problem with linear constant restriction.

Proposition 3.2 *The input system Γ_0 is solvable if and only if there exists a pair (Γ_1, Γ_2) in the output set such that Γ_1 and Γ_2 are solvable.*

A proof of this proposition is described in the next section. Obviously, if solvability of E_1- and E_2-unification problems with linear constant restrictions is decidable, the proposition implies decidability of elementary E-unifiability, which proves Theorem 2.1.

An example

We consider the theories E_1 and E_2 of Example 3.1, and the unification problem

$$\{h(x) \circ y = y \circ h(z_1 \circ z_2)\}.$$

Step 1: variable abstraction. This step results in the new system

$$\{x_1 \circ y = y \circ x_2, x_1 = h(x), x_2 = h(x_3), x_3 = z_1 \circ z_2\}.$$

Step 2: split non-pure equations. Since all equations are already pure, nothing is done in this step.

Step 3: variable identification. As an example, we consider the partition where x_1 and x_2 are in one class, and all the other variables are in singleton classes. Choosing x_1 as representative for its class, we obtain the new system

$$\{x_1 \circ y = y \circ x_1, x_1 = h(x), x_1 = h(x_3), x_3 = z_1 \circ z_2\}.$$

Step 4: choose ordering and theory indices. As an example, we take the linear ordering

$$z_1 < z_2 < x_3 < x < x_1 < y,$$

and the theory indices

$$\mathit{ind}(x_1) = \mathit{ind}(x) = \mathit{ind}(z_1) = \mathit{ind}(z_2) = 2 \text{ and } \mathit{ind}(x_3) = \mathit{ind}(y) = 1.$$

Step 5: split systems. On the one hand, we get the system

$$\Gamma_1 = \{x_1 \circ y = y \circ x_1, x_3 = z_1 \circ z_2\}$$

consisting of pure 1-equations. In this system the variables with index 1, i.e., x_3 and y, are still treated as variables, but the variables of index 2, i.e., x_1, z_1 and z_2, are treated as free constants. The linear constant restriction induced by the linear ordering is given by $V_{x_1} = \{x_3\}, V_{z_1} =$

$V_{z_2} = \emptyset$.
On the other hand, we obtain the system

$$\Gamma_2 = \{x_1 = h(x), x_1 = h(x_3)\}$$

consisting of pure 2-equations. Here x and x_1 are treated as variables, and x_3 is treated as free constant. The constant restriction is given by $V_{x_3} = \emptyset$.

This pair (Γ_1, Γ_2) is one element in the set which is the output of the algorithm. It is easy to see that Γ_1 has the solution $\{x_3 \mapsto z_1 \circ z_2, y \mapsto x_1\}$, and Γ_2 has the solution $\{x_1 \mapsto h(x_3), x \mapsto x_3\}$. Consequently, the proposition implies that the original system has a solution.

Combination of unifiers

The combination algorithm can also be used to compute complete sets of unifiers for elementary $(E_1 \cup E_2)$-unification problems, provided that one can compute finite complete sets of solutions for all E_i-unification problems with linear constant restriction ($i = 1, 2$). The reason is that solutions of the problems Γ_1, Γ_2 in the output of the algorithm can be combined to solutions of the original input system. This *combined solution* is defined inductively over the linear ordering chosen in Step 4 of the algorithm.

Assume that σ_1 is a solution of Γ_1 and σ_2 is a solution of Γ_2. Without loss of generality we may assume that the substitution σ_i maps all variables of index i to terms containing only variables of index $j \neq i$ (which are treated as free constants in Γ_i) or new variables, i.e., variables not occurring in Γ_0, Γ_1, or Γ_2. This can simply be achieved by renaming variables if necessary. First, we define the combined solution σ on the variables occurring in the system obtained after Step 4 of the algorithm. Note that the input system Γ_0 may contain additional variables which have been replaced during the variable identification step.

Let x be the least variable with respect to the linear ordering chosen in Step 4, and let i be its index. Since the solution σ_i of Γ_i satisfies the constant restriction induced by the linear ordering, the term $x\sigma_i$ does not contain any variables of index $j \neq i$ (Recall that these variables are treated as free constants in Γ_i.) Thus we can simply define $x\sigma := x\sigma_i$.

Now let x be an arbitrary variable with index i, and let $y_1, \ldots, y_m$ be the variables with index $j \neq i$ occurring in $x\sigma_i$. Since σ_i satisfies the constant restriction induced by the linear ordering, the variables $y_1, \ldots, y_m$ (which are treated as free constants in Γ_i) have to be smaller than x. That means that $y_1\sigma, \ldots, y_m\sigma$ are already defined. The term $x\sigma$ is now obtained from $x\sigma_i$ by replacing the y_k by $y_k\sigma$ $(k = 1, \ldots, m)$. Because we have assumed that the other variables occurring in $x\sigma_i$ are new variables, we thus have $x\sigma = x\sigma_i\sigma$.

Finally, let x be a variable of the input system which has been replaced by the variable y during the variable identification step. Thus $y\sigma$ is already defined, and we can simply set $x\sigma := y\sigma$.

For all variables z not occurring in the input system, or in Γ_1 or Γ_2, we define $z\sigma := z$.

Example 3.3 For the above example, the solutions $\sigma_1 = \{x_3 \mapsto z_1 \circ z_2, y \mapsto x_1\}$ and $\sigma_2 = \{x_1 \mapsto h(x_3), x \mapsto x_3\}$ of Γ_1, Γ_2 are combined to $\{z_1 \mapsto z_1, z_2 \mapsto z_2, x_3 \mapsto z_1 \circ z_2, x \mapsto z_1 \circ z_2, x_1 \mapsto h(z_1 \circ z_2), x_2 \mapsto h(z_1 \circ z_2), y \mapsto h(z_1 \circ z_2)\}$.

This construction can now be used to generate complete sets of unifiers for elementary $(E_1 \cup E_2)$-unification problems. For a given input system Γ_0, let $\{(\Gamma_{1,1}, \Gamma_{1,2}), \ldots, (\Gamma_{n,1}, \Gamma_{n,2})\}$ be the output of the combination algorithm. For $i = 1, \ldots, n$ and $j = 1, 2$, let $M_{i,j}$ be a complete set of solutions of the E_i-unification problem with linear constant restriction, $\Gamma_{i,j}$.

Proposition 3.4 *The set of substitutions*

$$\bigcup_{i=1}^{n} \{\sigma \mid \sigma \textit{ is the combined solution obtained from } \sigma_1 \in M_{i,1} \textit{ and } \sigma_2 \in M_{i,2}\}$$

is a complete set of $(E_1 \cup E_2)$-unifiers of the input system Γ_0.

A proof of this proposition will be given in the next section. Obviously, if all the sets $M_{i,j}$ are finite, then the complete set given by the proposition is also finite, which proves Theorem 2.2.

4 Correctness of the combination algorithm

In this section we shall prove Proposition 3.2 and Proposition 3.4, which shows that our combination method is correct when applied to decision problems. Before we can start with our task, we have to introduce a useful tool, which has first been utilized in connection with the combination problem in [BJ89], namely unfailing completion of the combined theory.

Let E_1, E_2 be equational theories over disjoint signatures Ω_1, Ω_2. We assume that both theories are consistent, that means, they have at least one model of cardinality greater than one, or equivalently, the identity $x =_{E_i} y$ does not hold in either theory. One can now apply unfailing completion (see e.g., [DJ87] for definitions and properties) to the combined theory $E = E_1 \cup E_2$. This yields a possibly infinite ordered-rewriting system R which is confluent and terminating on ground terms. In the following, we shall also apply this system to terms containing variables from a fixed countable set of variables X_0; but this is not a problem because these variables can simply be treated like constants. In particular, this means that the simplification ordering used during the completion must also take care of these additional "constants." The ordered-rewriting system R consists of (possibly infinitely many) equations $g = d$. Such an equation can be applied to a term $s \in T(\Omega_1 \cup \Omega_2, X_0)$ iff there exists an occurrence u in t and a substitution τ such that $s = s[u \leftarrow g\tau]$ ($s = s[u \leftarrow d\tau]$, resp.) and $g\tau$ is greater than $d\tau$ ($d\tau$ is greater than $g\tau$, resp.) with respect to the simplification ordering. This application results in the new term $s[u \leftarrow d\tau]$ ($s[u \leftarrow g\tau]$, resp.).

It is easy to see that, because the signatures of E_1 and E_2 are disjoint, the system R is the union of two systems R_1 and R_2, where the terms in R_i are built over the signature Ω_i ($i = 1, 2$). The R_i is just the system which would be obtained by applying unfailing completion to E_i. This is an easy consequence of the definition of critical pairs used for unfailing completion, and of the fact that E_1 and E_2 are assumed to be consistent.

Let $T(\Omega_1 \cup \Omega_2, X_0)$ be the set of terms built from function symbols in $\Omega_1 \cup \Omega_2$ and variables in X_0, and let $T_{\downarrow R}$ denote its R-irreducible elements. We consider an arbitrary bijection $\pi : T_{\downarrow R} \longrightarrow Y$ where Y is a set of variables which is disjoint to X_0. This bijection induces mappings π_1, π_2 of terms in $T(\Omega_1 \cup \Omega_2, X_0)$ to terms in $T(\Omega_1 \cup \Omega_2, Y)$ as follows. For variables $x \in X_0$, $x^{\pi_1} := \pi(x)$ (Note that variables are always R-irreducible.) If $t = f(t_1, \ldots, t_n)$ for a 1-symbol f, then $t^{\pi_1} := f(t_1^{\pi_1}, \ldots, t_n^{\pi_1})$. Finally, if t is a 2-term then $t^{\pi_1} := y$ where $y = \pi(s)$ for the unique R-irreducible element s of the $=_E$-class of t. The mapping π_2 is defined analogously. The mappings π_i may be regarded as projections which map a possibly mixed term to an i-pure term. We write these mappings as superscripts to distinguish them from substitutions. The inverse π^{-1} of π can be seen as a substitution which map the variables y in Y back to the terms $\pi^{-1}(y)$, and is the identity on all other variables. Obviously, we have $t^{\pi_i}\pi^{-1} =_E t$ for all terms $t \in T(\Omega_1 \cup \Omega_2, X_0)$, and if t is an R-irreducible term or an i-term such that all its alien subterms are R-irreducible, then $(t^{\pi_i})\pi^{-1} = t$.

A substitution σ is called R-normalized on a finite set of variables Z iff $z\sigma \in T_{\downarrow R}$ for all variables $z \in Z$. The next lemma will be important in the proof of Proposition 3.2.

Lemma 4.1 *Let s, t be pure i-terms or variables, and let σ be a substitution which is R-normalized on the variables occurring in s, t. Then*

$$s\sigma =_E t\sigma \quad \textit{iff} \quad (s\sigma)^{\pi_i} =_{E_i} (t\sigma)^{\pi_i}.$$

Proof. (1) The if-direction is easy to prove. Obviously, $(s\sigma)^{\pi_i} =_{E_i} (t\sigma)^{\pi_i}$ implies $(s\sigma)^{\pi_i} =_E (t\sigma)^{\pi_i}$, and thus $(s\sigma)^{\pi_i}\pi^{-1} =_E (t\sigma)^{\pi_i}\pi^{-1}$. By our assumptions on s, t and σ, the j-terms (for $j \neq i$) in $s\sigma$ and $t\sigma$ are R-irreducible, which finally yields $s\sigma = (s\sigma)^{\pi_i}\pi^{-1} =_E (t\sigma)^{\pi_i}\pi^{-1} = t\sigma$.

(2) From $s\sigma =_E t\sigma$ follows the existence of an R-irreducible term r which is a common R-descendant of $s\sigma$ and $t\sigma$. Let us now consider the derivation $s_0 := s\sigma \rightarrow_R s_1 \rightarrow_R \ldots r$ more closely. The goal is to show $s_0^{\pi_i} =_{E_i} s_1^{\pi_i} =_{E_i} \ldots r^{\pi_i}$. Symmetrically, we could then also deduce $(t\sigma)^{\pi_i} =_{E_i} r^{\pi_i}$, which finally would prove the lemma.

The case where s is a variable is trivial since then s_0 is R-irreducible, which yields $s_0 = r$. Thus assume that s is a pure i-term. Since all alien subterms of $s\sigma$ are R-irreducible, the first step of the derivation from s_0 to r must take place at an occurrence u which is not inside an alien subterm of $s_0 = s\sigma$. In particular, this means that it is done by applying a rule $g = d$ of R_i. To be more precise, there exists a substitution τ such that $s_0 = s_0[u \leftarrow g\tau]$, $s_1 = s_0[u \leftarrow d\tau]$, and $g\tau$ is greater than $d\tau$ with respect to the simplification ordering. From the fact that u is not inside an alien subterm of s_0 we get that $s_0^{\pi_i} = s_0^{\pi_i}[u \leftarrow (g\tau)^{\pi_i}]$ and $s_1^{\pi_i} = s_0^{\pi_i}[u \leftarrow (d\tau)^{\pi_i}]$.

In order to conclude $s_0^{\pi_i} =_{E_i} s_1^{\pi_i}$, it thus remains to be shown that $(g\tau)^{\pi_i} =_{E_i} (d\tau)^{\pi_i}$. To see this, we define the substitution $\tau^{\pi_i} := \{x \mapsto (x\tau)^{\pi_i} \mid x \text{ occurs in } g \text{ or } d\}$. Since g, d are pure i-terms or variables, we have $g(\tau^{\pi_i}) = (g\tau)^{\pi_i}$ and $d(\tau^{\pi_i}) = (d\tau)^{\pi_i}$. Because $g = d \in R_i$ implies $g =_{E_i} d$, we thus get $(g\tau)^{\pi_i} = g(\tau^{\pi_i}) =_{E_i} d(\tau^{\pi_i}) = (d\tau)^{\pi_i}$.

If we want to continue by induction, we have to know that all alien subterms of s_1 are R-irreducible. This need not be the case for arbitrary derivations from $s\sigma$ to r. The problem is that we only have an ordered-rewriting system which is terminating on ground terms. For this reason it may well be the case that d contains variables not contained in g; and in general we cannot be sure that the image of these variables under τ does not introduce reducible alien subterms into s_1. However, if we assume that the derivation from $s\sigma$ to r is a bottom-up derivation where all the matching substitutions (such as our τ) are R-normalized, then τ cannot introduce reducible alien subterms. This assumption can be made without loss of generality because it is easy to see that, whenever a term is not R-irreducible, then we can apply a rule of R to this term in a way that satisfies the constraints of the assumption. □

Proof of Proposition 3.2

First, we shall show *soundness of the combination algorithm*, that means, we have to demonstrate that Γ_0 is solvable if there exists a pair (Γ_1, Γ_2) in the output set such that Γ_1 and Γ_2 are solvable.

Assume that σ_1 is a solution of Γ_1 and σ_2 is a solution of Γ_2. In the previous section we have already described how these two solutions of the single problems can be combined to a substitution σ, which we have called the combined solution. It remains to be shown that σ is in fact a solution of Γ_0. Obviously, it is sufficient to prove that σ is a solution of the system Γ' which was obtained by Step 4 of the algorithm, and which in Step 5 was split into Γ_1 and Γ_2. Let $s \doteq t$ be an equation in Γ', and assume without loss of generality that this equation was put into Γ_1 in Step 5. Thus we know that $s\sigma_1 =_{E_1} t\sigma_1$. As an easy consequence of the definition of σ, one gets that $\sigma = \sigma_1\sigma$. Since $s\sigma_1 =_{E_1} t\sigma_1$ obviously implies $s\sigma_1\sigma =_{E_1} t\sigma_1\sigma$, and thus also $s\sigma_1\sigma =_E t\sigma_1\sigma$, this shows that $s\sigma =_E t\sigma$.

In the second part of the proof we have to show *completeness of the combination algorithm*, that means, we have to demonstrate that there exists a pair (Γ_1, Γ_2) in the output set such that Γ_1 and Γ_2 are solvable if Γ_0 is solvable.

Let σ be a solution of Γ_0. Without loss of generality we assume that σ is also a solution of the system obtained after the first two steps of the algorithm, that the set Y_0 of all variables occurring in this system is disjoint to X_0, and that σ is R-normalized on Y_0. In particular, this implies that the variables occurring in $y\sigma$ for $y \in Y_0$ are elements of X_0. The solution σ can be used to define the correct alternatives in the nondeterministic steps of the combination algorithm:

- The partition of the set of all variables, which has to be chosen in the third step, is defined as follows. Two variables y and z are in the same class iff $y\sigma = z\sigma$. Obviously, this means that σ is also a solution of the system obtained after the variable identification step corresponding to this partition.

- In the fourth step, the variable y gets index i if $y\sigma$ is an i-term. If $y\sigma$ is itself a variable, y gets index 1 (This is arbitrary, we could have taken index 2 as well.)

- In the fourth step, we also have to choose an appropriate linear ordering on the variables occurring in the system. Consider the strict partial ordering defined by $y < z$ iff $y\sigma$ is a strict subterm of $z\sigma$. We take an arbitrary extension of this partial ordering to a linear ordering on the variables occurring in the system.

The choices we have just described determine a system Γ' in the set of systems obtained after Step 4 of the algorithm, and thus a particular pair of systems (Γ_1, Γ_2) in the output set of the combination algorithm. It remains to be shown that Γ_1, Γ_2 are solvable. In order to define solutions σ_i of these systems, we consider a bijection π from the R-irreducible elements of $T(\Omega_1 \cup \Omega_2, X_0)$ onto a set of variables Y.

This bijection has to satisfy two conditions. First, Y should contain all the variables occurring in Γ'. Since σ is assumed to be R-normalized on Y_0, we have that $y\sigma$ is R-irreducible for all variables y occurring in Γ'. The second condition on π is that $\pi(y\sigma) = y$ for all these variables y. For the satisfiability of these conditions, the variable identification step is important. The reason is that only because of this step we can be sure that Γ' does not contain two different variables y, y' with $y\sigma = y'\sigma$.

As described above, the bijection π induces mappings π_1, π_2. These mappings will now be used to construct the solutions $\sigma_i, i = 1, 2$. The substitution σ_i is defined on the variables y occurring in Γ' by $y\sigma_i := (y\sigma)^{\pi_i}$.

If y is a variable of index $j \neq i$, the term $y\sigma$ is either a variable in X_0 or a j-term. In both cases we get $y\sigma_i = (y\sigma)^{\pi_i} = \pi(y\sigma) = y$ by definition of σ_i and π_i. This shows that σ_i really treats the variables of index j as constants.

Now assume that $s \doteq t$ is an equation in Γ_i. Since this equation is also contained in Γ', and since σ solves Γ', we know that $s\sigma =_E t\sigma$. Since σ was assumed to be R-normalized on Y_0, and since $s \doteq t$ is an i-equation, we can apply the lemma to get $(s\sigma)^{\pi_i} =_{E_i} (t\sigma)^{\pi_i}$. Using the definition of σ_i and the fact that $s \doteq t$ is an i-equation, it is easy to see that $(s\sigma)^{\pi_i} = s\sigma_i$ and $(t\sigma)^{\pi_i} = t\sigma_i$. Thus σ_i really solves the equation $s \doteq t$.

It remains to be shown that σ_i satisfies the constant restriction. Assume that x is a variable of index i, and that the variable y of index $j \neq i$ (which is treated as a constant in Γ_i) occurs in $x\sigma_i$. We have to show that x is not an element of V_y, i.e., that $x \not< y$. Recall that $x\sigma_i = (x\sigma)^{\pi_i}$, and that $x\sigma$ is R-irreducible. Thus, since $y \notin X_0$, the occurrence of y in $x\sigma_i$ must come from the occurrence of $y\sigma$ as a subterm of $x\sigma$. Because of the identification step, the fact that x and y are different variables also implies that $x\sigma$ and $y\sigma$ are different terms. Thus $y\sigma$ is a strict subterm of $x\sigma$, which yields $y < x$ because of the way the linear ordering was chosen. □

Proof of Proposition 3.4

In the first part of the proof of Proposition 3.2 we have already shown that the elements of the set of substitution defined in the formulation of Proposition 3.4 are solutions of Γ_0. It remains to be shown that this set is complete.

Let τ be a solution of Γ_0. Without loss of generality we assume that τ is also a solution of the system obtained after the first two steps of the algorithm, that the set Y_0 of all variables occurring in this system is disjoint to X_0, and that τ is R-normalized on Y_0. In the second part of the proof of Proposition 3.2 we have shown that τ can be used to find a pair of systems (Γ_1, Γ_2) in the output set of the combination algorithm, and to construct solutions τ_1 and τ_2 of these systems. This construction makes use of a bijection π and mappings π_1, π_2 induced by this bijection as described above.

Since τ_i is a solution of Γ_i, there exist an element σ_i in the complete set of solutions of Γ_i and a substitution λ_i such that $\tau_i =_{E_i} \sigma_i\lambda_i \ \langle Y_i \rangle$,[2] where Y_i denotes the set of variables occurring in Γ_i (i.e., the variables of index i). Without loss of generality we may assume that the substitution σ_i maps the variables in Y_i to terms containing only variables of index $j \neq i$ (which are treated as constants by λ_i) or new variables from a set of variables Z_i. We may assume that the domain of λ_i is Z_i, and that the sets X_0, Y_0, Z_1, Z_2 are pairwise disjoint.

As described in the first part of the proof of Proposition 3.2, the solutions σ_1, σ_2 of Γ_1, Γ_2 can be combined to a solution σ of Γ. Since this combined solution is an element of the set of substitution defined in the formulation of Proposition 3.4, it remains to be shown that there exists a substitution λ such that $\tau =_E \sigma\lambda \ \langle X \rangle$, where X denotes the set of variables occurring in Γ_0. We define $\lambda := (\lambda_1 \cup \lambda_2)\pi^{-1}$, where $(\lambda_1 \cup \lambda_2)$ is meant to denote the

[2] Recall that, for a finite set Z of variables, $\delta_1 =_E \delta_2 \ \langle Z \rangle$ means that $z\delta_1 =_E z\delta_2$ for all $z \in Z$.

substitution which is equal to λ_i on Z_i ($i = 1, 2$), and the identity on all variables not contained in $Z_1 \cup Z_2$.

First, we show $\tau =_E \sigma\lambda \; \langle Y_1 \cup Y_2 \rangle$. The proof is by induction on the linear ordering $<$ chosen in Step 4 of the combination algorithm. Without loss of generality, we consider a variable $y \in Y_1$. By the definition of σ, we have $y\sigma\lambda = (y\sigma_1)\sigma\lambda$. The variables occurring in the term $y\sigma_1$ are either variables of index 2, i.e., elements of Y_2, or new variables, i.e., elements of Z_1. We want to show that on these variables, the substitutions $\sigma\lambda$ and $\lambda_1\pi^{-1}$ coincide modulo E.

Let z_1 be an element of Z_1 occurring in $y\sigma_1$. Since we have assumed that the elements of Z_1 are new variables, z_1 is not in the domain of σ, which yields $z_1\sigma\lambda = z_1\lambda$. By definition of λ, and since $z_1 \in Z_1$, we get $z_1\lambda = z_1\lambda_1\pi^{-1}$.

If y is the least variable with respect to the linear ordering $<$, then the term $y\sigma_1$ does not contain a variable of Y_2. This is so because σ_1 satisfies the linear constant restriction induced by $<$. Now assume that y is an arbitrary variable in Y_1, and let y_2 be an element of Y_2 occurring in $y\sigma_1$. Since σ_1 satisfies the linear constant restriction, we know that $y_2 < y$. By induction, we thus get $y_2\sigma\lambda =_E y_2\tau$. We also have $y_2\tau = (y_2\tau)^{\pi_1}\pi^{-1} = y_2\tau_1\pi^{-1}$. Since τ_1 and λ_1 treat variables of index 2 as constants, we get $y_2\tau_1\pi^{-1} = y_2\pi^{-1} = y_2\lambda_1\pi^{-1}$. Thus we have shown $y_2\sigma\lambda =_E y_2\lambda_1\pi^{-1}$.

To sum up, we have just shown that, for all variables z occurring in $y\sigma_1$, we have $z\sigma\lambda =_E z\lambda_1\pi^{-1}$. Consequently, we get $y\sigma\lambda = (y\sigma_1)\sigma\lambda =_E (y\sigma_1)\lambda_1\pi^{-1} =_E y\tau_1\pi^{-1} = (y\tau)^{\pi_1}\pi^{-1} = y\tau$ as required.

Finally, assume that x is a variable occurring in Γ_0, but $x \notin Y_1 \cup Y_2$. This means that x has been substituted by a variable $y \in Y_1 \cup Y_2$ during the variable identification step of the algorithm. On the one hand, this means that $y\tau = x\tau$ (since this must have triggered the identification). On the other hand, because of this identification step, we have defined $x\sigma := y\sigma$. Thus we have $x\tau = y\tau =_E y\sigma\lambda = x\sigma\lambda$. This completes the proof of the proposition. □

5 Solving unification problems with constant restriction

We have seen that algorithms for E-unification with linear constant restriction may be used to obtain—via our combination method—algorithms for general unification. In the first part of this section we shall describe how, conversely, algorithms for general unification can be used to solve unification problems with *linear* constant restrictions. In the second part, constant elimination algorithms together with algorithms for unification with constants are used to solve unification problems with *arbitrary* constant restriction. In the following, F is assumed to be an arbitrary consistent equational theory.

5.1 Using algorithms for general unification

In this subsection we shall consider both the problem of deciding solvability and of generating complete sets of solutions of unification problems with linear constant restrictions.

The decision problem

Let Γ be an F-unification problem with a linear constant restriction, and let $<$ be the linear ordering by which this restriction is induced. In the following, let X denote the set of all variables and C denote the set of all free constants occurring in Γ. Our goal is to construct a general F-unification problem Γ' such that Γ is solvable iff Γ' is solvable.

In this new system Γ', the free constants of Γ will be treated as variables, i.e., the solutions are allowed to substitute terms for these "constants." For any free constant c of Γ we introduce a new (free) function symbol f_c of arity $|V_c|$. Recall that $V_c = \{x \in X \mid x < c\}$ is the set of variables in whose σ-image c must not occur for a solution σ of the problem Γ. The general F-unification problem—in which the free constants of Γ are treated as variables—is now defined as

$$\Gamma' := \Gamma \cup \{c \doteq f_c(x_1, \ldots, x_n) \mid c \in C \text{ and } V_c = \{x_1, \ldots, x_n\}\}.$$

Proposition 5.1 *The F-unification problem with linear constant restriction, Γ, is solvable iff the general F-unification problem Γ' is solvable.*

Please note that the proposition only holds for unification problems with *linear* constant restriction. The following example demonstrates that the construction described above cannot be used for unification problems with arbitrary constant restriction.

Example 5.2 Let F be the empty theory, and let x, y be variables and c, d be free constants. We consider the following F-unification problem with constant restriction:

$$\Gamma = \{x \doteq d, y \doteq c\}, \quad V_c = \{x\}, V_d = \{y\}.$$

It is easy to see that the restriction cannot be induced by a linear ordering on $\{x, y, c, d\}$. Obviously, the problem has the solution $\{x \mapsto d, y \mapsto c\}$.

The corresponding general F-unification problem is

$$\Gamma' = \{x \doteq d, y \doteq c, c \doteq f_c(x), d \doteq f_d(y)\},$$

where c, d are now treated as variables. It is easy to see that this problem does not have a solution.

However, as we shall prove below, our construction is correct for unification problems with linear constant restriction, Γ.

Generating complete sets of solutions

Now we shall describe how this construction can be used to get a finite complete set of solutions of Γ, provided that a finite complete set of F-unifiers of Γ' exists.

Let R be the possibly infinite ordered-rewriting system which is obtained when applying unfailing completion to F. We assume that the simplification ordering used during the completion also takes the additional symbols f_c and variables (which are however treated as constants by the ordering) out of a countable set X_0 of new variables into account. This means that we can apply R to terms built out of symbols in the signature of F, the additional symbols f_c, and variables in X_0. Let $T_{\downarrow R}$ be the R-irreducible elements of the set of these terms.

Now we shall show how an element σ' of a complete set of F-unifiers of Γ' can be used to define a solution σ of Γ. Without loss of generality we may assume that σ' is R-normalized on the variables occurring in Γ'. In fact, for any substitution there exists an $=_F$-equivalent substitution which is R-normalized on the variables occurring in Γ'; and exchanging an element of a complete set of F-unifiers of Γ' by an $=_F$-equivalent substitution still leaves us with a complete set of F-unifiers of Γ'.

Let π be a bijection from $T_{\downarrow R}$ onto a set of variables Y. This bijection has to satisfy two conditions. First, Y should contain all the free constants occurring in Γ (which are treated as variables in Γ'). Since σ' is assumed to be R-normalized on the variables occurring in Γ', we have that $c\sigma'$ is R-irreducible for all these constants c. The second condition on π is that $\pi(c\sigma') = c$ for these constants c. The two conditions are satisfiable because for $c \neq c'$ we have $c\sigma' \neq c'\sigma'$. In fact, since σ' solves Γ', we know that $c\sigma' =_F f_c(x_1\sigma', \ldots, x_n\sigma')$ and $c'\sigma' =_F f_{c'}(x_1\sigma', \ldots, x_n\sigma')$. But this implies that $c\sigma'$ has f_c as root symbol, and $c'\sigma'$ the different symbol $f_{c'}$.

As described in Section 4, the bijection π induces a mapping π_1. To this purpose we treat the symbols of the signature of F as 1-symbols and the symbols f_c as 2-symbols. The mapping π_1 is now used to define our solution σ of Γ. For all variables x occurring in Γ we define $x\sigma := (x\sigma')^{\pi_1}$. The constants c of Γ are really treated as constants by σ, i.e., $c\sigma = c$. However, note that $c = (c\sigma')^{\pi_1}$ holds, anyway.

Proposition 5.3 *Let $C(\Gamma')$ be a complete set of F-unifiers of Γ', which are (without loss of generality) assumed to be R-normalized on the variables occurring in Γ'. Then the set $C(\Gamma) := \{\sigma \mid \sigma' \in C(\Gamma')\}$, where σ is constructed out of σ' as described above, is a complete set of solutions of the F-unification problem with linear constant restriction, Γ.*

Again, the proposition only holds for unification problems with *linear* constant restriction.

Proof of Proposition 5.1

Recall that X denote the set of all variables and C denote the set of all free constants occurring in Γ.

To prove the "only-if" direction, assume that σ is a solution of Γ. Without loss of generality we may assume that for all $x \in X$ the variables occurring in $x\sigma$ are new variables (i.e., variables not contained in X), and that σ is the identity on all variables $y \notin X$. We define a substitution σ' on $X \cup C$ (where the elements of C are now treated as variables) by induction on the linear ordering $<$ which induces the constant restriction of Γ.

First, we consider the least element of $X \cup C$ with respect to $<$. If this is a variable $x \in X$, then for all $c \in C$ we have $x \in V_c$. This implies that $x\sigma$ does not contain any of these free constants, and we can define $x\sigma' := x\sigma$. If the least element of $X \cup C$ is a constant $c \in C$, then $V_c = \emptyset$. This means that f_c is a constant symbol, and we define $c\sigma' := f_c$.

Now let x be an arbitrary element of X, and let $c_1, \dots, c_m \in C$ be the free constants occurring in $x\sigma$. Since σ satisfies the constant restriction induced by the linear ordering, the constants $c_1, \dots, c_m$ (which are treated as variables in Γ') have to be smaller than x. That means that we may assume by induction that $c_1\sigma', \dots, c_m\sigma'$ are already defined. The term $x\sigma'$ is now obtained from $x\sigma$ by replacing the c_k by $c_k\sigma'$ ($k = 1, \dots, m$).

Finally, let c be an arbitrary element of C. By definition, the system Γ' contains the equation $c \doteq f_c(x_1, \dots, x_n)$, where $x_1, \dots, x_n$ are the elements of X which are smaller than c with respect to $<$. Thus we can assume by induction that $x_1\sigma', \dots, x_n\sigma'$ are already defined, and we set $c\sigma' := f_c(x_1\sigma', \dots, x_n\sigma')$.

It remains to be shown that σ' is a solution of Γ'. Obviously, the definition of σ' implies that it solves the equations $c \doteq f_c(x_1, \dots, x_n)$ in Γ'. Now let $s \doteq t$ be an equation of Γ. Since σ solves Γ, we get $s\sigma =_F t\sigma$. In addition, it is easy to see that for all $c \in C$ we have $c\sigma' = c\sigma\sigma'$ and for all $x \in X$ we have $x\sigma' = x\sigma\sigma'$. But then $s\sigma' = s\sigma\sigma' =_F t\sigma\sigma' = t\sigma'$.

To prove the "if" direction, assume that σ' is a solution of Γ'. Without loss of generality, we assume that σ' is R-normalized on the set $X \cup C$ of all variables occurring in Γ'. As described above, σ' can be used to define a new substitution σ as follows: For all variables x occurring in Γ one defines $x\sigma := (x\sigma')^{\pi_1}$. We have to show that σ solves Γ.

Let $s \doteq t$ be an equation of Γ. Since σ' solves Γ', we know that $s\sigma' =_F t\sigma'$. Now we can apply Lemma 4.1 to get $(s\sigma')^{\pi_1} =_F (t\sigma')^{\pi_1}$. Using the definition of σ and the fact that the terms s, t do not contain the symbols f_c, it is easy to see that $(s\sigma')^{\pi_1} = s\sigma$ and $(t\sigma')^{\pi_1} = t\sigma$. Thus σ really solves the equation $s \doteq t$.

It remains to be shown that σ satisfies the constant restriction. Let $c \in C$ be a free constant. Since σ' solves Γ', we know that $c\sigma' =_F f_c(x_1\sigma', \dots, x_n\sigma')$, where $\{x_1, \dots, x_n\} = V_c$. In addition, σ' was assumed to be R-normalized on $X \cup C$, which implies that $c\sigma', x_1\sigma', \dots, x_n\sigma'$ are R-irreducible terms; and since the symbol f_c does not occur in any rule of R, the term $f_c(x_1\sigma', \dots, x_n\sigma')$ is also R-irreducible. Thus we have $c\sigma' = f_c(x_1\sigma', \dots, x_n\sigma')$, which shows that $c\sigma'$ is not a subterm of any of the terms $x_1\sigma', \dots, x_n\sigma'$. But then c cannot occur in $x_i\sigma = (x_i\sigma')^{\pi_1}$ ($i = 1, \dots, n$). □

Proof of Proposition 5.3

Now we shall show that the set $C(\Gamma)$, as defined in the formulation of Proposition 5.3, is a complete set of solutions of Γ. In the second part of the proof of Proposition 5.1 we have already shown that the substitution σ constructed from a solution σ' of Γ' is itself a solution of Γ. Thus $C(\Gamma)$ is a set of solutions of Γ. It remains to be shown that it is a complete set. As before, let X denote the set of variables and C the set of free constants occurring in Γ. Recall that the elements of C are treated as variables in Γ'.

Let τ be a solution of Γ. Without loss of generality we assume that, for all $x \in X$, the variables occurring in $x\tau$ are elements of X_0. As shown in the first part of the proof of Proposition 5.1, τ can be used to define a solution τ' of Γ'. Because of our assumption on τ, it is easy to see that for all $z \in X \cup C$, the variables occurring in $z\tau'$ are elements of X_0.

Since $C(\Gamma')$ is a complete set of F-unifiers of Γ', there exists an element σ' of $C(\Gamma')$ and a substitution λ' such that $\tau' =_F \sigma'\lambda' \langle X \cup C\rangle$. Let σ be the element of $C(\Gamma)$ constructed out of σ'. Our aim is to define a substitution λ which satisfies $\tau =_F \sigma\lambda \langle X\rangle$. Let $Y_0 \subseteq Y$ denote the set of all variables occurring in the terms $x\sigma$ for $x \in X$. These terms may also contain elements of C, which however have to be treated as constants by λ.

In the construction of σ out of σ' we have used a bijection π from $T_{\downarrow R}$ onto a set of variables Y which contains C. In addition, this bijection had to satisfy $\pi(c\sigma') = c$ for all $c \in C$. The substitution σ was then defined by $x\sigma := (x\sigma')^{\pi_1}$ for all $x \in X$. In order to be able to reverse the construction of τ' out of τ we shall now consider an analogous bijection μ from $T_{\downarrow R}$ onto Y. The condition on μ is that $\mu((c\tau')_{\downarrow R}) = c$ for all $c \in C$. Here $(c\tau')_{\downarrow R}$ denotes the unique R-irreducible element of the $=_F$-class of $c\tau'$. As an easy consequence of this condition together with the definition of τ', we get that $x\tau = (x\tau')^{\mu_1}$ for all $x \in X$.

The substitution λ is now defined on all $y \in Y_0$ as $y\lambda := (y\pi^{-1}\lambda')^{\mu_1}$. To complete the proof of the proposition we have to show that $\tau =_F \sigma\lambda \langle X\rangle$.

In this proof we need a lemma which is a stronger version of Lemma 4.1 for the special case of the union of an arbitrary equational theory F with a disjoint free theory.

Lemma 5.4 *Let s, t be terms built out of symbols in the signature of F, the additional symbols f_c, and variables in X_0. Then $s =_F t$ iff $s^{\mu_1} =_F t^{\mu_1}$.*

Proof. (1) From $s^{\mu_1} =_F t^{\mu_1}$ we can deduce $s^{\mu_1}\mu^{-1} =_F t^{\mu_1}\mu^{-1}$, and we also have $s =_F s^{\mu_1}\mu^{-1}$ and $t =_F t^{\mu_1}\mu^{-1}$.

(2) Obviously, it is sufficient to prove the "only-if" direction for the case where t is obtained from s by one application of an identity of F. Thus assume that $g = d$ is an identity of F, u is an occurrence in s, and τ is a substitution such that $s = s[u \leftarrow g\tau]$, $t = s[u \leftarrow d\tau]$. If the occurrence u is strictly below an occurrence of a free function symbol, then it is easy to see that $s^{\mu_1} = t^{\mu_1}$. Otherwise, we have $s^{\mu_1} = s^{\mu_1}[u \leftarrow (g\tau)^{\mu_1}]$ and $s^{\mu_1} = s^{\mu_1}[u \leftarrow (d\tau)^{\mu_1}]$. What remains to be shown is $(g\tau)^{\mu_1} =_F (d\tau)^{\mu_1}$, and this can be done as in the proof of Lemma 4.1. □

We can now continue with the proof of the proposition. For all $x \in X$ we have $x\tau' =_F x\sigma'\lambda' = (x\sigma')^{\pi_1}\pi^{-1}\lambda' = x\sigma\pi^{-1}\lambda'$. But then the lemma yields $x\tau = (x\tau')^{\mu_1} =_F (x\sigma\pi^{-1}\lambda')^{\mu_1}$. It remains to be shown that $(x\sigma\pi^{-1}\lambda')^{\mu_1} =_F x\sigma\lambda$.

For this purpose, we define a substitution $(\pi^{-1}\lambda')^{\mu_1}$ on $Y_0 \cup C$ as follows: for all $z \in Y_0 \cup C$, $z(\pi^{-1}\lambda')^{\mu_1} := (z\pi^{-1}\lambda')^{\mu_1}$. Because the terms $x\sigma$ for $x \in X$ do not contain any of the free function symbols f_c, it is easy to see that $(x\sigma\pi^{-1}\lambda')^{\mu_1} = (x\sigma)(\pi^{-1}\lambda')^{\mu_1}$.

Obviously, the substitutions λ and $(\pi^{-1}\lambda')^{\mu_1}$ coincide on Y_0, but for $c \in C$ we need not have $c(\pi^{-1}\lambda')^{\mu_1} = c$. However, we can show that $c(\pi^{-1}\lambda')^{\mu_1} =_F c$. In fact, $c\pi^{-1}\lambda' = c\sigma'\lambda' =_F c\tau'$, and thus the lemma yields $(c\pi^{-1}\lambda')^{\mu_1} =_F (c\tau')^{\mu_1}$. But our assumption on μ yields $(c\tau')^{\mu_1} = c$. To sum up, we have just seen that $\lambda =_F (\pi^{-1}\lambda')^{\mu_1}\ \langle Y_0 \cup C\rangle$, which implies $(x\sigma)\lambda =_F (x\sigma)(\pi^{-1}\lambda')^{\mu_1}$. This completes the proof of the proposition. □

5.2 Using algorithms for constant elimination and for unification with constants

In this subsection we shall consider unification problems with *arbitrary* constant restrictions. It will be shown how to reduce solving this kind of problems to solving both unification problems with constants and constant elimination problems.

A *constant elimination problem* in the theory F is a finite set $\Delta = \{(c_1, t_1), ..., (c_n, t_n)\}$ where the c_i's are free constants (i.e., constant symbols not occurring in the signature of F) and the t_i's are terms (built over the signature of F, variables, and free constants). A solution to such a problem is called a *constant eliminator*. It is a substitution σ such that for all $i, 1 \leq i \leq n$, there exists a term t_i' not containing the free constant c_i

with $t'_i =_F t_i\sigma$. The notion *complete set of constant eliminators* is defined analogously to the notion complete set of unifiers.

Let Γ be an F-unification problem with *arbitrary* constant restriction. The goal is to construct a complete set of solutions of this problem. In the first step, we just ignore the constant restriction, and solve Γ as an ordinary F-unification problem with constants. Let $C(\Gamma)$ be a complete set of F-unifiers of this problem. In the second step, we define for all unifiers $\sigma \in C(\Gamma)$ a constant elimination problem Δ_σ as follows:

$$\Delta_\sigma := \{(c, x\sigma) \mid c \text{ is a free constant in } \Gamma \text{ and } x \in V_c\}.$$

For all $\sigma \in C(\Gamma)$, let C_σ be a complete set of solutions of the constant elimination problem Δ_σ.

Before we can describe the complete set of solutions of the F-unification problem with constant restriction, Γ, we have to define a slightly modified composition "$\otimes$" of substitutions. Let σ be an element of $C(\Gamma)$, and let τ be a constant eliminator in C_σ. Without loss of generality we assume that τ is the identity on variables not occurring in Δ_σ, and that the terms $y\tau$ for variables y occurring in Δ_σ contain only new variables. In particular, we will need for technical reasons that they do not contain variables occurring in terms $x\sigma$ for variables x occurring in Γ.

For a given variable x, let $\{c_1, \ldots, c_n\}$ be the set of all constants c_i occurring in Γ such that $x \in V_{c_i}$. If this set is empty (i.e., $n = 0$,) we define $x(\sigma \otimes \tau) := x\sigma\tau$. Now assume that $n > 0$. Obviously, we have $\{(c_1, x\sigma), \ldots, (c_n, x\sigma)\} \subseteq \Delta_\sigma$. Since τ is a solution of Δ_σ, there exist terms $s_1, \ldots, s_n$ such that for all $i, 1 \leq i \leq n$, $x\sigma\tau =_F s_i$ and c_i does not occur in s_i. It is easy to see that this also implies the existence of a single term s such $c_1, \ldots, c_n$ do not occur in s and $x\sigma\tau =_F s$. We define $x(\sigma\otimes\tau) := s$.

Proposition 5.5 *The set*

$$\bigcup_{\sigma \in C(\Gamma)} \{\sigma\otimes\tau \mid \tau \in C_\sigma\}$$

is a complete set of solutions of the F-unification problem with constant restriction, Γ.

Proof. First, we have to show that the elements $\sigma\otimes\tau$ of this set solve all the equations $s \doteq t$ of Γ. Since σ is an F-unifier of Γ, we have $s\sigma =_F t\sigma$, which implies $s\sigma\tau =_F t\sigma\tau$. But the definition of $\sigma\otimes\tau$ was such that $\sigma\tau =_F \sigma\otimes\tau$, and thus we get $s(\sigma\otimes\tau) =_F t(\sigma\otimes\tau)$.

Second, we must prove that $\sigma\otimes\tau$ satisfies the constant restriction. Assume that $x \in V_c$. Then the constant elimination problem Δ_σ contains the tuple $(c, x\sigma)$. By definition of $\sigma\otimes\tau$, we get that $x(\sigma\otimes\tau)$ is a term s not containing c.

Finally, it remains to be shown that the set is complete. Assume that θ is a solution of the F-unification problem with constant restriction, Γ. In particular, this means that θ solves the F-unification problem Γ (where the restrictions are ignored). Hence there exist an element σ of the complete set $C(\Gamma)$ and a substitution λ such that $\theta =_F \sigma\lambda \ \langle X\rangle$, where X denotes the set of all variables occurring in Γ. Thus we have

- for all $x \in X$: $x\theta =_F (x\sigma)\lambda$, and
- for all c with $x \in V_c$: c does not occur in $x\theta$,

which shows that λ solves the constant elimination problem Δ_σ. Consequently, there exist an element τ of the complete set C_σ and a substitution λ' such that $\lambda =_F \tau\lambda' \ \langle Y\rangle$, where Y denotes the set of all variables occurring in Δ_σ. Without loss of generality, we assume that $z\lambda' = z\lambda$ for all variables z not occurring in one of the terms $y\tau$ with $y \in Y$.

We want to show that for all $x \in X$ we have $x\theta =_F x(\sigma\otimes\tau)\lambda'$. For all $x \in X$, we know that $x\theta =_F x\sigma\lambda$, and since $\sigma\tau =_F \sigma\otimes\tau$ we also have $x(\sigma\otimes\tau)\lambda' =_F x\sigma\tau\lambda'$. Thus it remains to be shown that $x\sigma\lambda =_F x\sigma\tau\lambda'$. We have to distinguish two cases. First, assume that $(c, x\sigma) \in \Delta_\sigma$ for some c. In this case all variables occurring in $x\sigma$ are elements of Y, and thus $\lambda =_F \tau\lambda' \ \langle Y\rangle$ yields $x\sigma\lambda =_F x\sigma\tau\lambda'$. For the second case, assume that $x\sigma$ contains a variable z which is not an element of Y, the set of all variables occurring in Δ_σ. We are finished if we can show that, nevertheless, $z\lambda =_F z\tau\lambda'$ holds for all such variables z. Since τ was assumed to be the identity on variables not occurring in Δ_σ, we have $z\tau = z$. Since z occurs in $x\sigma$ for a variable $x \in X$, our second assumption on τ implies that z does not occur in any term $y\tau$ with $y \in Y$. But then $z\lambda' = z\lambda$ by our assumption on λ', which completes the proof of $x\sigma\lambda =_F x\sigma\tau\lambda'$. □

6 Conclusion

We have presented a new method for treating the problem of unification in the union of disjoint equational theories. Unlike most of the other methods developed for this purpose, it can be used to combine decision procedures as well as procedures computing finite complete sets of unifiers. Applicability of our method depends on a new type of prerequisite, namely on the solvability of unification problems with *linear* constant restrictions. Presupposing the existence of a constant elimination algorithm—as necessary for the method of Schmidt-Schauß—seems to be a stronger requirement. In fact, we have seen that constant elimination procedures can be used to solve unification problems with *arbitrary* constant restrictions. However, it is still an open problem whether there exists an equational theory for which solving unification problems with linear constant restrictions is finitary (or decidable) but solving unification problems with arbitrary constant restrictions is not.

Our main results together with the results described in the previous section show that there is a close correspondence between solving unification problems with linear constant restrictions and solving general unification problems. For a given equational theory, the first kind of problems is decidable (finitary solvable) if and only if the second kind of problems is. As an interesting open problem it remains to be shown whether there exists an equational theory for which unification with constants is decidable (finitary) but general unification—or equivalently, solving unification problems with linear constant restrictions—is not. One should note that there already exist such results for the case of single equations, i.e., unification problems of cardinality one. Narendran and Otto [NO90] have shown that there exists an equational theory E such that solvability is decidable for E-unification problems (with constants) of cardinality one, but is undecidable for E-unification problems of cardinality greater than one, and thus also for general E-unification problems.

To make the presentation and the proof of correctness of the combination method more concise, we did not consider possible optimizations which would rule out certain partitions in Step 3 and certain linear orderings in Step 4 of the algorithm.

References

[Ba91] F. Baader, "Unification Theory," Proceedings of the First International Workshop on Word Equations and Related Topics, to appear as Springer LNCS.

[Bc87] L. Bachmair, *Proof Methods for Equational Theories*, Ph.D. Thesis, Dept. of Comp. Sci., University of Illinois at Urbana-Champaign, 1987.

[Bo90] A. Boudet, "Unification in a Combination of Equational Theories: An Efficient Algorithm," *Proceedings of the 10th International Conference on Automated Deduction*, *LNCS* **449**, 1990.

[BJ89] A. Boudet, J.P. Jouannaud, M. Schmidt-Schauß, "Unification in Boolean Rings and Abelian Groups," *J. Symbolic Computation* **8**, 1989.

[Bü86] H.-J. Bürckert, "Some Relationships Between Unification, Restricted Unification, and Matching," *Proceedings of the 8th International Conference on Automated Deduction*, *LNCS* **230**, 1986.

[Bü90] H.-J. Bürckert, "A Resolution Principle for Clauses with Constraints," *Proceedings of the 10th International Conference on Automated Deduction*, *LNCS* **449**, 1990.

[Co90] A. Colmerauer, "An Introduction to PROLOG III," *C. ACM* **33**, 1990.

[DJ87] N. Dershowitz, J.P. Jouannaud, "Rewrite Systems," Unité Associée au CNRS UA 410: AL KHOWARIZMI, Rapport de Recherche $n°$ 478, 1989. To appear in Volume B of "Handbook of Theoretical Computer Science," North-Holland.

[Fa84] F. Fages, "Associative-Commutative Unification," *Proceedings of the 7th International Conference on Automated Deduction*, *LNCS* **170**, 1984.

[He86] A. Herold, "Combination of Unification Algorithms," *Proceedings of the 8th International Conference on Automated Deduction*, *LNCS* **230**, 1986.

[He87] A. Herold, *Combination of Unification Algorithms in Equational Theories*, Dissertation, Fachbereich Informatik, Universität Kaiserslautern, 1987.

[HS87] A. Herold, J.H. Siekmann, "Unification in Abelian Semigroups," *J. Automated Reasoning* **3**, 1987.

[Ho76] J.M. Howie, *An Introduction to Semigroup Theory*, London: Academic Press, 1976.

[JL84] J. Jaffar, J.L. Lassez, M. Maher, "A Theory of Complete Logic Programs with Equality," *J. Logic Programming* **1**, 1984.

[JL87] J. Jaffar, J.L. Lassez, "Constraint Logic Programming," *Proceedings of 14th POPL Conference*, Munich, 1987.

[JK86] J.P. Jouannaud, H. Kirchner, "Completion of a Set of Rules Modulo a Set of Equations," *SIAM J. Computing* **15**, 1986.

[JK90] J.P. Jouannaud, C. Kirchner, "Solving Equations in Abstract Algebras: A Rule-Based Survey of Unification," Preprint, 1990. To appear in the Festschrift to Alan Robinson's birthday.

[KN91] D. Kapur, P. Narendran, "Complexity of Unification Problems with Associative-Commutative Operators," Preprint, 1991. To appear in *J. Automated Reasoning*.

[Ki85] C. Kirchner, *Méthodes et Outils de Conception Systématique d'Algorithmes d'Unification dans les Théories equationnelles*, Thèse d'Etat, Univ. Nancy, France, 1985.

[KK89] C. Kirchner, H. Kirchner, "Constrained Equational Reasoning," *Proceedings of SIGSAM 1989 International Symposium on Symbolic and Algebraic Computation*, ACM Press, 1989.

[LS75] M. Livesey, J.H. Siekmann, "Unification of AC-Terms (bags) and ACI-Terms (sets)," Internal Report, University of Essex, 1975, and Technical Report 3-76, Universität Karlsruhe, 1976.

[Ma77] G.S. Makanin, "The Problem of Solvability of Equations in a Free Semigroup," *Mat. USSR Sbornik* **32**, 1977.

[NO90] P. Narendran, F. Otto, "Some Results on Equational Unification," *Proceedings of the 10th Conference on Automated Deduction, LNCS* **449**, 1990.

[Pl72] G. Plotkin, "Building in Equational Theories," *Machine Intelligence* **7**, 1972.

[Sc89] M. Schmidt-Schauß, "Combination of Unification Algorithms," *J. Symbolic Computation* **8**, 1989.

[Sh91] K. Schulz, "Makanin's Algorithm – Two Improvements and a Generalization," CIS-Report 91-39, CIS, University of Munich, 1991.

[Si89] J.H. Siekmann, “Unification Theory: A Survey,” in C. Kirchner (ed.), *Special Issue on Unification, Journal of Symbolic Computation* **7**, 1989.

[St75] M. Stickel, “A Complete Unification Algorithm for Associative-Commutative Functions,” *Proceedings of the International Joint Conference on Artificial Intelligence*, 1975.

[St81] M. Stickel, “A Unification Algorithm for Associative-Commutative Functions,” *J. ACM* **28**, 1981.

[St85] M. Stickel, “Automated Deduction by Theory Resolution,” *J. Automated Reasoning* **1**, 1985.

[Ti86] E. Tiden, “Unification in Combinations of Collapse Free Theories with Disjoint Sets of Function Symbols,” *Proceedings of the 8th International Conference on Automated Deduction*, *LNCS* **230**, 1986.

[Ye87] K. Yelick, “Unification in Combinations of Collapse Free Regular Theories,” *J. Symbolic Computation* **3**, 1987.