Non-Disjoint Combination with Forward-Closed Theories
(Extended Abstract)

Serdar Erbatur$^{1}$, Andrew M. Marshall$^{2}$, and Christophe Ringeissen$^{3}$

$^1$ Ludwig-Maximilians-Universität, München (Germany)
serdar.erbatur@ifi.lmu.de
$^2$ University of Mary Washington (USA)
marshall@umw.edu
$^3$ LORIA – INRIA Nancy-Grand Est (France)
Christophe.Ringeissen@loria.fr

1 Introduction

Equational unification is the problem of solving equations in structures of terms modulo an equational theory. In general, equational unification is undecidable, but specialized techniques have been developed to solve the problem for particular classes of equational theories.

When the equational theory has the Finite Variant Property (FVP) [4], equational unification reduces to syntactic unification via the computation of finitely many variants of terms. Nowadays, equational theories with the FVP have attracted a considerable interest, especially for their applications in the analysis of security protocols [2,5,10].

When the equational theory is given by a convergent term rewrite system, the concept of narrowing is a generalization of rewriting, where the matching process is replaced by some (syntactic) unification problem. Narrowing is complete for equational unification, but it terminates only in some very particular cases. Hence, a particular narrowing strategy, called folding variant narrowing, is complete and terminating for any equational theory with the FVP [10].

When the equational theory is syntactic, it is possible to apply a mutation-based unification procedure [11]. However, being syntactic is not a sufficient condition for a theory to admit a terminating mutation-based unification procedure. In the particular case of shallow theories, there exists a terminating mutation-based unification procedure [3].

Another important scenario is given by an equational theory defined as a union of component theories. To solve this case, it’s natural to proceed in a modular way and there are terminating and complete combination procedures for disjoint unions of theories [16]. These combination procedures can be extended to some particular non-disjoint unions of theories, but it is difficult to identify cases where these procedures terminate [6,15].

In this paper we investigate the unification problem in equational theories involving forward-closed convergent term rewrite systems [2]. In the class of forward-closed theories, unification is decidable and finitary since any convergent term rewrite system has a finite forward closure if and only if it has the FVP [2]. Furthermore, forward-closed theories are syntactic theories admitting a terminating mutation-based unification procedure. We first demonstrate this result by showing that a mutation-based unification algorithm, originally developed for equational theories saturated by paramodulation [13], remains sound and complete for forward-closed theories. Building on this result we use the new mutation-based algorithm to develop an algorithm that solves the unification problem in unions of forward-closed theories with non-disjoint theories. The resulting algorithm can be viewed as a terminating instance of a procedure initiated for hierarchical combination [7].
2 Preliminaries

We assume the reader is familiar with equational unification and term rewriting systems [1]. An axiom \( l = r \) is regular (also called variable-preserving) if \( \text{Var}(l) = \text{Var}(r) \). An axiom \( l = r \) is linear (resp., collapse-free) if \( l \) and \( r \) are linear (resp. non-variable terms). An equational theory is regular (resp., linear/collapse-free) if all its axioms are regular (resp., linear/collapse-free).

A theory \( E \) is syntactic if it has a finite resolvent presentation \( S \), that is a presentation \( S \) such that each equality \( t =_E u \) has an equational proof \( t \leftrightarrow^*_S u \) with at most one step \( \leftrightarrow_S \) applied at the root position. In the following, we often use tuples of terms, say \( \bar{u} = (u_1, \ldots, u_n) \).

An \( E \)-unification problem is a set of \( \Sigma \)-equations, \( G = \{ s_1 =_T t_1, \ldots, s_n =_T t_n \} \), or equivalently a conjunction of \( \Sigma \)-equations. The set of variables in \( G \) is denoted by \( \text{Var}(G) \). A solution to \( G \), called an \( E \)-unifier, is a substitution \( \sigma \) such that \( s_i \sigma =_E t_i \sigma \) for all \( 1 \leq i \leq n \). A substitution \( \sigma \) is more general modulo \( E \), called an \( E \)-unifier, if it is a conjunction of \( \Sigma \)-equations for \( E \)-unifiable \( \Sigma \)-terms.

Let us now define the notion of forward-closure [2]. For a given convergent TRS \( R \), assume a reduction order \( < \) such that \( r < l \) for any \( l \rightarrow r \in R \) and \( < \) is total on ground terms. Since (rewrite) rules are multisets of two terms, the multiset extension of \( < \) leads to an ordering on rules, also denoted by \( < \), which is total on ground instances of rules. A rule \( \rho \) is strictly redundant in \( R \) if any ground instance \( \rho \sigma \) of \( \rho \) follows from ground instances of \( R \) that are strictly smaller w.r.t \( < \) than \( \rho \sigma \). A rule \( \rho \) is redundant in \( R \) if \( \rho \) is strictly redundant in \( R \) or \( \rho \) is an instance of some rule in \( R \). A TRS \( R \) is forward-closed if any \textbf{Forward} inference (cf. Figure 1) with premises in \( R \) generates a rule which is redundant in \( R \).

![Figure 1: Forward inference](image)

3 Unification in Forward-Closed Rewrite Systems

We present a rule-based unification procedure for any forward-closed TRS. We basically reuse the unification procedure initially developed for any equational theory saturated by paramod-
ulation [13]. This procedure implements Basic Syntactic Mutation (BSM) by extending syntactic unification with some additional mutation rules (rules whose names include Mut in Figures 2 and 3) and cycle breaking rules (rules whose names include Cycle in Figure 3). These additional rules are applied in a don’t know non-deterministic way. Thus, the resulting BSM unification procedure is similar to the mutation-based unification procedures designed for syntactic theories [12, 14]. However, these mutation-based procedures are not terminating, whereas the BSM unification procedure always terminates. To get termination, the BSM rules depicted in Figures 2 and 3 make use of boxed terms. This particular annotation of terms works as follows: Subterms of boxed terms are also boxed, terms boxed in the premises of an inference rule remain boxed in the conclusion, and when the “box” status of a term is not explicitly given in an inference rule, it can be either boxed or unboxed. Given a unification problem $G$ and an $R$-normalized substitution $\sigma$, $(G, \sigma)$ is said to be $R$-normalized if $t\sigma$ is $R$-normalized whenever $t$ is boxed in $G$. For a forward-closed convergent TRS $R$, the set of equalities $S$ used in Figures 2 and 3 is defined as being equal to $RHS(R) = R^\neq \cup \{\sigma = g\sigma \mid l \rightarrow r \in R, g \rightarrow d \in R, \sigma \in mgu(r,d), l\sigma \neq g\sigma\}$, where $R^\neq = \{l = r \mid l \rightarrow r \in R\}$.

| Dec  | $f(\bar{u}) = f(\bar{v}) \cup G \vdash \{\bar{u} = \bar{v}\} \cup G$ |
| Mut | $f(\bar{u}) = g(\bar{v}) \cup G \vdash \{\bar{u} = \bar{v}\} \cup G$ where $f(\bar{u})$ is unboxed and $f(\bar{v}) = g(\bar{v}) \in S$. |
| Imit | $\bigcup_i \{x = f(\bar{u}_i)\} \cup G \vdash \{x = \{f(\bar{y})\}\} \cup \bigcup_i \{y = \bar{v}_i\} \cup G$ where $i > 1$ and there are no more equations $x = f(\ldots)$ in $G$. |
| MutImit | $\{x = f(\bar{u}), x = g(\bar{v})\} \cup G \vdash \{x = f(\bar{y}), \bar{y} = \{\bar{u}\} = \bar{u}, \bar{v} = \bar{v}\} \cup G$ where $f(\bar{u})$ is boxed, $g(\bar{v})$ is unboxed, and $f(\bar{y})$ is boxed; |
| 1. if $f(\bar{u})$ is boxed, $g(\bar{v})$ is unboxed, then $f(\bar{y})$ is boxed; |
| 2. if $f(\bar{u})$ and $g(\bar{v})$ are unboxed, then $f(\bar{y})$ is unboxed. |
| Coalesce | $\{x = y\} \cup G \vdash \{x = y\} \cup (G(x \rightarrow y))$ |
| where $x$ and $y$ are distinct variables occurring both in $G$. |

Figure 2: BSM rules

Given an $R$-unification problem $G$, the BSM unification procedure works as follows: apply the BSM rules (Figures 2 and 3) on $G$ until reaching normal forms. The procedure then only returns those sets of equations which are in dag solved form.

**Theorem 1.** If $R$ is a forward-closed convergent TRS, then the BSM unification procedure provides an $R$-unification algorithm.

## 4 Forward-Closed Combination

In previous papers [7,8], we have studied a form of non-disjoint combination, called hierarchical combination, which is defined as a convergent TRS $R_1$ combined with a base theory $E_2$. The TRS $R_1$ must satisfy some properties to ensure that $E = R_1 \cup E_2$ is a conservative extension of $E_2$. We are interested in combined theories $E$ where it is possible to reduce any $E$-equality between two terms ($s =_E t$) into the $E_2$-equality of their $R_1$-normal forms ($s \downarrow_{R_1} =_{E_2} t \downarrow_{R_1}$). Below, we assume that $R_1$ is forward-closed.
Example 1. Consider equational \( \Sigma \) TRS \( R \) such that:

\[ A \]

Non-Disjoint Combination with Forward-Closed Theories
Erbatur, Marshall and Ringeissen

collapse-free is true. Either,

\[ \text{Lemma 1. Assume} \]

definition of \( S \) can be lifted to take into account \( E_2 \). Hence, we can compute

![Figure 3: Additional BSM rules for a subterm collapsing theory](image)

<table>
<thead>
<tr>
<th>VarMut</th>
<th>( { f(\bar{u}) = v } \cup G \vdash { \bar{u} = \overline{\underbrace{\bar{x}}_{\text{unboxed}}} y = v } \cup G )</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImitCycle</td>
<td>( { x = f(\bar{v}) } \cup G \vdash { x = \overline{\underbrace{f(\bar{y})}_{\text{unboxed}}} y = \bar{v} } \cup G )</td>
</tr>
<tr>
<td>MutImitCycle</td>
<td>( { x = f(\bar{v}) } \cup G \vdash { x = \overline{\underbrace{g(\bar{t})}_{\text{unboxed}}} - \bar{v} } \cup G )</td>
</tr>
</tbody>
</table>

where \( f(\bar{u}) \) is unboxed, \( f(\bar{s}) = y \in S \) with a variable \( y \), and if \( v \) is a variable, then there is another equation \( v = t \in G \) with a non-variable term \( t \), or \( v = f(\bar{u}) \) occurs in a cycle.

If no other rule applies among \textbf{VarMut} and those in Figure 2, \( f(\bar{v}) \) is unboxed and \( x = f(\bar{v}) \) occurs in a cycle.

Figure 3: Additional BSM rules for a subterm collapsing theory

Definition 1. A forward-closed combination (FC-combination, for short) is a pair \((E_1, E_2)\) such that: \( \Sigma_1 \cap \Sigma_2 = \emptyset \); \( E_1 \) is an equational \( \Sigma_1 \cup \Sigma_2 \)-theory given by a forward-closed convergent TRS \( R_1 \) such that its left-hand sides are linear \( \Sigma_1 \)-terms; \( E_2 \) is a regular and collapse-free equational \( \Sigma_2 \)-theory. An FC-combination \((E_1, E_2)\) is said to be layer-preserving if \( R_1 \) is \( \Sigma_1 \)-rooted and regular.

Example 1. Consider \( R_1 = \{ \exp(a, y) \times \exp(a, z) \rightarrow \exp(a, y + z) \} \) with \( \Sigma_1 = \{ a, \exp, \times \} \) and \( \Sigma_2 = \{ + \} \). Then, an FC-combination can be obtained by considering any regular and collapse-free \( \Sigma_2 \)-theory \( E_2 \), such as Commutativity or Associativity-Commutativity.

From now on, we assume an \( E_2 \)-unification algorithm, a layer-preserving FC-combination \((E_1, E_2)\) such that \( E_1 \) is given by a forward-closed convergent TRS \( R_1 \), and a combined theory \( E = E_1 \cup E_2 \). An \( E_1 \)-unification algorithm is provided by BSM (cf. Section 3), where \textbf{VarMut} does not apply since \( E_1 \) is collapse-free.

5 Unification Procedure for Forward-Closed Combination

We study how the BSM unification procedure can be combined with an \( E_2 \)-unification algorithm to solve any \( E \)-unification problem. Remember that BSM is parameterized by a set of equalities \( S \) used in the mutation rules. In order to transform \( \Sigma_1 \)-rooted equations, the definition of \( S \) can be lifted to take into account \( E_2 \). Hence, we can compute

\[ S = RHS_{E_2}(R_1) = R_1^{\sigma} \cup \{ l \rightarrow r \in R_1, g \rightarrow d \in R_1, \sigma \in CSU_{E_2}(r = d), l\sigma \neq g\sigma \} \]

Lemma 1. Assume \( S = RHS_{E_2}(R_1) \). For each \( \Sigma_1 \)-rooted equality \( u =_E v \), one of the following is true. Either, \( u = f(\bar{u}) \), \( v = f(\bar{v}) \) and \( \bar{u} =_E \bar{v} \). Or, \( u = f(\bar{u}) \), \( v = g(\bar{v}) \) and there are \( f(\bar{s}) = g(\bar{t}) \in S \) and an \( R_1 \)-normalized substitution \( \sigma \) such that \( \bar{u} =_E \bar{s}\sigma \), \( \bar{v} =_E \bar{t}\sigma \) where \( \bar{s}\sigma \) and \( \bar{t}\sigma \) are \( R_1 \)-normalized.

Consider the inference system BSC defined by the set of rules in Figure 4 plus the BSM rules in Figures 2 and 3, parameterized by \( S = RHS_{E_2}(R_1) \), with the restriction that BSM rules are applied only if the input is in separate form, and by matching only \( \Sigma_1 \)-equations.

Lemma 2. Given an \( E \)-unification problem \( G \) as input, the repeated application of BSC rules always terminates and computes a set of normal forms in separate form denoted by BSC\((G)\).
Following Lemma 2, the BSC unification procedure works as follows: apply the BSC rules on a given \( E \)-unification problem \( G \) until reaching normal forms, and return all the dag solved forms in \( \text{BSC}(G) \). The completeness of the BSC unification procedure relies on the lemma given below. First, we state that an \( E_2 \)-unification algorithm can be reused without loss of completeness to \( E \)-unify any conjunction of \( \Sigma_2 \)-equations. This is a classical result, already used in hierarchical combination [7], which can be easily lifted to FC-combination.

\[
\begin{align*}
\text{VA} & \quad \{ s = t[u] \} \cup G \vdash \{ s = t[x], x = u \} \cup G \\
\text{IE} & \quad \{ s = t \} \cup G \vdash \{ x = s, x = t \} \cup G \\
\text{Solve2} & \quad G_1 \land G_2 \vdash V_{\sigma \in \text{CSU}_E(G_2)}(G_1 \land \sigma_2)
\end{align*}
\]

where \( u \) is an alien subterm of \( t \), \( x \) is a fresh variable, and \( u \) is boxed iff \( t[u] \) is boxed.

where \( s \) is a non-variable \( \Sigma_1 \)-term, \( t \) is a non-variable \( \Sigma_2 \)-term and \( x \) is a fresh variable.

if \( G_2 \) is \( E_2 \)-unifiable and unsolved, and no other rule applies.

Figure 4: Additional Rules for the combination with \( E_2 \)

**Lemma 3** (Completeness). If \( \sigma \) is an \( E \)-unifier of \( G \), \((G, \sigma)\) is \( R_1 \)-normalized, and \( G \) is not a separate form in dag solved form, then there exist some \( G' \) and a substitution \( \sigma' \) such that \( G' \) is obtained from \( G \) by applying some BSC rule, \( \sigma' \) is an \( E \)-unifier of \( G' \), \((G', \sigma')\) is \( R_1 \)-normalized, and \( \sigma' \leq V_{\text{Var}}(G) \sigma \).

According to Lemmas 2 and 3, we can conclude that BSC leads to a terminating and complete \( E \)-unification procedure: given an \( E \)-unification problem, the set of dag solved forms in \( \text{BSC}(G) \) provides a \( \text{CSU}_E(G) \). Then, this \( E \)-unification algorithm can be lifted to a general \( E \)-unification algorithm by assuming that \( E_2 \) includes the free symbols.

**Theorem 2.** Given any layer-preserving FC-combination \((E_1, E_2)\) and an \( E_2 \)-unification algorithm, BSC provides a general \( E_1 \cup E_2 \)-unification algorithm.

For sake of simplicity, we restrict us in this short paper to layer-preserving FC-combinations, where all conflicts between component theories have no solution just like in disjoint unions of regular and collapse-free theories [17]. The general case of arbitrary FC-combinations, where the TRS \( R_1 \) may contain non-regular or collapse axioms, will be addressed in a full version [9]. In the general case, we will investigate the possibility of considering a single mutation rule dedicated to equations between a variable and a \( \Sigma_1 \)-rooted term.

### 6 Connection with the Finite Variant Property

The computation of finite variants [4,10] is another way to reduce any unification problem modulo \( E = R_1 \cup E_2 \) into some \( E_2 \)-unification problems with free function symbols. Thus, there exists an interesting connection between the finite variant property and FC-combinations. It can be shown that a brute force method can be used to solve \( E \)-unification: compute all the finitely many \( R_1 \)-variants of an input \( E \)-unification problem, and solve them by general \( E_2 \)-unification, usually implemented by combining \( E_2 \)-unification and syntactic unification. This is a highly non-deterministic method. The brute force approach of computing all variants can be prohibitive due to the possible large number of variants and thus it is desirable to have alternatives to that approach. From our point of view, the approach described in Section 5 provides an interesting alternative where all the non-determinism is managed inside the combination procedure, when mutation rules and Solve2 have to be applied.
References


