Terminating Non-Disjoint Combined Unification

Serdar Erbatur, Andrew M Marshall, Christophe Ringeissen

Terminating Non-Disjoint Combined Unification

Serdar Erbatur[1], Andrew M. Marshall[2], and Christophe Ringeissen[3]

[1] University of Texas at Dallas, USA
[2] University of Mary Washington, USA
[3] Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France

Abstract. The equational unification problem, where the underlying equational theory may be given as the union of component equational theories, appears often in practice in many fields such as automated reasoning, logic programming, declarative programming, and the formal analysis of security protocols. In this paper, we investigate the unification problem in the non-disjoint union of equational theories via the combination of hierarchical unification procedures. In this context, a unification algorithm known for a base theory is extended with some additional inference rules to take into account the rest of the theory. We show theories for which a hierarchical approach is applicable and present a simple form of hierarchical unification procedure. The approach is particularly well-suited for any theory where a unification procedure can be obtained in a syntactic way using transformation rules to process the axioms of the theory. Hierarchical unification procedures are exemplified with various theories used in protocol analysis. Next, we look at modularity methods for combining theories already using a hierarchical approach. In addition, we consider a new complexity measure that allows us to obtain terminating (combined) hierarchical unification procedures.

1 Introduction

Unification is a critical tool in many fields such as automated reasoning, logic programming, declarative programming, and the formal analysis of security protocols. For many of these applications we want to consider equational unification, where the problem is defined modulo an equational theory E, such as Associativity-Commutativity. For example, one approach to the analysis of security protocols is based on deductive reasoning, as is done in the following tools [6,5,18,25]. In this approach protocols are usually represented by clauses in first-order logic with equality and equational theories are used to specify the capabilities of an intruder [1]. To support this reasoning approach we need to use E-unification procedures. Since equational unification is undecidable in general, specialized techniques have been developed to solve the problem for particular classes of equational theories, many of high practical interest. For instance, when the equational theory E has the Finite Variant Property (FVP) [11,19], there exists a reduction from E-unification to syntactic unification via the computation of finitely many variants of the unification problem. The class of equational

theories with the FVP has attracted a considerable interest since it contains theories that are crucial in protocol analysis [19,8,7,12,26].

Another ubiquitous scenario is given by an equational theory E involved in a union of theories $F \cup E$. To solve this case, it is quite natural to proceed in a modular way by reusing the unification algorithms available for F and for E. There are terminating and complete combination procedures for signature-disjoint unions of theories [29,3]. However, the non-disjoint case remains a challenging problem. One approach to the non-disjoint combination problem that has been successful in some cases is the hierarchical approach [14]. In this approach, $F \cup E$-unification can be considered as a conservative extension of E-unification. Then, a new inference system related to F, say U_F, can be combined with an E-unification algorithm to obtain an $F \cup E$ unification algorithm. While this hierarchical approach won't work for every $F \cup E$ it can be a very useful tool when applicable. However, up to now it could be complex to know if a combination $F \cup E$ could be solved via the hierarchical approach. For example, there is no general method for obtaining the inference system U_F, and the resulting hierarchical unification procedure may not terminate.

In this paper, we consider "syntactic" theories $F \cup E$ where U_F can be defined as a system of mutation rules, and we present new terminating instances of the hierarchical unification procedure. When an equational theory fulfills the syntacticness property [22,28], there exists a rule-based unification procedure in the same vein as the one known for syntactic unification, which is called a mutation-based unification procedure. Unfortunately, being syntactic is not a sufficient condition to ensure the termination of this mutation-based unification procedure. However, terminating mutation-based unification procedures are known for some particular theories such as one one-side distributivity [30,24], distributive exponentiation theories [15], shallow theories [10] and theories closed by paramodulation [23]. All the theories investigated here using the hierarchical approach are both *syntactic* and finitary: each of them is actually a syntactic theory for which a (finitary) unification algorithm is shown. On the one hand, we study theories which are both collapse-free and finitary, that is, finitary theories defined by axioms between non-variable terms. These theories are known to be syntactic [22]. On the other hand, we also examine forward-closed theories that are known to be both syntactic and finitary, just like theories closed by paramodulation [23]. The forward-closed theories we are interested in are actually examples of theories having the Finite Variant Property.

The contributions of the paper consist of several improvements to the hierarchical combination method [14,13] including: simplifying the method, clarifying the theories for which the approach is applicable, and reducing some of the restrictions. Furthermore, we develop several new results including general reduction procedures for certain types of theories, and modular termination results. More specifically:

– We better define theories for which a hierarchical approach is applicable, constructor-based theories, and simplify the hierarchical unification proce-

dure denoted here by $H_E(U_F)$, where U_F is an additional rule-based procedure to be combined with an E-unification algorithm (Section 3).
- We define the requirements for the U_F rule-based procedure, and develop new general rule-based procedures for subterm collapse-free and forward-closed theories (Section 3).
- Using the hierarchical approach, we develop new modularity results for the unification problem in unions of constructor-sharing theories. We define a new complexity measure to show terminating combinations of hierarchical unification algorithms. This allows us to obtain new (combined) unification algorithms for a wider variety of theories (Section 4).
- We show how the combination of hierarchical unification algorithms can be applied to unions of constructor-sharing forward-closed theories (Section 4).

The rest of the paper is organized as follows. Section 2 provides the background material. Section 2.3 contains an introduction to forward-closed theories. Section 3 introduces the notion of hierarchical unification and presents examples of theories admitting a hierarchical unification algorithm. Section 4 focuses on the combination of hierarchical unification algorithms. Finally, Section 5 contains the conclusions and future work.

2 Preliminaries

We use the standard notation of equational unification [4] and term rewriting systems [2]. Given a first-order signature Σ and a (countable) set of variables V, the set of Σ-terms over variables V is denoted by $T(\Sigma, V)$. The set of variables in a term t is denoted by $Var(t)$. A term t is *ground* if $Var(t) = \emptyset$. A term is *linear* if all its variables occur only once. For any position p in a term t (including the root position ϵ), $t(p)$ is the symbol at position p, $t|_p$ is the subterm of t at position p, and $t[u]_p$ is the term t in which $t|_p$ is replaced by u. A substitution is an endomorphism of $T(\Sigma, V)$ with only finitely many variables not mapped to themselves. A substitution is denoted by $\sigma = \{x_1 \mapsto t_1, \ldots, x_m \mapsto t_m\}$, where the domain of σ is $Dom(\sigma) = \{x_1, \ldots, x_m\}$. Application of a substitution σ to t is written $t\sigma$.

2.1 Equational Theories

Given a set E of Σ-axioms (i.e., pairs of Σ-terms, denoted by $l = r$), the *equational theory* $=_E$ is the congruence closure of E under the law of substitutivity (by a slight abuse of terminology, E is often called an equational theory). Equivalently, $=_E$ can be defined as the reflexive transitive closure $\leftrightarrow_E^*$ of an equational step $\leftrightarrow_E$ defined as follows: $s \leftrightarrow_E t$ if there exist a position p of s, $l = r$ (or $r = l$) in E, and substitution σ such that $s|_p = l\sigma$ and $t = s[r\sigma]_p$. An axiom $l = r$ is *regular* if $Var(l) = Var(r)$. 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 *subterm collapse-free* if and only if for all terms t it is not the case that $t =_E u$ where u is a strict subterm of t. A theory E is *syntactic* if it has finite *resolvent presentation* S, defined as a finite set of axioms S such that each equality $t =_E u$ has an equational proof $t \leftrightarrow_S^* u$ with at most one equational step $\leftrightarrow_S$ applied at the root position. One can easily check that $C = \{x * y = y * x\}$ (Commutativity) and $AC = \{x*(y*z) = (x*y)*z,\ x*y = y*x\}$ (Associativity-Commutativity) are regular, collapse-free, and linear. Moreover, C and AC are syntactic [22]. A Σ-equation is a pair of Σ-terms denoted by $s =^? t$ or simply $s = t$ when it is clear from the context that we do not refer to an axiom. A *flat* Σ-equation is either an equation between variables or a *non-variable flat* Σ-equation of the form $x_0 = f(x_1, \ldots, x_n)$ where $x_0, x_1, \ldots, x_n$ are variables and f is a function symbol in Σ. An E-unification problem is a set of Σ-equations, $G = \{s_1 =^? t_1, \ldots, s_n =^? t_n\}$, or equivalently a conjunction of Σ-equations. The set of variables in G is denoted by $Var(G)$. A solution to G, called an *E-unifier*, is a substitution σ such that $s_i\sigma =_E t_i\sigma$ for all $1 \leq i \leq n$, written $E \models G\sigma$. A substitution σ is *more general modulo E* than θ on a set of variables V, denoted as $\sigma \leq_E^V \theta$, if there is a substitution τ such that $x\sigma\tau =_E x\theta$ for all $x \in V$. A *Complete Set of E-Unifiers* of G, denoted by $CSU_E(G)$, is a set of substitutions such that each $\sigma \in CSU_E(G)$ is an E-unifier of G, and for each E-unifier θ of G, there exists $\sigma \in CSU_E(G)$ such that $\sigma \leq_E^{Var(G)} \theta$. An *$E$-unification algorithm* is an algorithm that computes a finite $CSU_E(G)$ for all E-unification problems G. An inference rule $G \vdash G'$ for E-unification is *sound* if each E-unifier of G' is an E-unifier of G; and *complete* if for each E-unifier σ of G, there exists an E-unifier σ' of G' such that $\sigma' \leq_E^{Var(G)} \sigma$. An inference system for E-unification is *sound* if all its inference rules are sound; and *complete* if for each E-unification problem G on which an inference applies and each E-unifier σ of G, there exist an E-unification problem G' inferred from G and an E-unifier σ' of G' such that $\sigma' \leq_E^{Var(G)} \sigma$. A set of equations $G = \{x_1 =^? t_1,\ \ldots,\ x_n =^? t_n\}$ is said to be in *tree solved form* if each x_i is a variable occurring once in G. Given an idempotent substitution $\sigma = \{x_1 \mapsto t_1,\ \ldots,\ x_n \mapsto t_n\}$ (such that $\sigma\sigma = \sigma$), $\hat{\sigma}$ denotes the corresponding tree solved form. A set of equations is said to be in *dag solved form* if they can be arranged as a list $x_1 =^? t_1,\ \ldots,\ x_n =^? t_n$ where (a) each left-hand side x_i is a distinct variable, and (b) $\forall\, 1 \leq i \leq j \leq n$: x_i does not occur in t_j. A set of equations $\{x_1 =^? t_1,\ \ldots,\ x_n =^? t_n\}$ is a *cycle* if for any $i \in [1, n-1], x_{i+1} \in Var(t_i)$, $x_1 \in Var(t_n)$, and there exists $j \in [1, n]$ such that t_j is not a variable. Given two disjoint signatures Σ_0 and Σ_1 and any $i = 1, 0$, Σ_i-terms (including the variables) and Σ_i-equations (including the equations between variables) are called *Σ_i-pure*. A term t is called a *Σ_i-rooted* term if its root symbol is in Σ_i. An *alien* subterm of a Σ_i-rooted term t is a Σ_j-rooted subterm s $(i \neq j)$ such that all superterms of s are Σ_i-rooted. We define *general E-unification* as the unification problem in the equational theory obtained by extending E with arbitrary free function symbols.

Given a Σ_0-theory E, a theory $F \cup E$ is a *conservative extension* of E if $=_{F\cup E}$ and $=_E$ coincide on Σ_0-terms. When $F \cup E$ is a conservative extension of E, E-unification is said to be *complete for solving the Σ_0-fragment of $F \cup E$-*

unification if for any Σ_0-pure $F \cup E$-unification problem G, any $CSU_E(G)$ is a $CSU_{F\cup E}(G)$. If F and E have disjoint signatures, E-unification is known to be complete for solving the Σ_0-fragment of $F \cup E$-unification.

2.2 Equational Term Rewrite Systems

Given a signature Σ, an *equational term rewrite system* (TRS) (R, E) over Σ is defined by a Σ-theory E and a finite set R of oriented Σ-axioms called rewrite rules and of the form $l \to r$ such that l, r are Σ-terms, l is not a variable and $Var(r) \subseteq Var(l)$. A term s *rewrites* to a term t w.r.t (R, E), denoted by $s \to_{R,E} t$, if there exist a position p of s, $l \to r \in R$, and substitution σ such that $s_{|p} =_E l\sigma$ and $t = s[r\sigma]_p$. The term $s_{|p}$ is called a redex. Given a TRS (R, E), $\longleftrightarrow_{R\cup E}$ denotes the symmetric relation $\leftarrow_{R,E} \cup \to_{R,E} \cup =_E$. A TRS (R, E) is Church-Rosser modulo E if $\longleftrightarrow^*_{R\cup E}$ is included in $\to^*_{R,E} \circ =_E \circ \leftarrow^*_{R,E}$. When $=_E \circ \to_{R,E} \circ =_E$ is terminating, the following properties are equivalent [20]:

1. (R, E) is Church-Rosser modulo E,
2. for any terms t, t', $t \longleftrightarrow^*_{R\cup E} t'$ if and only if $t{\downarrow} =_E t'{\downarrow}$, where $t{\downarrow}$ (resp., $t'{\downarrow}$) denotes any normal form of t (resp., t') w.r.t (R, E).

A TRS (R, E) is *E-convergent* if $=_E \circ \to_{R,E} \circ =_E$ is terminating and (R, E) is Church-Rosser modulo E. Let Σ_0 be the subsignature of Σ that consists of function symbols occurring in the axioms of E. An E-convergent TRS (R, E) is said to be *E-constructed* if $\Sigma_0 \cap \{l(\epsilon) \mid l \to r \in R\} = \emptyset$.

An E-convergent TRS (R, E) is said to be *subterm E-convergent* if for any $l \to r \in R$, r is either a strict subterm of l or a constant. When (R, E) is clear from the context, a normal form w.r.t (R, E) is said to be *normalized*. A substitution σ is *normalized* if, for every variable x in the domain of σ, $x\sigma$ is normalized. An instance $l\sigma \to r\sigma$ of a rule $l \to r \in R$ is a *right-reduced* instance if $\sigma_{|Var(r)}$ is normalized. A term t is an *innermost redex* if no subterm of t is a redex. An E-convergent TRS (R, E) is *IRR* if every innermost redex is R, E-reducible by a right-reduced instance of a rule in R. An E-convergent TRS (R, E) is *IR1* if every innermost redex is R, E-reducible to a normal form in one step.

To simplify the notation, we often use tuples of terms, say $\bar{u} = (u_1, \ldots, u_n)$, $\bar{v} = (v_1, \ldots, v_n)$. Applying a substitution σ to $\bar{u}$ is the tuple $\bar{u}\sigma = (u_1\sigma, \ldots, u_n\sigma)$. The tuples $\bar{u}$ and $\bar{v}$ are said to be E-equal, denoted by $\bar{u} =_E \bar{v}$, if $u_1 =_E v_1, \ldots, u_n =_E v_n$. Similarly, $\bar{u} \to^*_R \bar{v}$ if $u_1 \to^*_R v_1, \ldots, u_n \to^*_R v_n$, $\bar{u}$ is normalized if $u_1, \ldots, u_n$ are normalized, and $\bar{u} =^? \bar{v}$ is $u_1 =^? v_1 \wedge \cdots \wedge u_n =^? v_n$.

2.3 Forward Closure

In this section, we introduce the notion of finite forward closure, following the definition given in [21]. Consider the rule:

ForwardOverlap $g \to d[l'],\ l \to r \ \vdash\ (g \to d[r])\sigma$
where $g \to d[l'], l \to r \in R$, l' is not a variable and $\sigma \in CSU_E(l' =^? l)$

For this inference rule, the notion of redundancy is defined with respect to an ordering on terms. We assume the existence of a simplification ordering $>$ such that $>$ is E-compatible, meaning that $s' =_E s > t =_E t'$ implies $s' > t'$, and $l > r$ for any $l \to r \in R$. **ForwardOverlap** is said to be *redundant* in (R, E) if for each g' such that $g' =_E g\sigma$, g' is R, E-reducible by a right-reduced instance $s\mu \to t\mu$ of R and either $s\mu < g\sigma$ or ($s\mu =_E g\sigma$ and $t\mu < d[l']\sigma$).

Let $\mathcal{I}$ be an inference system generating rewrite rules and whose inferences are possibly redundant, like for instance $\mathcal{I} = \{\textbf{ForwardOverlap}\}$. Given an equational TRS (R, E), the saturation of (R, E) with respect to $\mathcal{I}$ is inductively defined as follows:

- $S^0_{\mathcal{I}}(R) = R$,
- $S^{k+1}_{\mathcal{I}}(R) = S^k_{\mathcal{I}}(R) \cup \{\rho\}$ where the rule ρ is obtained by applying an inference i in $\mathcal{I}$ using $(S^k_{\mathcal{I}}(R), E)$ as equational TRS and such that i is not redundant in $(S^k_{\mathcal{I}}(R), E)$.

Let $S_{\mathcal{I}}(R) = \bigcup_{k \geq 0} S^k_{\mathcal{I}}(R)$. When $S_{\mathcal{I}}(R)$ is finite, $S_{\mathcal{I}}(R)$ is called a finite $\mathcal{I}$-*saturation* of (R, E). An equational TRS (R, E) is $\mathcal{I}$-*saturated* if $S_{\mathcal{I}}(R) = R$. An equational TRS has a finite *forward closure* if it has a finite $\mathcal{I}$-saturation for $\mathcal{I} = \{\textbf{ForwardOverlap}\}$. An equational TRS is *forward-closed* if it is $\mathcal{I}$-saturated for $\mathcal{I} = \{\textbf{ForwardOverlap}\}$.

Example 1. Any subterm E-convergent TRS has a finite forward closure. Subterm convergent TRSs are often used in the verification of security protocols [1], e.g., $\{dec(enc(x, y), y) \to x\}$ and $\{fst(pair(x, y)) \to x,\ snd(pair(x, y)) \to y\}$. The equational TRSs $\{dec(enc(x, k), k * y) \to x\}$ and $\{rm(x * k, k) \to x\}$ are subterm E-convergent for $E = AC(*) = \{x * (y * z) = (x * y) * z, x * y = y * x\}$.

Forward closure can be connected to the notion of Finite Variant Property (FVP, for short) introduced in [11]. Given an E-convergent TRS (R, E), an (R, E)-variant of a term t is a pair $((t\theta)\downarrow, \theta)$ where θ is a normalized substitution whose domain is included in $Var(t)$. (R, E) has the FVP if for any term t there exists a finite set V of (R, E)-variants of t such that any (R, E)-variant of t is componentwise E-equal to an instance of some element in V. If (R, E) has the FVP, then any $R \cup E$-unification problem G reduces to E-unification problems via the computation of finitely many variants of G (viewed as a term with additional symbols). This computation can be performed using folding variant narrowing [19,12]. In [7], it was shown that for any TRS R, R has the FVP iff it has a finite forward closure. A similar equivalence holds for E-constructed TRSs:

Lemma 1. *Assume (R, E) is any E-constructed TRS and E is any regular and collapse-free equational theory such that E-unification is finitary. Then, (R, E) has a finite forward closure iff (R, E) has the FVP.*

Proof. We rely on some results that have been shown in [21] for an inference system $\mathcal{I}$ including **ForwardOverlap** plus an additional **Parallel** rule whose premises are $s \to t$, $l \to r \in R$, $v = u[l'] \in E$ such that l' is a non-variable strict subterm of u which is E-unifiable with l. The following statements are proved in [21]:

– (R, E) is *IR1* iff (R, E) is $\mathcal{I}$-saturated.
– If (R, E) is *IR1* and E-unification is finitary, then (R, E) has the FVP.
– If (R, E) has the FVP, then (R, E) has a finite $\mathcal{I}$-saturation.

When (R, E) is E-constructed and E is a regular and collapse-free equational theory, **Parallel** does not apply since a Σ_0-rooted term l' is not E-unifiable with a $\Sigma\backslash\Sigma_0$-rooted term l. Thus, $\mathcal{I}$-saturation reduces to forward closure, $\mathcal{I}$-satured means forward-closed, and the above statements can be reworded accordingly. To conclude the proof, notice that if R' is a finite forward closure of (R, E), then (R', E) is forward-closed and both (R', E) and (R, E) have the FVP. □

In this paper, (R, E) is assumed to be E-constructed and so the signature of (R, E) necessarily includes a non-empty set of function symbols that do not occur in the axioms of E. Thus, this means that we actually need general E-unification, i.e., E-unification with free function symbols, instead of E-unification. Fortunately, when E is regular and collapse-free, E-unification is finitary if and only if general E-unification is finitary. This equivalence is a consequence of a classical disjoint combination method for regular and collapse-free theories [31] that allows us to build a general E-unification algorithm as a combination of the syntactic unification algorithm and an E-unification algorithm.

From now on, the equational theory E is always assumed to be regular and collapse-free when (R, E) is E-constructed.

3 Hierarchical Unification

Consider now a union of theories $R \cup E$ where E is regular and collapse-free and (R, E) is assumed to be E-constructed. Thanks to this assumption, R and E are "sufficiently separated" and thus we can envision the problem of building an $R \cup E$-unification algorithm as a combination of two unification procedures: a mutation-based unification procedure processing some $R \cup E$-equalities, and an E-unification algorithm. The approach we will use for this problem is the *hierarchical* approach. Informally, the approach works as follows:

– The set of equations is processed to separate the terms over the shared signature, Σ_0, from terms over the non-shared one, $\Sigma\backslash\Sigma_0$.
– The mutation-based procedure is then used to simplify the $\Sigma\backslash\Sigma_0$-equations.
– The remaining equations over the shared signature Σ_0 are solved using the E-unification algorithm.
– The process can repeat. If the process terminates in a solved form then the problem is solvable and a unifier is produced.

A hierarchical unification procedure is parameterized by an E-unification algorithm and a mutation-based reduction procedure U. It applies some additional rules given in Figure 1: **Coalesce**, **Split**, **Flatten**, and **VA** are used to separate the terms, U is used to simplify the $\Sigma\backslash\Sigma_0$-equations, and finally, **Solve** calls the E-unification algorithm.

Coalesce $\{x = y\} \cup G \vdash \{x = y\} \cup (G\{x \mapsto y\})$
where x and y are distinct variables occurring both in G.

Split $\{f(\bar{v}) = t\} \cup G \vdash \{x = f(\bar{v}), x = t\} \cup G$
where $f \in \Sigma \backslash \Sigma_0$, t is a non-variable term and x is a fresh variable.

Flatten $\{v = f(\dots, u, \dots)\} \cup G \vdash \{v = f(\dots, x, \dots), x = u\} \cup G$
where $f \in \Sigma \backslash \Sigma_0$, v is a variable, u is a non-variable term, and x is a fresh variable.

VA $\{s = t[u]\} \cup G \vdash \{s = t[x], x = u\} \cup G$
where t is Σ_0-rooted, u is an alien subterm of t, and x is a fresh variable.

Solve $G \cup G_0 \vdash \bigvee_{\sigma_0 \in CSU_E(G_0)} G \cup \hat{\sigma}_0$
where G is a set of $\Sigma \backslash \Sigma_0$-equations, G_0 is a set of Σ_0-equations, G_0 is E-unifiable and not in tree solved form, $\hat{\sigma}_0$ is the tree solved form associated with σ_0, and w.l.o.g for any $x \in Dom(\sigma_0)$, $x\sigma_0 \in Var(G_0)$ if $x\sigma_0$ is a variable.

Fig. 1. H_E rules

Definition 1 (Hierarchical unification procedure). *Assume a Σ_0-theory E for which an E-unification algorithm is known, a Σ-theory $F \cup E$ for which E-unification is complete for solving the Σ_0-fragment of $F \cup E$-unification, and an inference system U satisfying the following assumptions: U transforms only non-variable flat $\Sigma \backslash \Sigma_0$-equations; U is sound and complete; and U is parameterized by some finite set S of $F \cup E$-equalities such that the soundness of each inference $\vdash_U$ follows from at most one equality in S. Under these assumptions, the $H_E(U)$ inference system is defined as the repeated application of some inference from H_E (cf. Figure 1) or U, using the following order of priority: **Coalesce**, **Split**, **Flatten**, **VA**, U, **Solve**. An $F \cup E$-unification problem is in* separate form *if it is a normal form with respect to $H_E \backslash \{$**Solve**$\}$.*

Note, that when we speak of an inference system, U, this is not just a set of rules but also a strategy for applying those rules. This could include, as in the $\mathcal{E}_{AC}$ case of Proposition 3, methods for detecting errors such as occur-checks and non-termination [15].

Proposition 1. *Let (R, E) be any E-constructed TRS such that an inference system U following Definition 1 is known for the equational theory $R \cup E$, in addition to an existing E-unification algorithm. Then E, $R \cup E$ and U satisfy the assumptions of Definition 1, and the $H_E(U)$ inference system provides a sound and complete $R \cup E$-unification procedure if the normal forms w.r.t $H_E(U)$ are either the dag solved forms or problems that are not $R \cup E$-unifiable. If $H_E(U)$ is terminating, then it is an $R \cup E$-unification algorithm.*

Proof. If (R, E) is E-constructed, then E-unification is complete for solving the Σ_0-fragment of $R \cup E$-unification, and so all the assumptions of Definition 1 are satisfied. By construction, $H_E(U)$ is sound and complete. Since the $R \cup E$-unifiable normal forms w.r.t $H_E(U)$ are assumed to be the dag solved forms, collecting all the dag solved forms reached by $H_E(U)$ suffices to get a complete set of $R \cup E$-unifiers. □

3.1 Subterm Collapse-Free Theories

Hierarchical unification algorithms are known for particular subterm collapse-free theories of particular interest for protocol analysis.

Proposition 2. *([30,15]) Let E be the empty Σ_0-theory where Σ_0 only consists of a binary function symbol $*$, $R_{\mathcal{D}} = \{h(x * y) \to h(x) * h(y)\}$ and $R_{\mathcal{D}1} = \{f(x * y, z) \to f(x, z) * f(y, z)\}$. The equational TRSs $(R_{\mathcal{D}}, E)$ and $(R_{\mathcal{D}1}, E)$ are E-constructed. Moreover, $R_{\mathcal{D}} \cup E$ (resp., $R_{\mathcal{D}1} \cup E$) is a subterm collapse-free theory admitting a unification algorithm of the form $H_E(U_{\mathcal{D}})$ (resp., $H_E(U_{\mathcal{D}1})$).*

Proof. Subterm collapse-freeness follows from the fact that both theories are *non-size-reducing*. The inference system $U_{\mathcal{D}1}$ can be derived following the approach developed in [15] and based on the one initiated in [30] for one-side distributivity. The same approach can be applied for $R_{\mathcal{D}}$ to get $U_{\mathcal{D}}$. □

Proposition 3. *([15]) Let $AC = AC(\circledast)$, $R_{\mathcal{E}} = \{exp(exp(x, y), z) \to exp(x, y \circledast z), exp(x * y, z) \to exp(x, z) * exp(y, z)\}$ and $R_{\mathcal{F}} = \{enc(enc(x, y), z) \to enc(x, y \circledast z)\}$. The equational TRSs $(R_{\mathcal{E}}, AC)$ and $(R_{\mathcal{F}}, AC)$ are AC-constructed. Moreover, $\mathcal{E}_{AC} = R_{\mathcal{E}} \cup AC$ (resp., $\mathcal{F}_{AC} = R_{\mathcal{F}} \cup AC$) is a subterm collapse-free theory admitting a unification algorithm of the form $H_{AC}(U_{\mathcal{E}})$ (resp., $H_{AC}(U_{\mathcal{F}})$).*

Proof. In [15] it is shown that both $\mathcal{E}_{AC}$ and $\mathcal{F}_{AC}$ are subterm collapse-free theories. Also in [15] a mutation-based inference system, say $U_{\mathcal{E}}$ (resp., $U_{\mathcal{F}}$), is developed for $\mathcal{E}_{AC}$ (resp., $\mathcal{F}_{AC}$): it reduces the $\Sigma \backslash \Sigma_0$-equations into solved forms after which a solving step applies AC-unification on Σ_0-equations. It is shown in [15] that the solving step needs only be applied once. Hence, the $\mathcal{E}_{AC}$-unification algorithm (resp., $\mathcal{F}_{AC}$-unification algorithm) given in [15] provides a unification algorithm of the form $H_{AC}(U_{\mathcal{E}})$ (resp., $H_{AC}(U_{\mathcal{F}})$). □

3.2 Forward-Closed *E*-Constructed TRSs

For any forward-closed E-constructed TRS (R, E) such that E is regular and collapse-free, an $R \cup E$-unification algorithm of the form $H_E(U)$ can be obtained by defining some inference system U, based on the *Basic Syntactic Mutation* approach initiated for the class of theories closed by paramodulation [23], and already applied in [13] to a particular class of forward-closed equational TRSs.

Let BSM_R be the inference system given in Figure 2. One can notice that each inference rule in BSM_R generates some boxed terms. This particular annotation of terms, detailed in [23,13], allows us to control the rules application, disregarding needless inferences on boxed terms, in such a way that BSM_R is terminating.

An $R \cup E$-unification algorithm combining BSM_R and an E-unification algorithm has been developed in [13] for the case of any forward-closed convergent TRS R such that the left-hand sides of R are linear and contain no symbols of E. In this paper, we extend [13] to any forward-closed E-constructed TRS (R, E), without any further restriction on R.

The soundness and completeness of BSM_R is shown by the following lemma.

Imit $\bigcup_i\{x = f(\bar{v}_i)\} \cup G \vdash \{x = \boxed{f(\bar{y})}\} \cup \bigcup_i\{\bar{y} = \bar{v}_i\} \cup G$
where $f \in \Sigma\backslash\Sigma_0$, $i > 1$, $\bar{y}$ are fresh variables and there are no more equations $x = f(\dots)$ in G.

MutConflict$_R$ $\{x = f(\bar{v})\} \cup G \vdash \{x = \boxed{t}, \boxed{\bar{s}} = \bar{v}\} \cup G$
where $f \in \Sigma\backslash\Sigma_0$, $f(\bar{s}) \to t$ is a fresh instance of a rule in R, $f(\bar{v})$ is unboxed, and there is another equation $x = u$ in G with a non-variable term u or $x = f(\bar{v})$ occurs in a cycle.

ImitCycle $\{x = f(\bar{v})\} \cup G \vdash \{x = \boxed{f(\bar{y})}, \bar{y} = \bar{v}\} \cup G$
where $f \in \Sigma\backslash\Sigma_0$, $f(\bar{v})$ is unboxed, $\bar{y}$ are fresh variables and $x = f(\bar{v})$ occurs in a cycle.

Fig. 2. *BSM*$_R$ rules

Lemma 2. *Let (R, E) be any forward-closed E-constructed TRS over the signature Σ. For each equality $u =_{R\cup E} v$ such that u is $\Sigma\backslash\Sigma_0$-rooted and v is normalized, one of the following is true:*

1. *$u = f(\bar{u})$, $v = f(\bar{v})$ and $\bar{u} =_{R\cup E} \bar{v}$.*
2. *$u = f(\bar{u})$, there exist $f(\bar{s}) \to t \in R$ and a normalized substitution σ such that $\bar{u} =_{R\cup E} \bar{s}\sigma$, $v =_E t\sigma$ and $\bar{s}\sigma, t\sigma$ are normalized.*

Proof. Let us analyze the possible rewrite proofs $\to^*_{R,E}$ of $u =_{R\cup E} v$.

First, if there is no step at the root position, then we get $u = f(\bar{u}) \to^*_{R,E} f(\bar{u'}) =_E v$ where $\bar{u} \to^*_{R,E} \bar{u'}$ and $\bar{u'}$ are normalized. Since f is a free symbol for E, we have that $v = f(\bar{v})$ and $\bar{u'} =_E \bar{v}$. Hence, $\bar{u} =_{R\cup E} \bar{v}$ since $\bar{u} =_{R\cup E} \bar{u'}$.

Second, if there is one step at the root position, then we have

$$u = f(\bar{u}) \to^*_{R,E} f(\bar{u'}) = f(\bar{s})\sigma \to_{R,E,\epsilon} t\sigma =_E v$$

where $f(\bar{s}) \to t \in R$, $\bar{u} \to^*_{R,E} \bar{u'}$, $\bar{u'}$ are normalized, $\bar{u'} =_E \bar{s}\sigma$, and so $\sigma, \bar{s}\sigma$ are normalized. Since $t\sigma =_E v$ and v is normalized, $t\sigma$ is also normalized. □

A unification procedure of the form $H_E(BSM_R)$ corresponds to the *BSC* unification procedure given in [13] except that **Solve** is applied in *BSC* before *BSM*$_R$ rules. However, the termination proof stated for *BSC* in [13] also holds when **Solve** is applied after the *BSM*$_R$ rules.

Lemma 3. *Assume E is any regular and collapse-free theory such that an E-unification algorithm is known. Let (R, E) be a forward-closed E-constructed TRS and BSM$_R$ the inference system given in Fig. 2. Then $H_E(BSM_R)$ is an $R \cup E$-unification algorithm.*

Example 2. Consider $R = \{h(x) \to a \times x\}$, $R' = \{f(x, y) \to a'(y) \times x\}$ and $E = \{x\times(y*z) = (x\times y)*(x\times z)\}$. The theory E corresponds to left-distributivity and an E-unification algorithm is given in [30]. Since (R, E) and (R', E) are forward-closed and E-constructed, $H_E(BSM_R)$ and $H_E(BSM_{R'})$ are unification algorithms for $R\cup E$ and $R'\cup E$, respectively. Notice that $h(x*y) =_{R\cup E} h(x)*h(y)$ and $f(x * y, z) =_{R'\cup E} f(x, z) * f(y, z)$.

Example 3. Consider $R = \{\pi_1(x.y) \to x, \pi_2(x.y) \to y, dec(enc(x,y),y) \to x\}$ and $E = \{enc(x.y,z) = enc(x,z).enc(y,z)\}$. An E-unification algorithm can be obtained following the approach developed in [30,15] and can be used in a hierarchical unification procedure of the form $H_E(BSM_R)$. Since (R,E) is forward-closed and E-constructed, $H_E(BSM_R)$ is an $R \cup E$-unification algorithm.

4 Combined Hierarchical Unification

We are now interested in combining hierarchical unification algorithms known for E-constructed TRSs. Given two E-constructed TRSs, say (R_1,E) and (R_2,E), the problem is to study the possible construction of a (combined) hierarchical unification algorithm for $(R_1 \cup R_2, E)$ using the two hierarchical unification algorithms known for (R_1,E) and (R_2,E). We investigate this combination problem for the two classes of E-constructed TRSs introduced in Section 3. First, we consider a class of E-constructed TRSs (R,E) such that $R \cup E$ is subterm collapse-free. Second, we study the class of forward-closed E-constructed TRSs (R,E) such that E is regular and collapse-free.

4.1 Combining Subterm Collapse-Free Theories

Let us first consider a technical lemma which is useful to get a hierarchical unification procedure.

Lemma 4. *Let (R_1,E) and (R_2,E) be two E-constructed TRSs over the signatures Σ_1 and Σ_2, respectively, such that $\Sigma_1 \cap \Sigma_2 = \Sigma_0$ for the signature Σ_0 of E, and for $i = 1,2$, $R_i \cup E$ admits a sound and complete unification procedure of the form $H_E(U_i)$. Assume that $R_1 \cup R_2 \cup E$ is subterm collapse-free, and for any $\Sigma_1 \backslash \Sigma_0$-rooted term t_1 and any $\Sigma_2 \backslash \Sigma_0$-rooted term t_2, t_1 cannot be equal to t_2 modulo $R_1 \cup R_2 \cup E$. Then, $H_E(U_1 \cup U_2)$ is a sound and complete $R_1 \cup R_2 \cup E$-unification procedure.*

Proof. According to the assumptions, any normal form w.r.t $H_E(U_1 \cup U_2)$ is $R_1 \cup R_2 \cup E$-unifiable iff it is in dag solved form. So, Proposition 1 applies. □

We study below a possible way to satisfy the assumptions of Lemma 4.

Definition 2 (Layer-preservingness). *Let (R,E) be an E-constructed TRS over the signature Σ, for which Σ_0 denotes the signature of E. A Σ-term t is said to be* E-capped *if there exist a constant-free Σ_0-term u and a substitution σ such that $t = u\sigma$, $Dom(\sigma) = Var(u)$ and $Ran(\sigma)$ is a set of $\Sigma \backslash \Sigma_0$-rooted terms. The TRS (R,E) is said to be* layer-preserving *if $R \cup E$ is subterm collapse-free and any normal form of any $\Sigma \backslash \Sigma_0$-rooted term is E-capped.*

Remark 1. An easy way to get layer-preservingness of (R,E) is to assume that $R \cup E$ is subterm collapse-free and the right hand-sides of rules in R are $\Sigma \backslash \Sigma_0$-rooted. In that case the term u in Definition 2 is simply a variable. Layer-preservingness generalizes this assumption used in [14].

The property of being E-constructed and layer-preserving is modular.

Lemma 5. *Assume E is a subterm collapse-free Σ_0-theory, for $i = 1, 2$, (R_i, E) is an E-constructed layer-preserving TRS over the signature Σ_i, and $\Sigma_1 \cap \Sigma_2 = \Sigma_0$. If $=_E \circ \to_{R_1 \cup R_2} \circ =_E$ is terminating, then $(R_1 \cup R_2, E)$ is an E-constructed layer-preserving TRS, and for any $\Sigma_1 \backslash \Sigma_0$-rooted term t_1 and any $\Sigma_2 \backslash \Sigma_0$-rooted term t_2, t_1 cannot be equal to t_2 modulo $R_1 \cup R_2 \cup E$.*

Proof. To show that $(R_1 \cup R_2, E)$ is layer-preserving, we have to prove that $R_1 \cup R_2 \cup E$ remains subterm collapse-free. The modularity of subterm collapse-freeness has been shown in [14] when the right-hand sides of R_i are $\Sigma_i \backslash \Sigma_0$-rooted, for $i = 1, 2$. Actually, a similar proof by contradiction can be performed in the case (R_i, E) is layer-preserving, for $i = 1, 2$. Let us consider the *height of layers* of a term t, inductively defined as follows:

- $ht(t) = 0$ if t is a variable,
- $ht(t) = 1$ if t is a non-variable pure term,
- $ht(t) = 1 + \max\{ht(u) \mid u \text{ is an alien subterm of } t\}$ if t is not pure.

Assume there exists a term t and a non-empty position p such that $t =_{R_1 \cup E_2 \cup E} t_{|p}$. If the path from ϵ to p contains only symbols from one theory, say $R_i \cup E$, this would lead to a contradiction with the subterm collapse-freeness of $R_i \cup E$. Consider now that the path from ϵ to p contains both a $\Sigma_1 \backslash \Sigma_0$-symbol and a $\Sigma_2 \backslash \Sigma_0$-symbol. Let $u = t|_p$ and let t' and u' be the respective normal forms of t and u w.r.t $(R_1 \cup R_2, E)$. Since $t' =_E u'$ and E is necessarily regular collapse-free, we have that t' and u' have the same height of layers. By the layer-preserving assumption, t and t' have the same height of layers, as well as u and u'. Thus t and u have the the same height of layers, which leads to a contradiction due to the considered path from ϵ to p.

Assume there exist some $\Sigma_1 \backslash \Sigma_0$-rooted term t_1 and some $\Sigma_2 \backslash \Sigma_0$-rooted term t_2 such that $t_1 =_{R_1 \cup R_2 \cup E} t_2$. Then, $t'_1 =_E t'_2$ where t'_1 and t'_2 are the respective normal forms of t_1 and t_2 w.r.t $(R_1 \cup R_2, E)$. The layer-preserving assumption implies that t'_i must still contain a symbol in $\Sigma_i \backslash \Sigma_0$ for $i = 1, 2$. Since E is necessarily regular and collapse-free, it is thus impossible to have $t'_1 =_E t'_2$. □

Remark 2. To satisfy the condition $=_E \circ \to_{R_1 \cup R_2} \circ =_E$ is terminating, it suffices to exhibit an E-compatible reduction ordering $>$ such that $l > r$ for any $l \to r \in R_1 \cup R_2$. In that case, $>$ is defined on terms built over $\Sigma_1 \cup \Sigma_2$.

By Lemma 5, the two assumptions of Lemma 4 can be satisfied, and this leads to a hierarchical unification procedure for the combined TRS. In the following, we consider a notion of decreasingness in order to study the termination of this unification procedure.

Definition 3 (Decreasingness). *Consider a complexity measure defined as a mapping C from separate forms to natural numbers. An $H_E(U)$ inference system is said to be C-*decreasing *if for any separate form $G \cup G_0$ we have that (1) for any G' such that $G \cup G_0 \vdash_U G' \cup G_0$, the separate form of $G' \cup G_0$ does not increase C; (2) for any G'_0 such that $G \cup G_0 \vdash_{\textbf{\textit{Solve}}} G \cup G'_0$, then either the separate form of $G \cup G'_0$ is in normal form w.r.t $H_E(U)$, or it decreases C.*

Consequently, $H_E(U)$ is terminating if there exists some C such that $H_E(U)$ is C-decreasing.

Theorem 1. *Assume E is a subterm collapse-free theory such that an E-unification algorithm is known, and C is a complexity measure defined on separate forms. Let (R_1, E) and (R_2, E) be two E-constructed TRSs sharing only symbols in E such that, for $i = 1, 2$, (R_i, E) is layer-preserving, and $R_i \cup E$ admits a C-decreasing unification algorithm of the form $H_E(U_i)$. If $=_E \circ \to_{R_1 \cup R_2} \circ =_E$ is terminating, then $(R_1 \cup R_2, E)$ is an E-constructed TRS such that $(R_1 \cup R_2, E)$ is layer-preserving, and $R_1 \cup R_2 \cup E$ admits a C-decreasing unification algorithm of the form $H_E(U_1 \cup U_2)$.*

Proof. $(R_1 \cup R_2, E)$ is layer-preserving by Lemma 5. In addition, a $\Sigma_1 \backslash \Sigma_0$-rooted term cannot be equal to a $\Sigma_2 \backslash \Sigma_0$-rooted term modulo $R_1 \cup R_2 \cup E$. Applying Lemma 4, $H_E(U_1 \cup U_2)$ provides a sound and complete $R_1 \cup R_2 \cup E$-unification procedure. Moreover, $H_E(U_1 \cup U_2)$ is C-decreasing and so it is terminating. □

Example 4. Consider the theories $\mathcal{E}_{AC}$ and $\mathcal{F}_{AC}$ introduced in Proposition 3 and the corresponding hierarchical unification algorithms $H_{AC}(U_{\mathcal{E}})$ and $H_{AC}(U_{\mathcal{F}})$ where the mutation rules defining $U_{\mathcal{E}}$ and $U_{\mathcal{F}}$ can be found in [15]. Let SVC be the complexity measure defined as follows: given an $R \cup E$-unification problem in separate form $G \cup G_0$, $SVC(G \cup G_0)$ is the number of equivalence classes of variables shared by G and G_0 that are variables abstracting $\Sigma \backslash \Sigma_0$-rooted terms.

Let us now check that the unification algorithms $H_{AC}(U_{\mathcal{E}})$ and $H_{AC}(U_{\mathcal{F}})$ are both SVC-decreasing. On the one hand, it is routine to verify that any (mutation) rule in $U_{\mathcal{E}}$ (resp., $U_{\mathcal{F}}$) does not lead, via a further possible application of **VA**, to new shared variables which are abstracting $\Sigma \backslash \Sigma_0$-rooted terms. Hence, the rules in $U_{\mathcal{E}}$ (resp., $U_{\mathcal{F}}$) cannot increase SVC. On the other hand, **Solve** leads to either a normal form w.r.t $H_{AC}(U_{\mathcal{E}})$ (resp., $H_{AC}(U_{\mathcal{F}})$), or it generates some equality $x =^? y$ between variables x and y for which there are $\Sigma \backslash \Sigma_0$-equations $x =^? s$ and $y =^? t$ in G. In the last case, the respective equivalence classes of x and y are merged into a single one by applying **Solve** and so, **Solve** strictly decreases SVC. By Theorem 1, we get that $\mathcal{E}_{AC} \cup \mathcal{F}_{AC}$ admits a SVC-decreasing unification algorithm of the form $H_{AC}(U_{\mathcal{E}} \cup U_{\mathcal{F}})$. Notice this means that we can use the termination strategy used in the individual $H_{AC}(U_{\mathcal{E}})$ and $H_{AC}(U_{\mathcal{F}})$ algorithms to obtain a termination strategy for the hierarchical combined algorithm, $H_{AC}(U_{\mathcal{E}} \cup U_{\mathcal{F}})$. We suspect that this complexity measure, SVC, could be useful for proving termination in other theories.

To conclude this section, let us mention the problem of combining two copies of the same E-constructed TRS, provided that only the symbols in E are possibly shared. In that very particular case, subterm collapse-freeness is sufficient and there is no need to find a decreasing complexity measure.

Theorem 2. *Consider (R, E) is an E-constructed TRS over the signature Σ. Assume $R \cup E$ is subterm collapse-free and $R \cup E$ admits a unification algorithm of the form $H_E(U)$. Let (R', E) be a copy of (R, E) obtained by renaming the*

$\Sigma\backslash\Sigma_0$-*symbols. Then,* $(R\cup R', E)$ *is an* E*-constructed TRS such that* $R\cup R'\cup E$ *is subterm collapse-free, and* $R\cup R'\cup E$ *admits a unification algorithm of the form* $H_E(U\cup U')$*, where* U' *is obtained from* U *by applying the same renaming as the one defining* (R', E)*.*

Proof. If $R\cup R'\cup E$ is not subterm collapse-free, then this would imply that $R\cup E$ is not subterm collapse-free, leading to a contradiction with the assumption. By construction, $H_E(U\cup U')$ is sound and complete. For each inference $P \vdash_{H_E(U\cup U')} Q$, there exists an inference $\iota(P) \vdash_{H_E(U)} \iota(Q)$ where ι denotes the morphism replacing each symbol $f' \in \Sigma'\backslash\Sigma_0$ by the corresponding function symbol $f \in \Sigma\backslash\Sigma_0$. Thus, the termination w.r.t $H_E(U)$ implies the termination w.r.t $H_E(U\cup U')$. □

Example 5. Consider the E-constructed TRSs $(R_{\mathcal{D}}, E)$ and $(R_{\mathcal{D}1}, E)$ defined in Proposition 2, and their copies $R'_{\mathcal{D}} = \{h'(x * y) \to h'(x) * h'(y)\}$ and $R'_{\mathcal{D}1} = \{f'(x * y, z) \to f'(x, z) * f'(y, z)\}$. The theories $R_{\mathcal{D}} \cup E$ and $R_{\mathcal{D}1} \cup E$ are subterm collapse-free and admit unification algorithms of the form $H_E(U_{\mathcal{D}})$ and $H_E(U_{\mathcal{D}1})$, respectively. By Theorem 2, $R_{\mathcal{D}} \cup R'_{\mathcal{D}} \cup E$ and $R_{\mathcal{D}1} \cup R'_{\mathcal{D}1} \cup E$ are subterm collapse-free and admit unification algorithms of the form $H_E(U_{\mathcal{D}}\cup U'_{\mathcal{D}})$ and $H_E(U_{\mathcal{D}1} \cup U'_{\mathcal{D}1})$, respectively.

4.2 Combining Forward-Closed *E*-Constructed TRSs

The union of two forward-closed E-constructed TRSs remains a forward-closed E constructed TRS. Thus, a hierarchical unification algorithm can be constructed in a modular way in unions of forward-closed E-constructed TRSs.

Theorem 3. *Assume* E *is a regular and collapse-free theory such that an* E*-unification algorithm is known. Let* (R_1, E) *and* (R_2, E) *be two forward-closed* E*-constructed TRSs sharing only symbols in* E*. Then* $R_1 \cup R_2 \cup E$ *admits a unification algorithm of the form* $H_E(BSM_{R_1} \cup BSM_{R_2})$*.*

Proof. $(R_1\cup R_2, E)$ is a forward-closed E-constructed TRS, and so by Lemma 3, $R_1 \cup R_2 \cup E$ admits a unification algorithm of the form $H_E(BSM_{R_1\cup R_2})$, which coincides with $H_E(BSM_{R_1} \cup BSM_{R_2})$. □

In the following, we investigate the case where E already admits a hierarchical unification algorithm of the form $H_{E'}(U')$ for a subtheory E' of E, like in Example 3 where E has a hierarchical unification algorithm of the form $H_{E'}(U')$ for $E' = \emptyset$. In that case, we can consider the following compositionality lemma:

Lemma 6. *Let* (R, E) *be an* E*-constructed TRS such that* $R\cup E$ *admits a unification algorithm of the form* $H_E(U)$*, and* E *admits a unification algorithm of the form* $H_{E'}(U')$*, where* E' *is a subtheory of* E*. Then* $R \cup E$ *also admits a unification algorithm of the form* $H_{E'}(U \cup U')$*.*

Proof. Consider $\Sigma' = \Sigma_0$ and E is a Σ'-theory of the form $E = F' \cup E'$. Assume $R \cup E$ (resp., $F' \cup E'$) has a unification algorithm of the form $H_E(U)$ (resp., $H_{E'}(U')$), where U (resp., U') is sound, complete, and parameterized by some finite set S (resp., S') of $R \cup E$-equalities (resp., $F' \cup E'$-equalities) such that the soundness of each inference $\vdash_U$ (resp., $\vdash_{U'}$) follows from at most one equality in S (resp., S').

Since E-unification is complete for solving the Σ'-fragment of $R \cup E$-unification, U' is also sound and complete for $R \cup F' \cup E'$. Hence, the inference system $U \cup U'$ is sound and complete. Moreover, $S \cup S'$ is a finite set of $R \cup F' \cup E'$-equalities such that the soundness of each inference $\vdash_{U \cup U'}$ follows from at most one equality in $S \cup S'$.

Since E-unification is complete for solving the Σ'-fragment of $R \cup E$-unification and E'-unification is complete for solving the Σ'_0-fragment of E-unification, we have that E' is also complete for solving the Σ'_0-fragment of $R \cup F' \cup E'$-unification.

Consequently, E', $R \cup F' \cup E'$ and $U \cup U'$ satisfy all the assumptions of Definition 1, and so $H_{E'}(U \cup U')$ is well-defined. Since $H_{E'}(U \cup U')$ corresponds to an "unfolding" of $H_E(U)$, it is terminating, sound and complete, just like $H_E(U)$. Thus, $H_{E'}(U \cup U')$ is a unification algorithm for $R \cup E = R \cup F' \cup E'$. □

Example 6. (Example 3 continued) $R \cup E$ admits a unification algorithm of the form $H_\emptyset(BSM_R \cup U')$ where $H_\emptyset(U')$ is a hierarchical E-unification algorithm.

Example 7. Let us consider a theory used in practice to model a group messaging protocol [9]. For this protocol, the theory modeling the intruder can be defined [27] as a combination $R_{ENC} \cup K$ where $K = \{keyexch(x, pk(x'), y, pk(y')) = keyexch(x', pk(x), y', pk(y))\}$ and (R_{ENC}, K) is the forward-closed K-constructed TRS where

$$R_{ENC} = \left\{ \begin{array}{r} adec(aenc(m, pk(sk)), sk) \to m \\ getmsg(sign(m, sk)) \to m \\ checksign(sign(m, sk), m, pk(sk)) \to ok \\ sdec(senc(m, k), k) \to m \end{array} \right\}$$

K is a theory closed by paramodulation and so K-unification is finitary [23]. By Lemma 3, $R_{ENC} \cup K$ has a hierarchical unification algorithm of the form $H_K(BSM_{R_{ENC}})$. The mutation-based unification algorithm known for theories closed by paramodulation [23] can be reworded as a hierarchical unification algorithm, of the form $H_\emptyset(U_K)$ for K. By Lemma 6, $H_\emptyset(BSM_{R_{ENC}} \cup U_K)$ is another $R_{ENC} \cup K$-unification algorithm.

Applying Lemma 6, we can easily obtain a hierarchical unification algorithm for a forward-closed E-constructed TRS combined with a regular and collapse-free E-constructed TRS.

Lemma 7. *Assume E is a regular and collapse-free theory such that an E-unification algorithm is known. Let (R_1, E) and (R_2, E) be two E-constructed TRSs sharing only symbols in E such that (R_1, E) is forward-closed, and $R_2 \cup E$*

is a regular and collapse-free theory E_2 admitting a unification algorithm of the form $H_E(U_2)$. Then (R_1, E_2) is a forward-closed E_2-constructed TRS and $R_1 \cup E_2$ admits a unification algorithm of the form $H_E(BSM_{R_1} \cup U_2)$.

Proof. (R_1, E_2) is forward-closed because (R_1, E) is forward-closed and the equational theory $=_E$ coincides with $=_{E_2}$ on Σ_1-terms. By Lemma 3, $R_1 \cup E_2$ admits a unification algorithm of the form $H_{E_2}(BSM_{R_1})$. According to Lemma 6, $R_1 \cup E_2$ also admits a unification algorithm of the form $H_E(BSM_{R_1} \cup U_2)$. □

Example 8. Let $(R, AC(\circledast))$ be a forward-closed $AC(\circledast)$-constructed TRS such that $\circledast$ is the only function symbol shared by $R \cup AC(\circledast)$ and $\mathcal{E}_{AC}$ (resp., $\mathcal{F}_{AC}$). By Lemma 7, $R \cup \mathcal{E}_{AC}$ (resp., $R \cup \mathcal{F}_{AC}$) admits a unification algorithm of the form $H_{AC}(BSM_R \cup U_{\mathcal{E}})$ (resp., $H_{AC}(BSM_R \cup U_{\mathcal{F}})$). According to Example 4, $\mathcal{E}_{AC} \cup \mathcal{F}_{AC}$ admits a unification algorithm of the form $H_{AC}(U_{\mathcal{E}} \cup U_{\mathcal{F}})$. Then, by Lemma 7, $R \cup \mathcal{E}_{AC} \cup \mathcal{F}_{AC}$ admits a unification algorithm of the form $H_{AC}(BSM_R \cup U_{\mathcal{E}} \cup U_{\mathcal{F}})$.

5 Conclusion

We have introduced a hierarchical unification framework as a generic tool to construct unification procedures for (combined) equational theories defined by E-constructed TRSs. We have presented new combination results for the simplest case of subterm collapse-free theories, and a natural follow-up would be to study the case of regular and collapse-free theories. A challenging future work is to investigate the general case of arbitrary theories.

Hierarchical unification allows us to handle syntactic theories $R \cup E$ while the E-unification algorithm can be arbitrary. According to this observation, we plan to study a weakening of syntacticness, in order to allow theories $R \cup E$ that are just *syntactic modulo* E.

We have also begun the implementation of the above hierarchical combination procedure. To begin with, we are using $E = AC$ as the background theory. However, we will explore expanding this to additional equational theories. In the short term, we plan to experiment the use of our variant-free hierarchical unification procedures (e.g., the ones introduced in Examples 3 and 7) as an alternative to variant-based unification procedures in modern protocol verification tools [6,18,25]. In the long term, we want to promote the use of non-disjoint combination procedures [16] and mutation-based procedures [17] in protocol verification tools, targeting unification problems as well as some decision problems related to the knowledge of an intruder, such as intruder deduction (a reachability problem) and indistinguishability (an equivalence problem) [1,8]. The goal is to improve automation of verification methods when theories share for instance AC symbols.

References

1. M. Abadi and V. Cortier. Deciding knowledge in security protocols under equational theories. *Theor. Comput. Sci.*, 367(1-2):2–32, 2006.

2. F. Baader and T. Nipkow. *Term rewriting and all that.* Cambridge University Press, New York, NY, USA, 1998.
3. F. Baader and K. U. Schulz. Unification in the union of disjoint equational theories: Combining decision procedures. *Journal of Symbolic Computation*, 21(2):211 – 243, 1996.
4. F. Baader and W. Snyder. Unification theory. In J. A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, pages 445–532. Elsevier and MIT Press, 2001.
5. D. A. Basin, S. Mödersheim, and L. Viganò. An on-the-fly model-checker for security protocol analysis. In E. Snekkenes and D. Gollmann, editors, *Computer Security - ESORICS 2003, 8th European Symposium on Research in Computer Security, Gjøvik, Norway, October 13-15, 2003, Proceedings*, volume 2808 of *Lecture Notes in Computer Science*, pages 253–270. Springer, 2003.
6. B. Blanchet. Modeling and verifying security protocols with the Applied Pi calculus and proverif. *Foundations and Trends in Privacy and Security*, 1(1-2):1–135, 2016.
7. C. Bouchard, K. A. Gero, C. Lynch, and P. Narendran. On forward closure and the finite variant property. In P. Fontaine, C. Ringeissen, and R. A. Schmidt, editors, *Frontiers of Combining Systems - 9th International Symposium, FroCoS 2013, Nancy, France, September 18-20, 2013. Proceedings*, volume 8152 of *Lecture Notes in Computer Science*, pages 327–342. Springer, 2013.
8. S. Ciobâcă, S. Delaune, and S. Kremer. Computing knowledge in security protocols under convergent equational theories. *J. Autom. Reasoning*, 48(2):219–262, 2012.
9. K. Cohn-Gordon, C. Cremers, L. Garratt, J. Millican, and K. Milner. On ends-to-ends encryption: Asynchronous group messaging with strong security guarantees. In D. Lie, M. Mannan, M. Backes, and X. Wang, editors, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 1802–1819. ACM, 2018.
10. H. Comon, M. Haberstrau, and J. Jouannaud. Syntacticness, cycle-syntacticness, and shallow theories. *Inf. Comput.*, 111(1):154–191, 1994.
11. H. Comon-Lundh and S. Delaune. The finite variant property: How to get rid of some algebraic properties. In J. Giesl, editor, *Rewriting Techniques and Applications*, volume 3467 of *Lecture Notes in Computer Science*, pages 294–307. Springer, 2005.
12. F. Durán, S. Eker, S. Escobar, N. Martí-Oliet, J. Meseguer, and C. L. Talcott. Built-in variant generation and unification, and their applications in Maude 2.7. In N. Olivetti and A. Tiwari, editors, *Automated Reasoning - 8th International Joint Conference, IJCAR 2016, Coimbra, Portugal, June 27 - July 2, 2016, Proceedings*, volume 9706 of *Lecture Notes in Computer Science*, pages 183–192. Springer, 2016.
13. A. K. Eeralla, S. Erbatur, A. M. Marshall, and C. Ringeissen. Rule-based unification in combined theories and the finite variant property. In C. Martín-Vide, A. Okhotin, and D. Shapira, editors, *Language and Automata Theory and Applications - 13th International Conference, LATA 2019, St. Petersburg, Russia, March 26-29, 2019, Proceedings*, volume 11417 of *Lecture Notes in Computer Science*, pages 356–367. Springer, 2019.
14. S. Erbatur, D. Kapur, A. M. Marshall, P. Narendran, and C. Ringeissen. Hierarchical combination. In M. P. Bonacina, editor, *Automated Deduction - CADE-24 - 24th International Conference on Automated Deduction, Lake Placid, NY, USA, June 9-14, 2013. Proceedings*, volume 7898 of *Lecture Notes in Computer Science*, pages 249–266. Springer, 2013.

15. S. Erbatur, A. M. Marshall, D. Kapur, and P. Narendran. Unification over distributive exponentiation (sub)theories. *Journal of Automata, Languages and Combinatorics (JALC)*, 16(2–4):109–140, 2011.
16. S. Erbatur, A. M. Marshall, and C. Ringeissen. Notions of knowledge in combinations of theories sharing constructors. In L. de Moura, editor, *Automated Deduction - CADE 26 - 26th International Conference on Automated Deduction, Gothenburg, Sweden, August 6-11, 2017, Proceedings*, volume 10395 of *Lecture Notes in Computer Science*, pages 60–76. Springer, 2017.
17. S. Erbatur, A. M. Marshall, and C. Ringeissen. Computing knowledge in equational extensions of subterm convergent theories. *Mathematical Structures in Computer Science*, pages 1–27, 2020. URL: `https://doi.org/10.1017/S0960129520000031`.
18. S. Escobar, C. A. Meadows, and J. Meseguer. Maude-NPA: Cryptographic protocol analysis modulo equational properties. In A. Aldini, G. Barthe, and R. Gorrieri, editors, *Foundations of Security Analysis and Design, Tutorial Lectures*, volume 5705 of *Lecture Notes in Computer Science*, pages 1–50. Springer, 2007.
19. S. Escobar, R. Sasse, and J. Meseguer. Folding variant narrowing and optimal variant termination. *J. Log. Algebr. Program.*, 81(7-8):898–928, 2012.
20. J. Jouannaud and H. Kirchner. Completion of a set of rules modulo a set of equations. *SIAM J. Comput.*, 15(4):1155–1194, 1986.
21. D. Kim, C. Lynch, and P. Narendran. Reviving basic narrowing modulo. In A. Herzig and A. Popescu, editors, *Frontiers of Combining Systems - 12th International Symposium, FroCoS, London, UK, Proceedings*, volume 11715 of *Lecture Notes in Computer Science*, pages 313–329. Springer, Sept. 2019.
22. C. Kirchner and F. Klay. Syntactic theories and unification. In *Logic in Computer Science, 1990. LICS '90, Proceedings, Fifth Annual IEEE Symposium on Logic in Computer Science*, pages 270–277, Jun 1990.
23. C. Lynch and B. Morawska. Basic syntactic mutation. In A. Voronkov, editor, *Automated Deduction - CADE-18, 18th International Conference on Automated Deduction, Copenhagen, Denmark, July 27-30, 2002, Proceedings*, volume 2392 of *Lecture Notes in Computer Science*, pages 471–485. Springer, 2002.
24. A. M. Marshall, C. A. Meadows, and P. Narendran. On unification modulo one-sided distributivity: Algorithms, variants and asymmetry. *Logical Methods in Computer Science*, 11(2), 2015.
25. S. Meier, B. Schmidt, C. Cremers, and D. A. Basin. The TAMARIN prover for the symbolic analysis of security protocols. In N. Sharygina and H. Veith, editors, *Computer Aided Verification - 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings*, volume 8044 of *Lecture Notes in Computer Science*, pages 696–701. Springer, 2013.
26. J. Meseguer. Variant-based satisfiability in initial algebras. *Sci. Comput. Program.*, 154:3–41, 2018.
27. K. Nguyen. Formal verification of a messaging protocol. Work done under the supervision of Vincent Cheval and Véronique Cortier.
28. T. Nipkow. Proof transformations for equational theories. In *Logic in Computer Science, 1990. LICS '90, Proceedings, Fifth Annual IEEE Symposium on Logic in Computer Science*, pages 278–288, Jun 1990.
29. M. Schmidt-Schauß. Unification in a combination of arbitrary disjoint equational theories. *Journal of Symbolic Computation*, 8:51–99, July 1989.
30. E. Tidén and S. Arnborg. Unification problems with one-sided distributivity. *Journal of Symbolic Computation*, 3(1/2):183–202, 1987.
31. K. A. Yelick. Unification in combinations of collapse-free regular theories. *Journal of Symbolic Computation*, 3(1-2):153–181, 1987.