Acta Cybernetica, Vol. 11, No. 3, Szeged, 1994

A Universal Unification Algorithm Based on Unification-Driven Leftmost Outermost Narrowing

Heinz Faßbender [†] [*] Heiko Vogler [†]

Abstract

We formalize a universal unification algorithm for the class of equational theories which is induced by the class of canonical, totally-defined, not strictly subunifiable term rewriting systems (for short: *ctn-trs*). For a *ctn-trs* $\mathcal{R}$ and for two terms t and s, the algorithm computes a ground-complete set of $(E_{\mathcal{R}}, \Delta)$-unifiers of t and s, where $E_{\mathcal{R}}$ is the set of rewrite rules of $\mathcal{R}$ viewed as equations and Δ is the set of constructor symbols. The algorithm is based on the *unification-driven leftmost outermost narrowing relation* (for short: *ulo narrowing relation*) which is introduced in this paper. The ulo narrowing relation interleaves leftmost outermost narrowing steps with decomposition steps taken from the usual unification of terms. In its turn, every decomposition step involves a consistency check on constructor symbols combined with a particular form of the occur check. Since decomposition steps are performed as early as possible, some of the nonsuccessful derivations can be stopped earlier than in other universal unification algorithms for ctn-trs's. We give a proof that our algorithm really is a universal unification algorithm.

1 Introduction

The *unification problem* is to determine whether or not, for two given terms t and s, there exists a unifier φ of t and s, i.e., a substitution φ such that $\varphi(t) = \varphi(s)$. It is well-known that the unification problem for first-order terms is decidable [27].

The problem of unification generalizes to the problem of E-unification, if one considers the equality modulo a set E of equations, denoted by $=_E$, rather than the usual equality; $=_E$ is also called the equational theory induced by E. The *E-unification problem* is to determine whether or not, for two given terms t and s, there exists a substitution φ such that $\varphi(t) =_E \varphi(s)$; then φ is called an E-unifier of

[*]The work of this author has been supported by the Deutsche Forschungsgemeinschaft (DFG).

[†]Dept. of Theoretical Computer Science, University of Ulm, D-89069 Ulm, Germany, e-mail: {fassbend,vogler}@informatik.uni-ulm.de

t and s. Clearly, the decidability of the E-unification problem depends on the set E of equations. If, e.g., E is the empty set, then the E-unification problem coincides with the unification problem and therefore it is decidable. As another example, if E consists of the algebraic laws of associativity and distributivity, then the E-unification problem becomes undecidable; if the law of associativity is dropped, then it is not known whether the problem is decidable. Surveys about the problem of E-unification can be found in [28,20,18].

For a class $\mathcal{E}$ of equational theories, a *universal unification algorithm for $\mathcal{E}$* (for short: uu-algorithm for $\mathcal{E}$) is a nondeterministic algorithm which takes as input an equational theory $=_E$ from the class $\mathcal{E}$ and two terms t and s, and which computes a complete set of E-unifiers of t and s (for the definition of complete set of E-unifiers cf., e.g., [28]). In this paper, we will concentrate on uu-algorithms for classes of equational theories which are induced by particular term rewriting systems (for short: trs's). A trs $\mathcal{R}$ induces the equational theory $=_{E_{\mathcal{R}}}$, where $E_{\mathcal{R}}$ is the set of rules of $\mathcal{R}$ viewed as equations.

Until now, a lot of research has been carried out to construct uu-algorithms for classes $\mathcal{E}$ of equational theories which are induced by trs's. There exist approaches which are extensions of the unification algorithm in [23] (cf. [19,12,18]). In these approaches there are additional transformation rules which perform the application of equations. Other approaches to construct uu-algorithms are based on the concept of narrowing [21]. More precisely, in every such investigation, a uu-algorithm is constructed for some particular class of trs's where the algorithm is based on a particular narrowing relation (plus some additional actions as, e.g., the usual unification of trees). Here we list some pairs (consisting of a class of trs's and a narrowing relation), for which uu-algorithms have been constructed.

- canonical trs's and narrowing [10,16]
- canonical trs's and basic narrowing [16,24]
- left-linear, non-overlapping trs's and D-narrowing [29]
- canonical, uniform trs's and leftmost outermost narrowing strategy [25]
- canonical, totally-defined, not strictly subunifiable trs's and any narrowing strategy [3].

We note that a narrowing strategy is a narrowing relation in which the narrowing occurrence is fixed. We also recall that a trs is *canonical*, if it is confluent and noetherian. A trs is *constructor-based*, if its ranked alphabet Ω is partitioned into sets F and Δ of function symbols and constructor symbols, respectively; moreover, the left-hand side of every rule is a linear term $f(t_1,\ldots,t_n)$ where f is a function symbol, $t_1,\ldots,t_n$ are terms over $\Delta \cup \mathcal{V}$ where $\mathcal{V}$ is the set of variables (cf. [30]). This particular structure of the left hand sides induces that every constructor term is irreducible. A trs is *totally-defined*, if it is constructor-based and every function symbol is completely defined over its domain or, equivalently: every normal form is a constructor term (cf., e.g., [3]). A trs which is *not strictly subunifiable* (cf. [3] and Subsection 3.1 of the present paper), satisfies a kind of local determinism, e.g., two rules cannot be applied at the same occurrence under the same substitution. In [25] totally-defined, not strictly subunifiable trs's are called *uniform* trs's.

In all mentioned narrowing-based approaches, the narrowing derivation results into two terms t' and s'; then, it has to be checked whether t' and s' are unifiable.

In [11] a uu-algorithm for totally-defined trs's is defined which interleaves unification with the narrowing derivation. More precisely, he considers any innermost narrowing strategy and interleaves decomposition steps without any occur check. Since the decomposition steps are performed as early as possible, it is clear that this can lead to a more efficient computation of $E_{\mathcal{R}}$-unifiers.

There exist some other narrowing relations as, e.g., *lazy narrowing* [26], *outer narrowing* [30] which were shown to be complete with respect to the unrestricted narrowing relation. It was not shown that a uu-algorithm which is based on one of the narrowing relations mentioned above, computes a complete set of $E_{\mathcal{R}}$-unifiers. However, for canonical trs's, this statement is clearly true (cf., e.g., [18] for a complete list of these narrowing relations).

In this paper we construct a uu-algorithm for the class of equational theories which are induced by canonical, totally-defined, not strictly subunifiable trs's (for short: ctn-trs's). This algorithm shall serve as a source for efficient implementations of $E_{\mathcal{R}}$-unification on deterministic abstract machines. Thus, we formalize our uu-algorithm in a way from which an operational approach can be derived easily. This is one of the reasons why we will introduce the uu-algorithm on the basis of a narrowing relation and not as a system of transition rules. The second reason for choosing the formalism of a narrowing relation is that we refine the uu-algorithm of [3] which, in its turn is based on a narrowing relation. The uu-algorithm in [3] improves the algorithm in [16] which is based on the unrestricted narrowing relation, by choosing an arbitrary narrowing strategy. For a particular narrowing strategy, our algorithm improves in its turn the uu-algorithm of [3] by following the idea of interleaving decomposition steps with the narrowing derivation as in [11]. However, we consider the leftmost outermost narrowing strategy and we implement a particular occur check. The relationships between the approaches of [16], [3], and [11], and our approach are illustrated in Figure 1.

More precisely, our uu-algorithm is based on the so-called *unification-driven leftmost outermost narrowing relation* (for short: ulo narrowing relation) which is introduced in this paper. For a trs $\mathcal{R}$, the ulo narrowing relation is denoted by $\overset{u}{\rightsquigarrow}_{\mathcal{R}}$. In $\overset{u}{\rightsquigarrow}_{\mathcal{R}}$ leftmost outermost narrowing is interleaved with the application of decomposition-rules (cf., e.g., [23]) which check the consistency of the root symbols of the terms to be unified. Moreover, the applicability of a decomposition-rule depends on a particular version of the occur check. Since decomposition-rules are applied as early as possible, the ulo narrowing relation is called 'unification-driven'.

Actually, for a ctn-trs $\mathcal{R}$ with some set Δ of constructors and two terms t and s, our uu-algorithm computes a *ground complete set of* $(E_{\mathcal{R}}, \Delta)$*-unifiers* of t and s. An $(E_{\mathcal{R}}, \Delta)$-unifier of t and s is an $E_{\mathcal{R}}$-unifier in which all the images are terms over $\Delta \cup \mathcal{V}$, where $\mathcal{V}$ is the set of variables; in particular, this means that we do not consider unifiers of the form $[z_1/f(t)]$ for some function symbol f. Roughly speaking, a set S of $(E_{\mathcal{R}}, \Delta)$-unifiers of t and s is ground complete, if, for every ground $(E_{\mathcal{R}}, \Delta)$-unifier φ of t and s (i.e., the images of φ do not contain variables), there is a $\psi \in S$ which is more general than φ. This notion will be formalized in Section 3.

Let us give an example at which we can illustrate the ulo narrowing relation. In Figure 2 a set R_1 of rules of the ctn-trs $\mathcal{R}_1$ is shown where we assume to have a ranked alphabet $F_1 = \{sh^{(2)}, mi^{(1)}\}$ of function symbols and a ranked alphabet $\Delta_1 = \{\sigma^{(2)}, \alpha^{(0)}\}$ of constructor symbols. Intuitively, $\mathcal{R}_1$ defines two functions *shovel* and *mirror* with arity 2 and 1, respectively; *mirror* reflects terms over Δ at the vertical center line, and *shovel* accumulates in its second argument the

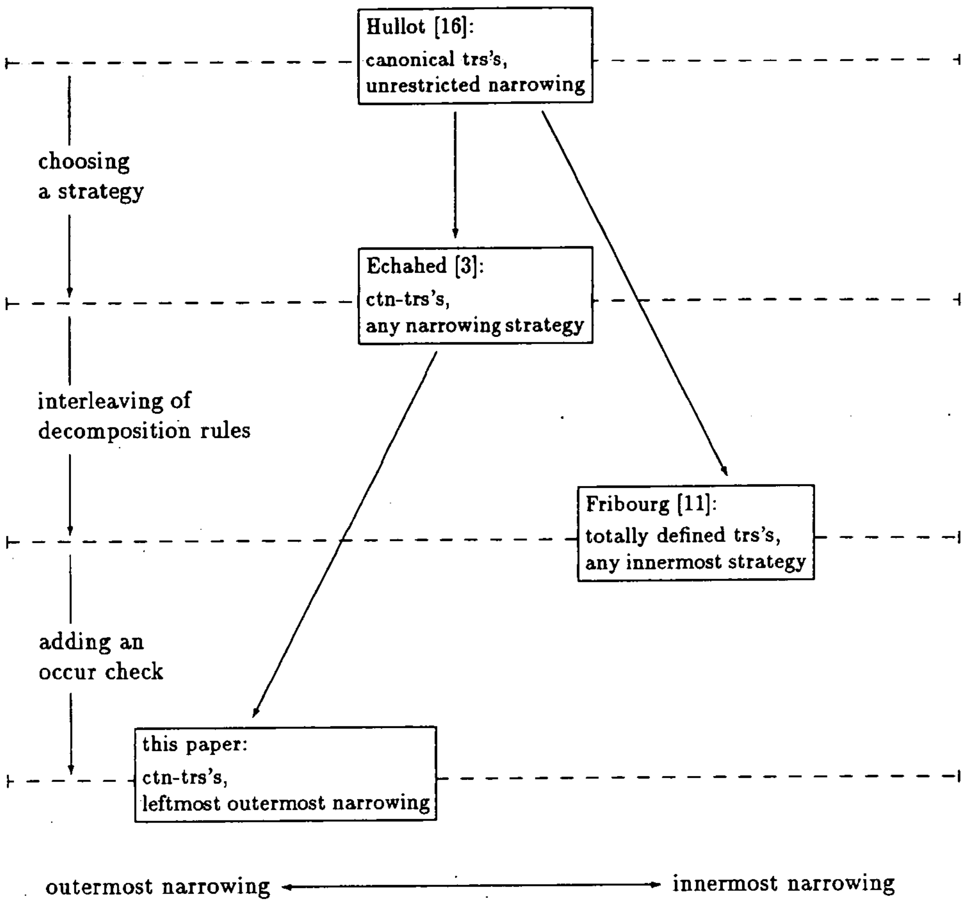


Figure 1: Relationship between some narrowing based approaches.

mirror-image of the second subterm of its first argument. If we consider, e.g., the term $t_1 = \sigma(\sigma(\alpha, s_1), s_2)$ for some terms s_1 and s_2, then for an arbitrary term t_2, $shovel(t_1, t_2)$ is the term $\sigma(mirror(s_1), \sigma(mirror(s_2), t_2))$.

$$\begin{array}{rcll}
sh(\alpha, y_1) & \rightarrow & y_1 & (1) \\
sh(\sigma(x_1, x_2), y_1) & \rightarrow & sh(x_1, \sigma(mi(x_2), y_1)) & (2) \\
mi(\alpha) & \rightarrow & \alpha & (3) \\
mi(\sigma(x_1, x_2)) & \rightarrow & \sigma(mi(x_2), mi(x_1)) & (4)
\end{array}$$

Figure 2: Set of rules of the ctn-trs $\mathcal{R}_1$.

Now we consider the $E_{\mathcal{R}_1}$-unification problem, where the set $E_{\mathcal{R}_1}$ of equations is obtained from R_1 by simply considering the rules as equations. In particular, we want to compute an $E_{\mathcal{R}_1}$-unifier for the terms $sh(z_1, \alpha)$ and $mi(\sigma(z_2, \alpha))$ in which z_1 and z_2 are free variables. Similar to Hullot in [16], we combine the two terms into one term $equ(sh(z_1, \alpha), mi(\sigma(z_2, \alpha)))$ with a new binary symbol equ (which is called H in [16]). Next we enrich R_1 by the set $R(\Delta)$ of *decomposition-rules of* Δ (cf. Figure 3). This enrichment yields the trs $\widehat{\mathcal{R}}_1$.

$$\begin{array}{rcll} equ(\alpha, \alpha) & \rightarrow & \alpha & (5) \\ equ(\sigma(x_1, x_2), \sigma(x_3, x_4)) & \rightarrow & \sigma(equ(x_1, x_3), equ(x_2, x_4)) & (6) \end{array}$$

Figure 3: Set of decomposition-rules of Δ_1.

Then a derivation by $\overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_1}$ starting from $equ(sh(z_1, \alpha), mi(\sigma(z_2, \alpha)))$ may look as follows where we have attached to $\overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_1}$ in every step the narrowing occurrence (in Dewey's notation), the applied rule, and the unifier as additional indices; $\varphi_\emptyset$ denotes the empty substitution; Λ denotes the empty word.

$$\begin{array}{lll}
 & & equ(sh(z_1, \alpha), mi(\sigma(z_2, \alpha))) \\
 & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_1, 1, (2), [z_1/\sigma(z_3, z_4)]} & equ(sh(z_3, \sigma(mi(z_4), \alpha)), mi(\sigma(z_2, \alpha))) \\
 & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_1, 1, (1), [z_3/\alpha]} & equ(\sigma(mi(z_4), \alpha), mi(\sigma(z_2, \alpha))) \\
 * & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_1, 2, (4), \varphi_\emptyset} & equ(\sigma(mi(z_4), \alpha), \sigma(mi(\alpha), mi(z_2))) \\
 ** & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_1, \Lambda, (6), \varphi_\emptyset} & \sigma(equ(mi(z_4), mi(\alpha)), equ(\alpha, mi(z_2))) \\
 & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_1, 11, (3), [z_4/\alpha]} & \sigma(equ(\alpha, mi(\alpha)), equ(\alpha, mi(z_2))) \\
 & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_1, 12, (3), \varphi_\emptyset} & \sigma(equ(\alpha, \alpha), equ(\alpha, mi(z_2))) \\
 & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_1, 1, (5), \varphi_\emptyset} & \sigma(\alpha, equ(\alpha, mi(z_2))) \\
 & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_1, 22, (3), [z_2/\alpha]} & \sigma(\alpha, equ(\alpha, \alpha)) \\
 & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_1, 2, (5), \varphi_\emptyset} & \sigma(\alpha, \alpha)
\end{array}$$

If we compose the unifiers which are involved in the narrowing steps, then we obtain the substitution $\varphi = [z_1/\sigma(\alpha, \alpha), z_2/\alpha]$; in fact, φ is a ground $(E_{\mathcal{R}_1}, \Delta_1)$-unifier of $sh(z_1, \alpha)$ and $mi(\sigma(z_2, \alpha))$. Note that φ is not an $E_{\widehat{\mathcal{R}}_1}$-unifier, because the equational theory is generated by $E_{\mathcal{R}_1}$. The narrowing step at $*$ shows how the ulo narrowing relation deviates from the leftmost outermost narrowing relation. For the latter relation, 11 is the narrowing occurrence in the term $equ(\sigma(mi(z_4), \alpha), mi(\sigma(z_2, \alpha)))$, and then the subterm $mi(z_4)$ has to be narrowed. Note that, since $\mathcal{R}_1$ is constructor-based, every normal form s_1' of the first argument $s_1 = \sigma(mi(z_4), \alpha)$ of equ has the root label σ. Thus, s_1' is unifiable with a normal form s_2' of the second argument $s_2 = mi(\sigma(z_2, \alpha))$ of equ only if the constructors at the root of s_1' and s_2' are identical. Because of reasons of efficiency, it is important to check this consistency as soon as possible. And since the root of s_1 is already a constructor symbol (i.e., s_1 is evaluated *in constructor head normal form*), we narrow s_2 at step $*$ and try to get it also into head normal form. Actually, this form is reached as the result of the application of rule (4). Then, at step $**$, the consistency of root symbols is checked by applying the decomposition-rule (6).

This paper is organized in five sections where the second section contains preliminaries. In Section 3 we recall the definitions of the leftmost outermost narrowing relation and of ctn-trs's; we recall the uu-algorithm of [3]. In Section 4 we define the ulo narrowing relation and an algorithm of which we prove that it is a uu-algorithm, i.e., that it computes a ground complete set of $(E_{\mathcal{R}}, \Delta)$-unifiers for the class of equational theories $E_{\mathcal{R}}$ which are characterized by ctn-trs's. Finally, Section 5 contains some concluding remarks and indicates further research topics.

2 Preliminaries

We recall and collect some notations, basic definitions, and terminology which will be used in the rest of the paper. We try to be in accordance with the notations in [14] and [2] as much as possible.

2.1 General Notations

We denote the set of nonnegative integers by $\mathbb{N}$. The empty set is denoted by $\emptyset$. For $j \in \mathbb{N}$, $[j]$ denotes the set $\{1, \dots, j\}$; thus $[0] = \emptyset$. For a finite set A, $\mathcal{P}(A)$ is the *set of subsets* of A and $card(A)$ denotes the *cardinality* of A. As usual for a set A, A^* denotes the set $\bigcup_{n \in \mathbb{N}} \{a_1 a_2 \dots a_n \mid \text{for every } i \in [n] : a_i \in A\}$ that is called the set of *words over A; λ denotes the empty word.*

2.2 Ranked Alphabets, Variables, and Terms

A pair $(\Omega, rank_\Omega)$ is called *ranked alphabet,* if Ω is an alphabet and $rank_\Omega : \Omega \longrightarrow \mathbb{N}$ is a total function. For $f \in \Omega$, $rank_\Omega(f)$ is called *rank of f; $maxrank\Omega$* denotes the maximal image of $rank_\Omega$. The subset $\Omega^{(m)}$ of Ω consists of all symbols of rank m $(m \geq 0)$. Note that, for $i \neq j$, $\Omega^{(i)}$ and $\Omega^{(j)}$ are disjoint. We can define a ranked alphabet $(\Omega, rank_\Omega)$ either by enumerating the finitely many subsets $\Omega^{(m)}$ that are not empty, or by giving a set of symbols that are indexed with their (unique) rank. For example, if $\Omega = \{a, b, c\}$ and $rank_\Omega : \Omega \longrightarrow \mathbb{N}$ with $rank_\Omega(a) = 0$, $rank_\Omega(b) = 2$, and $rank_\Omega(c) = 7$, then we can describe $(\Omega, rank_\Omega)$ either by $\Omega^{(0)} = \{a\}$, $\Omega^{(2)} = \{b\}$, and $\Omega^{(7)} = \{c\}$ or by $\{a^{(0)}, b^{(2)}, c^{(7)}\}$. If the ranks of the symbols are clear from the context, then we drop the function $rank_\Omega$ from the denotation of the ranked alphabet $(\Omega, rank_\Omega)$ and simply write Ω.

In the rest of the paper we let $\mathcal{V}$ denote a fixed enumerable set. Its elements are called *variables.* In the following we use the notations $x, x_1, x_2, \dots, y, y_1, y_2, \dots, z, z_1, z_2, \dots$ for variables.

Let Ω be a ranked alphabet and let S be an arbitrary set (in the sequel S will be instantiated by sets of variables). Then the set of *terms over Ω indexed by S,* denoted by $T\langle\Omega\rangle(S)$, is defined inductively as follows: (i) $S \subseteq T\langle\Omega\rangle(S)$ and (ii) for every $f \in \Omega^{(k)}$ with $k \geq 0$ and $t_1, \dots, t_k \in T\langle\Omega\rangle(S) : f(t_1, \dots, t_k) \in T\langle\Omega\rangle(S)$. The set $T\langle\Omega\rangle(\emptyset)$, denoted by $T\langle\Omega\rangle$, is called the set of *ground terms over* Ω.

For a term $t \in T\langle\Omega\rangle(\mathcal{V})$, the set of *occurrences of t,* denoted by $O(t)$, is a subset of $\mathbb{N}^*$ and it is defined inductively on the structure of t as follows:

(i) If $t = x$ where $x \in \mathcal{V}$, then $O(t) = \{\Lambda\}$,

(ii) if $t = f$ where $f \in \Omega^{(0)}$, then $O(t) = \{\Lambda\}$, and

(iii) if $t = f(t_1, \ldots, t_n)$ where $f \in \Omega^{(n)}$ and $n > 0$, and for every $i \in [n] : t_i \in T\langle\Omega\rangle(\mathcal{V})$, then $O(t) = \{\Lambda\} \cup \bigcup_{i \in [n]} \{iu \mid u \in O(t_i)\}$.

The prefix order on $O(t)$ is denoted by $<$ and the lexicographical order on $O(t)$ is denoted by $<_{lex}$. The reflexive closures of $<$ and $<_{lex}$ are denoted by $\leq$ and $\leq_{lex}$, respectively. Clearly, $\leq \subseteq \leq_{lex}$. Note that $<_{lex}$ is a total order, whereas, in general, $<$ is a partial order. The minimal element with respect to $\leq_{lex}$ in a subset S of $O(t)$ is denoted by $min_{lex}(S)$. For a term $t \in T\langle\Omega\rangle(\mathcal{V})$ and an occurrence u of t, t/u denotes the *subterm of t at occurrence u*, and $t[u]$ denotes the *label of t at occurrence u*. We use $\mathcal{V}(t)$ to denote the set of variables occurring in t; that is, $x \in \mathcal{V}(t)$, if $x \in \mathcal{V}$ and there exists a $u \in O(t)$ such that $t/u = x$. Finally, we define $t[u \leftarrow s]$ as the term t in which we have replaced the subterm at occurrence u by the term s.

2.3 Algebras, Substitutions, and Congruences

Let $(\Omega, rank_\Omega)$ be a ranked alphabet. An *Ω-algebra* is a pair (A, int_A), where A is a set and int_A is a mapping such that $int_A(f) \in A$, if $rank_\Omega(f) = 0$, and $int_A(f) : A^n \to A$, if $rank_\Omega(f) = n$.

The Ω-algebra $(T\langle\Omega\rangle(\mathcal{V}), int_T)$, where for every $f \in \Omega^{(n)}$ and for every $t_i \in T\langle\Omega\rangle(\mathcal{V})$ with $i \in [n] : int_T(f)(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$, is called the *$\Omega$-term algebra*. It is a free Ω-algebra (cf. [15]).

If (A, int_A) and (B, int_B) are two Ω-algebras, we say that $h : A \to B$ is a *homomorphism*, if for every $f \in \Omega^{(n)}$ with $n \geq 0$ and for every $a_i \in A$ with $i \in [n]$, we have

$$h(int_A(f)(a_1, \ldots, a_n)) = int_B(f)(h(a_1), \ldots, h(a_n)).$$

A mapping $\nu : \mathcal{V} \to A$ is called an *A-assignment*.

The property that every A-assignment can be extended in a unique way to a homomorphism from $T\langle\Omega\rangle(\mathcal{V})$ to A is called the *universal property for the free Ω-algebras* in [15]. We use ν to denote both the A-assignment and its extension.

A *$(\mathcal{V}, \Omega)$-substitution* is a $T\langle\Omega\rangle(\mathcal{V})$-assignment φ, where the set $\{x \mid \varphi(x) \neq x, x \in \mathcal{V}\}$ is finite. The set $\{x \mid \varphi(x) \neq x\}$ is denoted by $\mathcal{D}(\varphi)$ and it is called the *domain of* φ. If $\mathcal{D}(\varphi) = \{x_1, \ldots, x_n\}$, then φ is represented by $[x_1/\varphi(x_1), \ldots, x_n/\varphi(x_n)]$. If $\mathcal{D}(\varphi) = \emptyset$, then φ is denoted by $\varphi_\emptyset$. We say that φ is *ground*, if for every $x \in \mathcal{D}(\varphi) : \mathcal{V}(\varphi(x)) = \emptyset$. The set $\bigcup_{x \in \mathcal{D}(\varphi)} \mathcal{V}(\varphi(x))$ is denoted by $\mathcal{I}(\varphi)$ and is called the set of *variables introduced by* φ. The set of $(\mathcal{V}, \Omega)$-substitutions and the set of ground $(\mathcal{V}, \Omega)$-substitutions are denoted by $Sub(\mathcal{V}, \Omega)$ and $gSub(\mathcal{V}, \Omega)$, respectively. The *composition* of two $(\mathcal{V}, \Omega)$-substitutions φ and ψ is the $T\langle\Omega\rangle(\mathcal{V})$-assignment which is defined by $\psi(\varphi(x))$ for every $x \in \mathcal{V}$. It is denoted by $\varphi \circ \psi$.

An equivalence relation $\sim$ on $T\langle\Omega\rangle(\mathcal{V})$ is called an *Ω-congruence over $T\langle\Omega\rangle(\mathcal{V})$*, if for every $f \in \Omega^{(n)}$ with $n > 0$ and for every $t_1, s_1, \ldots, t_n, s_n \in T\langle\Omega\rangle(\mathcal{V})$ with $t_1 \sim s_1, \ldots, t_n \sim s_n$, the relation $f(t_1, \ldots, t_n) \sim f(s_1, \ldots, s_n)$ holds.

2.4 E-Unification

An *equation over* Ω *and* $\mathcal{V}$ is a pair (t, s), where $t, s \in T\langle\Omega\rangle(\mathcal{V})$. As usual we denote an equation (t, s) by $t = s$. Thus, we consider an equation as an ordered pair. In the rest of the paper, we let E denote a finite set of equations over Ω and $\mathcal{V}$. The *E-equality*, denoted by $=_E$, is the finest (i.e., smallest) congruence relation over $T\langle\Omega\rangle(\mathcal{V})$ containing every pair $(\psi(t), \psi(s))$, where $t = s \in E$ and ψ is an arbitrary $(\mathcal{V}, \Omega)$-substitution. If $t =_E s$, then t and s are called *E-equal* (cf. [15]). Two terms $t, s \in T\langle\Omega\rangle(\mathcal{V})$ are called *E-unifiable*, if there exists a $(\mathcal{V}, \Omega)$-substitution φ such that $\varphi(t) =_E \varphi(s)$. The set $\{\varphi \mid \varphi(t) =_E \varphi(s)\}$ is called the *set of E-unifiers of t and s*, and it is denoted by $\mathcal{U}_E(t, s)$ (cf. [28]). Let V be a finite subset of $\mathcal{V}$. We define the preorder $\preceq_E (V)$ on $(\mathcal{V}, \Omega)$-substitutions by $\varphi \preceq_E \varphi' \ (V)$, if there exists a $(\mathcal{V}, \Omega)$-substitution ψ such that for every $x \in V : \psi(\varphi(x)) =_E \varphi'(x)$ (cf. [28]).

2.5 TRS, Reduction, Narrowing, and Narrowing Trees

A *term rewriting system*, denoted by $\mathcal{R}$, is a pair (Ω, R), where Ω is a ranked alphabet and R is a finite set of rules of the form $l \to r$ such that $l, r \in T\langle\Omega\rangle(\mathcal{V})$ and $\mathcal{V}(r) \subseteq \mathcal{V}(l)$ (cf. [14]). For every term rewriting system $\mathcal{R} = (\Omega, R)$, the *related set of equations*, denoted by $E_\mathcal{R}$, is the set $\{l = r \mid l \to r \in R\}$ (cf. [24]).

The *reduction relation associated with* $\mathcal{R}$, denoted by $\Longrightarrow_\mathcal{R}$, is defined as follows: for every $t, s \in T\langle\Omega\rangle(\mathcal{V}) : t \Longrightarrow_\mathcal{R} s$, if there exist $u \in O(t)$ with $t/u \notin \mathcal{V}, \varphi \in Sub(\mathcal{V}, \Omega), l \to r \in R$ with $\varphi(l) = t/u$, and $s = t[u \leftarrow \varphi(r)]$ (cf. [14]). We use the standard notation $\Longrightarrow^*$ to denote the transitive-reflexive closure of $\Longrightarrow$.

A term rewriting system is *canonical*, if it is confluent and noetherian (cf. [15]). A term t is a *normal form of a term s*, if $s \Longrightarrow_\mathcal{R}^* t$ and t is irreducible, i.e., there does not exist any term t' such that $t \Longrightarrow_\mathcal{R} t'$. For a canonical term rewriting system $\mathcal{R}$, every term t has exactly one normal form (cf. [15]) which is denoted by $nf_\mathcal{R}(t)$. A $(\mathcal{V}, \Omega)$-substitution φ is *in normal form* if for every $x \in \mathcal{D}(\varphi)$, $\varphi(x)$ is irreducible.

The *set of narrowing interfaces for* $\mathcal{R}$ *and* $t \in T\langle\Omega\rangle(\mathcal{V})$, denoted by $narI(\mathcal{R}, t)$, is the set $\{(u, \varphi, l \to r, \rho) \mid u \in O(t), t/u \notin \mathcal{V}, l \to r \in R, \rho$ is a renaming of variables in l such that $\mathcal{V}(\rho(l)) \cap \mathcal{V}(t) = \emptyset, \varphi \in Sub(\mathcal{V}, \Omega)$ is the most general unifier of $\rho(l)$ and $t/u\}$. The *set of narrowing occurrences for* $\mathcal{R}$ *and* $t \in T\langle\Omega\rangle(\mathcal{V})$, denoted by $narO(\mathcal{R}, t)$, is the set $\{u \mid (u, \varphi, l \to r, \rho) \in narI(\mathcal{R}, t)\}$. The *narrowing relation associated with* $\mathcal{R}$, denoted by $\rightsquigarrow_\mathcal{R}$, is defined as follows. For every $t, s \in T\langle\Omega\rangle(\mathcal{V})$ and $\psi, \psi' \in Sub(\mathcal{V}, \Omega) : (t, \psi) \rightsquigarrow_\mathcal{R} (s, \psi')$, if the following three conditions hold:

1. There is a narrowing interface $(u, \varphi, l \to r, \rho) \in narI(\mathcal{R}, t)$.
2. $s = \varphi(t[u \leftarrow \rho(r)])$.
3. $\psi' = \psi \circ (\varphi|_{\mathcal{V}(t)})$ (cf. [24]), where composition is read from left to right.

It is obvious that there are two types of nondeterminism involved in the narrowing relation. Starting from a term t, first, there may be more than one narrowing occurrence in t, and second, for a fixed narrowing occurrence, there may be more than one narrowing interface. As usual, for a given starting term t and for given orders on the set of occurrences of t and on the set R of rules, one can collect all the possible narrowing sequences which start from t, into one tree which is called *narrowing tree for t*.

3 $E_{\mathcal{R}}$-Unification by LO Narrowing and Unification

As starting point of our considerations we recall the uu-algorithm which is induced by Theorem 3 in [3]. Here we impose the leftmost outermost narrowing strategy on the narrowing relation of the algorithm.

Before we recall the approach of [3], let us first state that the approach of [25] is technically a bit too complicated for the present purpose although it would theoretically also be a possible starting point. In [25] a uu-algorithm for equational theories induced by canonical, uniform trs's, is presented, where only leftmost outermost narrowing steps are allowed; in fact, ctn-trs's are canonical, uniform trs's.

Furthermore, we note that for ctn-trs, outer narrowing [30] is the same as outermost narrowing. But, in [30], there is no uu-algorithm presented, only a universal matching algorithm.

3.1 The Leftmost Outermost Narrowing Relation and CTN-TRS's

In the leftmost outermost narrowing relation, a pair (t,ψ) derives to a pair (t',ψ') at the minimal element (with respect to $\leq_{lex}$) of the set of narrowing occurrences in t.

Definition 3.1 Let $\mathcal{R} = (\Omega, R)$ be a term rewriting system and let $t \in T\langle\Omega\rangle(\mathcal{V})$.

- The *leftmost outermost narrowing occurrence for $\mathcal{R}$ and t*, denoted by $lo\text{-}narO(\mathcal{R},t)$, is the narrowing occurrence $min_{lex}(narO(\mathcal{R},t))$.
- The *set of leftmost outermost narrowing interfaces for $\mathcal{R}$ and t*, denoted by $lo\text{-}narI(\mathcal{R},t)$, is the set

$$\{(u,\varphi,l \rightarrow r,\rho) \mid (u,\varphi,l \rightarrow r,\rho) \in narI(\mathcal{R},t) \text{ and } u = lo\text{-}narO(\mathcal{R},t)\}.$$

- The *leftmost outermost narrowing relation associated with $\mathcal{R}$*, denoted by $\overset{lo}{\rightsquigarrow}_{\mathcal{R}}$, is defined as follows: for every $t,s \in T\langle\Omega\rangle(\mathcal{V})$ and $\psi,\psi' \in Sub(\mathcal{V},\Omega)$: (t,ψ) *derives to* (s,ψ') *by* $\overset{lo}{\rightsquigarrow}_{\mathcal{R}}$, denoted by $(t,\psi) \overset{lo}{\rightsquigarrow}_{\mathcal{R}} (s,\psi')$, if the following three conditions hold:
  1. there is a leftmost outermost narrowing interface $(u,\varphi,l \rightarrow r,\rho) \in lo\text{-}narI(\mathcal{R},t)$
  2. $s = \varphi(t[u \leftarrow \rho(r)])$

3. $\psi' = \psi \circ (\varphi|_{\mathcal{V}(t)})$ $\oplus$

It is obvious, that $\overset{lo}{\rightsquigarrow}_{\mathcal{R}} \subseteq \rightsquigarrow_{\mathcal{R}}$.

In Example 1 of [3] it is shown that the uu-algorithm of [16] which is based on the unrestricted narrowing relation, is not complete if one imposes a strategy on the narrowing relation. In particular, this negative result holds for the leftmost outermost narrowing relation.

However, Echahed also proves a positive result: the uu-algorithm of [16] stays complete for an arbitrary strategy imposed on the narrowing relation if one restricts to canonical trs's that have the *property of free strategies.* We call these trs's *canonical, totally defined, not strictly sub-unifiable term rewriting systems,* for short: *ctn-trs's.*

A ctn-trs $\mathcal{R} = (\Omega, R)$ is a canonical trs, where Ω is divided into two disjoint ranked alphabets, denoted by F and Δ. F is called the set of function symbols and Δ is called the set of working symbols or constructors. The left hand sides of the rewrite rules in R are linear in $\mathcal{V}$; function symbols only occur at the root of a left hand side. Thus, ctn-trs's are particular constructor-based trs's (cf. [30]). Furthermore, every function symbol in F is totally defined over its domain (cf. Definition 12 in [3]), i.e., if a term is in normal form, then it is in $T\langle\Delta\rangle(\mathcal{V})$. Finally, the left hand sides of the rules in R must be pairwise not strictly sub-unifiable.

Definition 3.2 (cf. [3] Definition 10 and Definition 11). Let $t, t' \in T\langle\Omega\rangle(\mathcal{V})$.

- t and t' are *sub-unifiable,* if there exists an occurrence u in $O(t) \cap O(t')$ such that the following two conditions hold:
  1. t/u and $\rho(t'/u)$ are unifiable with most general unifier σ_u where ρ is a variable-renaming such that $\mathcal{V}(t/u) \cap \mathcal{V}(\rho(t'/u)) = \emptyset$.
  2. For all occurrences w with $w < u$, t/w and t'/w have the same label at the root.
- t and t' are *strictly sub-unifiable,* if there exists an occurrence u where t and t' are sub-unifiable and the corresponding most general unifier σ_u is neither a variable renaming nor the empty substitution. $\oplus$

Example 3.3 Let $\mathcal{R} = (\Omega, R)$ be a canonical trs where $\Omega = \{f^{(2)}, \gamma^{(1)}, \alpha^{(0)}\}$ and let R contain the following rules:

$$\begin{array}{rcll} f(\alpha,\alpha) & \rightarrow & \alpha & (1) \\ f(\gamma(x),\alpha) & \rightarrow & \gamma(\alpha) & (2) \\ f(x,\gamma(y)) & \rightarrow & \gamma(\gamma(\alpha)) & (3) \end{array}$$

- For the trs $\mathcal{R}$, the left hand sides of rule 1 and rule 3 are strictly sub-unifiable at occurrence 1; the same holds for rule 2 and rule 3.
- The left hand sides of rule 1 and rule 2 are sub-unifiable at occurrence 2 but not strictly sub-unifiable, because the most general unifier σ_2 is the empty substitution.

- Let $\mathcal{R}' = (\Omega, R')$ be a trs where R' contains rules 1 and 2 in R and additionally the following two rules:

$$\begin{array}{rcll} f(\alpha, \gamma(y)) & \to & \gamma(\gamma(\alpha)) & (3) \\ f(\gamma(x), \gamma(y)) & \to & \gamma(\gamma(\alpha)) & (4) \end{array}$$

  The left hand sides of the rules in R' are pairwise not strictly sub-unifiable. Furthermore, the left hand sides of the rules 2 and 3 are not sub-unifiable and the left hand sides of the rules 1 and 4 are not sub-unifiable. $\oplus$

Now, we are able to define ctn-trs.

Definition 3.4 Let $\mathcal{R} = (\Omega, R)$ be a trs. $\mathcal{R}$ is a *canonical, totally defined, not strictly sub-unifiable term rewriting system,* for short *ctn-trs,* if the following conditions hold:

1. $\mathcal{R}$ is canonical.
2. $\Omega = F \cup \Delta$ and $F \cap \Delta = \emptyset$.
3. Every left hand side is linear in $\mathcal{V}$.
4. Every left hand side has the form $f(t_1, \ldots, t_n)$ where $f \in F^{(n)}$ and for every $i \in [n] : t_i \in T\langle\Delta\rangle(\mathcal{V})$.
5. For every $t \in T\langle\Omega\rangle(\mathcal{V}) : nf_{\mathcal{R}}(t) \in T\langle\Delta\rangle(\mathcal{V})$.
6. The left hand sides of the rewrite rules in R are pairwise not strictly sub-unifiable. $\oplus$

In the sequel we will denote a ctn-trs by the triple (F, Δ, R). In fact, the trs in Figure 2 is a ctn-trs. To give the reader an idea about the computational power of ctn-trs's, we mention that every primitive recursive tree function [17] can be described by a ctn-trs (which follows from [6]). But in fact, ctn-trs's are even more powerful.

In general, it is not decidable whether a trs is canonical (cf., e.g., [15]). However, if $\mathcal{R}$ is canonical, then the conditions (2)-(6) in Definition 3.4 are decidable.

3.2 The UU-Algorithm of Echahed

Here we recall the uu-algorithm of Echahed. This algorithm computes particular $E_{\mathcal{R}}$-unifiers which are called ground $(E_{\mathcal{R}}, \Delta)$-unifiers. The range of such a unifier is a subset of $T\langle\Delta\rangle$, i.e., function symbols and variables are not allowed. For a ctn-trs $\mathcal{R}$, this point of view is reasonable, because, in particular, $\mathcal{R}$ is totally defined and every function call can be evaluated into an element of $T\langle\Delta\rangle$. Thus, e.g., if we consider the ctn-trs $\mathcal{R}_1$ in Figure 2 and we want to compute $E_{\mathcal{R}_1}$-unifiers of the terms $mi(x)$ and z, then we are not interested in the minimal $E_{\mathcal{R}_1}$-unifier $[z/mi(x)]$; rather we should be able to compute the unifier $[z/\alpha, x/\alpha]$.

Definition 3.5 Let $\mathcal{R} = (F, \Delta, R)$ be a ctn-trs, let $t, s \in T\langle F \cup \Delta\rangle(\mathcal{V})$, and let $\varphi \in \mathcal{U}_{E_{\mathcal{R}}}(t, s)$ be an $E_{\mathcal{R}}$-unifier of t and s.

- φ is an $(E_{\mathcal{R}}, \Delta)$*-unifier of t and s,* if $\varphi \in Sub(\mathcal{V}, \Delta)$.
- φ is a *ground* $(E_{\mathcal{R}}, \Delta)$*-unifier of t and s,* if $\varphi \in gSub(\mathcal{V}, \Delta)$.

The sets of $(E_{\mathcal{R}}, \Delta)$-unifiers and of ground $(E_{\mathcal{R}}, \Delta)$-unifiers of t and s are denoted by $\mathcal{U}_{(E_{\mathcal{R}},\Delta)}(t,s)$ and $g\mathcal{U}_{(E_{\mathcal{R}},\Delta)}(t,s)$, respectively. $\oplus$

Similar to the situation of E-unifiers of two terms t and s, we do not have to compute the whole set $g\mathcal{U}_{(E_{\mathcal{R}},\Delta)}(t,s)$, but rather an approximation of it. It suffices to compute a ground complete set of $(E_{\mathcal{R}}, \Delta)$-unifiers of t and s.

Definition 3.6 (cf. [3] page 92) Let $\mathcal{R} = (F, \Delta, R)$ be a ctn-trs. Let $t, s \in T\langle F \cup \Delta\rangle(\mathcal{V})$ and let W be a finite set of variables containing $V = \mathcal{V}(t) \cup \mathcal{V}(s)$. A set S of $(\mathcal{V}, \Delta)$-substitutions is a *ground complete set of* $(E_{\mathcal{R}}, \Delta)$*-unifiers of t and s away from W,* if the following three conditions hold:

1. For every $\varphi \in S$: $\mathcal{D}(\varphi) \subseteq V$ and $\mathcal{I}(\varphi) \cap W = \emptyset$.
2. $S \subseteq \mathcal{U}_{(E_{\mathcal{R}},\Delta)}(t,s)$.
3. For every $\varphi \in g\mathcal{U}_{(E_{\mathcal{R}},\Delta)}(t,s)$ there is a $\psi \in S$ such that $\psi \preceq_{E_{\mathcal{R}}} \varphi\ (V)$. $\oplus$

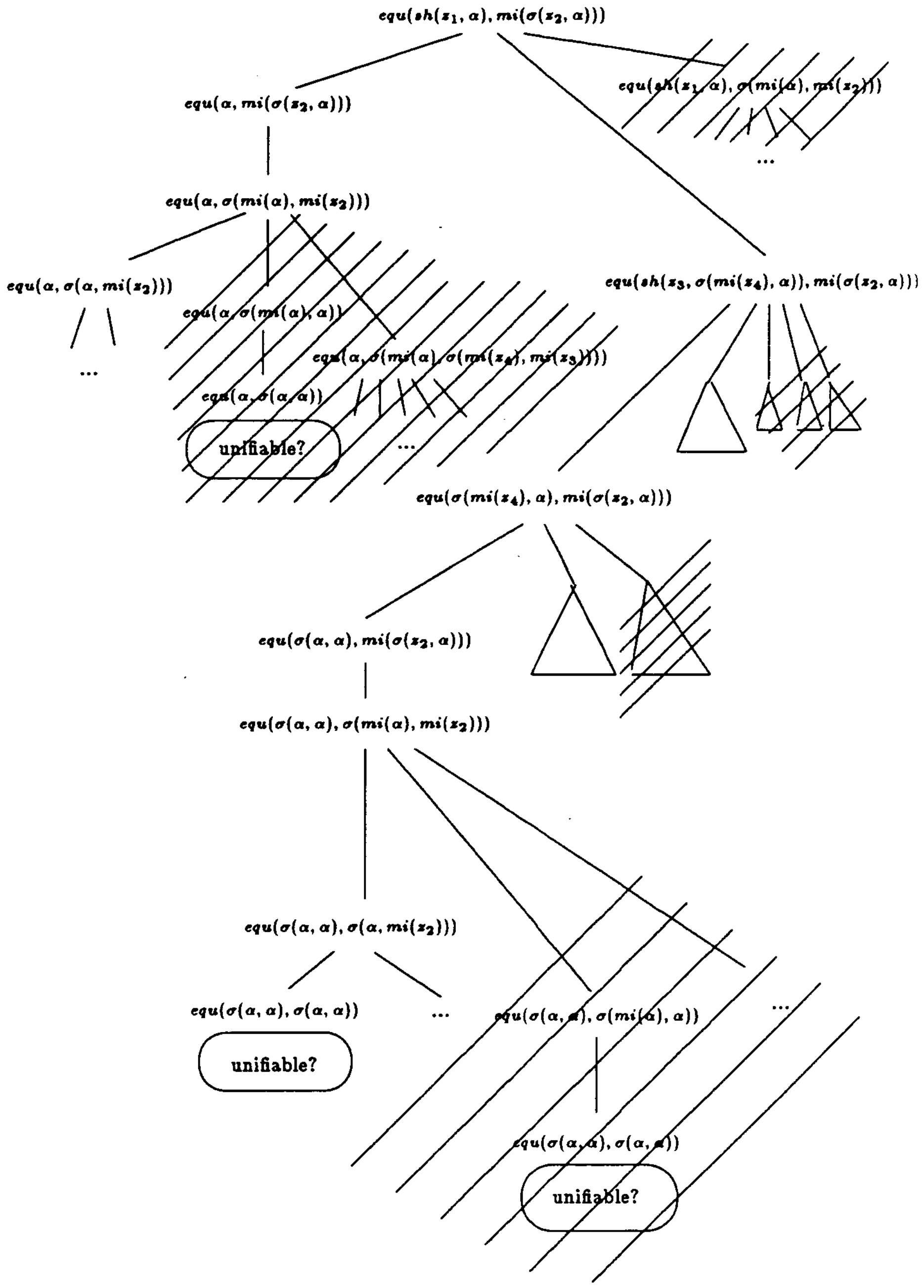
For ctn-trs's, Theorem 3 of [3] shows a uu-algorithm which computes a ground complete set of $(E_{\mathcal{R}}, \Delta)$-unifiers based on an arbitrary narrowing strategy. We present an instance of this theorem where we choose the leftmost outermost strategy. We assume that $\overset{lo}{\rightsquigarrow}_{\mathcal{R}}$ is extended to objects of the form $(equ(t,s), \varphi)$ where equ is a new binary symbol, in the way as it is done in, e.g., [16] and [3].

Theorem 3.7 (cf. [3] Theorem 3) Let $\mathcal{R} = (F, \Delta, R)$ be a ctn-trs. Let $t, s \in T\langle F \cup \Delta\rangle(\mathcal{V})$, and let V be the set $\mathcal{V}(t) \cup \mathcal{V}(s)$. Let S be the set of all $(\mathcal{V}, \Delta)$-substitutions φ such that φ is in S iff there exists a derivation by $\overset{lo}{\rightsquigarrow}_{\mathcal{R}}$:

$$\begin{aligned}(equ(t,s), \varphi_\emptyset) &\overset{lo}{\rightsquigarrow}_{\mathcal{R}} (equ(t_1,s_1), \varphi_1)\\ &\overset{lo}{\rightsquigarrow}_{\mathcal{R}} (equ(t_2,s_2), \varphi_2) \overset{lo}{\rightsquigarrow}_{\mathcal{R}} \cdots \overset{lo}{\rightsquigarrow}_{\mathcal{R}} (equ(t_n,s_n), \varphi_n),\end{aligned}$$

where for every $i \in [n]$: φ_i is in normal form, t_n and s_n are in normal form and unifiable with most general unifier μ, and $\varphi = (\varphi_n \circ \mu)|_V$. Then S is a ground complete set of $(E_{\mathcal{R}}, \Delta)$-unifiers of t and s away from V. $\oplus$

Clearly, in the leftmost outermost narrowing relation only one type of nondeterminism occurs, i.e, for a fixed narrowing occurrence, there may be more than one rule applicable. Thus, the leftmost outermost narrowing tree for a term $equ(t,s)$ results from the narrowing tree for $equ(t,s)$ by deleting the branches which do not correspond to derivations by $\overset{lo}{\rightsquigarrow}_{\mathcal{R}}$. In Figure 4 we illustrate the leftmost outermost narrowing tree for the term $equ(sh(z_1,\alpha), mi(\sigma(z_2,\alpha)))$ and we compare it with the narrowing tree for $equ(sh(z_1,\alpha), mi(\sigma(z_2,\alpha)))$. The latter one consists of the shaded and the non-shaded areas, whereas the former one only contains the non-shaded areas. We note that, for the computation of the $E_{\mathcal{R}}$-unifier, it must be checked after the computations of the narrowing derivations, whether the two subtrees contained in the labels of the leaves are unifiable.



equ(sh(z1, α), mi(σ(z2, α)))
equ(α, mi(σ(z2, α)))
equ(sh(z1, α), σ(mi(α), mi(z2)))
equ(α, σ(mi(α), mi(z2)))
equ(α, σ(α, mi(z2)))
equ(α, σ(mi(α), α))
equ(α, σ(mi(α), σ(mi(z4), mi(z3))))
equ(α, σ(α, α))
unifiable?
equ(sh(z3, σ(mi(z4), α)), mi(σ(z2, α)))
equ(σ(mi(z4), α), mi(σ(z2, α)))
equ(σ(α, α), mi(σ(z2, α)))
equ(σ(α, α), σ(mi(α), mi(z2)))
equ(σ(α, α), σ(α, mi(z2)))
equ(σ(α, α), σ(α, α))
unifiable?
equ(σ(α, α), σ(mi(α), α))
equ(σ(α, α), σ(α, α))
unifiable?

Figure 4: Leftmost outermost narrowing tree for $equ(sh(z_1, \alpha), mi(\sigma(z_2, \alpha)))$.

In general, by fixing one narrowing occurrence the breadth of the narrowing trees is reduced. Moreover, by choosing the leftmost outermost narrowing strategy, also the depth of narrowing trees is possibly reduced: arguments of functions are only evaluated on demand.

If we regard the shape of narrowing trees as a measure of the complexity of a uu-algorithm, then the uu-algorithm which is induced by Theorem 3.7, is as efficient as the uu-algorithm in [16] which is based on the unrestricted narrowing relation. But in some cases it is even more efficient. This is the reason for paying the price of a reduced expressiveness of ctn-trs's with respect to canonical trs's, because we want to introduce an efficient uu-algorithm.

4 $E_{\mathcal{R}}$-Unification by Unification-Driven LO-Narrowing

In this section we increase the efficiency of the uu-algorithm implied by Theorem 3.7 as follows. Consider a leaf n of some leftmost outermost narrowing tree. Now we view the unification which takes place at n, as a sequence of decomposition steps [23]. Next we split up this sequence and apply every decomposition step as early as possible. Moreover, whether a decomposition step is applicable or not depends on a particular occur check. By means of this technique, some of the derivations that do not yield unifiers, are blocked earlier than in the uu-algorithm of Echahed.

Every decomposition step is formalized as the application of one of the additional rules called *decomposition-rules.* The union of the decomposition-rules and $\mathcal{R}$ itself is called the *extension of* $\mathcal{R}$. Then the ulo narrowing relation is defined on the basis of the extension of $\mathcal{R}$.

We start this section with the definition of the ulo narrowing relation. As an intermediate result, we rephrase Theorem 3.7 by using the ulo narrowing relation (restricted to decomposition-rules) to unify two terms. Finally, based on the ulo narrowing relation, we present a uu-algorithm which computes a ground complete set of $(E_{\mathcal{R}}, \Delta)$-unifiers for every equational theory $=_{E_{\mathcal{R}}}$ where $\mathcal{R}$ is a ctn-trs.

4.1 The Unification-Driven Leftmost Outermost Narrowing Relation

Definition 4.1 Let $\mathcal{R} = (F, \Delta, R)$ be a ctn-trs.

- Let $\sigma \in \Delta^{(k)}$ with $k \geq 0$. The *decomposition-rule for* σ has the form

  $$equ(\sigma(x_1, \ldots, x_k), \sigma(x_{k+1}, \ldots, x_{2k})) \rightarrow \sigma(equ(x_1, x_{k+1}), \ldots, equ(x_k, x_{2k})).$$

- The *decomposition-part of* $\mathcal{R}$, denoted by $\mathcal{R}(\Delta)$, is the triple $(\widehat{F}, \Delta, R(\Delta))$ where $\widehat{F} = F \cup \{equ\}$ and equ is a new binary symbol, and $R(\Delta)$ is the set of all decomposition-rules for elements in Δ.

- The *extension of* $\mathcal{R}$, denoted by $\widehat{\mathcal{R}}$, is the triple $(\widehat{F}, \Delta, \widehat{R})$ where $\widehat{R}$ is the set $R \cup R(\Delta)$. $\oplus$

$$\begin{array}{rcll}
sh(\alpha, y_1) & \rightarrow & y_1 & (1)\\
sh(\sigma(x_1, x_2), y_1) & \rightarrow & sh(x_1, \sigma(mi(x_2), y_1)) & (2)\\
mi(\alpha) & \rightarrow & \alpha & (3)\\
mi(\sigma(x_1, x_2)) & \rightarrow & \sigma(mi(x_2), mi(x_1)) & (4)\\
equ(\alpha, \alpha) & \rightarrow & \alpha & (5)\\
equ(\sigma(x_1, x_2), \sigma(x_3, x_4)) & \rightarrow & \sigma(equ(x_1, x_3), equ(x_2, x_4)) & (6)
\end{array}$$

Figure 5: Set of rules of an extension.

In Figure 5 the rules of the extension $\widehat{\mathcal{R}}_1 = (\widehat{F}_1, \Delta_1, \widehat{R}_1)$ of $\mathcal{R}_1$ (cf. Figures 2 and 3) are shown where $\widehat{F}_1 = \{sh^{(2)}, mi^{(1)}, equ^{(2)}\}$ and $\Delta_1 = \{\sigma^{(2)}, \alpha^{(0)}\}$.

Roughly speaking, the ulo narrowing relation is almost the same as the leftmost outermost narrowing relation associated with $\widehat{\mathcal{R}}$. But there are the following three differences between the two relations. Let (t, φ) be the current derivation form.

1. Consider the term $t = equ(\alpha, \sigma(mi(\alpha), mi(z_2)))$ at occurrence 11 in the leftmost outermost narrowing tree of Figure 4. The leftmost outermost narrowing occurrence of t is 21. However, it is clear that none of the branches starting from t will yield an $E_{\mathcal{R}_1}$-unifier, because the two direct subterms α and $\sigma(mi(\alpha), mi(z_2))$ of t have different root symbols which cannot be changed in further derivation steps (this is due to the fact that $\mathcal{R}_1$ is constructor-based); hence, the terms α and $\sigma(mi(\alpha), mi(z_2))$ cannot be $E_{\mathcal{R}_1}$-unified. Thus, we will define the ulo narrowing relation in such a way that it blocks at this point. We realize this property by requiring that rules may only be applied at the leftmost occurrence of equ in the current derivation form t. This occurrence of equ is called *important occurrence of* t, denoted by $impO(t)$, because the nonunifiability of the two subterms of t is recognized exactly here. In our concrete situation, $impO(t) = \Lambda$ and none of the decomposition rules is applicable at $impO(t)$; hence, the derivation blocks.

2. If $t/impO(t) = equ(z_i, t')$ or $t/impO(t) = equ(t', z_i)$ where t' is a term the root of which is labelled by a construtor symbol, e.g. σ, then we can apply the decomposition-rule for σ. Clearly, this leads to an instantiation of z_i. Since, in this situation, the algorithm for usual unification of terms [23] would apply the rule for 'elimination of variables' and since this elimination rule requires an occur check, we also have to restrict the applicability of the decomposition-rules by an occur check. However, we may only check whether z_i occurs in the $(\Delta \cup \mathcal{V})$-skeleton of t' (note that the $(\Delta \cup \mathcal{V})$-skeleton is called *shell* in [22]) or not. For instance, the $(\Delta \cup \mathcal{V})$-skeleton of the tree $\sigma(\sigma(\sigma(z_1, \alpha), z_2), \sigma(sh(\alpha, z_1), \alpha))$ is the pattern $\sigma(\sigma(\sigma(z_1, \alpha), z_2), \sigma(., \alpha))$. In general, our algorithm would be incomplete if we would check the whole term t', e.g., if we have the following situation: $t' = \sigma(f(\alpha, z_i), \alpha)$ where f is a new function symbol of rank 2, and there exists a rule $f(\alpha, y_1) \rightarrow \alpha$, then $[z_i/\sigma(\alpha, \alpha)]$ is an $E_{\mathcal{R}}$-unifier of z_i and t' which would not be computed if we would apply the occur check to the whole term t', because z_i occurs in t'.

3. If $t/impO(t) = equ(z_i, z_j)$ for two variables z_i and z_j, then, using the leftmost outermost narrowing relation associated with $\widehat{\mathcal{R}}$ in a naive way, (t, φ) derives

to $(\varphi_s(t[impO(t) \leftarrow s]), \varphi \circ \varphi_s)$ for every $s \in T\langle\Delta\rangle$ where $\varphi_s = [z_i/s, z_j/s]$. That means, φ_s would be computed as the most general unifier of z_i and z_j which is certainly wrong. The most general unifier of z_i and z_j is $[z_i/z_k, z_j/z_k]$ where $k-1$ is the maximal index of a free variable in use (cf. [23]). Thus, we define the ulo narrowing relation in such a way that a derivation form (t, φ) with $t/impO(t) = equ(z_i, z_j)$ derives as follows: $t/impO(t)$ is replaced by z_k, every occurrence of z_i and z_j in t is replaced by z_k, and φ is composed with the substitution $[z_i/z_k, z_j/z_k]$.

Before we introduce the ulo narrowing relation, we define some auxiliary notions.

Definition 4.2 Let $\mathcal{R} = (F, \Delta, R)$ be a ctn-trs and let $t \in T\langle\widehat{F} \cup \Delta\rangle(\mathcal{V})$.

- The *important occurrence in t,* denoted by $impO(t)$, is the occurrence $min_{lex}(\{u \in O(t) \mid t[u] = equ\})$.
- t is in *binding mode,* if $t[impO(t)1], t[impO(t)2] \in \mathcal{V}$.
- The $(\Delta \cup \mathcal{V})$*-skeleton of t* is the set

  $$\{u \in O(t) \mid \text{there does not exist any } v \in O(t), v \leq u \text{ and } t[v] \in F\}.$$

- The *occur check for t succeeds,* if the following conditions hold:
  1. t is not in binding mode.
  2. there is an $i \in [2]$ such that $t[impO(t)i] \in \mathcal{V}$ and $t[impO(t)(3-i)] \notin \mathcal{V}$ and there exists an occurrence u in the $(\Delta \cup \mathcal{V})$-skeleton of $t/(impO(t)(3-i))$ such that $t/(impO(t)(3-i))[u] = t[impO(t)i]$. $\bigoplus$

Definition 4.3 Let $\mathcal{R} = (F, \Delta, R)$ be a ctn-trs. The *unification-driven leftmost outermost narrowing relation associated with* $\widehat{\mathcal{R}}$, denoted by $\overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}}$, is defined as follows: for every $t, s \in T\langle\widehat{F} \cup \Delta\rangle(\mathcal{V})$ and $\psi, \psi' \in Sub(\mathcal{V}, \Delta)$: (t, ψ) *derives to* (s, ψ') *by* $\overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}}$, denoted by $(t, \psi) \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}} (s, \psi')$, if $t/impO(t) = equ(t_1, t_2)$ where $t_1, t_2 \in T\langle F \cup \Delta\rangle(\mathcal{V})$ and one of the following four conditions holds:

1. $(t_1[\lambda], t_2[\lambda] \in \Delta$ and $t_1[\lambda] = t_2[\lambda])$ or $(((t_1[\lambda] \in \Delta$ and $t_2[\lambda] \in \mathcal{V})$ or $(t_1[\lambda] \in \mathcal{V}$ and $t_2[\lambda] \in \Delta))$ and the occur check fails for $t)$ and the following three conditions hold:
   - (a) $(equ(t_1, t_2), \varphi_\emptyset) \overset{lo}{\rightsquigarrow}_{\mathcal{R}(\Delta)} (t', \varphi')$.
   - (b) $s = \varphi'(t[impO(t) \leftarrow t'])$.
   - (c) $\psi' = \psi \circ \varphi'$.
2. - $t_1, t_2 \in \mathcal{V}$, $t_1 \neq t_2$, and the following three conditions hold where $k = min\{i \mid z_i \in \mathcal{V} \backslash (\mathcal{V}(t) \cup \mathcal{D}(\psi) \cup \mathcal{I}(\psi))\}$:
     - (a) $\varphi' = [t_1/z_k, t_2/z_k]$.
     - (b) $s = \varphi'(t[impO(t) \leftarrow z_k])$.
     - (c) $\psi' = \psi \circ \varphi'$.

- $t_1, t_2 \in \mathcal{V}$, $t_1 = t_2$, and the following two conditions hold:
  (a) $s = t[impO(t) \leftarrow t_1]$.
  (b) $\psi' = \psi$.

3. $t_1[\Lambda] \in F$ and the following three conditions hold:
   (a) $(t_1, \varphi_\emptyset) \overset{lo}{\leadsto}_{\mathcal{R}} (t', \varphi')$.
   (b) $s = \varphi'(t[impO(t)1 \leftarrow t'])$.
   (c) $\psi' = \psi \circ \varphi'$.
4. $t_1[\Lambda] \notin F$ and $t_2[\Lambda] \in F$ and the following three conditions hold:
   (a) $(t_2, \varphi_\emptyset) \overset{lo}{\leadsto}_{\mathcal{R}} (t', \varphi')$.
   (b) $s = \varphi'(t[impO(t)2 \leftarrow t'])$.
   (c) $\psi' = \psi \circ \varphi'$. $\oplus$

If a rule $l \to r \in \widehat{\mathcal{R}}$ is applied, i.e., in cases 1, 3, and 4, we write $\overset{u}{\leadsto}_{\widehat{\mathcal{R}}, l \to r}$. In case 2 we write $\overset{u}{\leadsto}_{\widehat{\mathcal{R}}, bm}$ to indicate that the current term is in binding mode.

In the following example we show three derivations by the ulo narrowing relation which illustrate the involved occur check.

Example 4.4 Consider the ctn-trs $\mathcal{R}_1$ and its extension $\widehat{\mathcal{R}}_1$ (cf. Figure 5).
(a) Consider the terms $sh(z_1, \sigma(\alpha, z_2))$ and $\sigma(mi(z_1), \sigma(z_2, \alpha))$. A possible derivation by $\overset{u}{\leadsto}_{\widehat{\mathcal{R}}_1}$ runs as follows:

$$\begin{array}{ll}
 & (equ(sh(z_1, \sigma(\alpha, z_2)), \sigma(mi(z_1), \sigma(z_2, \alpha))), \varphi_\emptyset) \\
\overset{u}{\leadsto}_{\widehat{\mathcal{R}}_1, 1, (1)} & (equ(\sigma(\alpha, z_2), \sigma(mi(\alpha), \sigma(z_2, \alpha))), [z_1/\alpha]) \\
\overset{u}{\leadsto}_{\widehat{\mathcal{R}}_1, \Lambda, (6)} & (\sigma(equ(\alpha, mi(\alpha)), equ(z_2, \sigma(z_2, \alpha))), [z_1/\alpha]) \\
\overset{u}{\leadsto}_{\widehat{\mathcal{R}}_1, 12, (3)} & (\sigma(equ(\alpha, \alpha), equ(z_2, \sigma(z_2, \alpha))), [z_1/\alpha]) \\
\overset{u}{\leadsto}_{\widehat{\mathcal{R}}_1, 1, (5)} & (\sigma(\alpha, \underline{equ(z_2, \sigma(z_2, \alpha))}), [z_1/\alpha])
\end{array}$$

Here the derivation stops, because the occur check succeeds.

(b) Consider the terms $sh(z_1, \sigma(\alpha, z_2))$ and $\sigma(mi(z_1), \sigma(z_3, \alpha))$. The first four derivation steps are analogous to those one in (a).

$$\begin{array}{ll}
 & (equ(sh(z_1, \sigma(\alpha, z_2)), \sigma(mi(z_1), \sigma(z_3, \alpha))), \varphi_\emptyset) \\
\overset{u}{\leadsto}{}^4_{\widehat{\mathcal{R}}_1} & (\sigma(\alpha, \underline{equ(z_2, \sigma(z_3, \alpha))}), [z_1/\alpha]) \\
\overset{u}{\leadsto}_{\widehat{\mathcal{R}}_1, 2, (6)} & (\sigma(\alpha, \sigma(equ(z_4, z_3), equ(z_5, \alpha))), [z_1/\alpha, z_2/\sigma(z_4, z_5)]) \\
\overset{u}{\leadsto}_{\widehat{\mathcal{R}}_1, 21, bm} & (\sigma(\alpha, \sigma(z_6, equ(z_5, \alpha))), [z_1/\alpha, z_2/\sigma(z_6, z_5), z_3/z_6]) \\
\overset{u}{\leadsto}_{\widehat{\mathcal{R}}_1, 22, (5)} & (\sigma(\alpha, \sigma(z_6, \alpha)), [z_1/\alpha, z_2/\sigma(z_6, \alpha), z_3/z_6])
\end{array}$$

Here the derivation yields the $E_{\mathcal{R}_1}$-unifier $[z_1/\alpha, z_2/\sigma(z_6,\alpha), z_3/z_6]$.

(c) Now enrich $\mathcal{R}_1$ by the rules $sh(\beta, y) \to \beta$ (with number (7)) and $mi(\beta) \to \beta$ (with number (8)) where $\beta \in \Delta^{(0)}$. Denote this ctn-trs by $\mathcal{R}_2$ and its extension by $\widehat{\mathcal{R}}_2$ where the decomposition-rule for β has the number (9). Consider the terms z_1 and $\sigma(\alpha, sh(z_2, z_1))$.

$$\begin{array}{lll}
 & & (equ(z_1, \sigma(\alpha, sh(z_2, z_1))), \varphi_\emptyset) \\
 & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_2,A,(6)} & (\sigma(equ(z_3,\alpha), equ(z_4, sh(z_2, \sigma(z_3, z_4)))), [z_1/\sigma(z_3,z_4)]) \\
 & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_2,1,(5)} & (\sigma(\alpha, equ(z_4, sh(z_2, \sigma(\alpha, z_4)))), [z_1/\sigma(\alpha, z_4)]) \\
(*) & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_2,22,(7)} & (\sigma(\alpha, equ(z_4, \beta)), [z_1/\sigma(\alpha, z_4), z_2/\beta]) \\
 & \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}_2,2,(9)} & (\sigma(\alpha,\beta), [z_1/\sigma(\alpha,\beta), z_2/\beta])
\end{array}$$

Hence, this derivation yields the $E_{\mathcal{R}_2}$-unifier $[z_1/\sigma(\alpha,\beta), z_2/\beta]$. Note that at (*) the occur check is only applied to the $(\Delta \cup \mathcal{V})$-skeleton of $sh(z_2, \sigma(\alpha, z_4))$. $\bigoplus$

4.2 Unification by $\overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}$

As an intermediate result between Theorem 3.7 and the intended uu-algorithm in Theorem 4.7 which is based on the ulo narrowing relation, we show in this subsection that the usual unification of two terms $t, s \in T\langle\Delta\rangle(\mathcal{V})$ can be realized by a derivation by the ulo narrowing relation associated with $\mathcal{R}(\Delta)$.

Lemma 4.5 Let $\mathcal{R} = (F, \Delta, R)$ be a ctn-trs and let $t, s \in T\langle\Delta\rangle(\mathcal{V})$. The terms t and s are unifiable with most general unifier φ iff there exists a derivation by $\overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}$ of the following form $(equ(t,s), \varphi_\emptyset) \overset{u}{\rightsquigarrow}{}^{*}_{\mathcal{R}(\Delta)} (t', \varphi)$ and $t' \in T\langle\Delta\rangle(\mathcal{V})$.

Proof: For the usual term unification, we consider the algorithm in [12] which transforms sets of unordered pairs. Let us briefly recall this algorithm. The unification of t and s starts with the set $P = \{\langle t, s\rangle\}$. Then, a finite number of transformations is applied step by step to this set. Every transformation is of one of the following three types:

1. If $\langle z_i, z_i\rangle \in P$, then P is transformed into the set $P\backslash\{\langle z_i, z_i\rangle\}$.
2. If $\langle\sigma(t_1, \ldots, t_k), \sigma(s_1, \ldots, s_k)\rangle \in P$, then P is transformed into the set $P\backslash\{\langle\sigma(t_1, \ldots, t_k), \sigma(s_1, \ldots, s_k)\rangle\} \cup \{\langle t_1, s_1\rangle, \ldots, \langle t_k, s_k\rangle\}$.
3. If $\langle z_i, s\rangle \in P$ such that z_i does not occur in s, then P is transformed into $\varphi(P\backslash\{\langle z_i, s\rangle\}) \cup \{\langle z_i, s\rangle\}$, where $\varphi = [z_i/s]$ and the φ-image of a set is defined as the set of the φ-images of its elements.

The algorithm stops, if P is in solved form, i.e., $P = \{\langle z_i, t_i\rangle \mid i \in [n]\}$ where for every $i, j \in [n] : z_i \neq z_j$ for $i \neq j$ and z_i does not occur in any t_j. Then, $[z_1/t_1, \ldots, z_n/t_n]$ is the most general unifier of t and s.

Let us note that the algorithm computes the same unifier (modulo variable renaming) if a strategy is imposed on the order in which the transformation steps are applied. Thus, we can choose the order which corresponds to leftmost outermost narrowing by $\overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}$.

Each transformation of the unification algorithm corresponds to the following derivations by $\overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}$ where $(t, \varphi) \in T\langle\Delta\rangle(\mathcal{V}) \times Sub(\mathcal{V}, \Delta)$.

1. A transformation of type 1 corresponds to the derivation step $(t, \varphi) \overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)} (t[impO(t) \leftarrow z_i], \varphi)$, because $t/impO(t) = equ(z_i, z_i)$. Then, the substitution φ is not changed.

2. A transformation of type 2 corresponds to the derivation step $(t, \varphi) \overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)} (t', \varphi)$, where $t' = t[impO(t) \leftarrow \sigma(equ(t_1, s_1), \ldots, equ(t_k, s_k))]$ and φ is not changed, because $t/impO(t) = equ(\sigma(t_1, \ldots, t_k), \sigma(s_1, \ldots, s_k))$. Thus, an application of an decomposition-rule covers the transformation of type 2.

3. The correspondence of a transformation of type 3 is split up into two cases.

   Case 1: If $s \notin \mathcal{V}$, then the transformation corresponds to the derivation $(t, \varphi) \overset{u}{\rightsquigarrow}{}^{*}_{\mathcal{R}(\Delta)} (t', \varphi \circ [z_i/s])$, where t' is the term that results from t by replacing every occurrence of z_i by s. The length of this derivation is $size(s)$, because decomposition-rules are applied node by node in s. Note that the applicability of decomposition-rules is subjected to an occur check (cf. Definition 4.3 1.).

   Case 2: If $s = z_j$ with $j \neq i$, then t is in binding form and the transformation corresponds the derivation step $(t, \varphi) \overset{u}{\rightsquigarrow}{}^{*}_{\mathcal{R}(\Delta),bm} (t', \varphi \circ [z_i/z_k, z_i/z_k])$ where z_k is a new variable.

Conversely, in the definition of $\overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}$, there occurs exactly one of the cases 1, 2, 3.1, and 3.2. In every of these cases, the derivation step by $\overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}$ corresponds to the transformation of the unification algorithm which is mentioned above. $\oplus$

The unification of the terms $t = \sigma(z_1, z_2)$ and $s = \sigma(\sigma(z_2, \alpha), \alpha)$ via a derivation by $\overset{u}{\rightsquigarrow}_{\mathcal{R}_1(\Delta_1)}$ is shown in Figure 6 (for $\mathcal{R}_1$ and Δ_1 cf. Figure 2). The most general unifier is $\theta = [z_1/\sigma(\alpha, \alpha), z_2/\alpha]$.

Now we rephrase Theorem 3.7 by replacing the unification by a derivation induced by $\overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}$.

Theorem 4.6 Let $\mathcal{R} = (F, \Delta, R)$ be a ctn-trs. Let $t, s \in T\langle F \cup \Delta\rangle(\mathcal{V})$, and let V be the set $\mathcal{V}(t) \cup \mathcal{V}(s)$. Let S be the set of all $(\mathcal{V}, \Delta)$-substitutions φ such that φ is in S iff there exists a derivation by $\overset{lo}{\rightsquigarrow}_{\mathcal{R}}$:

$$\begin{aligned}(equ(t, s), \varphi_\emptyset) &\overset{lo}{\rightsquigarrow}_{\mathcal{R}} (equ(t_1, s_1), \varphi_1)\\ &\overset{lo}{\rightsquigarrow}_{\mathcal{R}} (equ(t_2, s_2), \varphi_2) \overset{lo}{\rightsquigarrow}_{\mathcal{R}} \cdots \overset{lo}{\rightsquigarrow}_{\mathcal{R}} (equ(t_n, s_n), \varphi_n),\end{aligned}$$

$$\begin{array}{ll}
 & (equ(\sigma(z_1,z_2),\sigma(\sigma(z_2,\alpha),\alpha)),\varphi_\emptyset) \\
\overset{u}{\leadsto}_{\mathcal{R}_1(\Delta_1),(6)} & (\sigma(equ(z_1,\sigma(z_2,\alpha)),equ(z_2,\alpha)),\varphi_\emptyset) \\
\overset{u}{\leadsto}_{\mathcal{R}_1(\Delta_1),(6)} & (\sigma(\sigma(equ(z_3,z_2),equ(z_4,\alpha)),equ(z_2,\alpha)),[z_1/\sigma(z_3,z_4)]) \\
\overset{u}{\leadsto}_{\mathcal{R}_1(\Delta_1),bm} & (\sigma(\sigma(z_5,equ(z_4,\alpha)),equ(z_5,\alpha)),[z_1/\sigma(z_5,z_4),z_2/z_5]) \\
\overset{u}{\leadsto}_{\mathcal{R}_1(\Delta_1),(5)} & (\sigma(\sigma(z_5,\alpha),equ(z_5,\alpha)),[z_1/\sigma(z_5,\alpha),z_2/z_5]) \\
\overset{u}{\leadsto}_{\mathcal{R}_1(\Delta_1),(5)} & (\sigma(\sigma(\alpha,\alpha),\alpha),[z_1/\sigma(\alpha,\alpha),z_2/\alpha])
\end{array}$$

Figure 6: A unification by a derivation by $\overset{u}{\leadsto}_{\mathcal{R}_1(\Delta_1)}$.

where for every $i \in [n]$: φ_i is in normal form, t_n and s_n are in normal form, and there exists a derivation by $\overset{u}{\leadsto}_{\mathcal{R}(\Delta)}$:

$$(equ(t_n,s_n),\varphi_n) \overset{u}{\leadsto}{}^*_{\mathcal{R}(\Delta)} (t',\varphi'),$$

such that $t' \in T\langle\Delta\rangle(\mathcal{V})$ and $\varphi = \varphi'|_V$. Then S is a ground complete set of $(E_\mathcal{R},\Delta)$-unifiers of t and s away from V.

<u>*Proof:*</u> The correctness of Theorem 4.6 immediately follows from Theorem 3.7 and from Lemma 4.5. $\oplus$

4.3 $E_\mathcal{R}$-Unification by $\overset{u}{\leadsto}_{\hat{\mathcal{R}}}$

We finish this section by showing that we can compute a ground complete set of $(E_\mathcal{R},\Delta)$-unifiers of two terms t and s by derivations induced by the ulo narrowing relation.

Theorem 4.7 Let $\mathcal{R} = (F,\Delta,R)$ be a ctn-trs. Let $t,s \in T\langle F \cup \Delta\rangle(\mathcal{V})$, and let V be the set $\mathcal{V}(t) \cup \mathcal{V}(s)$. Let S be the set of all $(\mathcal{V},\Delta)$-substitutions φ such that φ is in S iff there exists a derivation by $\overset{u}{\leadsto}_{\hat{\mathcal{R}}}$:

$$(equ(t,s),\varphi_\emptyset) \overset{u}{\leadsto}_{\hat{\mathcal{R}}} (t_1,\varphi_1) \overset{u}{\leadsto}_{\hat{\mathcal{R}}} (t_2,\varphi_2) \overset{u}{\leadsto}_{\hat{\mathcal{R}}} \cdots \overset{u}{\leadsto}_{\hat{\mathcal{R}}} (t_n,\varphi_n),$$

where for every $i \in [n]$: φ_i is in normal form, $t_n \in T\langle\Delta\rangle(\mathcal{V})$, and $\varphi = \varphi_n|_V$. Then S is a ground complete set of $(E_\mathcal{R},\Delta)$-unifiers of t and s away from V.

<u>*Proof:*</u> We show that there exists a derivation

$$(equ(t,s),\varphi_\emptyset) \overset{lo}{\leadsto}{}^*_{\mathcal{R}} (equ(t',s'),\varphi') \overset{u}{\leadsto}{}^*_{\mathcal{R}(\Delta)} (t^*,\varphi^*), \tag{1}$$

where $t',s',t^* \in T\langle\Delta\rangle(\mathcal{V})$ and $\varphi',\varphi^* \in Sub(\mathcal{V},\Delta)$ iff there exists a derivation

$$(equ(t,s),\varphi_\emptyset) \overset{u}{\leadsto}{}^*_{\hat{\mathcal{R}}} (t^*,\varphi^*) \tag{2}$$

Then from Theorem 4.6 the correctness of the present theorem follows.

Derivation 1 $\Longrightarrow$ Derivation 2

First, we show that for every derivation 1, there exists a derivation 2. For this purpose, we introduce the function *eqpos* : $T\langle F \cup \Delta\rangle(\mathcal{V}) \times T\langle F \cup \Delta\rangle(\mathcal{V}) \rightarrow \mathbb{N}$ that yields, for two terms t_1 and t_2, the maximal number of steps which can be performed by $\overset{u}{\leadsto}_{\mathcal{R}(\Delta)}$ on the term $equ(t_1,t_2)$. In order to describe this function, we first have to find out the first occurrence $notequ(t_1,t_2)$ in t_1 or t_2 at which no decomposition-rule is applicable; $notequ(t_1,t_2)$ is defined by

$$\begin{aligned} min_{lex}(\{u \in O(t_1) \cup O(t_2) \mid\ & t_1[u] \in F \text{ or } t_2[u] \in F \text{ or} \\ & (t_1[u] \in \Delta \text{ and } t_2[u] \in \Delta \text{ and } t_1[u] \neq t_2[u]) \text{ or} \\ & \text{the occur check for } equ(t_1[u], t_2[u]) \text{ succeeds}\}) \end{aligned}$$

Then $eqpos(t_1,t_2)$ is defined by summing up the number of possible applications of decomposition-rules at occurrences which are common to t_1 and t_2.

$$\sum_{\{u \in O(t_1) \cap O(t_2)\ \mid\ u <_{lex} notequ(t_1,t_2)\}} equsteps(t_1,t_2,u)$$

$equsteps(t_1,t_2,u)$ is the number of possible applications of decomposition-rules at occurrence u. Let $t' = t_{3-i}/u$.

$$equsteps(t_1,t_2,u) = \begin{cases} 1 & \text{if } t_1[u], t_2[u] \in \Delta \text{ and } t_1[u] = t_2[u] \\ 1 & \text{if } t_1[u], t_2[u] \in \mathcal{V} \\ n & \text{if, for some } i \in [2] : t_i[u] \in \mathcal{V}, t' \in T\langle\Delta\rangle(\mathcal{V})\backslash\mathcal{V} \\ & \text{and } n = card(O(t')) \\ n & \text{if, for some } i \in [2] : t_i[u] \in \mathcal{V}, \\ & t' \in T\langle F \cup \Delta\rangle(\mathcal{V})\backslash T\langle\Delta\rangle(\mathcal{V}) \text{ and } n = \\ & card(\{w \in O(t') \mid w <_{lex} min_{lex}(\{v \mid t'[v] \in F\})\}) \end{cases}$$

To give an example, consider the following two terms $t_1 = \sigma(\sigma(\sigma(\alpha,\alpha),z_1), \sigma(f(\alpha),\alpha))$ and $t_2 = \sigma(\sigma(z_2,z_3), \sigma(\sigma(\alpha,\alpha),\alpha))$ (in Figure 7, the occurrences at which a decomposition-rule is applicable, are enclosed).

Obviously, $notequ(t_1,t_2) = 21$. Hence, $eqpos(t_1,t_2) = equsteps(t_1,t_2,\Lambda) + equsteps(t_1,t_2,1) + equsteps(t_1,t_2,11) + equsteps(t_1,t_2,12) + equsteps(t_1,t_2,2)$.

And $equsteps(t_1,t_2,\Lambda) = equsteps(t_1,t_2,1) = equsteps(t_1,t_2,12) = equsteps(t_1,t_2,2) = 1$, and $equsteps(t_1,t_2,11) = 3$. Thus, $eqpos(t_1,t_2) = 7$. This means that, starting from $equ(t_1,t_2)$, it is possible to perform exactly 7 applications of some decomposition-rule. The result after application of 7 decomposition-rules is the term

$$\sigma(\sigma(\sigma(\alpha,\alpha),z_4),\sigma(equ(f(\alpha),\sigma(\alpha,\alpha)),equ(\alpha,\alpha)))$$

which is shown in Figure 8, where z_4 results from the handling of the binding mode.

Furthermore, we prove the following Claim by induction on k.

Claim 1 For every $k \geq 0$, $\varsigma_t, \varsigma_s \in T\langle F \cup \Delta\rangle(\mathcal{V})$, $\varsigma \in T\langle \widehat{F} \cup \Delta\rangle(\mathcal{V})$, and for every $\varphi, \psi \in Sub(\mathcal{V},\Delta)$: If there exists a derivation

$$(equ(t,s),\varphi_\emptyset) \overset{lo}{\leadsto}{}^{k}_{\mathcal{R}} (equ(\varsigma_t,\varsigma_s),\varphi) \overset{u}{\leadsto}{}^{eqpos(\varsigma_t,\varsigma_s)}_{\mathcal{R}(\Delta)} (\varsigma,\psi),$$

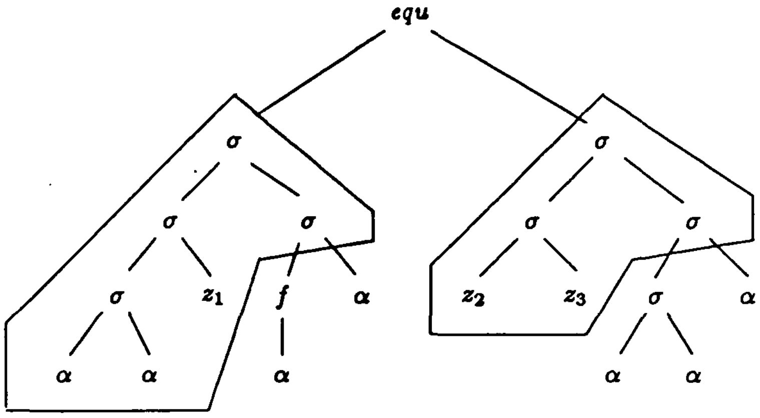


Figure 7: The term $equ(t_1, t_2)$.

then there exists a derivation

$$(equ(t,s),\varphi_\emptyset) \overset{u}{\rightsquigarrow}_{\widehat{R}}^{k+eqpos(\varsigma_t,\varsigma_s)} (\varsigma,\psi).$$

Induction on k:

$\underline{k=0:}$ $\varsigma_t = t$ and $\varsigma_s = s$. We have $(equ(t,s),\varphi_\emptyset) \overset{u}{\rightsquigarrow}_{R(\Delta)}^{eqpos(\varsigma_t,\varsigma_s)} (\varsigma,\psi)$.
From $R(\Delta) \subseteq \widehat{R}$ follows $(equ(t,s),\varphi_\emptyset) \overset{u}{\rightsquigarrow}_{\widehat{R}}^{eqpos(\varsigma_t,\varsigma_s)} (\varsigma,\psi)$.

$\underline{k \to k+1:}$ There exist $\varsigma_t', \varsigma_s' \in T\langle F \cup \Delta\rangle(\mathcal{V})$, $\varsigma' \in T\langle \widehat{F} \cup \Delta\rangle(\mathcal{V})$, $\varphi', \psi' \in Sub(\mathcal{V},\Delta)$, and there exists the following derivation:

$$(equ(t,s),\varphi_\emptyset) \overset{lo}{\rightsquigarrow}_{R}^{k} (equ(\varsigma_t,\varsigma_s),\varphi) \overset{lo}{\rightsquigarrow}_{R} (equ(\varsigma_t',\varsigma_s'),\varphi') \overset{u}{\rightsquigarrow}_{R(\Delta)}^{eqpos(\varsigma_t',\varsigma_s')} (\varsigma',\psi').$$

Now we split the derivation by $\overset{u}{\rightsquigarrow}_{R(\Delta)}$ into two derivations: There exist $\bar{\varsigma} \in T\langle \widehat{F} \cup \Delta\rangle(\mathcal{V})$, $\bar{\varphi} \in Sub(\mathcal{V},\Delta)$, and there exists the following derivation:

$$(equ(t,s),\varphi_\emptyset) \overset{lo}{\rightsquigarrow}_{R}^{k} (equ(\varsigma_t,\varsigma_s),\varphi) \overset{lo}{\rightsquigarrow}_{R} (equ(\varsigma_t',\varsigma_s'),\varphi') \overset{u}{\rightsquigarrow}_{R(\Delta)}^{eqpos(\varsigma_t,\varsigma_s)} (\bar{\varsigma},\bar{\varphi})$$
$$\overset{u}{\rightsquigarrow}_{R(\Delta)}^{eqpos(\varsigma_t',\varsigma_s')-eqpos(\varsigma_t,\varsigma_s)} (\varsigma',\psi').$$

There exist $\bar{\varsigma}' \in T\langle \widehat{F} \cup \Delta\rangle(\mathcal{V})$, $\bar{\varphi}' \in Sub(\mathcal{V},\Delta)$, and there exists the following derivation by changing the order of applications of rules in the previous derivation:

$$(equ(t,s),\varphi_\emptyset) \overset{lo}{\rightsquigarrow}_{R}^{k} (equ(\varsigma_t,\varsigma_s),\varphi) \overset{u}{\rightsquigarrow}_{R(\Delta)}^{eqpos(\varsigma_t,\varsigma_s)} (\bar{\varsigma}',\bar{\varphi}') \overset{u}{\rightsquigarrow}_{\widehat{R}} (\bar{\varsigma},\bar{\varphi})$$
$$\overset{u}{\rightsquigarrow}_{R(\Delta)}^{eqpos(\varsigma_t',\varsigma_s')-eqpos(\varsigma_t,\varsigma_s)} (\varsigma',\psi').$$

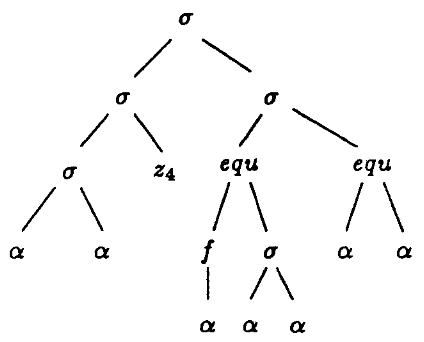


Figure 8: Resulting term t^* after application of seven decomposition-rules.

Changing the order of the derivation is correct, because in the derivation step $(equ(\varsigma_t, \varsigma_s), \varphi) \overset{lo}{\rightsquigarrow}_{\mathcal{R}} (equ(\varsigma_t', \varsigma_s'), \varphi')$, a function f is applied at the leftmost outermost narrowing occurrence. From the definition of *eqpos* it follows that f is also the label of the leftmost outermost narrowing occurrence in $\vec{\varsigma}'$. Furthermore, in the case of a function application, the relations $\overset{lo}{\rightsquigarrow}_{\mathcal{R}}$ and $\overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}}$ yield the same result.

<u>*Example:*</u> Let $\varsigma_t = t_1$ and $\varsigma_s = t_2$ in Figure 7 and let $f(\alpha) \to \alpha$ be a rule in R. The leftmost outermost narrowing occurrence in $equ(t_1, t_2)$ in Figure 7 is occurrence 121 which is labelled by the function symbol f. After $eqpos(t_1, t_2) = 7$ applications of decomposition-rules we get the term t^* in Figure 8 which is denoted by $\vec{\varsigma}'$ in the proof. The leftmost outermost narrowing occurrence in t^* is the occurrence 211 which is also labelled by f. Furthermore, the next step in the derivation by $\overset{lo}{\rightsquigarrow}_{\mathcal{R}}$ starting with $equ(t_1, t_2)$ in Figure 7 is analogous to the next step in the derivation by $\overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}}$ starting with t^* in Figure 8. □

The existence of the following derivation follows from the induction hypothesis:

$$(equ(t,s), \varphi_\emptyset) \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}}^{k+eqpos(\varsigma_t,\varsigma_s)} (\vec{\varsigma}', \bar{\varphi}') \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}} (\bar{\varsigma}, \bar{\varphi}) \overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}^{eqpos(\varsigma_t',\varsigma_s')-eqpos(\varsigma_t,\varsigma_s)} (\varsigma', \psi').$$

The existence of the following derivation follows from $R(\Delta) \subseteq \widehat{R}$:

$$(equ(t,s), \varphi_\emptyset) \overset{u}{\rightsquigarrow}_{\widehat{\mathcal{R}}}^{k+1+eqpos(\varsigma_t',\varsigma_s')} (\varsigma', \psi').$$

This finishes the proof of Claim 1.

Especially, if k is equal to the length of the derivation by $\overset{lo}{\rightsquigarrow}_{\mathcal{R}}$ in derivation 1, it follows that for every derivation 1, there exists a derivation 2.

Derivation 2 $\Longrightarrow$ Derivation 1

Now we show that for every derivation 2, there exists a derivation 1. For this purpose, we introduce the function *eqapp* : $T\langle\hat{F} \cup \Delta\rangle(\mathcal{V}) \rightarrow \mathbb{N}$ that yields, for a term t, the sum of applications of decomposition-rules and steps started by a term in binding mode, in the derivation by $\overset{u}{\leadsto}_{\hat{\mathcal{R}}}$ up to t.

$$eqapp(t) = card(\{u \in O(t) \mid u <_{lex} impO(t)\})$$

Furthermore, we prove the following claim by induction on k.

Claim 2 For every $k \geq 0$, $\varsigma \in T\langle\hat{F} \cup \Delta\rangle(\mathcal{V})$, and $\psi \in Sub(\mathcal{V}, \Delta)$: If there exists a derivation

$$(equ(t,s), \varphi_\emptyset) \overset{u}{\leadsto}{}^{k}_{\hat{\mathcal{R}}} (\varsigma, \psi),$$

then there exist $\varsigma_t, \varsigma_s \in T\langle F \cup \Delta\rangle(\mathcal{V})$, $\varphi \in Sub(\mathcal{V}, \Delta)$, and there exists a derivation

$$(equ(t,s), \varphi_\emptyset) \overset{lo}{\leadsto}{}^{k-eqapp(\varsigma)}_{\mathcal{R}} (equ(\varsigma_t, \varsigma_s), \varphi) \overset{u}{\leadsto}{}^{eqapp(\varsigma)}_{\mathcal{R}(\Delta)} (\varsigma, \psi).$$

<u>*Induction on k:*</u>
<u>$k = 0$:</u> $\varsigma = equ(t,s)$, $\psi = \varphi_\emptyset$. Thus, $eqapp(\varsigma) = 0$. We have

$$(equ(t,s), \varphi_\emptyset) \overset{lo}{\leadsto}{}^{0-0}_{\mathcal{R}} (equ(\varsigma_t, \varsigma_s), \varphi_\emptyset) \overset{u}{\leadsto}{}^{0}_{\mathcal{R}(\Delta)} (\varsigma, \psi).$$

<u>$k \rightarrow k+1$:</u> There exist $\varsigma' \in T\langle\hat{F} \cup \Delta\rangle(\mathcal{V})$, $\psi' \in Sub(\mathcal{V}, \Delta)$, and there exists the following derivation:

$$(equ(t,s), \varphi_\emptyset) \overset{u}{\leadsto}{}^{k}_{\hat{\mathcal{R}}} (\varsigma, \psi) \overset{u}{\leadsto}_{\hat{\mathcal{R}}} (\varsigma', \psi').$$

From the induction hypothesis it follows that there exists the following derivation:

$$(equ(t,s), \varphi_\emptyset) \overset{lo}{\leadsto}{}^{k-eqapp(\varsigma)}_{\mathcal{R}} (equ(\varsigma_t, \varsigma_s), \varphi) \overset{u}{\leadsto}{}^{eqapp(\varsigma)}_{\mathcal{R}(\Delta)} (\varsigma, \psi) \overset{u}{\leadsto}_{\hat{\mathcal{R}}} (\varsigma', \psi').$$

Now we have to distinguish the following two cases:

<u>*Case 1 :*</u> $eqapp(\varsigma') = eqapp(\varsigma)$. Then, the $k+1$th derivation step is a function application. The same function application can be applied to the term $equ(\varsigma_t, \varsigma_s)$ in a derivation step by $\overset{lo}{\leadsto}_{\mathcal{R}}$.

<u>*Example:*</u> Let $\varsigma_t = t_1$ and $\varsigma_s = t_2$ in Figure 7 and let ς be the term t^* in Figure 8; $eqapp(t^*) = 7$. The next derivation step in the derivation by $\overset{u}{\leadsto}_{\hat{\mathcal{R}}}$ starting with t^* is the application of the rule $f(\alpha) \rightarrow \alpha$ which simply replaces the subterm $f(\alpha)$ in t^* by α. The resulting term is denoted by ς' in the proof. The next step in the derivation by $\overset{lo}{\leadsto}_{\mathcal{R}}$ starting with $equ(t_1, t_2)$ is the application of the same rule. □

Furthermore, the $eqapp(\varsigma)$ derivation steps by $\overset{u}{\leadsto}_{\mathcal{R}(\Delta)}$ work only on occurrences that are less with respect to $<_{lex}$ than the occurrence where the function is applied. Thus, there exist $\varsigma_t', \varsigma_s' \in T\langle F \cup \Delta\rangle(\mathcal{V})$, $\varphi' \in Sub(\mathcal{V}, \Delta)$, and there exists a derivation:

$$(equ(t,s), \varphi_\emptyset) \overset{lo}{\rightsquigarrow}_{\mathcal{R}}^{k-eqapp(\varsigma)} (equ(\varsigma_t, \varsigma_s), \varphi) \overset{lo}{\rightsquigarrow}_{\mathcal{R}} (equ(\varsigma_t', \varsigma_s'), \varphi') \overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}^{eqapp(\varsigma')} (\varsigma', \psi').$$

Then we obtain the following derivation:

$$(equ(t,s), \varphi_\emptyset) \overset{lo}{\rightsquigarrow}_{\mathcal{R}}^{k+1-eqapp(\varsigma')} (equ(\varsigma_t', \varsigma_s'), \varphi') \overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}^{eqapp(\varsigma')} (\varsigma', \psi').$$

<u>*Case 2 :*</u> $eqapp(\varsigma') = eqapp(\varsigma) + 1$. One of the cases 1 and 2 in Definition 4.3 is applied in the added step. In these cases $\overset{u}{\rightsquigarrow}_{\hat{\mathcal{R}}}$ exactly works as $\overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}$ (e.g., suppose that the subterm $f(\alpha)$ is replaced by $\sigma(\alpha, \alpha)$ in Figure 8, then the decomposition-rule for σ is applied in the added step.). We get the following derivation:

$$(equ(t,s), \varphi_\emptyset) \overset{lo}{\rightsquigarrow}_{\mathcal{R}}^{k-eqapp(\varsigma)} (equ(\varsigma_t, \varsigma_s), \varphi) \overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}^{eqapp(\varsigma)} (\varsigma, \psi) \overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)} (\varsigma', \psi').$$

From $eqapp(\varsigma') = eqapp(\varsigma) + 1$ follows:

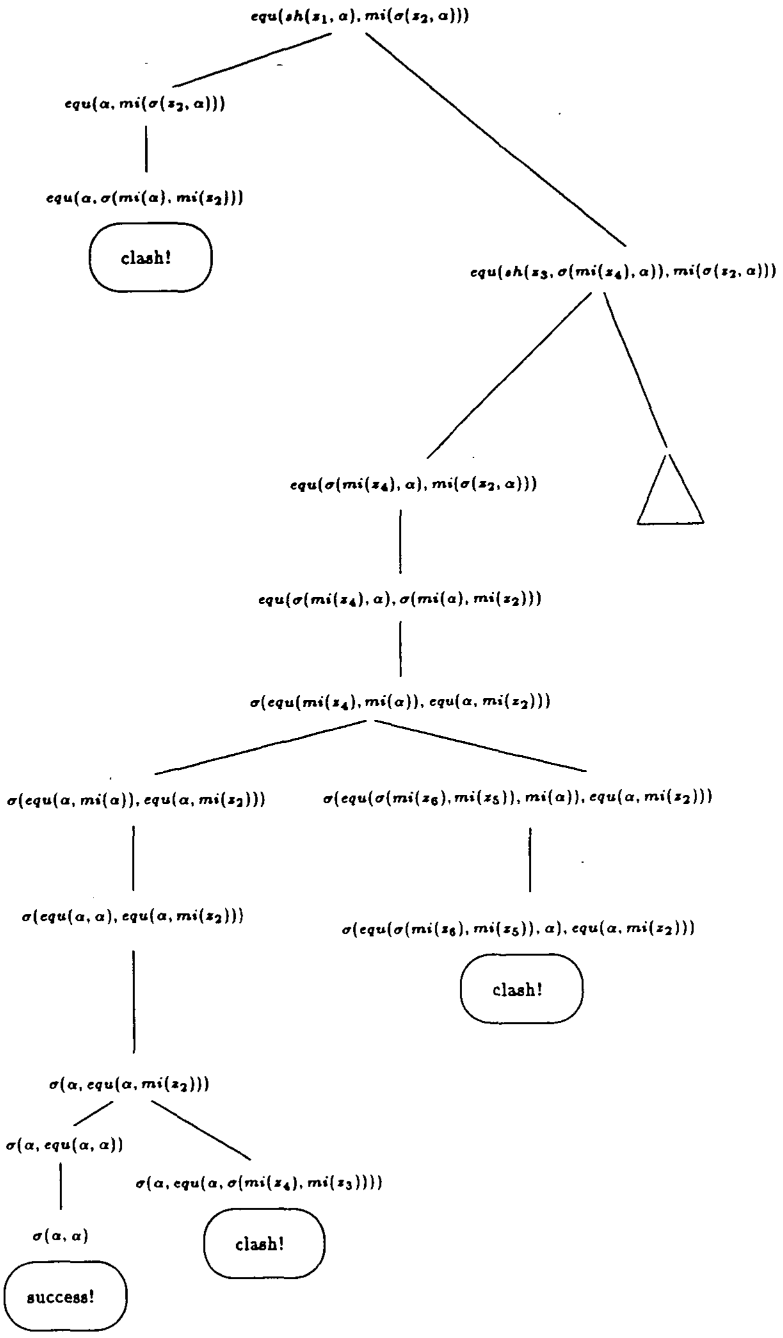
$$(equ(t,s), \varphi_\emptyset) \overset{lo}{\rightsquigarrow}_{\mathcal{R}}^{k-(eqapp(\varsigma')-1)} (equ(\varsigma_t, \varsigma_s), \varphi) \overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}^{eqapp(\varsigma')} (\varsigma', \psi').$$

Here we obtain the following derivation:

$$(equ(t,s), \varphi_\emptyset) \overset{lo}{\rightsquigarrow}_{\mathcal{R}}^{k+1-eqapp(\varsigma')} (equ(\varsigma_t, \varsigma_s), \varphi) \overset{u}{\rightsquigarrow}_{\mathcal{R}(\Delta)}^{eqapp(\varsigma')} (\varsigma', \psi').$$

Especially, if k is equal to the length of the derivation by $\overset{u}{\rightsquigarrow}_{\hat{\mathcal{R}}}$ in derivation 2, it follows that for every derivation 2, there exists a derivation 1. This finishes the proof of Claim 2. $\oplus$

The unification-driven leftmost outermost narrowing tree of $\mathcal{R}_1$ for the $E_{\mathcal{R}_1}$-unification of the terms $sh(z_1, \alpha)$ and $mi(\sigma(z_2, \alpha))$ is shown in Figure 9. At leaves which are labeled by 'clash!', the derivations are stopped by the ulo narrowing relation. Thus, in an intuitive sense, the uu-algorithm induced in Theorem 4.7 is more efficient than the uu-algorithm of Theorem 3.7. (Compare the ulo narrowing tree of Figure 9 with the leftmost outermost narrowing tree in Figure 4.)



$equ(sh(z_1,\alpha), mi(\sigma(z_2,\alpha)))$
$equ(\alpha, mi(\sigma(z_2,\alpha)))$
$equ(\alpha, \sigma(mi(\alpha), mi(z_2)))$
clash!
$equ(sh(z_3, \sigma(mi(z_4),\alpha)), mi(\sigma(z_2,\alpha)))$
$equ(\sigma(mi(z_4),\alpha), mi(\sigma(z_2,\alpha)))$
$equ(\sigma(mi(z_4),\alpha), \sigma(mi(\alpha), mi(z_2)))$
$\sigma(equ(mi(z_4), mi(\alpha)), equ(\alpha, mi(z_2)))$
$\sigma(equ(\alpha, mi(\alpha)), equ(\alpha, mi(z_2)))$
$\sigma(equ(\sigma(mi(z_6), mi(z_5)), mi(\alpha)), equ(\alpha, mi(z_2)))$
$\sigma(equ(\alpha,\alpha), equ(\alpha, mi(z_2)))$
$\sigma(equ(\sigma(mi(z_6), mi(z_5)), \alpha), equ(\alpha, mi(z_2)))$
clash!
$\sigma(\alpha, equ(\alpha, mi(z_2)))$
$\sigma(\alpha, equ(\alpha,\alpha))$
$\sigma(\alpha, equ(\alpha, \sigma(mi(z_4), mi(z_3))))$
$\sigma(\alpha,\alpha)$
clash!
success!

Figure 9: Ulo narrowing tree for $equ(sh(z_1,\alpha), mi(\sigma(z_2,\alpha)))$.

5 Conclusion

In this paper we have formalized a universal unification algorithm for equational theories which are characterized by ctn-trs's. This algorithm is at least as efficient as the algorithm which is implied by Theorem 3 in [3], but sometimes it is more efficient. The universal unification algorithm is based on the unification-driven leftmost outermost narrowing relation which is a combination of leftmost outermost narrowing and unification. It is inspired by the idea of the uu-algorithm in [11] which combines every innermost narrowing strategy with interleaving decomposition steps. The advantage of our uu-algorithm in comparison to the uu-algorithm in [11] is that arguments of function calls are only evaluated on demand which leads to a more efficient algorithm.

The conditions that the considered trs's are canonical and not strictly subunifiable, cannot be weakened, because the uu-algorithm would loose its completeness. Furthermore, the condition that the trs's are constructor-based, cannot be weakened. Otherwise, the decomposition-rules would make no sense. We are not sure, whether the condition that the trs's are totally-defined, can be weakened.

As mentioned in the introduction, there exist a lot of other uu-algorithms which are based on narrowing strategies. But none of them combines the narrowing strategy with interleaving decomposition steps. Thus, nonsuccessful derivations are computed up to the end, whereas they are immediately stopped in our algorithm.

Two implementations of leftmost outermost reduction for special ctn-trs's which are called macro tree transducers [1,4,5], are formalized in [8,13]. A nondeterministic implementation of the universal unification algorithm of the present paper which is an extension of the implementation in [8] by adding features for unification, is presented in [7]. In our current research [9] we construct a deterministic implementation of the universal unification algorithm by adding features for unification and backtracking to the implementation of leftmost outermost reduction shown in [13]. In the deterministic implementation a depth-first left-to-right traversal over the ulo narrowing tree is formalized. Clearly, this implementation does not produce a ground complete set of $(E_{\mathcal{R}}, \Delta)$-unifiers, because otherwise the $(E_{\mathcal{R}}, \Delta)$-unification problem would be decidable. Rather there are three possibilities:

- The machine stops and it has computed one $(E_{\mathcal{R}}, \Delta)$-unifier.
- The machine does not stop.
- The machine stops and it has computed no $(E_{\mathcal{R}}, \Delta)$-unifier. In fact, in this situation, the tree traversal has returned to the root and it is clear that there is no $(E_{\mathcal{R}}, \Delta)$-unifier at all.

As further research investigation, we will generalize the scope of this implementation from macro tree transducers to modular tree transducers [6]. Modular tree transducers are ctn-trs's which compute exactly the class of primitive recursive tree functions.

Acknowledgements

The authors would like to thank Alfons Geser for discussions on preliminary versions. The authors are grateful to the referee for communicating related work to us and for suggesting improvements of a previous version.

References

[1] B. Courcelle and P. Franchi-Zannettacci. Attribute grammars and recursive program schemes. *Theoret. Comput. Sci.*, 17:163–191 and 235–257, 1982.

[2] N. Dershowitz and J.P. Jouannaud. Notations for rewriting. *Bulletin of the EATCS*, 43:162–172, 1991.

[3] R. Echahed. On completeness of narrowing strategies. *CAAP 88, LNCS 299, 89-101*, 1988.

[4] J. Engelfriet. Some open questions and recent results on tree transducers and tree languages. In R.V. Book, editor, *Formal language theory; perspectives and open problems*. New York, Academic Press, 1980.

[5] J. Engelfriet and H. Vogler. Macro tree transducers. *J. Comput. System Sci.*, 31:71–146, 1985.

[6] J. Engelfriet and H. Vogler. Modular tree transducers. *Theoret. Comput. Sci.*, 78:267–304, 1991.

[7] H. Faßbender. Implementation of a universal unification algorithm for macro tree transducers. In *FCT'93*, pages 222–233. Springer-Verlag, 1993. LNCS 710.

[8] H. Faßbender and H. Vogler. An implementation of syntax directed functional programming on nested-stack machines. *Formal Aspects of Computing*, 4:341–375, 1992.

[9] H. Faßbender, H. Vogler, and A. Wedel. Implementation of a partial E–Unification algorithm for macro tree transducers. Technical report, University of Ulm, 1994. in preparation.

[10] M. Fay. First-order unification in an equational theory. In *Proceeding of the 4th workshop on automated deduction, Austin*, pages 161–167, 1979.

[11] L. Fribourg. Slog: A logic programming language interpreter based on clausual superposition and rewriting. In *Proceedings of the IEEE International Symposium on logic programming*, pages 172–184. IEEE Computer Society Press, 1985.

[12] J.H. Gallier and W. Snyder. A general complete E-unification procedure. In P. Lescanne, editor, *Rewriting techniques and applications RTA 87, LNCS 256*, pages 216–227, 1987.

[13] K. Gladitz, H. Faßbender, and H. Vogler. Compiler-based implementation of syntax directed functional programming. Technical Report 91-10, Technical University of Aachen, 1991.

[14] G. Huet. Confluent reductions: abstract properties and applications to term rewriting systems. *J. Assoc. Comput. Mach.*, 27:797–821, 1980.

[15] G. Huet and D.C. Oppen. Equations and rewrite rules: a survey. In R. Book, editor, *Formal Language Theory: Perspectives and Open Problems*. Academic Press, New York, 1980.

[16] J.M. Hullot. Canonical forms and unification. In *Proceedings of the 5th conference on automated deduction, LNCS 87*, pages 318–334. Springer-Verlag, 1980.

[17] U. Hupbach. Rekursive Funktionen in mehrsortigen Algebren. *Elektron. Informationsverarb. Kybernetik*, 15:491–506, 1978.

[18] J. Jouannaud and H. Kirchner. Solving equations in abstract algebras: a rule-based survey of unification. In *Computational Logic. Essays in the honour of Alan Robinson*, pages 257–321. MIT Press, Cambridge, 1991.

[19] C. Kirchner. A new equational unification method: a generalisation of Martelli-Montanari's algorithm. In *Conference on Automated Deduction*, pages 224–247. Springer-Verlag, 1984. LNCS 170.

[20] K. Knight. Unification: a multidisciplinary survey. *ACM Computing Surveys*, 21:93–124, 1989.

[21] D.S. Lankford. Canonical inference. Technical Report ATP-32, Department of Mathematics and Computer Science, University of Texas, 1975.

[22] R. Loogen, F. Lopez-Fraguas, and M. Rodriguez-Artalejo. A demand driven computation strategy for lazy narrowing. In *PLILP'93*, pages 184–200, 1993. LNCS 714.

[23] A. Martelli and U. Montanari. An efficient unification algorithm. *ACM Transactions on Programming Languages Systems*, 4:258–282, 1982.

[24] A. Middeldorp and E. Hamoen. Counterexamples to completeness results for basic narrowing. In H. Kirchner and G. Levi, editors, *Algebraic and Logic Programming*, pages 244–258. Springer-Verlag, 1992. LNCS 632.

[25] P. Padawitz. Strategy-controlled reduction and narrowing. In P. Lescanne, editor, *Rewriting Techniques and Applications*, pages 242–255. Springer-Verlag, 1987. LNCS 256.

[26] U. S. Reddy. Narrowing as the operational semantics of functional languages. *IEEE Comp. Soc. Press 1985, Symp. Log. Progr.*, 1985.

[27] J.A. Robinson. A machine-oriented logic based on the resolution principle. *J. Assoc. Comput. Mach.*, 20:23–41, 1965.

[28] J. H. Siekmann. Unification theory. *J. Symbolic Computation*, 7:207–274, 1989.

[29] J.H. You. Solving equations in an equational language. In *Conference on algebraic and logic programming*, pages 245–254. Springer-Verlag, 1988. LNCS 343.

[30] J.H. You. Enumerating outer narrowing derivations for constructor-based term rewriting systems. *J. Symbolic Computation 7 (1989), 319-341*, 1989.

Received March 5, 1993

Revised September 1, 1993