3 A Particular Matching Problem

We present a problem of matching under a set of equations. The set of equations defines the algebra of actions that we use in our application to contracts. The signature is formed of three functional symbols respecting properties similar to semiring structures with aditional features like commutativity or a restricted form of idempotence. The question that we put is to determine the decidability of our particular matching problem. We survey the existing methods and results in the literature and discuss how they are related to our problem. Our equational theory relates to undecidable problems like unification under commutative rings, or to hard problems like unification under left-right distributivity, or to simple problems like unification under Boolean rings or ACUI theories.

3.1 Termination, Confluence, and Normal Form of $\mathcal{CA}$

We want to have a *normal form* of actions of $\mathcal{CAT}$. For this we transform the axioms of the algebra (of Table 3) into a rewriting system. The rewriting system proves to be not a trivial one and we need to use the latest termination techniques (i.e. dependency pairs termination criterion [AG00]) and rewriting modulo equations (i.e. normalized rewriting [Mar96]) incorporated in the CiME tool [CM96, CMTU05]. In this section we prove the convergence of the rewriting system and give a characterization of the action normal form in terms of pruned trees.

We recall first classical notions and notations for term rewriting systems and equational unification from [BN98].

An *order-sorted signature*, denoted Σ, is a finite set of *functional symbols* with a finite partially prdered set of *sorts* $(S, \leq)$. In our setting consider the signature of the $\mathcal{CA}$ algebra which has three binary functional symbols $(+, \cdot, \&)$, two special constants $\mathbf{1}$, $\mathbf{0}$ (i.e. a constant is a functional symbol of arity 0), and another finite number of special constants $\mathcal{A}_B$ which we called *basic actions*. The ordered sorts were $\mathcal{A}_B \leq \mathcal{A}_B^{\&} \leq \mathcal{A}$ where $\mathcal{A}_B, \mathcal{A}_B^{\&}, \mathcal{A} \in S$ are the three elements (sorts) of the set of sorts S. We assume an S-sorted family $V = \{V_s\}_{s \in S}$ of disjoint sets of variables. $\mathcal{T}(\Sigma, V)$ is the set of *terms*, and $\mathcal{T}(\Sigma)$ is the set of *ground terms*. $V(t)$ with $t \in \mathcal{T}(\Sigma, V)$ a term represents the set of variables occurring in t. The set of ground terms corresponds to the carrier set $\mathcal{A}$ of the algebra $\mathcal{CA}$. We will be working in this section on the term algebra $T_{\mathcal{CA}}$. The set of positions of a term t is denoted $Pos(t)$ and the set of nonvariable positions $Pos_\Sigma(t)$. The subterm of t at position p is denoted $t|_p$ and the replacing of the subterm at position p of the term t with a new term u is denoted $t[u]_p$. A *substitution* ς is a sorted mapping from

a finite subset $Dom(\varsigma)$ of V into $\mathcal{T}(\Sigma, V)$. The set of variables introduced by ς is denoted $Ran(\varsigma)$. Substitutions can be extended homomorphicaly from variables to the whole set of terms. A substitution applied to a term is denoted $t\varsigma$. Composition of two substitutions is denoted $\varsigma\varsigma'$ and when applied to a term $t\varsigma\varsigma'$ it can also be understood as first applying ς and afterwards applying ς'.

An *identity* (or *equation*) is a pair of terms $t = t'$ with $t, t' \in \mathcal{T}(\Sigma, V)$. Intuitively, an identity holds in the term algebra iff the equality of the two terms is true for all the ways of replacing the variables. The *equational theory* induced by a set of identities E is the relation $=_E = \{(t, t') \in \mathcal{T}(\Sigma, V) \times \mathcal{T}(\Sigma, V) \mid E \models t = t'\}$ where the symbol $\models$ represents the *semantics consequence* relation. The axioms of the $\mathcal{CA}$ algebra is a set of identities, and the relation $=_{\mathcal{CA}}$ is a congruence over the set of actions from $\mathcal{A}$.

A *rewrite rule* is a directed identity $l \to r$ satisfying $l \notin V$ and $V(r) \subseteq V(l)$. A *variant* of a rule $l \to r$ is $l\varsigma \to r\varsigma$ where ς is a variable renaming substitution (i.e. a substitution which assigns to each variable from $Dom(\varsigma)$ a fresh variable not from $Dom(\varsigma)$). A *term rewrite system* TRS is a set of rewrite rules. A *rewrite relation* $\rightsquigarrow_R$ associated with a TRS R is defined as: $t \rightsquigarrow_R t'$ iff exists a variant $l \to r$ of a rewrite rule from R, $\exists p \in Pos_\Sigma(t)$, and $\exists\varsigma$ a substitution s.t. $t|_p = l\varsigma$ and $t' = t[r\varsigma]_p$. The transitive-reflexive closure of $\rightsquigarrow_R$ is denoted $\rightsquigarrow_R^*$. We denote by $=_R$ the transitive-reflexive-symmetric closure of $\rightsquigarrow_R$. The same as $=_E$ we have that $=_R$ is a congruence. A *derivation* of a term t is a sequaence $t \rightsquigarrow_R t_1 \rightsquigarrow_R t_2 \rightsquigarrow_R \ldots$ of rewrite rules (sometimes also called *rewrite steps*). A TRS is said to be *terminating* iff there is no infinite derivation from any term. A TRS is *confluent* iff for any terms t, t_1, t_2 with $t \rightsquigarrow_R t_1$ and $t \rightsquigarrow_R t_2$ then $\exists u$ s.t. $t_1 \rightsquigarrow_R^* u$ and $t_2 \rightsquigarrow_R^* u$. A TRS which is both terminating and confluent is called *convergent* (or complete).

We can associate to an equational system E an equivalent TRS R by directing in an arbitrary way the identities from E. It is easy to see that the two systems are equivalent (i.e. $t =_E t'$ iff $t =_R t'$) for any choice of direction of the identities. More important is to find a directioning s.t. R is convergent. Some identities are innerently nonterminating; for example the identity $f(t, t') = f(t', t)$ which defines the commutativity of the functional symbol f. For this purpose it is common to define a rewrite relation *modulo* a set of identities. The relation $\rightsquigarrow_{R/E}$ is the composition of the two relations $=_E \circ \rightsquigarrow_R \circ =_E$ where R is a set of rewrite rules and E is the set of "problematic" identities. Alternatively, $t \rightsquigarrow_{R/E} t'$ iff $\exists s, s' \in \mathcal{T}(\Sigma, V)$ s.t. $t =_E s$, $t' =_E s'$, and $s \rightsquigarrow_R s'$. The notions of termination, confluence, and convergence are defined naturally as for the $\rightsquigarrow_R$ relation. We consider an *order-sorted rewrite*

(1) $\alpha + \mathbf{0} \to \alpha$
(2) $\alpha + \alpha \to \alpha$
(3) $(\alpha \cdot \beta) \cdot \gamma \to \alpha \cdot (\beta \cdot \gamma)$
(4) $\alpha \cdot \mathbf{1} \to \alpha$
(5) $\mathbf{1} \cdot \alpha \to \alpha$
(6) $\alpha \cdot \mathbf{0} \to \mathbf{0}$
(7) $\mathbf{0} \cdot \alpha \to \mathbf{0}$
(8) $\alpha \cdot (\beta + \gamma) \to \alpha \cdot \beta + \alpha \cdot \gamma$
(9) $(\alpha + \beta) \cdot \gamma \to \alpha \cdot \gamma + \beta \cdot \gamma$
(10) $\alpha \& \mathbf{1} \to \alpha$
(11) $\alpha \& \mathbf{0} \to \mathbf{0}$
(12) $a \& a \to a$ for $a \in \mathcal{A}_B$
(13) $\alpha \& (\beta + \gamma) \to \alpha \& \beta + \alpha \& \gamma$
(14) $(\alpha_\& \cdot \alpha) \& (\beta_\& \cdot \beta) \to (\alpha_\& \& \beta_\&) \cdot (\alpha \& \beta)$

Table 4: A TRS equivalent modulo AC to $\mathcal{CA}$ and which is convergent modulo AC.

theory to be $R = (\Sigma, E, R)$ which has associated a rewrite relation $\rightsquigarrow_{R/E}$. The theory is said to be terminating (respectively confluent or convergent) iff the rewrite relation is terminating (respectively confluent or convergent).

For the equational system of the $\mathcal{CA}$ algebra from Table 3 we obtain the equivalent order-sorted rewrite theory $\mathcal{T}_{\mathcal{CA}} = (\Sigma, R_{\mathcal{CA}}, E_{AC+\&})$. The signature Σ is the same signature of the algebra $\mathcal{CA}$. The rules of the TRS $R_{\mathcal{CA}}$ are given in Table 4. The set of identities $E = \{A+, C+, A\&, C\&\}$ is the four axioms which define the associativity and commutativity of the functional operators + and &. The associated rewrite relation is done modulo AC (associativity and commutativity). These kind of rewriting as been well investigated and good algorithms are known for it. Note that the rewrite rule (12) is applied only to terms of sort $\mathcal{A}_B$ (i.e. only to basic actions denoted a). The rule (14) is applied to terms of sort $\mathcal{A}_B^\&$ which we denote by $\alpha_\&$. Terms of general sort $\mathcal{A}$ are denoted with α, β, γ.

To prove termination of the $R_{\mathcal{CA}}$ we use the *dependency pairs* termination criterion [AG00] which is implemented in CiME using the polynomial interpretations method in order to generate the required orderings [CMTU05]. We present now how the dependency pairs termination criterion applies to our term rewriting system $R_{\mathcal{CA}}$.

Let $R_{\mathcal{CA}}$ be a set of rewrite rules. The set of *defined symbols* is $D_{R_{\mathcal{CA}}} = \{root(l) \mid l \to r \in R_{\mathcal{CA}}\}$; and the set of *constructor symbols* is $C_{R_{\mathcal{CA}}} = \Sigma \setminus D_{R_{\mathcal{CA}}}$ ($root(t)$ returns the symbol ot the root of term t). For our con-

crete example $D_{R_{\mathcal{CA}}} = \{+, \cdot, \&\}$ and $C_{R_{\mathcal{CA}}} = \{\mathbf{1}, \mathbf{0}, \} \cup \mathcal{A}_B$. A *dependency pair* is a pair of terms $\langle f(s_1, \ldots, s_n), g(t_1, \ldots, t_m)\rangle$ s.t. $\exists f(s_1, \ldots, s_n) \rightarrow C[g(t_1, \ldots, t_m)] \in R_{\mathcal{CA}}$ a rewrite rule where $C[\,]$ is a context and $f, g \in D_{R_{\mathcal{CA}}}$ are defined symbols. The set of dependency pairs is finite if the TRS is finite (which is in our case).

Theorem 3.1 ([AG00])**.** *A TRS $R_{\mathcal{CA}}$ is terminating iff there exists a* well-founded weakly monotonic quasi-ordering $\geqslant$ *where both $\geqslant$ and $>$ are closed under substitution, s.t.*

- $l \geqslant r$ *for all* $l \rightarrow r \in R_{\mathcal{CA}}$*;*
- $s > t$ *for all dependency pairs* $\langle s, t\rangle$*.*

Finding the required quasi-ordering can be done automatically and rather efficiently using dependency graphs and polynomial interpretations [AG00]. The method has been implemented in the CiME tool [CMTU05].

For our $\mathcal{CA}$ algebra the TRS $R_{\mathcal{CA}}$ has a finite number of rules and thus a finite number of dependency pairs, therefore the method above can be applied. We have proven the term rewriting system $R_{\mathcal{CA}}$ of Table 4 to be terminating using CiME (see the implementation details in Section 3.1.1). Thus, we have that the rewriting relation $\rightsquigarrow_R$ is terminating. Note also that the equivalence classes generated by the four identifies in E of associativity and commutativity are finite. This and the fact that $\rightsquigarrow_R$ is terminating implies that $\rightsquigarrow_{R/E}$ is terminating.

Theorem 3.2 (termination of $\mathcal{T}_{\mathcal{CA}}$)**.** *The rewriting relation $\rightsquigarrow_{R_{\mathcal{CA}}/E_{AC+\&}}$ given by the order-sorted rewrite theory $\mathcal{T}_{\mathcal{CA}} = (\Sigma, R_{\mathcal{CA}}, E_{AC+\&})$ is* terminating*.*

Proof: The proof is done using the tool CiME and is based on the discussion above. See Section 3.1.1. □

Once the system is known to be terminating, proving *confluence* is done using the *critical pairs* method for proving local confluence. See in Section 3.1.1 how we use CiME to prove confluence.

Theorem 3.3 (confluence of $\mathcal{T}_{\mathcal{CA}}$)**.** *The rewriting relation $\rightsquigarrow_{R_{\mathcal{CA}}/E_{AC+\&}}$ given by the order-sorted rewrite theory $\mathcal{T}_{\mathcal{CA}} = (\Sigma, R_{\mathcal{CA}}, E_{AC+\&})$ is* local confluent*. Because it is also terminating (by Theorem 3.2) it implies that the rewriting relation is* confluent*.*

Proof: The proof is done using the tool CiME. See Section 3.1.1. □

For convenience we write $\rightsquigarrow_{\mathcal{CA}}$ to stand for the rewrite relation $\rightsquigarrow_{R_{\mathcal{CA}}/E_{AC+\&}}$ given by the order-sorted rewrite theory $\mathcal{T}_{\mathcal{CA}} = (\Sigma, R_{\mathcal{CA}}, E_{AC+\&})$ associated to the algebra of actions $\mathcal{CA}$.

We get as a corollary to Theorems 3.2 and 3.3 that there exists of a *normal form of actions* (NB: normal forms cannot be reduced anymore). The normal form is with respect to the rewriting relation $\rightsquigarrow_{\mathcal{CA}}$ and thus, there exists a whole equivalence class (given by the set of identifies $E_{AC+\&}$) of actions which denote the normal form. For an action $\alpha \in \mathcal{CA}$ we denote the normal form by $\alpha!$. By Theorem 2.13 the normal form of α is in canonical form (NB: the canonical form of an action is just a nice way of writing any action; a canonical form may still be reduced) which is special in the sense that: none of the concurrent actions $\alpha^i_{\&}$ is $\mathbf{0}$ (or $\bot$?) nor it contains a $\mathbf{0}$; and none of the α^i are $\mathbf{1}$ (or $\top$?).

Note: At this point we give more clear intuitions about some relations between trees and actions. Note that trees formed only by applying the tree operations $(\cup, \widehat{\ }, \|)$ are related to the *canonic* form of actions. Moreover, when applying the pruning procedure to a tree is the same as going from the canonical form of an action to its normal form.

We cannot show exactly how the normal form looks like, but guided by the completeness result of Theorem 2.10 we give a characterization of normal form of actions in terms of pruned trees in Theorem 3.4. Instead of working with the normal form of an action we work with the pruned tree that we obtain from the initial action. This gives us a clear *structure* of the normal form which is easy to work with.

Theorem 3.4 (Interpretation of normal form of actions)**.** *Consider an arbitrary action α. The tree interpreting the normal form of action α is isomorphic to the pruned tree $I_{\mathcal{CA}}(\alpha)$ interpreting α. Moreover, the pruning procedure cannot be applied to an action in normal form.*

Proof: The normal form of an action α is obtained from the action by applying the axioms of the algebra from Table 3 exhaustively. That means that to the normal form $\alpha!$ no more axioms can be applied.

We prove first the more easy part of the theorem which is part two. This can be rewritten as: $I_{\mathcal{CA}}(\alpha!) = Prune(I_{\mathcal{CA}}(\alpha!))$. We prove this by *reductio ad absurdum* and we suppose we can apply the pruning procedure to the tree interpreting action $\alpha!$. The contradiction will be that each time we show that the normal form $\alpha!$ can be rewritten even more using $\rightsquigarrow_{\mathcal{CA}}$; i.e. by applying one of the reduction rules of Table 4. This is impossible as a normal form by definition cannot be further reduced. We need to consider a case for each

possible way of applying the pruning procedure. That means we need to consider a case for each step of the pruning procedure in the Definition 2.9.

Case 1. Suppose there exists an edge $(m, \tau\gamma, n:\{\top\})$ such that step 1a of the pruning procedure can be applied. By using the inverse interpretation function as defined in the proof before of Theorem 2.10 we know that we can associate to the tree an action which by Theorem 2.13 is in canonical form: $\beta \cdot (I^{-1}_{\mathcal{CA}}((m, \tau\gamma, n:\{\top\})) \cdot \beta' + \ldots)$. It is clear that the action $I^{-1}_{\mathcal{CA}}((m, \tau\gamma, n:\{\top\}))$ comes from a concurrent composition and it should look like $\mathbf{1}\&\gamma$ and thus our initial action is $\beta \cdot (\mathbf{1}\&\gamma \cdot \beta' + \ldots) \overset{(10)}{\rightsquigarrow}_{\mathcal{CA}} \beta \cdot (\gamma \cdot \beta' + \ldots)$. Therefore, we have a contradiction because we can reduce even more.

Case 2. Suppose there exists an edge $(m, \tau, n:\{\top\})$ where n is not a leaf node with siblings, such that step 1b of the pruning procedure can be applied. This means that we can have two kinds of actions: one $\beta \cdot (I^{-1}_{\mathcal{CA}}((m, \tau, n:\{\top\})))$ when n is a leaf node but it has no siblings, and the second $\beta \cdot (I^{-1}_{\mathcal{CA}}((m, \tau, n:\{\top\})) \cdot \beta' + \ldots)$ when n is not a leaf node (i.e. $\beta' \neq \mathbf{1}$). The tree $(m, \tau, n:\{\top\})$ comes from the action $\mathbf{1}$. The first action is thus $\beta \cdot \mathbf{1} \overset{(4)}{\rightsquigarrow}_{\mathcal{CA}} \beta$. The second action is $\beta \cdot (\mathbf{1} \cdot \beta' + \ldots) \overset{(5)}{\rightsquigarrow}_{\mathcal{CA}} \beta \cdot (\beta' + \ldots)$. Therefore, we have a contradiction in for each of the two actions.

Case 3. Suppose that the edge $(m, \gamma, n:\{\bot\})$ ends in a bottom node and γ can be any label (even τ). A tree containing this edge can come from any action; i.e. of the form: $\beta \cdot (I^{-1}_{\mathcal{CA}}((m, \gamma, n:\{\bot\})) \cdot \beta' + \ldots)$ where the bottom node can come only from combination with the tree $(m, \tau, n:\{\bot\})$ modelling $\mathbf{0}$. Now suppose $\gamma \neq \tau$ then the initial action must be of the form $\beta \cdot (\mathbf{0}\&\gamma \cdot \beta' + \ldots) \overset{(11)}{\rightsquigarrow}_{\mathcal{CA}} \beta \cdot (\mathbf{0} \cdot \beta' + \ldots)$. On the other hand, when $\gamma = \tau$ then we have the following situations for the action $\beta \cdot (\mathbf{0} \cdot \beta' + \ldots)$: suppose β' exists then we have the derivation $\beta \cdot (\mathbf{0} \cdot \beta' + \ldots) \overset{(7)}{\rightsquigarrow}_{\mathcal{CA}} \beta \cdot (\mathbf{0} + \ldots)$; suppose that β' does not exist, but we still have a derivation $\beta \cdot (\mathbf{0} + \ldots) \overset{(1)}{\rightsquigarrow}_{\mathcal{CA}} \beta \cdot (+ \ldots)$. If the β' does not exists and also there is no sibling for the node n then we still have the derivation $\beta \cdot \mathbf{0} \overset{(6)}{\rightsquigarrow}_{\mathcal{CA}} \mathbf{0}$. In the case when β does not exist either, we are in the special case when the tree is only the edge $(m, \tau, n:\{\bot\})$ interpreting action $\mathbf{0}$ which is allready in normal form.

The three cases above finish the proof of the second part of the theorem and thus $I_{\mathcal{CA}}(\alpha!) = Prune(I_{\mathcal{CA}}(\alpha!))$.

For the first part, the statement of the theorem basically says that given α we need to prove that $\hat{I}_{\mathcal{CA}}(\alpha) \doteq I_{\mathcal{CA}}(\alpha!)$; i.e. the pruned tree interpreting the action α is isomorphic to the tree interpreting its normal form. Having

the second part of the theorem proven before we can rewrite this statement as: $\hat{I}_{\mathcal{CA}}(\alpha) \doteq \hat{I}_{\mathcal{CA}}(\alpha!)$. Without loss of generality we can consider α and its normal form $\alpha!$ as having a minimal difference which is: $\alpha!$ is obtained from α by application of only one reduction rule of the $\mathcal{T}_{\mathcal{CA}}$ from Table 4. For example when looking at rule (4) we consider $\alpha = \alpha' \cdot \mathbf{1}$ and $\alpha! = \alpha'$. Therefore, the rest of the proof needs only to consider one case for each rule of Table 4.

Case given by rule (1). We consider $\alpha = \alpha' + \mathbf{0}$ and $\alpha! = \alpha'$. The tree $\hat{I}_{\mathcal{CA}}(\alpha) = \hat{I}_{\mathcal{CA}}(\alpha' + \mathbf{0}) = Prune(I_{\mathcal{CA}}(\alpha' + \mathbf{0}))$. On the other hand $I_{\mathcal{CA}}(\alpha' + \mathbf{0}) = I_{\mathcal{CA}}(\alpha') \cup I_{\mathcal{CA}}(\mathbf{0})$. This means that the operator $\cup$ adds to the first level of the tree $I_{\mathcal{CA}}(\alpha')$ one edge $(r, \tau, n : \{\bot\})$ interpreting the tree $I_{\mathcal{CA}}(\mathbf{0})$. The pruning procedure that must be applied after the joining of the two trees will apply step 3b and remove the newly added edge thus remaining with the tree $I_{\mathcal{CA}}(\alpha')$. We know from the second part of the theorem that $Prune(I_{\mathcal{CA}}(\alpha')) = I_{\mathcal{CA}}(\alpha')$ (and because our supposition states that $\alpha' = \alpha!$) and thus we have finished the proof.

Case given by rule (2). We consider $\alpha = \alpha' + \alpha'$ and $\alpha! = \alpha'$. This case is take care of only by the definition of the $\cup$ operator, which combines each two edges which are labeled with the same label and then continues downwards in the same manner. Therefore, the two identical trees $I_{\mathcal{CA}}(\alpha')$ are united into only one $I_{\mathcal{CA}}(\alpha')$. Thus we have our proof that $\hat{I}_{\mathcal{CA}}(\alpha) = \hat{I}_{\mathcal{CA}}(\alpha!)$.

Case given by rule (3). We consider $\alpha = (\alpha' \cdot \beta) \cdot \gamma$ and $\alpha! = \alpha' \cdot (\beta \cdot \gamma)$. This case is taken care by the definition of the tree operators. More precisely $I_{\mathcal{CA}}((\alpha' \cdot \beta) \cdot \gamma)$ is obtained as $I_{\mathcal{CA}}(\alpha' \cdot \beta) \widehat{} I_{\mathcal{CA}}(\gamma) = (I_{\mathcal{CA}}(\alpha') \widehat{} I_{\mathcal{CA}}(\beta)) \widehat{} I_{\mathcal{CA}}(\gamma)$, which from the associativity of the $\widehat{}$ operation on trees (see Proposition 2.6) we conclude that it is the same as the tree $I_{\mathcal{CA}}(\alpha') \widehat{} (I_{\mathcal{CA}}(\beta) \widehat{} I_{\mathcal{CA}}(\gamma)) = I_{\mathcal{CA}}(\alpha' \cdot (\beta \cdot \gamma))$. Thus we have proven this case of not prunde trees; i.e. $I_{\mathcal{CA}}(\alpha) \doteq I_{\mathcal{CA}}(\alpha!)$, and because the pruning procedure behaves like a function we have the proof for the pruned versions $\hat{I}_{\mathcal{CA}}(\alpha) \doteq \hat{I}_{\mathcal{CA}}(\alpha!)$.

Case given by rules (4) and (5). We consider $\alpha = \mathbf{1} \cdot \alpha'$ and $\alpha! = \alpha'$. The symmetric case when $\alpha = \alpha' \cdot \mathbf{1}$ is treated similarly. The tree $\hat{I}_{\mathcal{CA}}(\alpha) = \hat{I}_{\mathcal{CA}}(\mathbf{1} \cdot \alpha') = Prune(I_{\mathcal{CA}}(\mathbf{1} \cdot \alpha'))$. The tree $I_{\mathcal{CA}}(\mathbf{1} \cdot \alpha')$ is obtained by concatenation of the tree $I_{\mathcal{CA}}(\alpha')$ to the tree formed by only one edge $(m, \tau, n : \{\top\})$. On the other hand the pruning procedure applies step 1b and removes the first edge $(m, \tau, n : \{\top\})$ thus remaining with the tree $I_{\mathcal{CA}}(\alpha')$, and the proof is finished.

Case given by rules (6) and (7). We consider $\alpha = \mathbf{0} \cdot \alpha'$ and $\alpha! = \mathbf{0}$. The tree $\hat{I}_{\mathcal{CA}}(\alpha) = \hat{I}_{\mathcal{CA}}(\mathbf{0} \cdot \alpha') = Prune(I_{\mathcal{CA}}(\mathbf{0} \cdot \alpha'))$. The tree $I_{\mathcal{CA}}(\mathbf{0} \cdot \alpha')$

is obtained by concatenation of the tree $I_{\mathcal{CA}}(\alpha')$ to the tree formed by only one edge $(m, \tau, n : \{\bot\})$. Now the pruning procedure applies step 2a and removes the whole tree $I_{\mathcal{CA}}(\alpha)$ coming after the bottom node n and we are left with the tree formed of only the edge $(m, \tau, n : \{\bot\})$ which is $I_{\mathcal{CA}}(\mathbf{0})$. The symmetric case when $\alpha = \alpha' \cdot \mathbf{0}$ is treated similarly only that the other steps then 2a of the pruning procedure that deal with bottom nodes are applied. The concatenation operation spreads the bottom node of the tree $I_{\mathcal{CA}}(\mathbf{0})$ to all leaf nodes of the tree $I_{\mathcal{CA}}(\alpha')$ and thus any path in the tree has a bottom node on it. Because of this the step 3b can never be applied. Therefore the bottom node propagates upwards until reaching the topmost edge. Thus we have now the tree $I_{\mathcal{CA}}(\mathbf{0})$.

Case given by rule (8). We consider $\alpha = \alpha' \cdot (\beta + \gamma)$ and $\alpha! = \alpha' \cdot \beta + \alpha' \cdot \gamma$. This case follows immediately from the definition of the *join* tree operation $\cup$; more precisely see Proposition 2.6 for a proof of this general property on trees. Particularly, we need to prove that the tree $\hat{I}_{\mathcal{CA}}(\alpha) \doteq \hat{I}_{\mathcal{CA}}(\alpha!)$, for which it is enough to prove this for the not prunned versions; i.e. $I_{\mathcal{CA}}(\alpha' \cdot (\beta + \gamma)) \doteq I_{\mathcal{CA}}(\alpha' \cdot \beta + \alpha' \cdot \gamma)$. But the above equality translates to $I_{\mathcal{CA}}(\alpha') \widehat{} (I_{\mathcal{CA}}(\beta) \cup I_{\mathcal{CA}}(\gamma)) \doteq (I_{\mathcal{CA}}(\alpha') \widehat{} I_{\mathcal{CA}}(\beta)) \cup (I_{\mathcal{CA}}(\alpha') \widehat{} I_{\mathcal{CA}}(\gamma))$. This follows directly from the distributivity of the concatenation operation $\widehat{}$ over the joining operation $\cup$.

Case given by rule (9). This case follows similarly as the one above from the general property on tree which is proven in Proposition 2.6.

Case given by rule (10). We consider $\alpha = \alpha' \& \mathbf{1}$ and $\alpha! = \alpha'$. This case is treated similarly to the case for axiom (4) before only that now the step 1a of the pruning procedure is applied.

Case given by rule (11). We consider $\alpha = \alpha' \& \mathbf{0}$ and $\alpha! = \mathbf{0}$. This case is similar to the case for axiom (7) before. In a first state the $\|$ operation spreads the edge $(m, \tau, n : \{\bot\})$ all over the first level of the tree $I_{\mathcal{CA}}(\alpha')$. Therefore, each edge on the first level of the resulting tree will be ending in a bottom node because $\|$ makes the conjoining of the nodes (i.e. $\bot \wedge \phi = \bot$ or $\bot \wedge \top = \bot$). Now it is easy to see that the pruning procedure applies first step 1a to remove the newly added τ labels, and then applies step 2a to remove all edges on the levels starting below the bottom nodes. Then step 2b is applied and transforms all the labels of the remaining edges into τ. The last application is of the step 3b several times until we are left with only one edge denoting the tree $I_{\mathcal{CA}}(\mathbf{0})$. The proof is finished.

Case given by rule (12). We consider $\alpha = a \& a$ and $\alpha! = a$ where $a \in \mathcal{A}_B$ is a basic action. We need to prove $\hat{I}_{\mathcal{CA}}(a \& a) \doteq \hat{I}_{\mathcal{CA}}(a)$ for which it suffices to prove $I_{\mathcal{CA}}(a \& a) \doteq I_{\mathcal{CA}}(a)$. More precisely we have to proove that

$I_{\mathcal{CA}}(a) \parallel I_{\mathcal{CA}}(a) \doteq I_{\mathcal{CA}}(a)$. This case is trivial as the operation $\parallel$ takes care that duplicated entries on a label of a tree are removed.[15]

Case given by rule (13). We consider $\alpha = \alpha'\&(\beta + \gamma)$ and $\alpha! = \alpha'\&\beta + \alpha'\&\gamma$. We prove this case for the not-pruned trees, which can by automatically extended to pruned trees. Therefore, we prove that $I_{\mathcal{CA}}(\alpha'\&(\beta + \gamma)) \doteq I_{\mathcal{CA}}(\alpha'\&\beta + \alpha'\&\gamma)$. The basic idea for this case is that the $\parallel$ operation works on each level of the tree at a time and makes the cartesian product of the labels from both trees on each level. It is easy to see how the two trees are the same: $I_{\mathcal{CA}}(\alpha') \parallel (I_{\mathcal{CA}}(\beta) \cup I_{\mathcal{CA}}(\gamma)) \doteq (I_{\mathcal{CA}}(\alpha') \parallel I_{\mathcal{CA}}(\beta)) \cup (I_{\mathcal{CA}}(\alpha') \parallel I_{\mathcal{CA}}(\gamma))$. The details are left to the reader.

Case given by rule (14). We consider $\alpha = (\alpha_{\&} \cdot \alpha')\&(\beta_{\&} \cdot \beta)$ and $\alpha! = (\alpha_{\&}\&\beta_{\&}) \cdot (\alpha'\&\beta)$. We prove this case for the not-pruned trees, which can by automatically extended to pruned trees. Therefore, we prove that $I_{\mathcal{CA}}((\alpha_{\&} \cdot \alpha')\&(\beta_{\&} \cdot \beta)) \doteq I_{\mathcal{CA}}((\alpha_{\&}\&\beta_{\&}) \cdot (\alpha'\&\beta))$. This case uses arguments similar to the case before. This time they refer to the fact that the $\parallel$ operation is applied on each level of the tree at a time. The details are felt to the reader.

The remaining rules and the identifies of $E_{AC+\&}$ are dealt with by the definition of the guarded rooted trees and the definition of the operations on trees. For the identifies defining associativity and commutativity of the $+$ constructor (i.e. axioms (1) and (2) of Table 3), these are taken care of by the definition of guarded rooted trees. In a guarded rooted tree there is no order on the branches. For the identifies of associativity and commutativity of $\&$ constructor (i.e. axioms (10) and (11) of Table 3) this is taken care of by the associativity and commutativity of union of sets of labels of the edges, and the classical definition of cartesian product of two sets. □

3.1.1 Implementation in CiME

CiME is a tool for checking termination and confluence of term rewriting systems. It can do a lot more, like completion or unification. The latest version of CiME is 2.02 and incorporates the termination checking.[16] The input syntax is natural for a term rewriting systems. We have specified the term rewriting system from Table 4 in CiME.[17]

We need to define first the signature of our TRS and a set of variables

[15] Recall that the labels of the edges of the tree ar *sets* of basic actions of $\mathcal{A}_B$.

[16] The CiME web site is: `http://cime.lri.fr/`

[17] See input specification files for the CiME tool on the COSoDIS project homepage: `http://www.ifi.uio.no/cosodis/software.shtml`

to work with. CiME does not support order-sorting (like Maude does f.ex.). But we are lucky because our sorts are finite

```
let F = signature "
0,1,a,b : constant;
+ : AC;
. : infix binary;
& : AC;
";

let X = vars "x y z";

let R = TRS F X "
x + 0 -> x;
x + x -> x;
x . 1 -> x;
1 . x -> x;
x . 0 -> 0;
0 . x -> 0;
(x . y) . z -> x . (y . z);
(x . y) + (x . z) -> x . (y + z);
(x + y) . z -> (x . z) + (y . z);
x & 1 -> x;
x & 0 -> 0;
a & a -> a;
b & b -> b;
x & (y + z) -> (x & y) + (x & z);
(a . x) & (a . y) -> (a & a) . (x & y);
(a . x) & (b . y) -> (a & b) . (x & y);
(a . x) & (a & b . y) -> (a & a & b) . (x & y);
(b . x) & (b . y) -> (b & b) . (x & y);
(b . x) & (a & b . y) -> (b & a & b) . (x & y);
(a & b . x) & (a & b . y) -> (a & b & a & b) . (x & y);
a & (a . x) -> (a & a) . x;
a & (b . x) -> (a & b) . x;
a & (a & b . x) -> (a & a & b) . x;
b & (a . x) -> (b & a) . x;
b & (b . x) -> (b & b) . x;
b & (a & b . x) -> (b & a & b) . x;
a & b & (a & b . x) -> (a & b & a & b) . x;
a & b & (a . x) -> (a & b & a) . x;
a & b & (b . x) -> (a & b & b) . x;
";
```

3.2 Simple matching modulo $\mathcal{CA}$

The problem presented in this section is important in giving the branching semantics of obligations in Section 4.1. We present a particular matching problem modulo the set of equations of the $\mathcal{CA}$ algebra.

(1)	$x + (y + z) = (x + y) + z$	(A+)
(2)	$x + y = y + x$	(C+)
(3)	$x + \mathbf{0} = \mathbf{0} + x = x$	(U+0)
(4)	$\forall x,\ \exists x^{-}$ s.t. $x + x^{-} = 0$	(In+)
(5)	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$	(A.)
(6)	$x \cdot \mathbf{1} = \mathbf{1} \cdot x = x$	(U.1)
(7)	$x \cdot (y + z) = x \cdot y + x \cdot z$	(D.+l)
(8)	$(x + y) \cdot z = x \cdot z + y \cdot z$	(D.+r)
(9)	$x \cdot y = y \cdot x$	(C.)
(10)	$x \cdot x = x$	(I.)

Table 5: Axioms for commutative ring and Boolean ring

Our problem comes from the broad field of unification theory. Before presenting the problem we give preliminary related definitions.

3.2.1 Commutative ring compared with Boolean ring

We would like to understand a rather strange situation: elementary unification modulo a commutative ring is proven to be undecidable; elementary unification modulo Boolean ring equations is NP-compete.

A *ring* structure is defined by the axioms (1)-(8). If one adds the commutativity property (9) for the multiplicative operation then we get a *commutative ring* (denoted CR henceforth). On the other hand if one adds the identity property (10) for the multiplicative operation then one gets a *Boolean ring* structure (denoted BR henceforth).

The proof that unification in commutative rings is undecidable is given in [BS94] and is based on the proof of undecidability of Hilbert's 10^{th} problem [Mat73, Mat93]. It is known that the integers with the operations $\{+, \cdot, ^{-}, \mathbf{0}, \mathbf{1}\}$ form a ring; i.e. the *ring of integers* (where $^{-}$ is the inverse with respect to the aditive operation). The ring of integers is the initial model of the commutative ring, therefore any constructor term over the signature of the CR can be viewed as an integer; any ground term over CR can be viewed as an operation with integers (i.e. addition, multiplication); and any term with n variables over CR can be viewed as a polinomial over the integers with n unknowns. Conversely, every such polinomial can be viewed as a term in CR. A unification problem over CR has a solution iff it has a solution that substitutes only ground terms for the variables. Substituting ground terms for variables can be viewed in the integers as evaluating the polinomial with the unknowns substituted for integers. Thus, a unification

problem over the CR can be translated into the ring of integers and it reduces to solving equations between polinomials with solutions in the ring of integers. But this problem is known to be undecidable which implies that also elementary unification in CR is undecidable.

Let us now remind basic results for Boolean rings [MN89]. The BR has the extra axiom of idempotence for the multiplicative operation, which turns out to be stronger than the commutativity of CR. Indeed, we can prove the commutativity of $\cdot$ in BR. First we have to prove that each element is its own invers w.r.t. $+$; that is $x + x = 0$. The proof is $x + x = (x + x)^2 = x^2 + x^2 + x^2 + x^2 = x + x + x + x$ now we add in both parts of the equation ^{-}x to obtain $0 = x + x$. Note that we use powers to write multiplication of the same element several times. To prove the commutativity $x \cdot y = y \cdot x$ we do: $x + y = (x + y)^2 = x^2 + x \cdot y + y \cdot x + y^2 = x + x \cdot y + y \cdot x + y$. We again use the inverse of the elements x and y (which we know to exist) and obtain that $0 = x \cdot y + y \cdot x$. By using the property before we have $x \cdot y + x \cdot y = x \cdot y + y \cdot x$ and we add the inverse of $x \cdot y$ and the fact that 0 is unity element for $+$ to get $x \cdot y = y \cdot x$. Moreover, in Boolean rings we can prove $x \cdot 0 = 0$ which is the axiom added in the *semiring* structure instead of the inverse property of the rings. In conclusion, any Boolean ring is also a commutative ring. On the other hand, the Boolean ring has not as a model the ring of integers. This breakes the undecidability proof we did before. Even more, the extra properties on the elements of a Boolean ring make the unification problem decidable. Note that the above results are based not only on the idempotency of the multiplicative operation, but also on the equation for the inverse of each element with respect to the aditive operation.

Examples of models of Boolean ring are the powerset $\mathcal{P}(S)$ of a set S with symmetric difference for $+$ and intersection for $\cdot$, and S for $\mathbf{1}$ and $\emptyset$ for $\mathbf{0}$. A second example of Boolean ring is the set of well-formed formulas of propositional calculus under the operations of exclusive or and conjunction for respectively $+$ and $\cdot$, where $\mathbf{1}, \mathbf{0}$ represent $true, false$.

At an intuitive level, the undecidability of unification in commutative ring comes from the fact that terms may be expanded indefinitely, creating an infinite number of different elements. This comes from terms like $x+x+\ldots$ or $x \cdot x \cdot \ldots$. On the other hand, in Boolean rings this is not the case because of equations $x + x = \mathbf{0}$ and $x \cdot x = x$. Moreover, Boolean rings have a finite number of elements.

The algorithms for unification in Boolean ring are based on ideas from Bool or Löwenheim and are presented in [MN89] (see also [Rud74]). The insights of these algorithms are based on the finiteness of the Boolean algebras generated by a finite set of generators. The following theorem is at the basis

of the unification algorithms for Boolean ring.

Theorem 3.5 ([MN89])**.**

1. *Any unification problem in Boolean rings can be reduced to a (set of) problem(s) written in the following schema:*
$$f(x) = \mathbf{0}.$$
2. *Let D be the (finite) subalgebra of BR generated by the set of constant symbols that appear in the term $f(x)$. The following hold:*
   (a) *The equation $f(x) = \mathbf{0}$ has a solution in BR iff it has a solution in D;*
   (b) *Any most general unifyier (mgu) of $f(x) = \mathbf{0}$ w.r.t. D is also a mgu w.r.t. BR.*

The theorem shows that we need to be concerned with only solving equations in the finite Boolean subalgebra generated by the constants found in the term $f(x)$; which becomes a simpler problem. A method for finding a mgu is called "successive variable elimination" and was proposed by Bool and then further extended by Rudeanu [Rud74].

We now take a look at the axioms of the $\mathcal{CA}$ algebra from Table 3.

3.2.2 Narrowing and related recent results

The *narrowing* relation was introduced by Hullot in [Hul80]; the definition that we give here is taken from Middeldorp and Hamoen [MH94].

Definition 3.1 (narrowing relation)**.** *Consider two terms t, t', a nonvariable position $p \in Pos_\Sigma(t)$, a variant of a rewrite rule $l \to r$, and a substitution ς such that:*

- *ς is a* most general unifyier *of $t|_p$ and l,*
- *$t' = (t[r]_p)\varsigma$*

In this case we say that t and t' are in the narrowing relation *(or equivaletly we say that t is narrowable into t') and we denote it by $t \overset{\varsigma}{\rightsquigarrow}_{[p,l\to r]} t'$ (or simply by $t \overset{\varsigma}{\rightsquigarrow} t'$).*

We denote by $t_1 \overset{\varsigma}{\rightsquigarrow}^* t_n$ a *narrowing derivation* to stand for $t_1 \overset{\varsigma_1}{\rightsquigarrow} t_2 \overset{\varsigma_2}{\rightsquigarrow} \dots t_{n-1} \overset{\varsigma_{n-1}}{\rightsquigarrow} t_n$ where $\varsigma = \varsigma_1\varsigma_2 \dots \varsigma_{n-1}$.

Note that the rewriting relation is a special case of the narrowing relation where the rewrite rule $l \to r$ has no variables in common with t and ς is

restricted only to the variables occuring in l and thus ς becomes a matcher and not a mgu.

Narrowing was designated to solve equational unification problems. We show how can this be done. We first need to extend the signature with a binary symbol $=^?$ and a new special constant **true**. Moreover, we add to the set of rewrite rules the special rule $x =^? x \to \mathbf{true}$. The new symbols are restricted even more (such that to not interfeer with the normal rewriting process):

1. we consider only terms which do not contain any occurrences of $=^?$ or **true**;
2. or terms $t =^? t'$ (called *goals*) where t, t' are terms as defined in 1.;
3. the constant **true** is a term.

The next result gives a suficient condition for determining an unifyier of two terms. This is also called the *soundness* of the narrowing procedure.

Lemma 3.6 ([MH94])**.** *Let R be a term rewriting system. If $t =^? t' \overset{\varsigma}{\rightsquigarrow}{}^* \mathbf{true}$ then ς is an R-unifyier of t and t'.*

This lemma alone does not provide a decision procedure for the unification problem because there may be unification problems for which there exists a unifyier which is not found by the narrowing procedure (i.e. the term $t =^? t'$ is not narrowed to **true**). Therefore, the *completeness* property for the narrowing procedure is desirable.

Lemma 3.7 ([MH94])**.** *Let R be a convergent term rewriting system. If there exists ς s.t. $t\varsigma =_R t'\varsigma$ then there exists a narrowing derivation $t =^? t' \overset{\varsigma'}{\rightsquigarrow}{}^* \mathbf{true}$ s.t. ς' is more general than ς.*

Moreover, we obtain as a corollary that the narrowing procedure generates a *complete set of R-unifyiers.*

The main problem with the narrowing procedure is that we do not know if it terminates; in fact it seldom does. For this reason Hullot [Hul80] devised a termination criterion for narrowing, which is called *basic narrowing.* In short, basic narrowing is a restricted form of narrowing which restricts the positions p at which the narrowing relation can be applied to only *basic positions.* The purpose of basic narrowing is to not allow the application of narrowing on terms introduced by the substitution in an earlier narrowing step. The following result ensures terminaiton of the narrowing procedure.

Lemma 3.8 ([Hul80]). *If all basic narrowing derivations starting at the right-hand side of a rewrite rule terminates, then any narrowing derivation starting at any term terminates.*

Unfortunately, our equational system which defines the $\mathcal{CA}$ algebra does not allow for application of the narrowing procedure because it is not a convergent system. Therefore, more elaborated techniques for narrowing are required which may allow more general equational systems.

We now see a natural extension of narrowing which is called *narrowing modulo a set of equations* [JKK83]. Consider as before a rewrite theory $\mathcal{T} = (\Sigma, E, R)$ which satisfies the following properties (some of which we formally defined before also):

(a) A set of equations E is called *regular* iff for each $t = t'$ we have that $V(t) = V(t')$ and is *sort preserving* which means that $\forall \varsigma$ the terms $t\varsigma$ and $t'\varsigma$ have the same sort.

(b) E has a finitary and complete unification algorithm; which implies that E-mathcing is also finitary and complete.

(c) For each rewrite rule $l \to r \in R$ we have that $V(l) \subseteq V(r)$.

(d) R is *sort-decreasing.*

(e) R is *convergent modulo* E which means that the relation $\rightsquigarrow_{R/E}$ is convergent.

On another hand the relation $\rightsquigarrow_{R/E}$ is undecidable in general because the E-congruence classes may be infinite; therefore, an equivalen relation is used in practice [PS81].

Definition 3.2 (R, E rewrite relation). *Consider a rewrite theory (Σ, E, R) with the properties (a)-(e) from above. We define the rewrite relation $\rightsquigarrow_{R,E}$ as $t \rightsquigarrow_{R,E} t'$ iff $\exists p \in Pos_\Sigma(t)$, $\exists l \to r \in R$, $\exists \varsigma$, such that $t|_p =_E l\varsigma$ and $t' = t[r\varsigma]_p$ (i.e. ς is an E-matcher of $t|_p$ and l).*

We give on important property of $\rightsquigarrow_{R,E}$ (first defined in [JKK83]):

(f) $\rightsquigarrow_{R,E}$ is *E-coherent* iff $\forall t_1, t_2, t_3$ we have that if $t_1 \rightsquigarrow_{R,E} t_2$ and $t_1 =_E t_3$ then $\exists t_4, t_5$ s.t. $t_2 \rightsquigarrow^*_{R,E} t_4$ and $t_3 \rightsquigarrow^+_{R,E} t_5$ and $t_4 =_E t_5$.

Definition 3.3 (R,E narrowing relation). *Consider an order-sorted rewrite theory $\mathcal{T} = (\Sigma, E, R)$ which has the properties (a)-(f) from above. The R, E-*narrowing relation *is denoted by $\overset{S}{\rightsquigarrow}_{R,E}$ and is defined as $t \overset{S}{\rightsquigarrow}_{R,E} t'$ iff $\exists p \in Pos_\Sigma(t)$, $\exists l \to r \in R$, and $\exists \varsigma$ a substitution from the complete set of unifyiers of $t|_p$ and l, and $t' = t[r\varsigma]_p$.*

Note that in the case of (simple) narrowing of Definition 3.1 ς is just a mgu because we do syntactic unification of the two terms $t|_p$ and l. In the case of R, E-narrowing of Definition 3.3 we do E-unification and, as expected when going from sntactic to equational unification, we change from mgu to a (minimal) complete set of unifyiers. The property (b) of $\mathcal{T}$ ensures that there exists a *compelte set of unifyiers* for each E-unification problem.

The definition of R, E-narrowing requires that the rewrite theory (Σ, E, R) has the six properties (a)-(f), therefore testing for these properties must be decidable in order for R, E-narrowing relation to be useful. Testing (a) for regularity and sort preserving of E is decidable as there are finitely many identifies in E. The unification algorithm for E is considered given. Testing (c) and (d) is clearly simple. Testing (e) for convergence of R modulo E is in general undecidable, but techniques exists for many of the interesting systems; thus we can decide this property using some of these classical techniques.

Not too clear is how to (and if we can) test for the property (f). Fortunately, in [JKK83] it has also been given a sufficient and decidable condition for testing E-coherence.

Definition 3.4 ([JKK83])**.** *Consider two rules $l \to r$ and $l' \to r'$ with $V(l) \cap V(l') = \emptyset$, s.t. l E-overlaps l' at position p with a complete set S of E-overlappings. We say that l E-overlaps l' at $p \in Pos_\Sigma(l')$ with a complete set S of E-overlappings iff S is a complete set of E-unifyiers of l and $l'|_p$.*

We call a complete set of E-critical pairs *of the two rules $l \to r$ and $l' \to r'$ the set $\{(P, Q) \mid P = r'\varsigma, Q = (l'[r]_p)\varsigma, \forall \varsigma \in S\}$.*

The complete set of E-critical pairs associated to a rewrite theory (Σ, E, R) is the union of all the nontrivial E-critical pairs from the complete sets obtained from the rules of R together with the rules $t \to t'$ with $t = t' \in E$.

Lemma 3.9 ([JKK83])**.** *Consider a rewrite theory $\mathcal{T} = (\Sigma, E, R)$ with properties (a)-(e). The rewrite relation $\rightsquigarrow_{R,E}$ is E-coherent if any E-critical pair $(P = r'\varsigma, Q)$ from the complete set of E-critical pairs associated to the rewrite theory $\mathcal{T}$ satisfies $P \rightsquigarrow_{R,E} P'$ at some position $p \in Pos_\Sigma(r')$ and $P'! =_E Q!$ (i.e. the two normal forms w.r.t. $\rightsquigarrow_{R,E}$ are equal modulo the set of identities in E).*

It is easy to see that the condition of the lemma above is decidable. This is because there is a finite number of rules in R and a finite number of identities in E and each term has a finite number of positions on which E-overlappings may occur; therefore there is a finite number of E-critical pairs to test. The test for each E-critical pair involves the normalization of the two terms which is decidable as the rewrite theory is convergent.

R, E-narrowing procedure is proven sound and complete in the same manner as we presented before. We get a similar decision procedure for equational unification, only that in this case we work with complete sets of unifyiers. To avoid the same nontermination problem as with (general) narrowing [JKK83] defined *basic* R, E*-narrowing* in a similar manner as Hullot and proved it to be sound and complete.

Unfortunately, in [CLD05] it was shown by counterexample that the completeness result for basic R, E-narrowing was wrong. It seams that this has happend because [JKK83] based their proofs on the similarities with the lifting lemma of [Hul80] which was later shown (and corrected in [MH94]) to have some notorious flaw in its proof.

Nevertheless in [EMS08] all was repaired by introducing the *variant narrowing* relation which builds up on the ideas of R, E-narrowing and on the *variant property* of [CLD05].

3.2.3 The solution