# Sentence-Normalized Conditional Narrowing Modulo in Rewriting Logic and Maude*

Luis Aguirre, Narciso Martí-Oliet, Miguel Palomino, and Isabel Pita

Facultad de Informática, Universidad Complutense de Madrid, Spain
{luisagui, narciso, miguelpt, ipandreu}@ucm.es

**Abstract.** This work studies the relationship between verifiable and computable answers for reachability problems in rewrite theories with an underlying membership equational logic. These problems have the form

$$(\exists \bar{x})t(\bar{x}) \rightarrow^* t'(\bar{x})$$

with $\bar{x}$ some variables, or a conjunction of several of these subgoals. A calculus that solves this kind of problems working always with *normalized terms and substitutions* has been developed. Given a reachability problem in a rewrite theory, this calculus can compute any normalized answer that can be checked by rewriting, or one that can be instantiated to that answer. Special care has been taken in the calculus to keep membership information attached to each term, to make use of it whenever possible.

**Keywords:** Maude, narrowing, reachability, rewriting logic, unification, membership equational logic

*Dedicated to José Meseguer on occasion of his 65th birthday. This paper is based on work he has been developing during the last 30 years, from order-sorted algebra to narrowing, going through membership equational logic, rewriting logic, and much more.*

## 1 Introduction

Rewriting logic is a computational logic that has been around for more than twenty years [20], whose semantics [6] has a precise mathematical meaning allowing mathematical reasoning for proving properties, providing a flexible framework for the specification of concurrent systems. It turned out that it can express both concurrent computation and logical deduction, allowing its application in many areas such as automated deduction, software and hardware specification and verification, security, etc. [19, 22].

A deductive system is specified in rewriting logic as a rewrite theory $\mathcal{R} = (\Sigma, \mathcal{E}, R)$, with $(\Sigma, \mathcal{E})$ an underlying equational theory (in this paper we will consider *membership equational logic*), where terms are given an algebraic data

type, allowing us to identify as semantically equal two syntactically different terms, and $R$ a set of rules that specify how the deductive system can derive one term from another. *Order-sorted*, *many-sorted*, and *unsorted* theories can be formulated as special cases of membership equational logic (MEL) theories.

Reachability problems have the form

$$(\exists \bar{x}) t(\bar{x}) \rightarrow^* t'(\bar{x})$$

with $t$, $t'$ terms with variables in $\bar{x}$, or a conjunction of several of these subgoals. They can be solved by model-checking methods for finite state spaces. When the initial term $t$ has no variables, i.e. it is a ground term, and under certain admissibility conditions, rewriting can be used in a breadth-first way to traverse the state space, trying to find a suitable matching of $t'(\bar{x})$ in each traversed node. In the general case where $t(\bar{x})$ is not a ground term, a technique known as *narrowing* [14] that was first proposed as a method for solving equational goals (*unification*), has been extended to cover also reachability goals [24], leaving equational goals as a special case. The strength of narrowing can be found in that it enables us to manage complex concurrent and deductive systems that cannot be handled by faster, but more limited, specialized methods. Under the admissibility conditions for rewrite theories, which allow for conditional rules and equations with extra variables in the conditions under some requirements, and the assumption of the existence of an $\mathcal{E}$-unification algorithm, we can use narrowing modulo $\mathcal{E}$ to perform symbolic analysis of the possibly infinite set of initial states $t(\bar{x})$ in the state space and determine the actual values of $\bar{x}$ that allow us to derive $t'(\bar{x})$ from $t(\bar{x})$.

What is most striking is the fact that an $\mathcal{E}$-unification algorithm can itself make use of narrowing at another level for finding the solution to its equational goals. Specific $\mathcal{E}$-unification algorithms exist for a small number of equational theories, but if the equational theory $(\Sigma, \mathcal{E})$ can be decomposed as $E \cup A$, where $A$ is a set of axioms having a unification algorithm, and the equations $E$ can be turned into a set of rules $\overrightarrow{E}$, by orienting them, such that the rewrite theory $\overrightarrow{\mathcal{E}} = (\Sigma, A, \overrightarrow{E})$ is admissible in the above sense, then narrowing can be used on $\overrightarrow{\mathcal{E}}$ to solve the $\mathcal{E}$-unification goals generated by performing narrowing on $\mathcal{R}$. For these equational goals the idea of *variants of a term* has been applied in recent years to narrowing. A strategy known as *folding variant narrowing* [13], which computes a complete set of variants of any term, has been developed by Escobar, Sasse, and Meseguer, allowing unification modulo a set of unconditional equations and axioms. The strategy terminates on any input term on those systems enjoying the *finite variant property*, and it is optimally terminating. It is being used for cryptographic protocol analysis [24], with tools like Maude-NPA [12], termination algorithms modulo axioms [9], algorithms for checking confluence and coherence of rewrite theories modulo axioms [10], and infinite-state model checking [4]. Recent development in conditional narrowing have been made for order-sorted equational theories [7] and also for narrowing with constraint solvers [26].

Conditional narrowing without axioms for rewrite theories with an order-sorted type structure has been thoroughly studied for increasingly complex categories of term rewriting systems. A wide survey can be found in [25]. The literature that can be found is scarce when we allow for extra variables in conditions (e.g. [15,16]), conditional narrowing modulo axioms (e.g. [7]), or conditional narrowing modulo a set of equations(e.g. [5]). Conditional narrowing modulo axioms for Mel theories has not been addressed, to the best of our knowledge, one of the main reasons being the lack of fast and effective unification algorithms modulo axioms for Mel theories. Nonetheless, there are plenty of algebraic data types, including all types that imply some kind of order between subterms, that are better expressed inside a Mel theory, so there is a need to give an answer to these cases. In this paper we focus on conditional narrowing modulo axioms for rewrite theories with an underlying equational Mel theory.

Our main contribution in this work is the proposal of two narrowing calculi for computing normalized answers to unification and reachability problems in membership equational logic and membership conditional rewrite theories, respectively. These calculi normalize all terms before applying any unification or reachability rule, and only generate normalized instantiations of reachability terms and intermediate (*matching*) variables, greatly reducing the state space. They have been proved sound and weakly complete, i.e., complete with respect to idempotent normalized answers.

The work is structured as follows: in Section 2 all needed definitions and properties for rewriting and narrowing are introduced. In Section 3 we present a rewrite theory, that only generates normalized substitutions on matching variables, for checking normalized solutions to unification problems. Section 4 introduces the narrowing calculus for equational unification. Section 5 introduces the narrowing calculus for reachability. Section 6 shows the calculi at work. In Section 7, related work, conclusions, and future lines of investigation for this work are presented. This paper is a continuation of a previous one [1], where non-normalized terms were allowed by the calculus.

## 2    Preliminaries

We assume familiarity with term rewriting and rewriting logic [6]. Rewriting logic is always parameterized by an underlying equational logic. This work is focused in membership equational logic [21], an equational logic that generalizes both many-sorted and order-sorted equational theories and that can also handle partial functions. There are several language implementations of rewriting logic, one of them being Maude [8], a language whose underlying logic is membership equational logic.

### 2.1    Search tree example

A search tree implementation will be used as running example to explain the definitions in a less abstract way. We review the needed terms, enclosing shortcuts for the definitions between brackets. We have Keys (abbreviated to k)

a, ..., f lexicographically ordered, and Values (v) 1, ..., 9. A pair Key Value forms a Record (r). A SearchTree (st) can be an EmptyTree (et), which we call empty, or non-empty, NeSearchTree (nt), containing in this case a Record at its root and two sub-SearchTrees, left and right. All the Keys in the left sub-SearchTree must be smaller than the key in the root and all the Keys in the right sub-SearchTree must be greater than the key in the root. t is the Boolean (b) result of a valid comparison (<), max and min return the Record on a NeSearchTree with the highest and lowest Key respectively, and key returns the Key of a Record. We also have a set of records RecordSet (rs) with none as identity atom, which are intended to be nondeterministically inserted (ins) in the SearchTree, hence yielding a NeSearchTree, deleted (del) (we admit deletion attempts on EmptyTrees), or omitted. Finally, there is a list Path (p), with nil as identity atom, holding the history - inserted, deleted, or omitted (i, d, o) - of already processed records. A triple SearchTree, RecordSet, Path forms a State (s).

## 2.2   Membership equational logic

Let's take a partially ordered set $(S, \leq)$ of *sorts*, whose *connected components* are the equivalence classes corresponding to the least equivalence relation $\equiv_\leq$ containing $\leq$.

A *membership equational logic* (MEL) *signature* [6] is defined by a *kind-complete* triple $\Sigma = (K, \Omega, S)$ meaning that:

- $K$ is a set of *kinds*.
- $S$ is split into a $K$-kinded family of disjoint sets of sorts $S_k$, i.e. $S = \bigcup_{k \in K} S_k$, such that if $s_i \leq s_j$ and $s_i \in S_k$ then $s_j \in S_k$. We write $[s_i] = k$ and say that the kind of $s_i$ is $k$, i.e., each connected component of $(S, \leq)$ has the same kind. $\leq$ is extended so that $s_i \leq k$ iff $s_i \in S_k$, i.e., $k$ is the top sort of its connected component (we also define $[k] = k$ if $k \in K$ for simplicity of notation).
- $\Omega = \{\Sigma_{\bar{\kappa}, \kappa}\}_{(\bar{\kappa}, \kappa) \in (K \cup S)^* \times (K \cup S)}$ is an algebraic signature of *function symbols*, where for each symbol $f \in \Sigma_{\kappa_1 \ldots \kappa_n, \kappa}$ if $n \geq 1$ and at least one of the subindices is not a kind, then there is another function symbol $f \in \Sigma_{[\kappa_1] \ldots [\kappa_n], [\kappa]}$.

When $f \in \Sigma_{\epsilon, \kappa}$ ($\epsilon$ is the empty word), we say that $f$ is a *constant* with *type* (meaning sort or kind) $\kappa$. We write $f \in \Sigma_\kappa$ instead of $f \in \Sigma_{\epsilon, \kappa}$.

If $f \in \Sigma_{\kappa_1 \ldots \kappa_n, \kappa}$, then we display $f$ as $f : \kappa_1 \ldots \kappa_n \to \kappa$, and say that $f$ has *arity n*. We call this a *rank* declaration for symbol $f$. Constant symbols have only one rank declaration $f : \to \kappa$ (plus the mandatory $f : \to [\kappa]$ if $\kappa$ is not a kind). We extend the order $\leq$ on $K \cup S$ to $(K \cup S)^*$, component-wise. Then $\Omega$ must also satisfy a *monotonicity condition*: $f \in \Sigma_{\kappa_1 \ldots \kappa_n, \kappa} \bigcap \Sigma_{\kappa'_1 \ldots \kappa'_n, \kappa'}$ and $\kappa_1 \ldots \kappa_n \leq \kappa'_1 \ldots \kappa'_n$ imply $\kappa \leq \kappa'$. If $f \in \Sigma_{\kappa_1 \ldots \kappa_n, \kappa}$ and $t_1, \ldots, t_n$ have type $\kappa_1$, ..., $\kappa_n$ respectively, then the term $f(t_1, \ldots, t_n)$ has type $\kappa$. If $\kappa \leq \kappa'$ and the term $t$ has type $\kappa$, then $t$ has also type $\kappa'$. This means that a term may have

several types. In fact, as for every sort $s$ we have that $s \leq [s]$, if a term has only one type then it must be a kind.

A MEL $\Sigma$-algebra $\mathcal{A}$ contains a set $\mathcal{A}_k$ for each kind $k \in K$, an $n$-ary function $\mathcal{A}_f : \mathcal{A}_{\kappa_1} \dots \mathcal{A}_{\kappa_n} \to \mathcal{A}_\kappa$ for each function $f \in \Sigma_{\kappa_1 \dots \kappa_n, \kappa}$, and a subset $\mathcal{A}_s \subseteq \mathcal{A}_k$ for each sort $s \in S_k$ such that if $s_i \leq s_j$ then $\mathcal{A}_{s_i} \subseteq \mathcal{A}_{s_j}$, and if $f \in \Sigma_{\kappa_1 \dots \kappa_n, \kappa} \bigcap \Sigma_{\kappa'_1 \dots \kappa'_n, \kappa'}$ and $\kappa_1 \dots \kappa_n \leq \kappa'_1 \dots \kappa'_n$ then $\mathcal{A}_f : \mathcal{A}_{\kappa_1} \dots \mathcal{A}_{\kappa_n} \to \mathcal{A}_\kappa$ equals $\mathcal{A}_f : \mathcal{A}_{\kappa'_1} \dots \mathcal{A}_{\kappa'_n} \to \mathcal{A}_{\kappa'}$ on $\mathcal{A}_{\kappa_1} \dots \mathcal{A}_{\kappa_n}$.

In membership equational logic the elements in a sort are well-defined, while the elements in a kind that don't belong to any sort are usually meant to refer to error or undefined elements. Kinds also provide a general way of dealing with partial functions in equational specifications. For instance, in the search tree example a `NeSearchTree` must have its `Keys` correctly ordered. Otherwise we have an error term with kind `[NeSearchTree]` (we haven't defined a sort `Tree`), so the constructor function for `NeSearchTrees` becomes total on `[NeSearchTree]`.

We allow *mix-fix* notation in $\Omega$, where the symbol _ is used to identify the position of each $\kappa_i \in \bar{\kappa}$. For instance, $\_ < \_ : Int\ Int \to Bool$ is a rank declaration stating that $5 < 4$ is a term with sort *Bool*. If omitted we assume the usual functional notation $f(\kappa_1, \dots, \kappa_n)$, which is an alternative notation admitted for all functions. We call $\mathcal{X} = \bigcup_{\kappa \in (K \cup S)} \mathcal{X}_\kappa$, where $\mathcal{X}_\kappa = \{x^i_\kappa\}$ for $\kappa \in (K \cup S)$, is a family of pairwise disjoint infinitely countable sets of variables. If $\kappa$ is a sort then $x^i_\kappa$ has sort $\kappa$ (and kind $[\kappa]$), otherwise $x^i_\kappa$ has kind $\kappa$ but no sort. The set of variables is potentially infinite, but any computation will only require a finite number of variables. A term that has no variables in it is said to be *ground*. A term where each variable occurs only once is said to be *linear* (ground terms are linear).

The sets $T_{\Sigma, \kappa}$, $T_\Sigma(\mathcal{X})_\kappa$ denote, respectively, the set of ground $\Sigma$-terms with sort or kind $\kappa$ and the set of $\Sigma$-terms with sort or kind $\kappa$ over $\mathcal{X}$. We ambiguously use the notation $T_\Sigma$ to refer to the initial $\Sigma$-algebra and as a shortcut for $\bigcup_{\kappa \in (K \cup S)} T_{\Sigma, \kappa}$. We also ambiguously use the notation $T_\Sigma(\mathcal{X})$ to refer to the free $\Sigma$-algebra on $\mathcal{X}$ and as a shortcut for $\bigcup_{\kappa \in (K \cup S)} T_\Sigma(\mathcal{X})_\kappa$. $Var(t) \subseteq \mathcal{X}$ denotes the set of variables in $t \in T_\Sigma(\mathcal{X})$. $\Sigma$ is assumed to be *sensible* meaning that if $f \in \Sigma_{\kappa_1 \dots \kappa_n, \kappa}$, $f \in \Sigma_{\kappa'_1 \dots \kappa'_n, \kappa'}$ and $[\kappa_i] = [\kappa'_i]$ for $i = 1, \dots, n$ then $[\kappa] = [\kappa']$. We also assume that $\Sigma$ has non-empty sorts, i.e., $T_{\Sigma, s} \neq \emptyset$ for all $s \in S$.

In the search tree example we have, omitting the implied kinded definition for each function in $\Omega$, that $\Sigma = (K, \Omega, S)$ is:

$K = \{[\mathtt{rs}], [\mathtt{st}], [\mathtt{k}], [\mathtt{v}], [\mathtt{b}], [\mathtt{p}], [\mathtt{s}]\}, S = \{\mathtt{r}, \mathtt{et}, \mathtt{st}, \mathtt{nt}, \mathtt{k}, \mathtt{v}, \mathtt{b}, \mathtt{rs}, \mathtt{p}, \mathtt{s}\},$

$S_{[\mathtt{rs}]} = \{\mathtt{r}, \mathtt{rs}\}, S_{[\mathtt{st}]} = \{\mathtt{et}, \mathtt{nt}, \mathtt{st}\}, S_{[\mathtt{k}]} = \{\mathtt{k}\}, S_{[\mathtt{v}]} = \{\mathtt{v}\}, S_{[\mathtt{b}]} = \{\mathtt{b}\},$

$S_{[\mathtt{p}]} = \{\mathtt{p}\}, S_{[\mathtt{s}]} = \{\mathtt{s}\},$

$\Omega = \{\{\_ < \_\}_{\mathtt{k\ k,b}}, \{\_\_\}_{\mathtt{k\ v,r}}, \{\_ ; \_\}_{\mathtt{rs\ rs,rs}}, \{\mathtt{key}\}_{\mathtt{r,k}}, \{\_\mathtt{i}\_, \_\mathtt{d}\_, \_\mathtt{o}\_\}_{\mathtt{p\ r,p}},$

$\{\mathtt{ins}\}_{\mathtt{st\ k\ v,nt}}, \{\mathtt{del}\}_{\mathtt{st\ k,st}}, \{\_|\_|\_\}_{\mathtt{st\ rs\ p,s}}, \{\_[\_]\_\}_{\mathtt{st\ r\ st,nt}},$

$\{\mathtt{min}, \mathtt{max}\}_{\mathtt{nt,r}}, \{\mathtt{a}, \dots, \mathtt{f}\}_{\mathtt{k}}, \{1, \dots, 9\}_{\mathtt{v}}, \{\mathtt{t}\}_{\mathtt{b}}, \{\mathtt{empty}\}_{\mathtt{et}}, \{\mathtt{none}\}_{\mathtt{rs}}, \{\mathtt{nil}\}_{\mathtt{p}}\}.$

We explain the notation used in $\Omega$: $\{\_[\_]\_\}_{\mathtt{st\ r\ st,nt}}$ means that there is a mix-fix function symbol $\_[\_]\_$ such that if $t_1, t_3$ are terms with sort `SearchTree` and $t_2$ is a term with sort `Record` then $t_1[t_2]t_3$ is a term with sort `NeSearchTree`.

*Positions* in a term $t$: as previously said, a term $t$ can be always expressed in functional notation as $f(t_1, \ldots, t_n)$. Then we can picture $t$ as a tree with root $f$ and children $t_1$, $\ldots$, $t_n$. We refer to the root position of $t$ as $\epsilon$ and to the other positions of $t$ as strings of nonzero natural numbers, $i_1 \ldots i_m$, meaning the position $i_2 \ldots i_m$ of $t_{i_1}$. The set of positions of a term is written $Pos(t)$. The set of nonvariable positions of a term is written $Pos_\Sigma(t)$. $t|_p$ is the subtree of $t$ below position $p$. $t[u]_p$ is the replacement in $t$ of the subterm at position $p$ with a term $u$. As an example, if $t$ is $f(g(a,b),c)$, then $t|_1$ is $g(a,b)$, $t|_{12}$ is $b$, and $t[d]_{12}$ is $f(g(a,d),c)$.

A MEL signature $\Sigma$ is said to be *preregular* iff for each $n$, for every function symbol $f$ with arity $n$, and for every $\kappa_1 \ldots \kappa_n \in (K \cup S)^n$, if the set $S_f$ containing all the sorts $s'$ that appear in rank declarations in $\Sigma$ of the form $f : \kappa'_1 \ldots \kappa'_n \to \kappa'$ such that $\kappa_i \leq \kappa'_i$, for $1 \leq i \leq n$, is not empty (so a term $f(t_1, \ldots, t_n)$ where $t_i$ has type $\kappa_i$ for $1 \leq i \leq n$ would be a $\Sigma$-term), then $S_f$ has a least sort. Preregularity guarantees that every $\Sigma$-term $t$ has a *least sort*, denoted $ls(t)$, among all the sorts that $t$ has because of the different rank declarations that can be applied to $t$, which is the most accurate classification for $t$.

A *substitution* $\sigma : \mathcal{X} \to T_\Sigma(\mathcal{X})$ is a function that matches the identity function in all $\mathcal{X}$ except for a finite set of variables $\mathcal{Y} \subseteq \mathcal{X}$, verifying that for each variable $y_\kappa \in \mathcal{Y}$ we have that $ls(y_\kappa \sigma) \leq \kappa$. Substitutions are written as $\sigma = \{y^1_{\kappa_1} \mapsto t_1, \ldots, y^n_{\kappa_n} \mapsto t_n\}$ where $Dom(\sigma) = \{y^1_{\kappa_1}, \ldots, y^n_{\kappa_n}\}$ and $Ran(\sigma) = \bigcup_{i=1}^n Var(t_i)$. The identity substitution is displayed as $id$. Substitutions are homomorphically extended to terms in $T_\Sigma(\mathcal{X})$ (and also to the rest of syntactic structures introduced along this paper, such as equations, goals, etc.). The restriction $\sigma|_\mathcal{V}$ of $\sigma$ to a set of variables $\mathcal{V}$ is defined as $x\sigma|_\mathcal{V} = x\sigma$ if $x \in \mathcal{V}$ and $x\sigma|_\mathcal{V} = x$ otherwise. Composition of two substitutions is denoted by $\sigma\sigma'$, with $x(\sigma\sigma') = (x\sigma)\sigma'$. If $\sigma\sigma = \sigma$ we say that $\sigma$ is *idempotent*. For substitutions $\sigma$ and $\sigma'$ where $Dom(\sigma) \cap Dom(\sigma') = \emptyset$, we denote their union by $\sigma \cup \sigma'$.

A *$\Sigma$-equation* is an expression of the form $t = t'$. A *$\Sigma$-equation* $t = t'$ is said to be:

- *Regular* if $Var(t) = Var(t')$.
- *Sort-preserving* if for each substitution $\sigma$, we have $t\sigma \in T_\Sigma(\mathcal{X})_\kappa$ ($\kappa \in K \cup S$) implies $t'\sigma \in T_\Sigma(\mathcal{X})_\kappa$ and vice versa.
- *Left (or right) linear* if $t$ (resp. $t'$) is linear.
- *Linear* if it is both left and right linear.

A set of equations $E$ is said to be regular, or sort-preserving, or (left or right) linear, if each equation in it is so.

A MEL theory [6] is a pair $(\Sigma, \mathcal{E})$, where $\Sigma$ is a MEL signature and $\mathcal{E}$ is a finite set of (possibly labeled) MEL sentences, either conditional equations or conditional memberships of the forms:

$$t = t' \; if \; A_1 \wedge \ldots \wedge A_n, \qquad t : s \; if \; A_1 \wedge \ldots \wedge A_n,$$

where $t = t'$ is a $\Sigma$-equation, $t : s$, $s \in S$, is a unary membership predicate stating that $t$ is a term with sort $s$, provided that the condition holds, and each

$A_i$ can be of the form $t = t'$, $t : s$ or $t := t'$ (a *matching* equation). Matching equations are treated as ordinary $\Sigma$-equations. They are a warning that new *extra* variables appear in $t$, imposing a limitation in the syntax of admissible MEL theories, as we will see. We also admit unconditional sentences in $\mathcal{E}$. $x_{s_1} : s_2$ is an unconditional membership expressing $s_1 \leq s_2$. For each variable $x_s \in \mathcal{X}_s$, where $s \in S$, we have that $x_s : s \in \mathcal{E}$. As an exception, there are two types of unconditional memberships over kinds, instead of sorts, that are implied by the MEL signature: if $f \in \Sigma_{\kappa_1 \ldots \kappa_n, k}$, $k \in K$ then $f(x_{\kappa_1}, \ldots, x_{\kappa_n}) : k \in \mathcal{E}$; also for each variable $x_\kappa \in \mathcal{X}_\kappa$ such that $[\kappa] = k$, $x_\kappa : k \in \mathcal{E}$.

Throughout this paper we will assume that all signatures are preregular and all their equations and memberships $t = t'$, $t := t'$ and $t : s$, $s \in S$, satisfy the conditions $[ls(t)] = [ls(t')]$ and $[ls(t)] = [s]$, that is, they are well-formed.

Given a MEL sentence $\phi$, we denote by $\mathcal{E} \vdash \phi$ the fact that $\phi$ can be deduced from $\mathcal{E}$ using the rules in Figure 1 [3,6]; for an equation $t = t'$, $\mathcal{E} \vdash t = t'$ is also written $t =_{\mathcal{E}} t'$. These rules, where the symbol = stands for = or := indistinctly, specify a sound and complete calculus. A MEL theory $(\Sigma, \mathcal{E})$ has an *initial algebra* $(T_{\Sigma/\mathcal{E}})$, whose elements are equivalence classes $[t]_{\mathcal{E}} \subseteq T_\Sigma$ of ground terms identified by the equations in $\mathcal{E}$.

$$\frac{t \in T_\Sigma(\mathcal{X})}{t = t} \text{ Reflexivity} \qquad \frac{t = t'}{t' = t} \text{ Symmetry}$$

$$\frac{t':s \quad t = t'}{t:s} \text{ Membership} \qquad \frac{t_1 = t_2 \quad t_2 = t_3}{t_1 = t_3} \text{ Transitivity}$$

$$\frac{f \in \Sigma_{k_1 \ldots k_n, k} \quad t_i = t_i' \quad t_i, t_i' \in T_\Sigma(X)_{k_i}, 1 \leq i \leq n}{f(t_1, \ldots, t_n) = f(t_1', \ldots, t_n')} \text{ Congruence}$$

$$\frac{(\; A_0 \text{ if } \bigwedge_i A_i) \in E \quad \sigma : X \to T_\Sigma(Y) \quad A_1\sigma \ldots A_n\sigma}{A_0\sigma} \text{ Replacement}$$

**Fig. 1.** Deduction rules for membership equational logic.

The MEL theory for the search tree example consists of $\Sigma = (K, \Omega, S)$ and the following set $\mathcal{E}$ of MEL sentences, one of them labeled (i1), where the first line of MEL sentences represents the subsort ordering in $S$. We omit the implicit subsorts for each kind, and the implicit memberships for each variable and kinded function. For executability requirements of the theory, that will be later defined, associativity, commutativity, and identity axioms are defined over kinds:

$x_{\mathtt{nt}} : \mathtt{st}$, $x_{\mathtt{et}} : \mathtt{st}$, $x_{\mathtt{r}} : \mathtt{rs}$,
$(x_{[\mathtt{rs}]}; y_{[\mathtt{rs}]}); z_{[\mathtt{rs}]} = x_{[\mathtt{rs}]}; (y_{[\mathtt{rs}]}; z_{[\mathtt{rs}]})$ (associativity),
$x_{[\mathtt{rs}]}; y_{[\mathtt{rs}]} = y_{[\mathtt{rs}]}; x_{[\mathtt{rs}]}$ (commutativity), $x_{[\mathtt{rs}]}; \mathtt{none} = x_{[\mathtt{rs}]}$ (identity),
$\mathtt{empty}[x_{\mathtt{r}}]\mathtt{empty} : \mathtt{nt}$, $\mathtt{empty}[x_{\mathtt{r}}]r_{\mathtt{nt}} : \mathtt{nt}$ *if* $\mathtt{key}(x_{\mathtt{r}}) < \mathtt{key}(\mathtt{min}(r_{\mathtt{nt}})) = \mathtt{t}$,
$l_{\mathtt{nt}}[x_{\mathtt{r}}]\mathtt{empty} : \mathtt{nt}$ *if* $\mathtt{key}(\mathtt{max}(l_{\mathtt{nt}})) < \mathtt{key}(x_{\mathtt{r}}) = \mathtt{t}$,
$l_{\mathtt{nt}}[x_{\mathtt{r}}]r_{\mathtt{nt}} : \mathtt{nt}$ *if* $\mathtt{key}(\mathtt{max}(l_{\mathtt{nt}})) < \mathtt{key}(x_{\mathtt{r}}) = \mathtt{t} \wedge \mathtt{key}(x_{\mathtt{r}}) < \mathtt{key}(\mathtt{min}(r_{\mathtt{nt}})) = \mathtt{t}$,
$\mathtt{a} < \mathtt{b} = \mathtt{t} \ldots \mathtt{a} < \mathtt{f} = \mathtt{t}$, $\mathtt{b} < \mathtt{c} = \mathtt{t} \ldots \mathtt{b} < \mathtt{f} = \mathtt{t} \ldots \mathtt{e} < \mathtt{f} = \mathtt{t}$, $\mathtt{key}(y_{\mathtt{k}}z_{\mathtt{v}}) = y_{\mathtt{k}}$,
$\mathtt{min}(\mathtt{empty}[x_{\mathtt{r}}]r_{\mathtt{st}}) = x_{\mathtt{r}}$, $\mathtt{min}(l_{\mathtt{nt}}[x_{\mathtt{r}}]r_{\mathtt{nt}}) = \mathtt{min}(l_{\mathtt{nt}})$,

$\mathtt{max}(l_{\mathrm{st}}[x_{\mathrm{r}}]\mathtt{empty}) = x_{\mathrm{r}}$, $\mathtt{max}(l_{\mathrm{nt}}[x_{\mathrm{r}}]r_{\mathrm{nt}}) = \mathtt{max}(r_{\mathrm{nt}})$,
$\mathtt{ins}(\mathtt{empty}, y_{\mathrm{k}}, z_{\mathrm{v}}) = \mathtt{empty}[y_{\mathrm{k}}, z_{\mathrm{v}}]\mathtt{empty}$,
$\mathtt{ins}(l_{\mathrm{st}}[y_{\mathrm{k}}z_{\mathrm{v}}']r_{\mathrm{st}}, y_{\mathrm{k}}, z_{\mathrm{v}}) = l_{\mathrm{st}}[y_{\mathrm{k}}z_{\mathrm{v}}]r_{\mathrm{st}}$,
(i1) $\mathtt{ins}(l_{\mathrm{st}}[x_{\mathrm{r}}]r_{\mathrm{st}}, y_{\mathrm{k}}, z_{\mathrm{v}}) = \mathtt{ins}(l_{\mathrm{st}}, y_{\mathrm{k}}, z_{\mathrm{v}})[x_{\mathrm{r}}]r_{\mathrm{st}}$ $if$ $y_{\mathrm{k}} < \mathtt{key}(x_{\mathrm{r}}) = \mathtt{t}$,
$\mathtt{ins}(l_{\mathrm{st}}[x_{\mathrm{r}}]r_{\mathrm{st}}, y_{\mathrm{k}}, z_{\mathrm{v}}) = l_{\mathrm{st}}[x_{\mathrm{r}}]\mathtt{ins}(r_{\mathrm{st}}, y_{\mathrm{k}}, z_{\mathrm{v}})$ $if$ $\mathtt{key}(x_{\mathrm{r}}) < y_{\mathrm{k}} = \mathtt{t}$,
$\mathtt{del}(\mathtt{empty}, y_{\mathrm{k}}) = \mathtt{empty}$, $\mathtt{del}(\mathtt{empty}[y_{\mathrm{k}}z_{\mathrm{v}}]r_{\mathrm{st}}, y_{\mathrm{k}}) = r_{\mathrm{st}}$,
$\mathtt{del}(l_{\mathrm{st}}[y_{\mathrm{k}}z_{\mathrm{v}}]\mathtt{empty}, y_{\mathrm{k}}) = l_{\mathrm{st}}$,
$\mathtt{del}(l_{\mathrm{nt}}[y_{\mathrm{k}}z_{\mathrm{v}}]r_{\mathrm{nt}}, y_{\mathrm{k}}) = l_{\mathrm{nt}}[\mathtt{min}(r_{\mathrm{nt}})]\mathtt{del}(r_{\mathrm{nt}}, \mathtt{key}(\mathtt{min}(r_{\mathrm{nt}})))$,
$\mathtt{del}(l_{\mathrm{st}}[x_{\mathrm{r}}]r_{\mathrm{st}}, y_{\mathrm{k}}) = \mathtt{del}(l_{\mathrm{st}}, y_{\mathrm{k}})[x_{\mathrm{r}}]r_{\mathrm{st}}$ $if$ $y_{\mathrm{k}} < \mathtt{key}(x_{\mathrm{r}}) = \mathtt{t}$,
$\mathtt{del}(l_{\mathrm{st}}[x_{\mathrm{r}}]r_{\mathrm{st}}, y_{\mathrm{k}}) = l_{\mathrm{st}}[x_{\mathrm{r}}]\mathtt{del}(r_{\mathrm{st}}, y_{\mathrm{k}})$ $if$ $\mathtt{key}(x_{\mathrm{r}}) < y_{\mathrm{k}} = \mathtt{t}$.

These axioms correspond to an algebraic specification of search trees similar to the one found in [8].

### 2.3   Unification

Given a MEL theory $(\Sigma, \mathcal{E})$, the $\mathcal{E}$-*subsumption* preorder $\ll_{\mathcal{E}}$ on $T_{\Sigma}(\mathcal{X})_k$ is defined by $t \ll_{\mathcal{E}} t'$ if there is a substitution $\sigma$ such that $t =_{\mathcal{E}} t'\sigma$. For substitutions $\sigma, \rho$ and a set of variables $\mathcal{V}$ we define $\sigma|_{\mathcal{V}} \ll_{\mathcal{E}} \rho|_{\mathcal{V}}$ if there is a substitution $\eta$ such that $\sigma|_{\mathcal{V}} =_{\mathcal{E}} (\rho\eta)|_{\mathcal{V}}$. Then we say that $\rho$ is more general than $\sigma$ with respect to $\mathcal{V}$. When $\mathcal{V}$ is not specified, we assume that $Dom(\rho) \subseteq Dom(\sigma)$ and say that $\rho$ is more general than $\sigma$.

A *system of equations* $F$ is a conjunction of the form $t_1 = t_1' \wedge \ldots \wedge t_n = t_n'$ where for $1 \leq i \leq n$, $t_i = t_i'$ is a $\Sigma$-equation. We define $Var(F) = \bigcup_i (Var(t_i) \cup Var(t_i'))$. An $\mathcal{E}$-*unifier* for $F$ is a substitution $\sigma$ such that $t_i\sigma =_{\mathcal{E}} t_i'\sigma$ for $1 \leq i \leq n$. For $\mathcal{V} = Var(F) \subseteq \mathcal{W}$, a set of substitutions $CSU_{\mathcal{E}}^{\mathcal{W}}(F)$ is said to be a *complete set of unifiers modulo $\mathcal{E}$* of $F$ away from $\mathcal{W}$ if

- each $\sigma \in CSU_{\mathcal{E}}^{\mathcal{W}}(F)$ is an $\mathcal{E}$-unifier of $F$;
- for any $\mathcal{E}$-unifier $\rho$ of $F$ there is a $\sigma \in CSU_{\mathcal{E}}^{\mathcal{W}}(F)$ such that $\rho|_{\mathcal{V}} \ll_{\mathcal{E}} \sigma|_{\mathcal{V}}$;
- for all $\sigma \in CSU_{\mathcal{E}}^{\mathcal{W}}(F)$, $Dom(\sigma) \subseteq \mathcal{V}$ and $Ran(\sigma) \cap \mathcal{W} = \emptyset$.

An $\mathcal{E}$-unification algorithm is *complete* if for any given system of equations it generates a complete set of $\mathcal{E}$-unifiers, which may not be finite. A unification algorithm is said to be *finite* and complete if it terminates after generating a finite and complete set of solutions.

### 2.4   Rewriting logic

A rewrite theory $\mathcal{R} = (\Sigma, \mathcal{E}, R)$ consists of a MEL theory $(\Sigma, \mathcal{E})$ together with a finite set $R$ of *conditional rewrite rules* each of which has the form

$$l \to r \; if \; \bigwedge_h p_h = q_h \wedge \bigwedge_i u_i := v_i \wedge \bigwedge_j w_j : s_j \wedge \bigwedge_k l_k \to r_k,$$

where $l, r$, and also each pair $l_k, r_k$, are $\Sigma$-terms of the same kind, and the rest of conditions fulfill the same requirements pointed out for MEL sentences. We will sometimes write $l \to r \; if \; c$ as a shortcut. Rewrite rules can also be

unconditional. Equational and membership conditions are intended to be solved within the MEL theory $(\Sigma, \mathcal{E})$, i.e., no rewriting with rules from $R$ is allowed on those conditions, whereas *reachability conditions* $l_k \rightarrow r_k$ are intended to be inferred using the deduction rules for rewrite theories below.

Such a rewrite rule specifies a *one-step transition* from a state $t[l\theta]_p$ to the state $t[r\theta]_p$, denoted by $t[l\theta]_p \rightarrow^1_R t[r\theta]_p$, provided the condition holds. The subterm $t|_p$ is called a *redex*.

In the search tree example, $R$ has as elements the nondeterministic (labeled) rewrite rules:

(I1) $x_{\mathrm{st}} \mid y_{\mathrm{k}} z_{\mathrm{v}}; v_{\mathrm{rs}} \mid w_{\mathrm{p}} \rightarrow \mathrm{ins}(x_{\mathrm{st}}, y_{\mathrm{k}}, z_{\mathrm{v}}) \mid v_{\mathrm{rs}} \mid w_{\mathrm{p}} \, \mathrm{i} \, y_{\mathrm{k}} \, z_{\mathrm{v}}$.

(D1) $x_{\mathrm{st}} \mid y_{\mathrm{k}} z_{\mathrm{v}}; v_{\mathrm{rs}} \mid w_{\mathrm{p}} \rightarrow \mathrm{del}(x_{\mathrm{st}}, y_{\mathrm{k}}) \mid v_{\mathrm{rs}} \mid w_{\mathrm{p}} \, \mathrm{d} \, y_{\mathrm{k}} \, z_{\mathrm{v}}$.

(O1) $x_{\mathrm{st}} \mid y_{\mathrm{k}} z_{\mathrm{v}}; v_{\mathrm{rs}} \mid w_{\mathrm{p}} \rightarrow x_{\mathrm{st}} \mid v_{\mathrm{rs}} \mid w_{\mathrm{p}} \, \mathrm{o} \, y_{\mathrm{k}} \, z_{\mathrm{v}}$.

All three rules have the same left term $x_{\mathrm{st}} \mid y_{\mathrm{k}} z_{\mathrm{v}}; v_{\mathrm{rs}} \mid w_{\mathrm{p}}$, hence the nondeterminism of its application. As an example, rule (I1) states that from a `State` formed by any `SearchTree` ($x_{\mathrm{st}}$), a non-empty `RecordSet` ($y_{\mathrm{k}} z_{\mathrm{v}}; v_{\mathrm{rs}}$), and any `Path` ($w_{\mathrm{p}}$), we can reach the `State` formed by the `NeSearchTree` obtained by inserting the `Record` $y_{\mathrm{k}} z_{\mathrm{v}}$ into $x_{\mathrm{st}}$, the `RecordSet` $v_{\mathrm{rs}}$, and the `Path` formed with the function symbol `_i_` applied to the `Path` $w_{\mathrm{p}}$ and the `Record` $y_{\mathrm{k}} z_{\mathrm{v}}$. Recall that as `RecordSets` are commutative and associative, the `Record` is also nondeterministically chosen.

The inference rules for rewrite theories [3, 6] in Figure 2 specify a sound and complete calculus for the system specified by $\mathcal{R}$. We can reach a state $v$ from a state $u$, written $\mathcal{R} \vdash u \rightarrow v$, if $u \rightarrow v$ can be inferred from $\mathcal{R}$ using these rules.

$$\frac{t \in T_\Sigma(\mathcal{X})}{t \rightarrow t} \; \texttt{Reflexivity} \qquad \frac{t_1 \rightarrow t_2, t_2 \rightarrow t_3}{t_1 \rightarrow t_3} \; \texttt{Transitivity}$$

$$\frac{f \in \Sigma_{k_1 \cdots k_n, k} \quad t_i \rightarrow t'_i \quad t_i, t'_i \in T_\Sigma(\mathcal{X})_{k_i}, 1 \leq i \leq n}{f(t_1, \ldots, t_n) \rightarrow f(t'_1, \ldots, t'_n)} \; \texttt{Congruence}$$

$$\frac{\begin{array}{c}(l \rightarrow r \; \texttt{if} \; \bigwedge_i p_i = q_i \wedge \bigwedge_j w_j : s_j \wedge \bigwedge_k l_k \rightarrow r_k) \in R \quad \theta : X \rightarrow T_\Sigma(Y) \\ \mathcal{E} \vdash p_i\theta = q_i\theta \; \text{for all } i, \quad \mathcal{E} \vdash w_j\theta : s_j \; \text{for all } j, \quad l_k\theta \rightarrow r_k\theta \; \text{for all } k\end{array}}{l\theta \rightarrow r\theta} \; \texttt{Replacement}$$

**Fig. 2.** Deduction rules for rewrite theories.

The relation $\rightarrow^1_{R/\mathcal{E}}$ on $T_\Sigma(\mathcal{X})$ is $=_\mathcal{E} \circ \rightarrow^1_R \circ =_\mathcal{E}$. $\rightarrow^1_{R/\mathcal{E}}$ on $T_\Sigma(\mathcal{X})$ induces a relation $\rightarrow^1_{R/\mathcal{E}}$ on $T_{\Sigma/\mathcal{E}}(\mathcal{X})$, the equivalence relation modulo $\mathcal{E}$, by $[t]_\mathcal{E} \rightarrow^1_{R/\mathcal{E}} [t']_\mathcal{E}$ iff $t \rightarrow^1_{R/\mathcal{E}} t'$. The transitive (resp. transitive and reflexive) closure of $\rightarrow^1_{R/\mathcal{E}}$ is denoted $\rightarrow^+_{R/\mathcal{E}}$ (resp. $\rightarrow^*_{R/\mathcal{E}}$).

A rewrite rule $l \rightarrow r \; \texttt{if} \; c$, is *sort-decreasing* if for each substitution $\sigma$ we have that $l\sigma \in T_\Sigma(\mathcal{X})_\kappa$ ($\kappa \in K \cup S$) and $c\sigma$ is verified implies $r\sigma \in T_\Sigma(\mathcal{X})_\kappa$.

For any relation $\rightarrow^1_R$ we say that a term $t$ is $\rightarrow_R$-*irreducible* (or just $R$-irreducible) if there is no term $t'$ such that $t \rightarrow^1_R t'$ and we say that a substitution

is *R-normalized* (or normalized if $R$ can be deduced from the context) if $x\sigma$ is $R$-irreducible for all $x \in Dom(\sigma)$. We also say that a term $t$ is *strongly R-irreducible* if for every $R$-normalized substitution $\sigma$ the term $t\sigma$ is $R$-irreducible.

The relation $\to^1_R$ is *terminating* if there are no infinite rewriting sequences in $\to^1_R$. The relation $\to^1_R$ is *confluent* if whenever $t\to^*_R t'$ and $t\to^*_R t''$, there exists a term $t'''$ such that $t'\to^*_R t'''$ and $t''\to^*_R t'''$. In a confluent, terminating, sort-decreasing, membership rewrite theory, for each term $t \in T_\Sigma(\mathcal{X})$, there is a unique (up to $\mathcal{E}$-equivalence) $R/\mathcal{E}$-irreducible term $t'$ obtained by rewriting to *canonical* form, denoted by $t \to^!_{R/\mathcal{E}} t'$, or $t \downarrow_{R/\mathcal{E}}$ when $t'$ is not relevant, which we call $can_{R/\mathcal{E}}(t)$.

### 2.5   Executable rewrite theories

For a rewrite theory $\mathcal{R} = (\Sigma, \mathcal{E}, R)$, whether a one step rewrite $t \to^1_{R/\mathcal{E}} t'$ holds is undecidable in general, because it involves searching a potentially infinite, and even non-computable, set $[t]_\mathcal{E}$ and checking if for any of its elements $t_i$ we have that $t_i \to^1_R t''$ and $t'' =_\mathcal{E} t'$. The approach taken to solve this problem is to decompose $\mathcal{E}$ into a disjoint union $E \cup A$, with $A$ a set of equational axioms (such as associativity, and/or commutativity, and/or identity) and define a new relation on $T_\Sigma(\mathcal{X})$ which, under certain assumptions on $\mathcal{R}$, will make $t \to^1_{R/\mathcal{E}} t'$ decidable.

**Associated rewrite theory.** Any MEL theory $(\Sigma, E \cup A)$ has a corresponding rewrite theory $\mathcal{R}_E = (\Sigma', A, R_E)$ associated to it [9], that allows us to check solutions for MEL conditions using rewriting instead of the deduction rules. Under certain restrictions this will be a finite process. The associated rewrite theory is constructed in the following way: we add a new connected component with sort *Truth* and a constant $tt$ of this sort to $\Sigma$, and for each kind $k \in K$ a function symbol $eq_{k,k} : k\ k \to Truth$. We will use the symbols $\to^1$ and $\to$ to refer to single rewrite steps or full rewriting paths in this rewrite theory. There are rules $eq_{k,k}(x_k, x_k)\to tt$ in $R_E$ for each kind $k \in K$. For each equation or membership in $E$

$$t=t' \ if \ A_1 \wedge \ldots \wedge A_n \qquad t{:}s \ if \ A_1 \wedge \ldots \wedge A_n,$$

$R_E$ has a conditional rule or membership of the form

$$t\to t' \ if \ A_1' \wedge \ldots \wedge A_n' \qquad t{:}s \ if \ A_1' \wedge \ldots \wedge A_n'$$

where if $A_i \equiv t_i{:}s_i$ then $A_i'$ is $t_i{:}s_i$, if $A_i \equiv t_i{:}=t_i'$ then $A_i'$ is $t_i'\to t_i$, and if $A_i \equiv t_i=t_i'$ then $A_i'$ is $eq_{k_i,k_i}(t_i, t_i')\to tt$, where $k_i = [ls(t_i)]$.

The *inference rules for membership rewriting in $\mathcal{R}_E$* are the ones in Figure 3, adapted from [9, Fig. 4, p. 71], where the rules are defined for context-sensitive membership rewriting.

**Definition 1 (E,A Rewriting).** *The relation $\to^1_{E,A}$ is defined as: $t \to^1_{E,A} t'$ if there is a position $p \in Pos(t)$, a rule $l \to r \ if \ \bigwedge_{i\in I} A_i'$ in $R_E$, and a substitution $\sigma$ such that $t|_p =_A l\sigma$ (A-matching), $t' = t[r\sigma]_p$, and for all $i \in I$:*

$$\frac{t_1\to^1 t_2,\ t_2\to t_3}{t_1\to t_3}\ \texttt{Transitivity} \qquad \frac{t\to^1 t',t':s}{t:s}\ \texttt{Subject Reduction}$$

$$\frac{t=_A t'}{t\to t'}\ \texttt{Reflexivity} \qquad \frac{t_i\to^1 t_i'}{f(t_1,\ldots,t_i,\ldots,t_n)\to^1 f(t_1,\ldots,t_i',\ldots,t_n)}\ \texttt{Congruence}$$

$$l\to r\ if\ A_1'\wedge\ldots\wedge A_n'\in R_E\ and\ t=_A l\sigma$$
$$\frac{A_1'\sigma\ldots A_n'\sigma}{t\to^1 r\sigma}\ \texttt{Replacement}$$

$$l:s\ if\ A_1'\wedge\ldots\wedge A_n'\in R_E\ and\ t=_A l\sigma$$
$$\frac{A_1'\sigma\ldots A_n'\sigma}{t:s}\ \texttt{Membership}$$

**Fig. 3.** Inference rules for membership rewriting.

*i) If $A_i'$ is of the form $t_i\to t_i'$ then there is a term $t_i''$ such that $t_i\sigma\to^*_{E,A} t_i''=_A t_i'\sigma$.*

*ii) If $A_i'$ is of the form $t_i:s_i$ then there is a term $t_i''$, a conditional membership $u:s_i\ if\ \bigwedge_{j\in J} B_j'$ in $R_E$, and a substitution $\rho$ such that $t_i\sigma\to^*_{E,A} t_i''$, $t_i''=_A u\rho$, and $B_j'\rho$ satisfies one of these same two conditions for all $j\in J$.*

The relation should have been called $\to^1_{R_E,A}$, but we prefer $\to^1_{E,A}$ for simplicity. It is important to point out that after zero or more rewrite steps from $t_i\sigma$ in $\to^1_{E,A}$ we must check for $A$-equality the resulting term $t_i''$ against $t_i'\sigma$ in case *i*, and against $u\rho$ in case *ii*. This corresponds to an application of the reflexivity inference rule in $\mathcal{R}_E$, which allows us to derive a rewrite step from an $A$-equality without applying any rewrite rule from $R_E$. The substitutions $\sigma$ and $\rho$ are difficult to find, but the conditions that we are going to impose on the rewrite theories, in order to make them executable, will make this task decidable.

**Definition 2 ($R$,$A$ Rewriting).** *The relation $\to^1_{R,A}$ is defined as: $t\to^1_{R,A} t'$ if there is a position $p\in Pos(t)$, a rule $l\to r\ if\ \bigwedge_{i\in I} A_i$ in $R$, and a substitution $\sigma$ such that $t|_p=_A l\sigma$, $t'=t[r\sigma]_p$, and for all $i\in I$:*

- *if $A_i$ is of the form $t_i\to t_i'$ then there is a term $t_i''$ such that $t_i\sigma\to^*_{R,A} t_i''=_A t_i'\sigma$.*
- *if $A_i$ is of the form $t_i:s_i$ then there is a term $t_i''$, a conditional membership $u:s_i\ if\ \bigwedge_{j\in J} B_j'$ in $R_E$, and a substitution $\rho$ such that $t_i\sigma\to^*_{E,A} t_i''$, $t_i''=_A u\rho$, and $B_j'\rho$ satisfies this same condition or the following one for all $j\in J$.*
- *else we consider $A_i'$, as in $R_E$, which is of the form $t_i\to t_i'$. Then there must be a term $t_i''$ such that $t_i\sigma\to^*_{E,A} t_i''=_A t_i'\sigma$.*

We define: $\to_{E,A}$ as $\to^*_{E,A}\circ=_A$; $\to_{R,A}$ as $\to^*_{R,A}\circ=_A$; $\to^1_{R\cup E,A}$ as $\to^1_{R,A}\cup\to^1_{E,A}$; $\to_{R\cup E,A}$ as $\to^*_{R\cup E,A}\circ=_A$.

We have replaced searching in $\mathcal{E}$ with matching modulo and rewriting with $\to^1_{E,A}$. There are some problems with $\to^1_{E,A}$, and also with $\to^1_{R,A}$, that must be solved to make this approach executable. Consider a rewrite theory $\mathcal{R}$ with only one sort $s$, and whose only rule is $f(a,b)\to c$, where $f$ is associative and commutative. The term $f(f(a,a),b)$ is a normal form in $\to^1_{R,A}$, but $f(f(a,a),b)\to^1_{R/A}$

$f(a, c)$, because $f(f(a, a), b) =_A f(a, f(a, b))$, so the relations are different. This problem would not happen if $\mathcal{R}$ had another rule $f(x_s, f(a, b)) \to f(x_s, c)$ that could be applied on top of the term $f(f(a, a), b)$ with matching $x_s \mapsto a$, modulo associativity and commutativity, leading to $f(f(a, a), b) \to^1_{R,A} f(a, c)$. Rewrite theories that have these rules, avoiding such problems, are called *closed under A-extensions* [23].

A problem that can arise when trying to decide $t \to^*_R t'$ in a rewrite theory is that although $\to^1_R$ is terminating the proof of a condition may generate a recursive infinite check of conditions. This leads us to the notion of *operational termination*.

**Definition 3 (Operational Termination of $\to^1_R$).** *The relation $\to^1_R$ is operationally terminating if there are no infinite well-formed proof trees [18].*

This notion of operational termination was presented by Lucas, Marché and Meseguer [17] in an attempt to exclude those conditional term rewriting systems like the one consisting of the single conditional rule:

$$a \to b \ \ if \ f(a) \to b$$

The absence of unconditional rules makes the relation $\to$ trivially empty, hence terminating. Nevertheless, when trying to reduce the term $a$, most implementations will loop because of the following infinite derivation tree:

$$\frac{\dfrac{\dfrac{\cdots}{a \to b}}{f(a) \to b}}{a \to b}$$

The condition of operational termination states that such derivation trees don't exist. It may be argued that implementations can be enhanced to identify repeated terms in the derivation tree and block further derivations. This would not solve the problem. For instance, let's consider a specification of natural numbers in a MEL theory, which is easy to develop. In this specification we call the sort for natural numbers Nat, and the successor operation $s \in \Sigma_{\text{Nat,Nat}}$. Now we extend this specification with the declaration of a new sort Inf, with $\text{Inf} \leq \text{Nat}$, and add the following equations:

$$x_{\text{Inf}} : \text{Nat} \ (i.e. \ \text{Inf} \leq \text{Nat})$$

$$s(x_{\text{Nat}}) : \text{Inf} \ if \ s(s(x_{\text{Nat}})) : \text{Inf}$$

If we try to derive $s(y_{\text{Nat}}) : \text{Inf}$, we get the following infinite derivation tree:

$$\frac{\dfrac{\dfrac{\cdots}{s(s(s(y_{\text{Nat}}))) : \text{Inf}}}{s(s(y_{\text{Nat}})) : \text{Inf}}}{s(y_{\text{Nat}}) : \text{Inf}}$$

Now, implementations usually get stuck here and there are no repeated terms that they can use to end the infinite loop. To avoid this problem, we will restrict ourselves to operationally terminating rewrite theories.

Another problem with rewrite theories when trying to apply a rule $l \to r$ *if* $c$ is the value given to new *extra* variables in $c$, i.e., variables appearing in $c$ and not in $l$. In order not to have to "guess" these values, which may make a rewrite theory untractable, rewrite theories must be *admissible*. We will first define *deterministic* and *strongly deterministic* rewrites theories, which is admissibility for rewrite theories with no $\Sigma$-equations at all, and later extend the definition to cover all rewrite theories.

**Definition 4 (Deterministic Rewrite Theory).** *Let* $\mathcal{R} = (\Sigma, A, R)$ *be a rewrite theory. We call* $\mathcal{R}$ *deterministic iff for each* $l \to r$ *if* $\bigwedge_{i=1}^{n} A_i \in R$, $A_i$ *is of the form* $u_i \to v_i$ *or* $u_i : s$, *and for each* $i$, $1 \leq i \leq n$, *we have* $Var(u_i) \subseteq Var(l) \cup \bigcup_{j=1}^{i-1} Var(v_j)$.

In a deterministic rewrite theory, a condition $u_i \to v_i$ in a rule is satisfied if before attempting a rewriting step there exists a substitution $\sigma$ such that $u_i =_A v_i\sigma$ ($A$-matching). For executability purposes of rewriting and efficiency of narrowing we limit this checking only to normal forms by using strongly deterministic rewrite theories.

**Definition 5 (Strongly Deterministic Rewrite Theory).** *A deterministic rewrite theory* $\mathcal{R} = (\Sigma, A, R)$ *is called* strongly deterministic *iff for each* $l \to r$ *if* $\bigwedge_{i=1}^{n} A_i \in R$, *and for each* $i$, $1 \leq i \leq n$ *where* $A_i$ *is of the form* $u_i \to v_i$, $v_i$ *is strongly* $R, A$-*irreducible.*

As previously said $R/\mathcal{E}$-rewriting may be non-computable. Under certain conditions we can replace it with $R \cup E, A$-rewriting and $A$-matching, and make a rewrite theory *executable*.

**Definition 6 (Executable Rewrite Theory).** *A rewrite theory* $\mathcal{R} = (\Sigma, E \cup A, R)$ *is* executable, *and also its underlying* MEL *theory* $(\Sigma, E \cup A)$, *if* $\Sigma$ *is preregular modulo* $A$; $E$, $A$, *and* $R$ *are finite, and the following conditions hold:*

1. *$E$ and $R$ are admissible [8] and closed under $A$-extensions, and $\to_{E/A}^{1}$ and $\to_{R/A}^{1}$ are operationally terminating. Admissibility is a translation of strong determinism to* MEL *theories forcing all matching variables in conditions to be instantiated by $A$-matching.*
2. *The axioms in $A$ are regular, linear, and sort-preserving. Any variable appearing in an axiom must be a kinded variable. Furthermore, equality modulo $A$ must be decidable and there must exist a finite matching algorithm modulo $A$ producing a finite number of $A$-matching substitutions, $Match_A(t_1, t_2) = \{\sigma_i\}_{i=1}^{n}$ meaning that $t_1 =_A t_2\sigma_i$ for $i = 1, \ldots, n$, or failing otherwise.*
3. *The relation $\to_{E/A}^{1}$ is sort-decreasing, terminating, and confluent (where we again prefer to use $\to_{E/A}^{1}$ instead of $\to_{R_E/A}^{1}$).*
4. *$\to_{R,A}^{1}$ is $\mathcal{E}$-consistent with $\to_{E,A}^{1}$, i.e., for all $t_1, t_2, t_3$ we have $t_1 \to_{R,A}^{1} t_2$ and $t_1 \to_{E,A}^{*} t_3$ implies that there exist $t_4, t_5$ such that $t_3 \to_{E,A}^{*} t_4$, $t_4 \to_{R,A}^{1} t_5$, and $t_2 =_{\mathcal{E}} t_5$. We represent this property by using a diagram with filled lines for universal quantification and dotted lines for existential quantification:*

$$t_1 \xrightarrow[R,A]{1} t_2$$

$$E,A \Big\downarrow * \qquad \vdots \varepsilon$$

$$t_3 \xrightarrow[E,A]{*} t_4 \xrightarrow[R,A]{1} t_5$$

$\mathcal{E}$-consistency of $\to_{R,A}^1$ with $\to_{E,A}^1$

Under the above assumptions $\to_{E,A}^1$ is *strictly coherent*, i.e., for all $t_1, t_2, t_3$ if $t_1 \to_{E,A}^1 t_2$ and $t_1 =_A t_3$ then there exists $t_4$ such that $t_3 \to_{E,A}^1 t_4$ and $t_2 =_A t_4$ [23]. Also $\to_{R,A}^1$ is *strictly coherent*, i.e., for all $t_1, t_2, t_3$ we have $t_1 \to_{R,A}^1 t_2$ and $t_1 =_A t_3$ implies that there exists $t_4$ such that $t_3 \to_{R,A}^1 t_4$ and $t_2 =_A t_4$.

$$t_1 \xrightarrow[E,A]{1} t_2 \qquad\qquad\qquad t_1 \xrightarrow[R,A]{1} t_2$$

$$A \| \qquad \vdots A \qquad\qquad\qquad A \| \qquad \vdots A$$

$$t_3 \xrightarrow[E,A]{1} t_4 \qquad\qquad\qquad t_3 \xrightarrow[R,A]{1} t_4$$

(a) strict coherence of $\to_{E,A}^1$         (b) strict coherence of $\to_{R,A}^1$

Technically, strict coherence means that the weaker relations $\to_{E,A}^1$ and $\to_{R,A}^1$ become semantically equivalent to the stronger relations $\to_{E/A}^1$ and $\to_{R/A}^1$. Under these conditions we can implement $\to_{R/\mathcal{E}}^1$ on terms using $\to_{R\cup E,A}^1$.

The following lemma links $\to_{R/\mathcal{E}}^1$ with $\to_{E,A}^1$ and $\to_{R,A}^1$.

**Lemma 1.** *Let $\mathcal{R} = (\Sigma, \mathcal{E}, R)$ be an executable rewrite theory, that is, it has all the properties above. Then $t_1 \to_{R/\mathcal{E}}^1 t_2$ if and only if $t_1\!\downarrow \to_{R,A}^1 t_3$ for some $t_3 =_{\mathcal{E}} t_2$, which can be verified by checking $t_3\!\downarrow =_A t_2\!\downarrow$.*

**Lemma 2.** *The rewrite theory $\mathcal{R}_E = (\Sigma', A, R_E)$ associated to any executable* MEL *theory $(\Sigma, E \cup A)$ is strongly deterministic.*

The rewrite theory for the search tree example is executable if we decompose $\mathcal{E}$ in the following way: the set $A$ contains the associative, commutative, and identity equations in $\mathcal{E}$; the set $E$ contains the rest of equations and all memberships in $\mathcal{E}$.

## 2.6   Reachability goals

Given a rewrite theory $\mathcal{R} = (\Sigma, \mathcal{E}, R)$, a *reachability goal* $G$ is a conjunction of the form $t_1 \to^* t_1' \wedge \ldots \wedge t_n \to^* t_n'$ where for $1 \le i \le n$, $t_i, t_i' \in T_\Sigma(\mathcal{X})_{\kappa_i}$ for appropriate $\kappa_i$. We define $Var(G) = \bigcup_i Var(t_i) \cup Var(t_i')$. A substitution $\sigma$ is a *solution* of $G$ if $t_i\sigma \to_{R/\mathcal{E}}^* t_i'\sigma$ for $1 \le i \le n$. If the substitution is idempotent we also say that the solution is *idempotent*. We define $\mathrm{E}(G)$ to be the system of equations $t_1 = t_1' \wedge \ldots \wedge t_n = t_n'$. We say $\sigma$ is a *trivial solution* of $G$ if it is an $\mathcal{E}$-unifier for $\mathrm{E}(G)$. We say $G$ is trivial if the identity substitution $id$ is a trivial solution of $G$.

For goals $G : t_1 \to^* t_2 \wedge \ldots \wedge t_{2n-1} \to^* t_{2n}$ and $G' : t'_1 \to^* t'_2 \wedge \ldots \wedge t'_{2n-1} \to^* t'_{2n}$ we say $G =_{\mathcal{E}} G'$ if $t_i =_{\mathcal{E}} t'_i$ for $1 \leq i \leq 2n$. We say $G \to^1_R G'$ if there is an odd $i$ such that $t_i \to_R t'_i$ and for all $j \neq i$ we have $t_j = t'_j$. That is, $G$ and $G'$ differ only in one subgoal ($t_i \to t_{i+1}$ vs $t'_i \to t_{i+1}$), but $t_i \to t'_i$, so when we rewrite $t_i$ in $G$ to $t'_i$ we get $G'$. The relation $\to^1_{R/\mathcal{E}}$ over goals is defined as $=_{\mathcal{E}} \circ \to^1_R \circ =_{\mathcal{E}}$.

Systems of equations in $(\Sigma, E \cup A)$ with form $G \equiv \bigwedge_{i=1}^m (t_i = t'_i)$, with $[ls(t_i)] = k_i$, become reachability goals in $\mathcal{R}_E$ of the form $\bigwedge_{i=1}^m (eq_{k_i,k_i}(t_i, t'_i) \to tt)$. Using $\mathcal{R}_E$ we can verify whether a substitution $\sigma$ is a solution of $G$ by checking $eq_{k_i,k_i}(t_i\sigma, t'_i\sigma) \to tt$ for $i = 1, \ldots, m$. We will extend our notion of system of equations to include well-formed sentences $\bigwedge_{j=1}^n t_j : \kappa_j$, i.e. with $\kappa_j \in (S \cup K)$ and $[ls(t_j)] = [\kappa_j]$ for $j = 1, \ldots, n$, equivalent to an equation $t_j = x_{\kappa_j}$ with $x_{\kappa_j}$ a fresh variable, which become reachability goals of the same form. Then $\sigma$ is a solution of $G$ if $t_j\sigma : \kappa_j$ is derivable in $\mathcal{R}_E$ for $j = 1, \ldots, n$, and $eq_{k_i,k_i}(t_i\sigma, t'_i\sigma) \to tt$ for $i = 1, \ldots, m$.

### 2.7   Narrowing

Let $t$ be a $\Sigma$-term and $\mathcal{W}$ be a set of variables such that $Var(t) \subseteq \mathcal{W}$. The $R, A$-*narrowing* relation on $T_\Sigma(\mathcal{X})$ is defined as follows: $t \leadsto_{p,\sigma,R,A} t'$ if there is a non-variable position $p \in Pos_\Sigma(t)$, a rule $l \to r$ `if` $c \in R$, properly renamed, such that $Var(l) \cap \mathcal{W} = \emptyset$, and a unifier $\sigma' \in CSU_A^{\mathcal{W}'}(t|_p = l)$ for $\mathcal{W}' = \mathcal{W} \cup Var(l)$, such that $\sigma = \sigma'\sigma''$ for some $\sigma''$, $t' = (t[r]_p)\sigma$ and $c\sigma''$ holds. Similarly $E, A$-narrowing and $R \cup E, A$-narrowing relations are defined.

The substitution $\sigma''$ appears because the use of conditional rules usually forces the recursive resolution of $c\sigma'$ with narrowing, returning some $\sigma''$ as solution. Then $\sigma = \sigma'\sigma''$ is the desired substitution such that $t \leadsto_{p,\sigma,R,A} t'$.

*Example 1.* Consider a rewrite theory $\mathcal{R} = (\Sigma, E \cup A, R)$, where $S = \{s\}$, $\Omega = \{\{a, b, c\}_s, \{f, g\}_{ss,s}\}$, with $A = E = \emptyset$, and $R = \{g(b, c) \to c, f(a, z_s) \to b$ `if` $g(b, z_s) \to c\}$.

If we try to narrow the term $f(x_s, y_s)$ with rule $f(a, z_s) \to b$ `if` $g(b, z_s) \to c$ and unifier $\sigma' = \{x_s \mapsto a, y_s \mapsto w_s, z_s \mapsto w_s\}$ we have to prove the condition $g(b, w_s) \to c$, which can be narrowed with rule $g(b, c) \to c$ and substitution $\sigma'' = \{w_s \mapsto c\}$, so $g(b, z_s) \leadsto_{\sigma'',R,A} c$. Then, by composition of the substitutions $\sigma'$ and $\sigma''$, we get $\sigma = \{x_s \mapsto a, y_s \mapsto c, z_s \mapsto c\}$ and we have $f(x_s, y_s) \leadsto_{\sigma,R,A} b$. As a consequence, that will be later proved, $f(x_s, y_s)\sigma \to^1_{R,A} b$.

## 3   Sentence-normalized rewriting

In this paper we are going to prove properties of our calculus with respect to $R \cup E, A$-normalized solutions. Executable rewrite and MEL theories force us to check whether there is a match between $u_i$ and $v_i$ or not before each single rewriting step for any condition $u_i \to_{E,A} v_i$ or $u_i \to_{R,A} v_i$. We are going to restrict ourselves to *FPP (Fresh Pattern Property)* executable rewrite and MEL theories [7], meaning that any variable appearing in the left side of a matching

equation is new. These theories allow us to reduce every term to normal form before each narrowing step in a safe way.

When we check whether some substitution $\gamma$ is a solution of a reachability goal $G$ we can assume that $G\gamma$ is a ground term. If $Var(G\gamma) \neq \emptyset$ we can extend the signature $\Sigma'$ in a way that all these variables become constants in the extended signature. By doing so, the only variables that will appear at any rewriting step will be matching variables.

We are going to incrementally construct the substitution used in the replacement and membership rules on FPP executable MEL theories. From now on we will write $\downarrow$ instead of $\downarrow_{E,A}$. If the ground term $t$ matches $l$ using $\sigma_0$ ($t =_A l\sigma_0$), then $l\sigma_0$ is a ground term and $\sigma_0$ is a ground substitution. If $A_i$ has no matching variables then $\sigma_i = id$. If $A_i \equiv t'_i \rightarrow t_i$ has matching variables, then $Dom(\bigcup_{j=0}^{i-1} \sigma_j) \cap Var(t_i) = \emptyset$ and $eq_{k_i,k_i}(t_i \bigcup_{j=0}^{i-1} \sigma_j, t'_i \bigcup_{j=0}^{i-1} \sigma_j) = eq_{k_i,k_i}(t_i, t'_i \bigcup_{j=0}^{i-1} \sigma_j)$. Let $\sigma_i$ be an $A$-matching of $t_i$ with $(t'_i \bigcup_{j=0}^{i-1} \sigma_j)\downarrow$, that is $t_i\sigma_i =_A (t'_i \bigcup_{j=0}^{i-1} \sigma_j)\downarrow$, which is a ground term. As we are matching against an $E,A$-normalized ground term and $t_i$ is strongly irreducible, $\sigma_i$ must be ground $E,A$-normalized. The only exception is the first substitution $\sigma_0$ which is ground but may not be $E,A$-normalized. The *extended* substitution $\sigma$ that we need to apply the replacement rule or the membership rule is $\sigma = \bigcup_{i=0}^{n} \sigma_i$, where the instantiation of all matching variables, $\bigcup_{i=1}^{n} \sigma_i$, is ground $E,A$-normalized.

**Definition 7 (Sentence-Normalized Substitution).** *Given an FPP executable MEL theory $(\Sigma, E \cup A)$, its associated rewrite theory $\mathcal{R}_E = (\Sigma', A, R_E)$, for any conditional sentence, $c \equiv l \rightarrow r$ if $(\bigwedge_{i=1}^{n} A'_i)$ or $c \equiv l : s$ if $(\bigwedge_{i=1}^{n} A'_i)$, and substitution $\sigma$, the* sentence-normalized substitution $\sigma_c$ *is defined as $\sigma_c = \sigma|_{Var(l)} \cup \sigma\downarrow|_{Mat(c)}$, where $Mat(c)$ is the set of matching variables in $c$.*

**Proposition 1.** *Given an FPP executable MEL theory $(\Sigma, E \cup A)$, its associated rewrite theory $\mathcal{R}_E = (\Sigma', A, R_E)$, and a ground term $t \in T_\Sigma$, if $t \rightarrow^1 r\sigma$ or $t : s$ is derived with a conditional sentence $c \equiv l \rightarrow r$ if $\bigwedge_{i=1}^{n} A'_i$ or $c \equiv l : s$ if $\bigwedge_{i=1}^{n} A'_i$ and a substitution $\sigma$, then $t \rightarrow^1 r\sigma_c$ or $t : s$ is derivable using the same sentence $c$ and $\sigma_c$.*

We are interested in derivations for reachability goals where we only rewrite with sentence-normalized substitutions, hence reducing the state space.

**Definition 8 (Sentence-Normalized Rewriting).** *When we want to imply that only sentence-normalized substitutions are applied in a rewrite step or a derivation with $\rightarrow^1_{E,A}$, will use the terms* sentence-normalized rewriting (SNR) *or* SNR-derivable, *and write $t :_N s$, $t \rightarrow^1_N t'$, or $t \rightarrow_N t'$, .*

**Lemma 3 (Completeness of Sentence-Normalized Rewriting).** *Given an FPP executable MEL theory $(\Sigma, E \cup A)$ and its associated rewrite theory $\mathcal{R}_E = (\Sigma', A, R_E)$ if $eq_k(t,t') \rightarrow tt$, $t' \rightarrow t''$ or $t : s$, with $t''$ a normal form, are derivable in $\mathcal{R}_E$, then $eq_k(t,t') \rightarrow_N tt$, $t' \rightarrow_N t''$ or $t :_N s$ are SNR-derivable in $\mathcal{R}_E$ respectively.*

As a direct consequence we get the following theorem telling us that with respect to reachability goals the solutions are the same using $\rightarrow^1_{E,A}$ or $\rightarrow^1_N$.

**Theorem 1 (Equivalence of SNR for Reachability Goal Solutions).**
*Given an FPP executable* MEL *theory* $(\Sigma, E \cup A)$ *and its associated rewrite theory* $\mathcal{R}_E = (\Sigma', A, R_E)$, $\sigma$ *is a solution of* $\bigwedge_{i=1}^{m}(eq_{k_i,k_i}(t_i, t'_i) \to tt) \wedge \bigwedge_{j=1}^{n} t''_j : s_j$ *if and only if* $eq_{k_i,k_i}(t_i\sigma, t'_i\sigma) \to_N tt$, $i = 1, \ldots, m$, *and* $t''_j\sigma :_N s_j$, $j = 1, \ldots, n$, *are SNR-derivable.*

Now we prove that conditions and reduced conditions have the same solutions. This result is important because it will allow us to reduce the state space in our narrowing problems.

**Proposition 2.** *Given an FPP executable* MEL *theory* $(\Sigma, E \cup A)$ *and its associated rewrite theory* $\mathcal{R}_E = (\Sigma', A, R_E)$, *for any conditional* MEL *sentence* $c \equiv s$ *if* $\bigwedge_{i=1}^{n} A_i \in E$ *and corresponding rule or membership* $c' \equiv s'$ *if* $\bigwedge_{i=1}^{n} A'_i \in \mathcal{R}_E$ *if there is a substitution* $\sigma$ *such that* $A'_1 \downarrow \sigma_c, \ldots, A'_n \downarrow \sigma_c$ *are derivable in* $\mathcal{R}_E$ *then* $A'_1\sigma_c, \ldots, A'_n\sigma_c$ *are SNR-derivable in* $\mathcal{R}_E$.

**Proposition 3.** *Given an FPP executable* MEL *theory* $(\Sigma, E \cup A)$ *and its associated rewrite theory* $\mathcal{R}_E = (\Sigma', A, R_E)$, *for any conditional* MEL *sentence* $c \equiv s$ *if* $\bigwedge_{i=1}^{n} A_i \in E$ *and corresponding rule or membership* $c' \equiv s'$ *if* $\bigwedge_{i=1}^{n} A'_i \in \mathcal{R}_E$ *if there is a substitution* $\sigma$ *such that* $A'_1\sigma_c, \ldots, A'_n\sigma_c$ *are derivable in* $\mathcal{R}_E$ *then* $A'_1 \downarrow \sigma_c, \ldots, A'_n \downarrow \sigma_c$ *are SNR-derivable in* $\mathcal{R}_E$.

As SNR-derivations are derivations, we have proved the most strict properties in both directions. Again, as reachability goals are a special case of sentence conditions, we have as a direct consequence of the last two propositions the desired result.

**Lemma 4 (Equivalence of Solutions for Reduced Reachability Goals).**
*Given an FPP executable* MEL *theory* $(\Sigma, E \cup A)$ *and its associated rewrite theory* $\mathcal{R}_E = (\Sigma', A, R_E)$, $\sigma$ *is a solution of* $\bigwedge_{i=1}^{m}(eq_{k_i,k_i}(t_i, t'_i) \to tt) \wedge \bigwedge_{j=1}^{n} t''_j : s_j$ *if and only if it is a solution of* $\bigwedge_{i=1}^{m}(eq_{k_i,k_i}(t_i, t'_i)\downarrow \to tt) \wedge \bigwedge_{j=1}^{n} t''_j\downarrow : s_j$.

## 4  Conditional narrowing modulo unification

Narrowing allows us to assign values to variables in such a way that a reachability goal holds. We implement narrowing using a calculus with the following properties:

1. The calculus is *weakly complete*, i.e., for any idempotent $E, A$-normalized solution of a reachability goal $G$, the calculus can compute a more general answer for $G$.
2. The calculus is *sound*, i.e., if the calculus computes an answer $\sigma$ for a reachability goal $G$, then $\sigma$ is a solution of $G$.

We are going to split the calculus into two parts: the one that solves unification problems and the one that solves reachability problems.

### 4.1   Transformations and calculus rules for unification

We assume that we have an $A$-unification algorithm that returns a *complete set of unifiers modulo $A$ ($CSU_A$)* for any pair of terms. A unification equation has the form $t : \kappa = t' : \kappa'$, as a shorthand for $t = t' \wedge t : \kappa \wedge t' : \kappa'$. A unification goal is a conjunction of unification equations.

First we introduce an extension for the signature of the transformation described in Section 2.5 in which we add syntax useful for the calculus, to force left to right solving of subgoals, and change the rules in $R_E$ by turning conditional memberships into conditional rules, adding a new transformation for matching equations in conditions. Both changes will be useful in the context of the narrowing calculus.

**Transformations of the associated rewrite theory.** We associate to any FPP executable MEL theory $(\Sigma, E \cup A)$ another rewrite theory $\tilde{\mathcal{R}}_E = (\tilde{\Sigma}, A, \tilde{R}_E)$, where $\tilde{\Sigma}$ is an extension of $\Sigma'$ in $\mathcal{R}_E$. We add in $\tilde{\Sigma}$:

- A function symbol $\_ \wedge \_ : [Truth] \, [Truth] \to [Truth]$.
- A function symbol $eq_{s,s'} : [s] \, [s] \to Truth$ for each pair of sorts $s, s' \in S$ such that $[s] = [s']$
- A function symbol $\_ : \kappa : [\kappa] \to Truth$ for each $\kappa \in (S \cup K)$.
- For each conditional equation $(l = r \text{ if } \bigwedge_{i=1}^{n} A_i)$ or conditional membership $(l : \kappa \text{ if } \bigwedge_{i=1}^{n} A_i)$ in $E$, we add to $\tilde{R}_E$ a conditional rule $l \to r \text{ if } A_1^\bullet \wedge \ldots \wedge A_n^\bullet$ or $l : \kappa \to tt \text{ if } A_1^\bullet \wedge \ldots \wedge A_n^\bullet$, where each condition has an implicit $\to tt$, if $A_i \equiv t : s$ then $A_i^\bullet$ is $t : s$, and if $A_i \equiv t = t'$ or $A_i \equiv t := t'$ then $A_i^\bullet$ is $eq_{k,k}(t, t')$ with $k = [ls(t)]$.

A unification goal $\bigwedge_{i=1}^{n} (t_i : \kappa_i = t_i' : \kappa_i')$ has an associated narrowing problem $\bigwedge_{i=1}^{n} eq_{\kappa_i,\kappa_i'}(t_i, t_i') \downarrow \wedge \, tt$ (with an implicit $\rightsquigarrow tt$). The trailing $\wedge \, tt$ will allow us to work with a simpler set of calculus rules.

**Calculus rules for unification.** As we are computing $R \cup E, A$-normalized solutions, one general rule in our calculus is that we only apply a calculus rule if the composition of all computed substitutions remains $E, A$-normalized with respect to all matching variables and $R \cup E, A$-normalized with respect to the variables in the initial reachability problem. Our calculus is defined by the following inference rules, where $t'$ has sort $Truth$ and $t''$ has kind $[Truth]$:

- $[t]$ *transitivity*

$$\frac{t' \wedge t''}{t' \rightsquigarrow^1 x_{[Truth]}, x_{[Truth]} \wedge t''}$$

- $[n]$ *narrowing*

$$\frac{t \rightsquigarrow^1 x_k, t''}{(((c) \wedge t'')\rho\theta)\downarrow}$$

where $l \to r \ (if \ c) \in \tilde{R}_E$, $\theta \in CSU_A(t = l)$, $\rho = \{x_k \mapsto r\}$

- [c] *congruence*

$$\frac{f(\bar{t}) \rightsquigarrow^1 x_k, t''}{t_i \rightsquigarrow^1 x'_{k_i}, t''\theta}$$

  where $t_i \notin \mathcal{X}$, $\theta = \{x_k \mapsto f(t_1, \ldots, t_{i-1}, x'_{k_i}, t_{i+1}, \ldots, t_n)\}$, $k_i = [ls(t_i)]$

- [e] *elimination*

$$\frac{tt \wedge t''}{t''}$$

- [u] *unification*

$$\frac{eq_{\kappa,\kappa'}(t_1, t_2) \wedge t''}{((t_1 : \kappa'' \wedge t'')\theta)\downarrow}$$

  where $\theta \in CSU_A(t_1 = t_2)$, $\kappa''$ maximal such that $\kappa'' \leq \kappa, \kappa'' \leq \kappa'$

**Definition 9 (Computed Answer).** *Given a unification goal $u$ and the corresponding narrowing problem $G$, if there is a narrowing path from $G$ to $tt$, $G = G_0 \rightsquigarrow_{\sigma_1} G_1 \rightsquigarrow_{\sigma_2} \ldots \rightsquigarrow_{\sigma_n} tt$, where if in the step $i$ we apply the transitivity rule or the elimination rule then $\sigma_i = id$, and $\sigma = \sigma_1\sigma_2 \ldots \sigma_n$, then we write $G \rightsquigarrow^*_\sigma tt$ and call $\sigma|_{Var(G)}$ a computed answer for $u$.*

**Theorem 2.** *The calculus for unification is sound and weakly complete, i.e., complete with respect to $E, A$-normalized solutions.*

## 5  Reachability by conditional narrowing

In this part of the calculus we define a new type of reachability goal $G \equiv \bigwedge_{i=1}^n t_i{:}\kappa_i \Rightarrow t'_i{:}\kappa'_i$, where the $\kappa$'s are meant to carry sort information throughout the calculus in order to make use of it. Given an FPP executable rewrite theory with an underlying FPP MEL theory $\mathcal{R} = (\Sigma, E \cup A, R)$, we try to find an idempotent $R \cup E, A$-normalized solution $\sigma$ (ground or not) such that $t_i\sigma \rightarrow_{R \cup E,A} t'_i\sigma$, $t_i\sigma$ has type $\kappa_i$ and $t'_i\sigma$ has type $\kappa'_i$ for all $i = 1, \ldots, n$. Since we need additional syntax to handle reachability goals, we extend $\tilde{R}_E$ to $\tilde{R}$ in the following way:

- For each pair $\kappa, \kappa' \in (S \cup K)$ such that $[\kappa] = [\kappa']$ we add the function symbol $\_{:}\kappa \Rightarrow \_{:}\kappa'$ to $\Sigma_{\kappa\kappa', Truth}$ to handle the new reachability goals. As rewrite theories are not sort-decreasing in general, we need a function symbol for each pair of types with same kind.
- For each kind $k \in K$ we add the function symbol $\_\Rightarrow^1 \_$ to $\Sigma_{kk, Truth}$. This function symbol is *ad hoc* overloaded, i.e. the arguments of each overloaded function symbol have different kind, and stands for an actual narrowing step performed using the rules in $R$.

A conditional rewrite rule $l \rightarrow r$ *if* $\bigwedge_i A_i$ becomes $l \Rightarrow r$ *if* $\bigwedge_i A_i^\bullet$, where conditions of the form $l_i \rightarrow r_i$ become $l_i{:}k_i \Rightarrow r_i{:}k_i$, with $k_i = [ls(t_i)]$ and the rest of conditions are transformed as in the calculus for unification.

We turn the reachability goal $G$ into a narrowing problem $G' \equiv \bigwedge_i t_i\downarrow{:}\kappa_i \Rightarrow t'_i\downarrow{:}\kappa'_i \wedge tt \rightsquigarrow tt$. Admissible goals, or simply goals, are now extended to be a

conjunction of terms $t{:}\kappa \Rightarrow t'{:}\kappa'$, $t \Rightarrow^1 t'$, $t \rightsquigarrow tt$, $t \rightsquigarrow^1 x_k$ and $tt$, ending with $tt$. As in the calculus for unification we will use $t$ as a shortcut for $t \rightsquigarrow tt$.

Any reachability subgoal in our calculus of the form $t{:}\kappa \Rightarrow t'{:}\kappa'$ is equivalent to $t \Rightarrow t' \wedge t{=}x_\kappa \wedge t'{=}y_{\kappa'}$ (or $t \Rightarrow t' \wedge t{:}\kappa \wedge t'{:}\kappa'$). We will solve normalized reachability goals $t_i{\downarrow}{:}\kappa \Rightarrow t'_i{\downarrow}{:}\kappa'$, which have the same solutions that the reachability goals $t_i{:}\kappa \Rightarrow t'_i{:}\kappa'$ have, because if $t_i\sigma \Rightarrow t'_i\sigma$, as $t_i =_\mathcal{E} t_i{\downarrow}$ and $t'_i =_\mathcal{E} t'_i{\downarrow}$ then $t_i{\downarrow}\sigma \Rightarrow t'_i{\downarrow}\sigma$ and vice versa. Also if $t_i{\downarrow}\sigma$ has type $\kappa$, as $t_i\sigma \rightarrow^* t_i{\downarrow}\sigma$, by subject reduction $t_i\sigma$ has type $\kappa$ (and $t'_i\sigma$ has type $\kappa'$), and if $t_i\sigma$ has type $\kappa$, as $\rightarrow$ is sort-decreasing then $t_i{\downarrow}\sigma$ has type $\kappa$ (and $t'_i{\downarrow}\sigma$ has type $\kappa'$).

### 5.1   Calculus rules for reachability

*Reachability by conditional narrowing* is achieved using the calculus rules presented in Section 4, extended with the following calculus rules based on the deduction rules for rewrite theories in Figure 2. It is very important to point out that we only apply replacement on $t|_p$ with substitution $\theta$ if **the whole term** $t\theta$ is $E,A$-normalized, achieving another reduction in the state space:

- $[X]$ *reflexivity*

$$\frac{t : \kappa \Rightarrow t' : \kappa' \wedge G'}{eq_{\kappa,\kappa'}(t,t') \wedge G'}$$

- $[N]$ *narrowing*

$$\frac{t : \kappa \Rightarrow t' : \kappa' \wedge G'}{t \rightsquigarrow^1 x_{[\kappa]}, x_{[\kappa]} : \kappa \Rightarrow t' : \kappa' \wedge G'}$$
$$\text{where } t \notin \mathcal{X}$$

- $[T]$ *transitivity*

$$\frac{t : \kappa \Rightarrow t' : \kappa' \wedge G'}{t \Rightarrow^1 x_{[\kappa]}, x_{[\kappa]} : [\kappa] \Rightarrow t' : \kappa' \wedge t : \kappa \wedge G'}$$
$$\text{where } t \notin \mathcal{X}$$

- $[I]$ *imitation*

$$\frac{t|_p \Rightarrow^1 x^p_{k_p}, t[x^p_{k_p}]_p : [\kappa] \Rightarrow t' : \kappa' \wedge t : \kappa \wedge G'}{t_j \Rightarrow^1 x^{pj}_{k_{pj}}, t[x^{pj}_{k_{pj}}]_{pj} : [\kappa] \Rightarrow t' : \kappa' \wedge t : \kappa \wedge G'}$$
$$\text{where } t|_p = f(t_1,\ldots,t_n),\ t_j \notin \mathcal{X},\ k_{pj} = [ls(t_j)],\ 1 \le j \le n$$

- $[R]$ *replacement*

$$\frac{t|_p \Rightarrow^1 x^p_{k_p}, t[x^p_{k_p}]_p : [\kappa] \Rightarrow t' : \kappa' \wedge t : \kappa \wedge G'}{(((c) \wedge t[r]_p \Rightarrow t' : \kappa' \wedge t : \kappa \wedge G')\theta){\downarrow}}$$
$$\text{where } l \Rightarrow r\ (if\ c) \in \tilde{R},\ \theta \in CSU_A(t|_p = l),\ t\theta\ E,A\text{-normalized}$$

**Theorem 3.** *The calculus for reachability is sound and weakly complete, i.e., complete with respect to $R \cup E,A$-normalized solutions.*

## 6    Example

As an example of application of our calculus we use the specification of search trees in Section 2. We will abbreviate `empty` to $\epsilon$, and consider the reachability goal $\epsilon \, [x_{\mathtt{k}}^1 \, y_{\mathtt{v}}^1] \, \epsilon \mid \mathtt{f} \, y_{\mathtt{v}}^2 \, ; \, x_{\mathtt{k}}^2 \, 2 \mid \mathtt{nil} : \mathtt{s} \Rightarrow (\epsilon \, [x_{\mathtt{k}}^3 \, 2] \, \epsilon) \, [\mathtt{c} \, y_{\mathtt{v}}^3] \, \epsilon \mid w_{\mathtt{rs}} \mid z_{\mathtt{p}} : \mathtt{s}$, where from a `State` composed of a `SearchTree` with only one `Record` $x_{\mathtt{k}}^1 y_{\mathtt{v}}^1$ on the root, a `RecordSet` with two `Records`, $\mathtt{f} \, y_{\mathtt{v}}^2$ and $x_{\mathtt{k}}^2 \, 2$ and the `nil Path`, we want to reach a `State` composed of a `SearchTree` with two `Records`, $x_{\mathtt{k}}^3 \, 2$ on the left branch of the root and $\mathtt{c} \, y_{\mathtt{v}}^2$ on the root, some `RecordSet`, $w_{\mathtt{rs}}$, and some `Path`, $z_{\mathtt{p}}$. The reachability goal is already normalized, so we only have to add $\wedge \, tt$ to get the reachability problem (where we call $T_1 = (\epsilon \, [x_{\mathtt{k}}^3 \, 2] \, \epsilon) \, [\mathtt{c} \, y_{\mathtt{v}}^3] \, \epsilon \mid w_{\mathtt{rs}} \mid z_{\mathtt{p}}$ and $T_2 = \epsilon \, [x_{\mathtt{k}}^1 \, y_{\mathtt{v}}^1] \, \epsilon \mid \mathtt{f} \, y_{\mathtt{v}}^2 \, ; \, x_{\mathtt{k}}^2 \, 2 \mid \mathtt{nil})$:

1. $\epsilon \, [x_{\mathtt{k}}^1 \, y_{\mathtt{v}}^1] \, \epsilon \mid \mathtt{f} \, y_{\mathtt{v}}^2 \, ; \, x_{\mathtt{k}}^2 \, 2 \mid \mathtt{nil} : \mathtt{s} \Rightarrow T_1 : \mathtt{s} \wedge tt \rightsquigarrow_{[T]}$
   The transitivity rule is always needed before a narrowing step.
2. $\epsilon \, [x_{\mathtt{k}}^1 \, y_{\mathtt{v}}^1] \, \epsilon \mid \mathtt{f} \, y_{\mathtt{v}}^2 \, ; \, x_{\mathtt{k}}^2 \, 2 \mid \mathtt{nil} : \mathtt{s} \Rightarrow^1 x_{[s]}, x_{[s]} : [s] \Rightarrow T_1 : \mathtt{s} \wedge T_2 : s \wedge tt \rightsquigarrow_{[R],\mathtt{I1}}$
   Rule `I1` is able to insert the record $x_{\mathtt{k}}^2 \, 2$ thanks to the commutative axiom on `RecordSets`. We call $P_1 = \mathtt{nil} \, \mathtt{i} \, x_{\mathtt{k}}^2 \, 2$.
3. $\mathtt{ins}(\epsilon \, [x_{\mathtt{k}}^1 \, y_{\mathtt{v}}^1] \, \epsilon, x_{\mathtt{k}}^2, 2) \mid \mathtt{f} \, y_{\mathtt{v}}^2 \mid P_1 : [s] \Rightarrow T_1 : \mathtt{s} \wedge tt \wedge tt \rightsquigarrow_{[N]}$
   Now the calculus generates a narrowing for unification step.
4. $\mathtt{ins}(\epsilon \, [x_{\mathtt{k}}^1 \, y_{\mathtt{v}}^1] \, \epsilon, x_{\mathtt{k}}^2, 2) \mid \mathtt{f} \, y_{\mathtt{v}}^2 \mid P_1 \rightsquigarrow^1 x_{[s]}^4, x_{[s]}^4 : [s] \Rightarrow T_1 : \mathtt{s} \wedge tt \wedge tt \rightsquigarrow_{[\mathtt{c}]}$
   Congruence chooses the first subterm.
5. $\mathtt{ins}(\epsilon \, [x_{\mathtt{k}}^1 \, y_{\mathtt{v}}^1] \, \epsilon, x_{\mathtt{k}}^2, 2) \rightsquigarrow^1 x_{[t]}^5, x_{[t]}^5 \mid \mathtt{f} \, y_{\mathtt{v}}^2 \mid P_1 : [s] \Rightarrow T_1 : \mathtt{s} \wedge tt \wedge tt \rightsquigarrow_{[n],\mathtt{i1}}$
   We apply narrowing for unification with conditional rule `i1` :
   $\mathtt{ins}(l_{\mathtt{st}}[x_{\mathtt{r}}]r_{\mathtt{st}}, y_{\mathtt{k}}, z_{\mathtt{v}}) = \mathtt{ins}(l_{\mathtt{st}}, y_{\mathtt{k}}, z_{\mathtt{v}})[x_{\mathtt{r}}]r_{\mathtt{st}} \; if \; y_{\mathtt{k}} < \mathtt{key}(x_{\mathtt{r}}) = \mathtt{t}.$
6. $eq_{[b],[b]}(x_{\mathtt{k}}^2 < x_{\mathtt{k}}^1, t) \wedge (\epsilon \, [x_{\mathtt{k}}^2 \, 2] \, \epsilon) \, [x_{\mathtt{k}}^1 \, y_{\mathtt{v}}^1] \, \epsilon \mid \mathtt{f} \, y_{\mathtt{v}}^2 \mid P_1 : [s] \Rightarrow T_1 : \mathtt{s} \wedge tt \wedge tt \rightsquigarrow_{[t]}$
   Transitivity generates another narrowing for unification step.
7. $eq_{[b],[b]}(x_{\mathtt{k}}^2 < x_{\mathtt{k}}^1, t) \rightsquigarrow^1 x_{[Truth]}^6, x_{[Truth]}^6 \wedge (\epsilon \, [x_{\mathtt{k}}^2 \, 2] \, \epsilon) \, [x_{\mathtt{k}}^1 \, y_{\mathtt{v}}^1] \, \epsilon \mid \mathtt{f} \, y_{\mathtt{v}}^2 \mid P_1 : [s] \Rightarrow$
   $T_1 : \mathtt{s} \wedge tt \wedge tt \rightsquigarrow_{[\mathtt{c}]}$
   Congruence chooses the first subterm.
8. $x_{\mathtt{k}}^2 < x_{\mathtt{k}}^1 \rightsquigarrow^1 x_{[b]}^7, eq_{[b],[b]}(x_{[b]}^7, t) \wedge (\epsilon \, [x_{\mathtt{k}}^2 \, 2] \, \epsilon) \, [x_{\mathtt{k}}^1 \, y_{\mathtt{v}}^1] \, \epsilon \mid \mathtt{f} \, y_{\mathtt{v}}^2 \mid P_1 : [s] \Rightarrow$
   $T_1 : \mathtt{s} \wedge tt \wedge tt \rightsquigarrow_{[n],\mathtt{b}<\mathtt{c}=\mathtt{t}}, \{x_{\mathtt{k}}^2 \mapsto \mathtt{b}, x_{\mathtt{k}}^1 \mapsto \mathtt{c}\}$
   We apply narrowing for unification.
9. $tt \wedge (\epsilon \, [\mathtt{b} \, 2] \, \epsilon) \, [\mathtt{c} \, y_{\mathtt{v}}^1] \, \epsilon \mid \mathtt{f} \, y_{\mathtt{v}}^2 \mid \mathtt{nil} \, \mathtt{i} \, \mathtt{b} \, 2 : [s] \Rightarrow T_1 : \mathtt{s} \wedge tt \wedge tt \rightsquigarrow_{[e]}$
   Elimination removes leading $tt$'s.
10. $(\epsilon \, [\mathtt{b} \, 2] \, \epsilon) \, [\mathtt{c} \, y_{\mathtt{v}}^1] \, \epsilon \mid \mathtt{f} \, y_{\mathtt{v}}^2 \mid \mathtt{nil} \, \mathtt{i} \, \mathtt{b} \, 2 : [s] \Rightarrow T_1 : \mathtt{s} \wedge tt \wedge tt \rightsquigarrow_{[X]}$
    The reflexivity rule turns the reachability problem into a unification one.
11. $eq_{[s],s}((\epsilon \, [\mathtt{b} \, 2] \, \epsilon) \, [\mathtt{c} \, y_{\mathtt{v}}^1] \, \epsilon \mid \mathtt{f} \, y_{\mathtt{v}}^2 \mid \mathtt{nil} \, \mathtt{i} \, \mathtt{b} \, 2, (\epsilon \, [x_{\mathtt{k}}^3 \, 2] \, \epsilon) \, [\mathtt{c} \, y_{\mathtt{v}}^3] \, \epsilon \mid w_{\mathtt{rs}} \mid z_{\mathtt{p}}) \wedge tt \wedge tt$
    $\rightsquigarrow_{[u],\{x_{\mathtt{k}}^3 \mapsto \mathtt{b}, y_{\mathtt{v}}^1 \mapsto y_{\mathtt{v}}^4, y_{\mathtt{v}}^2 \mapsto y_{\mathtt{v}}^5, y_{\mathtt{v}}^3 \mapsto y_{\mathtt{v}}^4, w_{\mathtt{rs}} \mapsto \mathtt{f} \, y_{\mathtt{v}}^5, z_{\mathtt{p}} \mapsto \mathtt{nil} \, \mathtt{i} \, \mathtt{b} \, 2\}}$
    The unification rule solves the problem.

12. $((\epsilon\,[\mathtt{b}\,2]\,\epsilon)\,[\mathtt{c}\,y_{\mathtt{v}}^4]\,\epsilon\,|\,\mathtt{f}\,y_{\mathtt{v}}^5\,|\,\mathtt{nil}\,\mathtt{i}\,\mathtt{b}\,2:\mathtt{s}\wedge tt\wedge tt)\!\downarrow\,\equiv$
    $tt\wedge tt\wedge tt\rightsquigarrow_{[e]}$

13. $tt\wedge tt\rightsquigarrow_{[e]}$

14. $tt$

$\sigma=\{x_{\mathtt{k}}^1\mapsto\mathtt{c},x_{\mathtt{k}}^2\mapsto\mathtt{b},x_{\mathtt{k}}^3\mapsto\mathtt{b},y_{\mathtt{v}}^1\mapsto y_{\mathtt{v}}^4,y_{\mathtt{v}}^2\mapsto y_{\mathtt{v}}^5,y_{\mathtt{v}}^3\mapsto y_{\mathtt{v}}^4,w_{\mathtt{rs}}\mapsto\mathtt{f}\,y_{\mathtt{v}}^5,z_{\mathtt{p}}\mapsto$ $\mathtt{nil}\,\mathtt{i}\,\mathtt{b}\,2\}$ is the computed answer (substitutions in steps 8 and 11). The calculus has found for the given reachability goal an instance $\epsilon\,[\mathtt{c}\,y_{\mathtt{v}}^4]\,\epsilon\,|\,\mathtt{f}\,y_{\mathtt{v}}^5,\mathtt{b}\,2\,|\,\mathtt{nil}:\mathtt{s}\Rightarrow$ $(\epsilon\,[\mathtt{b}\,2]\,\epsilon)\,[\mathtt{c}\,y_{\mathtt{v}}^4]\,\epsilon\,|\,\mathtt{f}\,y_{\mathtt{v}}^5\,|\,\mathtt{nil}\,\mathtt{i}\,\mathtt{b}\,2:\mathtt{s}$ where in the final State the SearchTree has two Records, $\mathtt{b}\,2$ and $\mathtt{c}\,y_{\mathtt{v}}^4$, there is still one Record, $\mathtt{f}\,y_{\mathtt{v}}^5$, left in the RecordSet to process, and the Path $\mathtt{nil}\,\mathtt{i}\,\mathtt{b}\,2$ tells us that the system has chosen to insert the Record $\mathtt{b}\,2$ in the SearchTree when building the computed answer.

## 7    Related work, conclusions and future work

A classic reference in equational conditional narrowing modulo is the work of Bockmayr [5]. The topic is addressed here for Church-Rosser equational conditional term rewriting systems with empty axioms, but non-terminating axioms (like ACU) are not allowed. Non-conditional narrowing modulo order-sorted equational logics is covered by Meseguer and Thati [24] and it is being used for cryptographic protocol analysis. Equivalence of $R/\mathcal{E}$ and $R\cup E,A$ rewriting was proved by Viry [27] for unsorted rewrite theories. Membership equational logic was defined by Meseguer [21]. A rewrite system for MEL theories that allows unification by rewriting is presented by Durán, Lucas et al. [9]. Strategies, which also play a main role in narrowing, have been studied by Antoy, Echahed, and Hanus [2]. Their needed narrowing strategy, for inductively sequential rewrite systems, generates only narrowing steps leading to a computed answer. Recently Escobar, Sasse, and Meseguer [13] have developed the concepts of variant and folding variant, a narrowing strategy for order-sorted unconditional rewrite theories that terminates on those theories having the *finite variant property*. Foundations for order-sorted conditional rewriting have been published by Meseguer [23]. Cholewa, Escobar, and Meseguer [7] have defined a new hierarchical method, called *layered constraint narrowing*, to solve narrowing problems in order-sorted conditional equational theories and given new theoretical results on that matter. Order-sorted conditional narrowing with constraint solvers has been addressed by Rocha and Meseguer [26].

In this work we have developed narrowing calculi for unification in membership equational logic and reachability in rewrite theories with an underlying membership equational logic. The main features in these calculi are that membership information is taken into account, all terms are normalized after each calculus step and only normalized instantiations are allowed for reachability terms and matching variables, greatly reducing the state space. The calculi have been proved sound and weakly complete.

Previous work with non-normalized terms and substitutions is available at http://maude.sip.ucm.es/cnarrowing/, where a strategy for applying the calculi was shown and implemented using Maude. This new version of the calculi has not been implemented yet, but we plan to make use of it in our current line of investigation, that concerns the extension of the calculi to handle constraints and their connection with external constraint solvers for domains such as finite domains, integers, Boolean values, etc., that could greatly improve the performance of any implementation.

**Acknowledgments.** We are very grateful to the anonymous referees for all their helpful comments.

# References

1. L. Aguirre, N. Martí-Oliet, M. Palomino, and I. Pita. Conditional narrowing modulo in rewriting logic and Maude. In Escobar [11], pages 80–96.
2. S. Antoy, R. Echahed, and M. Hanus. A needed narrowing strategy. In H. Boehm, B. Lang, and D. M. Yellin, editors, *Conference Record of POPL'94: 21st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Portland, Oregon, USA, January 17-21, 1994*, pages 268–279. ACM Press, 1994.
3. K. Bae and J. Meseguer. Model checking LTLR formulas under localized fairness. In F. Durán, editor, *Rewriting Logic and Its Applications - 9th International Workshop, WRLA 2012, Held as a Satellite Event of ETAPS, Tallinn, Estonia, March 24-25, 2012, Revised Selected Papers*, volume 7571 of *Lecture Notes in Computer Science*, pages 99–117. Springer, 2012.
4. K. Bae and J. Meseguer. Infinite-state model checking of LTLR formulas using narrowing. In Escobar [11], pages 113–129.
5. A. Bockmayr. Conditional narrowing modulo a set of equations. *Applicable Algebra in Engineering, Communication and Computing*, 4:147–168, 1993.
6. R. Bruni and J. Meseguer. Semantic foundations for generalized rewrite theories. *Theoretical Computer Science*, 360(1-3):386–414, 2006.
7. A. Cholewa, S. Escobar, and J. Meseguer. Constrained Narrowing for Conditional Equational Theories Modulo Axioms. Technical report, http://hdl.handle.net/2142/50289, C.S. Department, University of Illinois at Urbana-Champaign, August 2014.
8. M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and C. Talcott. *All About Maude - A High-Performance Logical Framework: How to Specify, Program, and Verify Systems in Rewriting Logic*, volume 4350 of *Lecture Notes in Computer Science*. Springer, 2007.
9. F. Durán, S. Lucas, C. Marché, J. Meseguer, and X. Urbain. Proving operational termination of membership equational programs. *Higher-Order and Symbolic Computation*, 21(1-2):59–88, 2008.
10. F. Durán and J. Meseguer. On the Church-Rosser and coherence properties of conditional order-sorted rewrite theories. *The Journal of Logic and Algebraic Programming*, 81(7-8):816–850, 2012.
11. S. Escobar, editor. *Rewriting Logic and Its Applications - 10th International Workshop, WRLA 2014, Held as a Satellite Event of ETAPS, Grenoble, France, April 5-6, 2014, Revised Selected Papers*, volume 8663 of *Lecture Notes in Computer Science*. Springer, 2014.

12. S. Escobar, C. Meadows, and J. Meseguer. Maude-NPA: Cryptographic protocol analysis modulo equational properties. In A. Aldini, G. Barthe, and R. Gorrieri, editors, *Foundations of Security Analysis and Design V*, volume 5705 of *Lecture Notes in Computer Science*, pages 1–50. Springer, 2009.
13. S. Escobar, R. Sasse, and J. Meseguer. Folding variant narrowing and optimal variant termination. *The Journal of Logic and Algebraic Programming*, 81(7-8):898–928, 2012.
14. M. Fay. *First-order Unification in an Equational Theory*. University of California, 1978.
15. E. Giovannetti and C. Moiso. A completeness result for e-unification algorithms based on conditional narrowing. In M. Boscarol, L. C. Aiello, and G. Levi, editors, *Foundations of Logic and Functional Programming, Workshop, Trento, Italy, December 15-19, 1986, Proceedings*, volume 306 of *Lecture Notes in Computer Science*, pages 157–167. Springer, 1986.
16. M. Hamada. Strong completeness of a narrowing calculus for conditional rewrite systems with extra variables. *Electr. Notes Theor. Comput. Sci.*, 31:89–103, 2000.
17. S. Lucas, C. Marché, and J. Meseguer. Operational termination of conditional term rewriting systems. *Inf. Process. Lett.*, 95(4):446–453, 2005.
18. S. Lucas and J. Meseguer. Operational termination of membership equational programs: the order-sorted way. *Electr. Notes Theor. Comput. Sci.*, 238(3):207–225, 2009.
19. N. Martí-Oliet and J. Meseguer. Rewriting logic: roadmap and bibliography. *Theor. Comput. Sci.*, 285(2):121–154, 2002.
20. J. Meseguer. Rewriting as a unified model of concurrency. In J. Baeten and J. Klop, editors, *CONCUR '90 Theories of Concurrency: Unification and Extension*, volume 458 of *Lecture Notes in Computer Science*, pages 384–400. Springer, 1990.
21. J. Meseguer. Membership algebra as a logical framework for equational specification. In F. Parisi-Presicce, editor, *Recent Trends in Algebraic Development Techniques, 12th International Workshop, WADT'97, Tarquinia, Italy, June 1997, Selected Papers*, volume 1376 of *Lecture Notes in Computer Science*, pages 18–61. Springer, 1997.
22. J. Meseguer. Twenty years of rewriting logic. *J. Log. Algebr. Program.*, 81(7-8):721–781, 2012.
23. J. Meseguer. Strict Coherence of Conditional Rewriting Modulo Axioms. Technical report, http://hdl.handle.net/2142/50288, C.S. Department, University of Illinois at Urbana-Champaign, August 2014.
24. J. Meseguer and P. Thati. Symbolic reachability analysis using narrowing and its application to verification of cryptographic protocols. *Higher-Order and Symbolic Computation*, 20(1-2):123–160, 2007.
25. A. Middeldorp and E. Hamoen. Completeness results for basic narrowing. *Appl. Algebra Eng. Commun. Comput.*, 5:213–253, 1994.
26. C. Rocha. *Symbolic reachability analysis for rewrite theories*. PhD thesis, http://hdl.handle.net/2142/42200, C.S. Department, University of Illinois at Urbana-Champaign, February 2013.
27. P. Viry. Rewriting: An effective model of concurrency. In C. Halatsis, D. G. Maritsas, G. Philokyprou, and S. Theodoridis, editors, *PARLE '94: Parallel Architectures and Languages Europe, 6th International PARLE Conference, Athens, Greece, July 4-8, 1994, Proceedings*, volume 817 of *Lecture Notes in Computer Science*, pages 648–660. Springer, 1994.