# Natural Rewriting for General Term Rewriting Systems*

Santiago Escobar[1], José Meseguer[2], and Prasanna Thati[2]

[1] Universidad Politécnica de Valencia, Spain
sescobar@dsic.upv.es
[2] University of Illinois at Urbana-Champaign, USA
{meseguer,thati}@cs.uiuc.edu

**Abstract.** We address the problem of an efficient rewriting strategy for general term rewriting systems. Several strategies have been proposed over the last two decades for rewriting, the most efficient of all being the natural rewriting strategy [7]. All the strategies so far, including natural rewriting, assume that the given term rewriting system is left-linear and constructor-based. Although these restrictions are reasonable for some functional programming languages, they limit the expressive power of equational languages, and they preclude certain applications of rewriting to equational theorem proving and to languages combining equational and logic programming. In this paper, we propose a conservative generalization of natural rewriting that does not require the rules to be left-linear and constructor-based. We also establish the soundness and completeness of this generalization.

## 1 Introduction

A challenging problem in modern programming languages is the discovery of sound and complete evaluation strategies which are: (i) optimal w.r.t. some efficiency criterion, (ii) easily implementable, and (iii) applicable for a large class of programs.

The evaluation strategies for programming languages so far, can be classified into two classical families: *eager strategies* (also know as *innermost* or *call-by-value*) and *lazy strategies* (also know as *outermost* or *call-by-need*). The choice of the strategy can have a big impact on performance of programming languages: for instance, lazy rewriting typically needs more resources than eager rewriting [14], but the former improves termination of the program w.r.t. the latter. To define a lazy rewriting strategy, we have to define what a needed computation is and also have to provide an efficient procedure to determine whether some computation is needed. These two problems were first addressed for rewriting in a seminal paper by Huet and Levy [11], where the *strongly needed reduction*

* Work partially supported by CICYT TIC2001-2705-C03-01, MCyT Acción Integrada HU 2003-0003, Agencia Valenciana de Ciencia y Tecnología GR03/025, and EU-India Cross-Cultural Dissemination project ALA/95/23/2003/077-054.

strategy was proposed. Several refinements of this strategy have been proposed over the last two decades, the most significant ones being Sekar and Ramakrishnan's *parallel needed reduction* [15], and Antoy, Echahed and Hanus' *(weakly) outermost-needed rewriting* [1,2,3]. Recently, (weakly) outermost-needed rewriting has been improved by Escobar by means of the *natural rewriting strategy* [7,8]. Natural rewriting is based on a suitable extension of the demandedness notion associated to (weakly) outermost-needed rewriting; see [4] for a survey on demandedness in programming languages. Moreover, the strategy enjoys good computational properties such as soundness and completeness w.r.t. head-normal forms, and it preserves optimality w.r.t. the number of reduction steps for sequential parts of the program.

A typical assumption of the above rewriting strategies, including the natural rewriting strategy, is that the rewrite rules are left-linear and constructor-based. These restrictions are reasonable for some functional programming languages, but they limit the expressive power of equational languages such as OBJ [10], CafeOBJ [9], ASF+SDF [16], and Maude [6], where non-linear left-hand sides are perfectly acceptable. This extra generality is also necessary for applications of rewriting to equational theorem proving, and to languages combining equational and logic programming, since in both cases assuming left-linearity is too restrictive. Furthermore, for rewrite systems whose semantics is not equational but is instead rewriting logic based, such as rewrite rules in ELAN [5], or Maude system modules, the constructor-based assumption is unreasonable and almost never holds.

In summary, generalizing natural rewriting to *general* rewriting systems will extend the scope of applicability of the strategy to more expressive equational languages and to rewriting logic based languages, and will open up a much wider range of applications. In the following, we give the reader a first intuitive example of how the generalized natural rewriting strategy will work.

*Example 1.* Consider the following TRS for proving equality of arithmetic expressions built using division ($\div$), modulus or remainder ($\%$), and subtraction ($-$) operations on natural numbers.

(1) $\mathtt{0} \div \mathtt{s(N)} \to \mathtt{0}$
(2) $\mathtt{s(M)} \div \mathtt{s(N)} \to \mathtt{s((M-N)} \div \mathtt{s(N))}$
(3) $\mathtt{M} \,\%\, \mathtt{s(N)} \to \mathtt{(M-s(N))} \,\%\, \mathtt{s(N)}$
(4) $\mathtt{(0} - \mathtt{s(M))} \,\%\, \mathtt{s(N)} \to \mathtt{N} - \mathtt{M}$
(5) $\mathtt{M} - \mathtt{0} \to \mathtt{M}$
(6) $\mathtt{s(M)} - \mathtt{s(N)} \to \mathtt{M-N}$
(7) $\mathtt{X} \approx \mathtt{X} \to \mathtt{True}$

Note that this TRS is not left-linear because of rule (7) and it is not constructor-based because of rule (4). Therefore, it is outside the scope of all the strategies mentioned above. Furthermore, note that the TRS is neither terminating nor confluent due to rule (3).

Consider the term[3] $t_1 = \mathtt{10!} \div \mathtt{0}$. If we only had rules (1), (2), (5) and (6), the natural rewriting strategy [7] would be applicable and no reductions on $t_1$

[3] The subterm `10!` represents factorial of 10 but we do not include the rules for `!` because we are only interested in the fact that it has a remarkable computational cost, and therefore we would like to avoid its reduction whenever possible.

would be performed, since $t_1$ is a head-normal form. In contrast, the other strategies mentioned above, for example, outermost-needed rewriting, would force the evaluation of the computationally expensive subterm `10!` (see [7, Example 21]). Hence, we would like to generalize natural rewriting to a version that enjoys this optimality and that can also handle non-left-linear and non-constructor-based rules such as (7) and (4).

Now, consider the term $t_2 = \texttt{10!} \ \% \ \texttt{(1-1)} \approx \texttt{10!} \ \% \ \texttt{0}$. We would like the generalized natural rewriting strategy to perform only the optimal computation:

$$\begin{aligned}&\texttt{10!} \ \% \ \underline{\texttt{(s(0)-s(0))}} \approx \texttt{10!} \ \% \ \texttt{0}\\ &\quad \to \texttt{10!} \ \% \ \underline{\texttt{(0-0)}} \approx \texttt{10!} \ \% \ \texttt{0}\\ &\quad \to \underline{\texttt{10!} \ \% \ \texttt{0} \approx \texttt{10!} \ \% \ \texttt{0}} \to \texttt{True}\end{aligned}$$

that avoids unnecessary reduction of the subterm `10!` % `0` at the final rewrite step, and also avoids reductions on the computationally expensive term `10!` during the whole rewrite sequence.

Since natural rewriting [7] uses a more refined demandedness notion for redexes in comparison with other strategies such as needed rewriting [1,3], it leads to a very efficient lazy evaluation strategy. In this paper, we propose a conservative generalization of this demandedness notion that drops the assumptions that the rewrite rules are left-linear and constructor-based, while retaining soundness and completeness w.r.t. head-normal forms.

After some preliminaries in Section 2, we present our generalization of natural rewriting strategy in Section 3, and formally define its properties. We show soundness and completeness of the generalized rewrite strategy w.r.t. head-normal forms. In Section 4, we further refine the strategy to obtain a more optimal one, without loosing the soundness and completeness properties. Finally, we conclude in Section 5.

## 2 Preliminaries

For a binary relation $R \subseteq A \times A$, we denote its reflexive and transitive closure by $R^*$. An element $a \in A$ is an $R$-normal form, if there exists no $b$ such that $a \ R \ b$. We say that $b$ is an $R$-normal form of $a$ (written $a \, R^! \, b$), if $b$ is an $R$-normal form and $a \ R^* b$.

We assume a finite alphabet (function symbols) $\mathcal{F} = \{\mathtt{f}, \mathtt{g}, \ldots\}$, and a countable set of variables $\mathcal{X} = \{\mathtt{X}, \mathtt{Y}, \ldots\}$. We denote the set of terms built from $\mathcal{F}$ and $\mathcal{X}$ by $\mathcal{T}(\mathcal{F}, \mathcal{X})$. We write $\mathcal{V}ar(t)$ for the set of variables occurring in $t$. A term is said to be linear if it has no multiple occurrences of a single variable. We let finite sequences of integers to denote access paths in a term, and denote the set of positions of a term $t$ by $\mathcal{P}os(t)$. Given a set $S \subseteq \mathcal{F} \cup \mathcal{X}$, $\mathcal{P}os_S(t)$ denotes positions in $t$ where symbols in $S$ occur. We write $\mathcal{P}os_f(t)$ as a shorthand for $\mathcal{P}os_{\{f\}}(t)$. We denote the *root position* by $\Lambda$. Given positions $p, q$, we denote its concatenation as $p.q$ and define $p/q = p'$ if $p = q.p'$. Positions are ordered by the standard prefix ordering $\leq$. We say $p$ and $q$ are *disjoint positions* and write $p \parallel q$, if $p \not\leq q$ and $q \not\leq p$. For sets of positions $P, Q$ we define $P.Q = \{p.q \mid p \in P \wedge q \in Q\}$. We

write $P.q$ as a shorthand for $P.\{q\}$ and similarly for $p.Q$. The subterm of $t$ at position $p$ is denoted as $t|_p$, and $t[s]_p$ is the term $t$ with the subterm at position $p$ replaced by $s$. We define $t|_P = \{t|_p \mid p \in P\}$. The symbol labeling the root of $t$ is denoted as $root(t)$. Given a set of positions $P$, we call $p \in P$ an *outermost position* in $P$ if there is no $q \in P$ such that $q < p$.

A *substitution* is a function $\sigma : X \to \mathcal{T}(\mathcal{F}, \mathcal{X})$ which maps variables to terms, and which is different from the identity only for a finite subset $\mathcal{D}om(\sigma)$ of $X$. We denote the homomorphic extension of $\sigma$ to $\mathcal{T}(\mathcal{F}, \mathcal{X})$ also by $\sigma$. The set of variables introduced by $\sigma$ is $\mathcal{R}an(\sigma) = \cup_{x \in \mathcal{D}om(\sigma)} \mathcal{V}ar(\sigma(x))$. We denote by $id$ the "identity" substitution: $id(x) = x$ for all $x \in \mathcal{X}$. Terms are ordered by the preorder $\leq$ of "relative generality", i.e. $s \leq t$ if there exists $\sigma$ s.t. $\sigma(s) = t$. Similarly, a substitution $\sigma$ is said to be more general than $\theta$, denoted by $\sigma \leq \theta$, if there is $\gamma$ such that $\theta = \gamma \circ \sigma$.

A rewrite rule is an ordered pair $(l, r)$ of terms, also written $l \to r$, with $l \notin \mathcal{X}$. The left-hand side (*lhs*) of the rule is $l$, and $r$ is the right-hand side (*rhs*). A TRS is a pair $\mathcal{R} = (\mathcal{F}, R)$ where $R$ is a set of rewrite rules. $L(\mathcal{R})$ denotes the set of *lhs*'s of $\mathcal{R}$. A TRS $\mathcal{R}$ is left-linear if for all $l \in L(\mathcal{R})$, $l$ is a linear term. Given $\mathcal{R} = (\mathcal{F}, R)$, we take $\mathcal{F}$ as the disjoint union $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$ of symbols $c \in \mathcal{C}$, called *constructors*, and symbols $f \in \mathcal{D}$, called *defined symbols*, where $\mathcal{D} = \{root(l) \mid l \to r \in R\}$ and $\mathcal{C} = \mathcal{F} - \mathcal{D}$. A pattern is a term $f(l_1, \ldots, l_k)$ where $f \in \mathcal{D}$ and $l_i \in \mathcal{T}(\mathcal{C}, \mathcal{X})$, for $1 \leq i \leq k$. A TRS $\mathcal{R} = (\mathcal{C} \uplus \mathcal{D}, R)$ is a constructor system (CS) if every $l \in L(\mathcal{R})$ is a pattern.

A term $t$ rewrites to $s$ at position $p \in \mathcal{P}os(t)$ using the rule $l \to r \in R$, written $t \to_{\langle p, l \to r \rangle} s$ (or simply $t \xrightarrow{p} s$ or $t \to s$), if $t|_p = \sigma(l)$ and $s = t[\sigma(r)]_p$. The pair $\langle p, l \to r \rangle$ is called a redex; and often, we also refer to the subterm $t|_p$ in $t$ as a redex. A term $t$ is a head-normal form, or root-stable, if it cannot be reduced to a redex, i.e. there are no $t', t''$ such that $t \to^* t' \xrightarrow{\Lambda} t''$. We denote by $\xrightarrow{>\Lambda}$ a rewrite step at a position $p > \Lambda$, and thus by $\xrightarrow{>\Lambda}{}^*$ a sequence of rewrites all of which occur at positions $p > \Lambda$.

A *sequential rewrite strategy* $\mathcal{S}$ for a TRS $\mathcal{R}$ is a subrelation $\xrightarrow{\mathcal{S}}_{\mathcal{R}} \subseteq \to_{\mathcal{R}}$. More generally, a *rewrite strategy* is defined to be a subset of all possible rewrite sequences [13], but we are only concerned with sequential strategies. Furthermore, we are only concerned with *root-normalizing* rewrite strategies [13], and therefore we are interested in the following correctness and completeness criteria for $\mathcal{S}$:

1. (Correctness) If a term $t$ is a $\xrightarrow{\mathcal{S}}_{\mathcal{R}}$-normal form, then $t$ is root-stable.
2. (Completeness) If $t \to^*_{\mathcal{R}} s$, then $\exists s'$ s.t. $t \xrightarrow{\mathcal{S}}{}^* s'$, $root(s') = root(s)$ and $s' \xrightarrow{>\Lambda}{}^*_{\mathcal{R}} s$.

## 3 Generalizing Natural Rewriting

As mentioned earlier, we are interested in rewrite strategies for computing head-normal forms of a given term. If a term $t$ is not root-normal; then we know that

after a (possibly empty) sequence of rewrites at positions other than the root, a rule $l \to r$ can be applied at the root. We are interested in a strategy that performs only those rewrites that are essential for the rule $l \to r$ to be applied at the root.

**Definition 1.** *For a term $s$ and a set of terms $T = \{t_1, \ldots, t_n\}$ we say that $s$ is a context of the terms in $T$ if $s \leq t_i$ for all $1 \leq i \leq n$. It is the case that there is a least general context $s$ of $T$, i.e. for any other context $s'$ we have $s \leq s'$; further $s$ is unique up to renaming of variables. For $1 \leq i \leq n$ let the substitution $\sigma_i$ be such that $\sigma_i(s) = t_i$ and $\mathcal{D}om(\sigma_i) \subseteq \mathcal{V}ar(s)$. We define the set $\mathcal{P}os_{\neq}(T)$ of disagreeing positions between the terms in $T$ as $p \in \mathcal{P}os_{\mathcal{X}}(s)$ such that there is an $i$ with $\sigma_i(s|_p) \neq s|_p$.*

*Example 2.* Consider the set of terms $T = \{\texttt{10!} \ \% \ \texttt{(1-1)}, \ \texttt{10!} \ \% \ \texttt{0}\}$ borrowed from Example 1. The least general context of $T$ is the term $s = \texttt{10!} \ \% \ \texttt{Z}$ and the set of disagreeing positions between terms in $T$ is $\mathcal{P}os_{\neq}(T) = \{2\}$.

**Definition 2 (Demanded positions).** *For terms $l, t$ let $s$ be the least general context of $l$ and $t$, and let $\sigma$ be the substitution such that $\sigma(s) = l$. We define the set of demanded positions in $t$ w.r.t. to $l$ as*

$$DP_l(t) = \bigcup_{x \in \mathcal{V}ar(s)} \begin{array}{l} \text{if } \sigma(x) \notin \mathcal{X} \text{ then } \mathcal{P}os_x(s) \text{ else } Q.\mathcal{P}os_{\neq}(t|_Q) \\ \text{where } Q = \mathcal{P}os_{\sigma^{-1}(\sigma(x))}(s) \end{array}$$

Let us dissect the definition above. Intuitively, the set $DP_l(t)$ returns a set of positions in $t$ that necessarily have to be "changed" for the rule $l \to r$ to be applied at the root position, i.e., for $l$ to be able to match the term under consideration. Suppose, $s$ is the least general context of $l$ and $t$, and $\sigma$ is such that $\sigma(s) = l$. Note that for every non-variable position $p$ in $s$, it is the case that $t$ and $l$ have the same symbol at $p$. Now, if $\sigma$ maps a variable $x \in \mathcal{V}ar(s)$ to a non-variable term, then $t$ and $l$ disagree (have a different symbol) at every position $p \in \mathcal{P}os_x(s)$; this is a consequence of the fact that $s$ is the least general context of $l$ and $t$. The other case is where a variable $x \in \mathcal{V}ar(s)$ is mapped to a possibly non-linear variable of $l$. In this case, consider the positions of all the variables in $s$ that are mapped to the same variable as $x$, namely $Q = \mathcal{P}os_{\sigma^{-1}(\sigma(x))}(s)$. Now, $l$ matches $t$ only if all the terms in $t|_Q$ are identical. Thus, we compute the disagreeing positions $\mathcal{P}os_{\neq}(t|_Q)$, if any, and add $Q.\mathcal{P}os_{\neq}(t|_Q)$ to the set $DP_l(t)$. Finally, note that when $l \leq t$, it is the case that $l$ is the least general context of $l$ and $t$, and $DP_l(t) = \varnothing$.

*Example 3.* Consider the left-hand side $l = \texttt{X} \approx \texttt{X}$ and the term $t_2 = \texttt{10!} \ \% \ \texttt{(1-1)} \approx \texttt{10!} \ \% \ \texttt{0}$ of Example 1. The least general context of $l$ and $t_2$ is $s = \texttt{W} \approx \texttt{Y}$. Now, for $\sigma = \{\texttt{W} \mapsto \texttt{X}, \texttt{Y} \mapsto \texttt{X}\}$, we have $\sigma(s) = l$. While computing $DP_l(t_2)$, we obtain the set of disagreeing positions between the subterms in $t_2$ corresponding to the non-linear variable $\texttt{X}$ in $l$, i.e. the set $\mathcal{P}os_{\neq}(\texttt{10!} \ \% \ \texttt{(1-1)}, \texttt{10!} \ \% \ \texttt{0}) = \{2\}$. Thus, $DP_l(t_2) = \{1, 2\}.\{2\} = \{1.2, 2.2\}$.

Note that the symbol at a position $p \in DP_l(t)$ in $t$ can be changed by not only a rewrite at $p$, but also by a rewrite at a position $q < p$. Thus, besides considering the positions in $DP_l(t)$ as examinable, we also need to consider the positions $q$ in $t$ that are above some position in $DP_l(t)$. However, we only need to consider those positions $q$ at which $t$ has a defined symbol, because otherwise a rewrite can never be performed at $q$ without first changing the symbol at $q$. Thus, for a position $q$ in a term $t$, we define the set of defined positions above $q$ as $D_t^{\uparrow}(q) = \{p \mid p \leq q \ \wedge \ p \in \mathcal{P}os_{\mathcal{D}}(t)\}$. We lift this to sets of positions as $D_t^{\uparrow}(Q) = \cup_{q \in Q} D_t^{\uparrow}(q)$. Finally, note that for constructor-based TRS such as those considered by the natural rewriting strategy of [7], we have $D_t^{\uparrow}(Q) \subseteq Q$.

*Example 4.* Consider the TRS in Example 1, the term

$$t = \texttt{0} \div \texttt{s(10!)} \approx \texttt{0} \div \texttt{s(s(10!))}$$

and the rule $l = \texttt{X} \approx \texttt{X} \to \texttt{True}$. We have $DP_l(t) = \{1.2.1, 2.2.1\}$, and the subterms at positions 1.2.1 and 2.2.1 should be identical for the above rule to be applied at the root position. Now, reductions in only the subterm `10!` at position 1.2.1 would never result in `s(10!)`, which is the subterm at position 2.2.1, and vice versa. The right reduction sequence leading to constant `True` is the one reducing the symbols $\div$ above the demanded positions 1.2.1 and 2.2.1:

$$\underline{\texttt{0} \div \texttt{s(10!)}} \approx \texttt{0} \div \texttt{s(s(10!))}$$
$$\to \texttt{0} \approx \underline{\texttt{0} \div \texttt{s(s(10!))}} \to \underline{\texttt{0} \approx \texttt{0}} \to \texttt{True}$$

The essential idea behind our natural rewriting strategy is to compute a *needed* set of redexes $NR_{\mathcal{R}}(t)$ such that for any root stable term $t'$ that $t$ can be reduced to, there is a reduction sequence $t \xrightarrow{q}_{l \to r} \to^* t'$ that begins with a redex $\langle q, l \to r\rangle \in NR_{\mathcal{R}}(t)$. Then the strategy can consider only those reduction sequences from $t$ that begin with a redex in $NR_{\mathcal{R}}(t)$, and still be able to find all the head-normal forms of $t$.

**Definition 3 (Needed set of redexes).** *We define the needed set of redexes of a term $t$ w.r.t. TRS $\mathcal{R}$ as*

$$NR_{\mathcal{R}}(t) = \{\langle \Lambda, l \to r\rangle \mid l \in L(\mathcal{R}) \wedge l \leq t\} \ \cup \bigcup_{q \in SP_{\mathcal{R}}(t) \setminus \{\Lambda\}} q.NR_{\mathcal{R}}(t|_q)$$

*where for a set of redexes $S$, we define $q.S = \{\langle q.p, l \to r\rangle \mid \langle p, l \to r\rangle \in S\}$, and*

$$SP_{\mathcal{R}}(t) = \bigcup_{l \in L(\mathcal{R}) \wedge l \leq t} D_t^{\uparrow}(\mathcal{P}os_{\mathcal{X}}(l)) \ \cup \ D_t^{\uparrow}\Big( \bigcup_{l \in L(\mathcal{R}) \wedge l \not\leq t} DP_l(t) \Big)$$

The set $NR_{\mathcal{R}}(t)$ is recursively computed as follows. Whenever $l \leq t$ for a rule $l \to r$, the redex $\langle \Lambda, l \to r\rangle$ is included in $NR_{\mathcal{R}}(t)$. When $l \not\leq t$, we recursively compute $NR_{\mathcal{R}}(t|_q)$ for each position $q \in D_t^{\uparrow}(DP_l(t))$. The case $l \leq t$ has an additional subtlety; specifically, in this case, we also have to recursively compute $NR_{\mathcal{R}}(t|_q)$ for the positions $q$ in $t$ that have a defined symbol and that are above a variable position in $l$. This is necessary for the strategy to be complete, as illustrated by the following example.

*Example 5.* Consider the TRS

`(i) first(pair(X,Y)) → X` `(ii) pair(X,Y) → pair(Y,X)`

and the term $t =$ `first(pair(a,b))`. Suppose we simply define

$$SP_{\mathcal{R}}(t) = D_t^{\uparrow}(\cup_{l \in L(\mathcal{R}) \wedge l \not\lesssim t} DP_l(t))$$

Then $NR_{\mathcal{R}}(t) = \{\langle \Lambda, (i) \rangle\}$, and the only rewrite sequence starting from $t$ and beginning with a redex in $NR_{\mathcal{R}}(t)$ would be

$\underline{\texttt{first(pair(a,b))}} \rightarrow$ `a`

But the term $t$ can also be reduced to the head-normal form `b` as follows

`first(`$\underline{\texttt{pair(a,b)}}$`)` $\rightarrow$ $\underline{\texttt{first(pair(b,a))}}$ $\rightarrow$ `b` `(*)`

Hence, although the left-hand side of rule `(i)` matches $t$, for the strategy to be complete, we also have to consider the subterm `pair(a,b)` of $t$ at position `1` (which is above variable positions 1.1 and 1.2 in the left-hand side of rule (i)), and recursively compute $NR_{\mathcal{R}}($`pair(a,b)`$)$. Then we will have $NR_{\mathcal{R}}(t) = \{\langle \Lambda, (i) \rangle, \langle 1, (ii) \rangle\}$, which enables us to account for the rewrite sequence $(*)$ above.

From now on, while displaying the sets $NR_{\mathcal{R}}(t)$ in examples, we will omit the rule $l \rightarrow r$ in a redex $\langle \Lambda, l \rightarrow r \rangle$ and simply write $\langle \Lambda \rangle$, whenever there is no scope for ambiguity about the rule.

*Example 6.* Consider again the following term from Example 1:

$$t_2 = \texttt{10! \% (1−1)} \approx \texttt{10! \% 0}$$

and consider the computation of $NR_{\mathcal{R}}(t_2)$. Since $t_2$ is not a redex, we have that $NR_{\mathcal{R}}(t_2) = \cup_{q \in SP_{\mathcal{R}}(t_2) \setminus \{\Lambda\}} q.NR_{\mathcal{R}}(t_2|_q)$. Consider now $SP_{\mathcal{R}}(t_2)$. Since $t_2$ is not a redex, we have $SP_{\mathcal{R}}(t_2) = D_{t_2}^{\uparrow}(\cup_{l \in L(\mathcal{R}) \wedge l \not\lesssim t} DP_l(t_2))$. Now, from Example 3, we have that $DP_l(t_2) = \{1.2, 2.2\}$ for the rule $l =$ `X` $\approx$ `X` and $DP_{l'}(t_2) = \{\Lambda\}$ for any other rule $l'$ in $\mathcal{R}$. So then, $D_{t_2}^{\uparrow}(\cup_{l \in L(\mathcal{R})} DP_l(t_2)) = \{\Lambda, 1, 2, 1.2\}$ where the position 2.2 has been removed since it is not rooted by a defined symbol. Hence, we have that $NR_{\mathcal{R}}(t_2) = 1.NR_{\mathcal{R}}(t_2|_1) \cup 2.NR_{\mathcal{R}}(t_2|_2) \cup 1.2.NR_{\mathcal{R}}(t_2|_{1.2})$.

First, we consider $NR_{\mathcal{R}}(t_2|_{1.2})$. Subterm `1 − 1` at position 1.2 is a redex and thus $\langle \Lambda \rangle \in NR_{\mathcal{R}}(t_2|_{1.2})$. Furthermore, $SP_{\mathcal{R}}(t_2|_{1.2}) \setminus \{\Lambda\} = \varnothing$ because all symbols under root position in `1 − 1` are constructor symbols. Thus, we have $NR_{\mathcal{R}}(t_2|_{1.2}) = \{\ \langle \Lambda \rangle\ \}$.

Now, we consider $NR_{\mathcal{R}}(t_2|_1)$. The subterm `10! % (1−1)` is not a redex, and thus $NR_{\mathcal{R}}(t_2|_1) = \cup_{q \in SP_{\mathcal{R}}(t_2|_1) \setminus \{\Lambda\}} q.NR_{\mathcal{R}}(t_2|_{1.q})$. Now consider $SP_{\mathcal{R}}(t_2|_1)$. Since `10! % (1−1)` is not a redex, we have $SP_{\mathcal{R}}(t_2|_1) = D_{t_2|_1}^{\uparrow}(\ \cup_{l \in L(\mathcal{R}) \wedge l \not\lesssim t} DP_l(t_2|_1)\ )$. Now we consider $DP_l(t_2|_1)$ and $DP_{l'}(t_2|_1)$ for left-hand sides $l =$ `M % s(N)` and $l' =$ `(0 − s(M)) % s(N)`; note that $DP_{l''}(t_2|_1) = \{\Lambda\}$ for any other rule $l''$ in $\mathcal{R}$. Then, we have $DP_l(t_2|_1) = \{2\}$ and $DP_{l'}(t_2|_1) = \{1, 2\}$ and we can conclude $NR_{\mathcal{R}}(t_2|_1) = 1.NR_{\mathcal{R}}(t_2|_{1.1}) \cup 2.NR_{\mathcal{R}}(t_2|_{1.2})$. Now, this implies that we have to compute recursively $NR_{\mathcal{R}}(t_2|_{1.1})$ and $NR_{\mathcal{R}}(t_2|_{1.2})$. Now, $NR_{\mathcal{R}}(t_2|_{1.2})$ was already

computed before, and the reader can check that $NR_{\mathcal{R}}(t_2|_{1.1}) = \{\langle\Lambda\rangle\}$. So, we can conclude $NR_{\mathcal{R}}(t_2|_1) = \{\ \langle 1\rangle, \langle 2\rangle\ \}$.

Now, we consider $NR_{\mathcal{R}}(t_2|_2)$. Subterm `10! % 0` is not a redex, thus $NR_{\mathcal{R}}(t_2|_2) = \cup_{q \in SP_{\mathcal{R}}(t_2|_2)\setminus\{\Lambda\}} q.NR_{\mathcal{R}}(t_2|_{2.q})$. But using a similar reasoning that in the previous term $t_2|_1$, we can conclude $NR_{\mathcal{R}}(t_2|_2) = \{\ \langle 1\rangle\ \}$. Putting everything together, we have that $NR_{\mathcal{R}}(t_2) = \{\langle 1.1\rangle, \langle 1.2\rangle, \langle 2.1\rangle\}$.

The following is a property of $NR_{\mathcal{R}}(t)$ that will be useful later on.

*Remark 1.* If $\langle q, l \to r\rangle \in NR_{\mathcal{R}}(t)$ and $p \leq q.q'$ for $q' \in D_t^{\uparrow}(\mathcal{P}os_{\mathcal{X}}(l))$, then $p.NR_{\mathcal{R}}(t|_p) \subseteq NR_{\mathcal{R}}(t)$.

We are now ready to formally define the natural rewriting strategy.

**Definition 4 (Natural rewriting).** *We say term $t$ reduces by* natural rewriting *to term $s$, denoted by $t \xrightarrow{\mathtt{n}}_{\langle p, l\to r\rangle} s$ (or simply $t \xrightarrow{\mathtt{n}} s$) if $t \to_{\langle p,l\to r\rangle} s$ and $\langle p, l \to r\rangle \in NR_{\mathcal{R}}(t)$.*

*Example 7.* Continuing Example 6, we have three possible natural rewriting steps from the term $t_2$: (i) a rewriting step reducing the subterm `1−1` at position 1.2, (ii) a rewriting step reducing the subterm `10!` at position 1.1, and (iii) a rewriting step reducing the subterm `10!` at position 2.1. The last two rewriting steps are undesirable and unnecessary for obtaining the normal form `True`, as shown in Example 1. Using the further refinements to the natural rewriting strategy presented in Section 4, we will be able to avoid reducing these unnecessary redexes.

It is worthy to note that although some refinements are still necessary to obtain the rewrite strategy we want, we are already able to avoid some unnecessary rewrite steps, as shown in the following example.

*Example 8.* Consider Example 1 and the term $t =$ `0 ÷ s(10!)`. The term is a redex, so we have $NR_{\mathcal{R}}(t) = \{\ \langle\Lambda\rangle\ \} \cup \cup_{q \in SP_{\mathcal{R}}(t)\setminus\{\Lambda\}} q.NR_{\mathcal{R}}(t|_q)$. Now consider $SP_{\mathcal{R}}(t)$. Since $t$ is a redex but the only defined symbol is at root, we have $SP_{\mathcal{R}}(t) = D_t^{\uparrow}(\ \cup_{l \in L(\mathcal{R}) \wedge l \not\leq t} DP_l(t)\ )$. Now, we have that $DP_l(t) = \varnothing$ for $l =$ `0 ÷ s(M)`, $DP_{l'}(t) = \{1\}$ for $l' =$ `s(M) ÷ s(N)`, and $DP_{l''}(t) = \{\Lambda\}$ for any other rule $l''$ in $\mathcal{R}$. Then, $SP_{\mathcal{R}}(t) = \{\Lambda\}$, since position 1 corresponds to a constructor, and therefore $NR_{\mathcal{R}}(t) = \{\ \langle\Lambda\rangle\ \}$. So, our natural rewriting strategy performs only the sequence: `0 ÷ s(10!)` $\to$ `0` and avoids any reduction on the computational expensive term `10!`.

In the remaining part of this section, we show that the natural rewriting strategy defined above, satisfies the correctness and completeness criteria w.r.t to head-normal forms, that are described in Section 2.

**Theorem 1 (Correctness).** *If a term $t$ is a $\xrightarrow{\mathtt{n}}$-normal form, then $t$ is root-stable.*

*Proof.* By structural induction on $t$. Consider $|t| = 1$. If $t \in \mathcal{X}$ or $t \in \mathcal{C}$, then $t$ is obviously root-stable. Consider $t = f \in \mathcal{D}$. Since $NR_{\mathcal{R}}(t) = \varnothing$, by Definition 3, there is no lhs rooted by symbol $f$ and the conclusion follows.

Consider $|t| > 1$. We are done if for each rule $l \to r \in \mathcal{R}$ we show that (i) $t$ is not a redex and (ii) for each $p \in D_t^{\uparrow}(DP_l(t))$ the term $t|_p$ is root-stable. The idea behind (ii) is that before a rule $l \to r \in \mathcal{R}$ can be applied at the root position in $t$, the demanded positions between $t$ and $l$ have first to be "changed," which is possible only via reductions at some $p \in D_t^{\uparrow}(DP_l(t))$. Now, (i) is clear from the fact that $NR_{\mathcal{R}}(t) = \varnothing$, and (ii) follows by the induction hypothesis, because, $NR_{\mathcal{R}}(t) = \varnothing$ implies $NR_{\mathcal{R}}(t|_p) = \varnothing$ by Definition 3. □

In order to prove completeness of the natural rewriting strategy, we introduce some auxiliary notation. Given two rewrite sequences $\pi = t \to^* s$ and $\pi' = s \to^* w$, we write $\pi;\pi'$ for the sequence $t \to^* s \to^* w$. Given a rewrite sequence $\pi = t_0 \xrightarrow{p_1} t_1;\pi'$ with $\pi' = t_1 \xrightarrow{p_2} t_2 \cdots \xrightarrow{p_n} t_n$ and given an outermost position $p_k$ amongst $p_1, \ldots, p_n$, we define the projection $\pi|_{p_k}$ as follows:

$$\pi|_{p_k} = \begin{cases} \pi & \text{if } n = 0 \\ \pi'|_{p_k} & \text{if } p_1 \parallel p_k \\ t_0|_{p_k} \xrightarrow{p_1/p_k} t_1|_{p_k}; \pi'|_{p_k} & \text{otherwise} \end{cases}$$

We now establish a few key properties of the set $NR_{\mathcal{R}}(t)$ that will be useful in proving the completeness result.

**Lemma 1.** *Consider a rewrite sequence*

$$t \xrightarrow{p_1}_{l_1 \to r_1} t_1 \cdots \xrightarrow{p_n}_{l_n \to r_n} t_n$$

*such that $p_n = \Lambda$. Then, there is a $k$ s.t. $1 \leq k \leq n$ and $\langle p_k, l_k \to r_k \rangle \in NR_{\mathcal{R}}(t)$.*

*Proof.* We prove the lemma by induction on $n$. The base case where $n = 1$ is obvious, because then $\langle p_1, l_1 \to r_1 \rangle \in NR_{\mathcal{R}}(t)$. For the induction step, we may assume that $\langle p_n, l_n \to r_n \rangle$ is not a redex in $t$, because otherwise it is in $NR_{\mathcal{R}}(t)$, and the statement follows. Then, we have $DP_{l_n}(t) \neq \varnothing$, and for each $q \in DP_{l_n}(t)$ there is a $k < n$ such that $p_k \in D_t^{\uparrow}(q)$. Consider an outermost such $p_k$, and the rewrite sequence

$$\pi' = \pi|_{p_k} = t' \xrightarrow{p'_1}_{l'_1 \to r'_1} t'_1 \cdots \xrightarrow{p'_m}_{l'_m \to r'_m} t'_m$$

where $\pi = t \xrightarrow{p_1}_{l_1 \to r_1} t_1 \cdots \xrightarrow{p_k}_{l_k \to r_k} t_k$ and $p'_m = \Lambda$. By induction hypothesis, there is a $j$ such that $1 \leq j \leq m$ and $\langle p'_j, l'_j \to r'_j \rangle \in NR_{\mathcal{R}}(t')$. Now, note that $t' = t|_{p_k}$, $p'_j = p_i/p_k$, and $l'_j \to r'_j = l_i \to r_i$ for some $i$ s.t. $1 \leq i \leq k$. But since $p_k \in D_t^{\uparrow}(DP_{l_n}(t))$, and hence in $SP_{\mathcal{R}}(t)$, is follows that $\langle p_i, l_i \to r_i \rangle \in NR_{\mathcal{R}}(t)$ and we are done. □

**Lemma 2.** *Consider a rewrite sequence $t \xrightarrow{p_1}_{l_1 \to r_1} t_1 \cdots \xrightarrow{p_n}_{l_n \to r_n} t_n$ such that $\langle p_i, l_i \to r_i \rangle \notin NR_{\mathcal{R}}(t)$ for all $i$ s.t. $1 \leq i \leq n$. Then, there is no $i$ and $\langle q, l \to r \rangle \in NR_{\mathcal{R}}(t)$ such that $p_i \leq q.q'$ for $q' \in \mathcal{P}os_{\mathcal{X}}(l)$.*

*Proof.* We prove the contrapositive. Suppose there is an $i$ such that $1 \leq i \leq n$ and $p_i \leq q.q'$ for $q' \in \mathcal{P}os_{\mathcal{X}}(l)$. Consider an outermost such $p_k$, and the rewrite sequence

$$\pi' = \pi|_{p_k} = t' \xrightarrow{p'_1}_{l'_1 \to r'_1} t'_1 \cdots \xrightarrow{p'_m}_{l'_m \to r'_m} t'_m$$

where $\pi = t \xrightarrow{p_1}_{l_1 \to r_1} t_1 \cdots \xrightarrow{p_k}_{l_k \to r_k} t_k$ and $p'_m = \Lambda$. Now, by Lemma 1, there is $1 \leq j \leq m$ such that $\langle p'_j, l'_j \to r'_j \rangle \in NR_{\mathcal{R}}(t')$. Now, note that $t' = t|_{p_k}$, $p'_j = p_i/p_k$, and $l'_j \to r'_j = l_i \to r_i$ for some $i$ s.t. $1 \leq i \leq k$. But by Remark 1, it follows that $\langle p_i, l_i \to r_i \rangle \in NR_{\mathcal{R}}(t)$ and we are done. □

Recall that the main idea behind our natural rewriting strategy is that if a term $t$ can be reduced to a head-normal form $t'$, say via a rewrite sequence $\pi = t \to^* t'$, then there is also a rewrite sequence $t \xrightarrow{q}_{l \to r} s \to^* t'$ that begins with a redex $\langle q, l \to r \rangle \in NR_{\mathcal{R}}(t)$. Furthermore, we will show that the rewrite sequence $\pi' = s \to^* t'$ is "smaller" in an appropriate sense in comparison to $\pi$. Specifically, we will define a well-founded metric on rewrite sequences, and show that the metric of $\pi'$ is strictly smaller than that of $\pi$. The completeness result will then follow by noetheranian induction on this metric.

**Definition 5.** *Given a rewrite sequence $\pi = t_0 \xrightarrow{p_1} t_1 \cdots \xrightarrow{p_n} t_n$, we define a metric $\mu(\pi)$ as follows.*

- *Let $k$ be the smallest integer such that $p_k = \Lambda$, if any. Then,*

  $$\mu(\pi) = \mu(\pi_1).1.\mu(\pi_2)$$

  *where $\pi_1 = t_0 \to^* t_{k-1}$ and $\pi_2 = t_k \to^* t_n$.*
- *If $p_i \neq \Lambda$ for all $i$, then let $q_1, \ldots, q_k$ be the outermost positions in $p_1, \ldots, p_n$. We define*

  $$\mu(\pi) = \sum_{i=1}^{k} \mu(\pi|_{q_i})$$

  *where $+$ is inductively defined as: $n_1.v_1 + n_2.v_2 = (n_1+n_2).(v_1+v_2)$ $\epsilon + v = v$, and $v + \epsilon = v$.*

*We define the ordering $<$ on metrics as follows $v_1 < v_2$ if (i) $|v_1| < |v_2|$, or (ii) $|v_1| = |v_2|$, $v_1 = v.n_1.v'_1$, $v_2 = v.n_2.v'_2$, and $n_1 < n_2$. Note that $<$ is a well-ordering.*

The metric $\mu(\pi)$ essentially represents the parallelism that is implicit in the rewrite sequence $\pi$. Specifically, consider the rewrite sequence in Definition 5. If $p_k = \Lambda$, then the first $k-1$ rewrites in $\pi$ have to be performed before the $k^{th}$ rewrite, and similarly the $k^{th}$ rewrite has to be performed before any of the remaining $n-k$ rewrites. On the other hand, if $p_i$ and $p_j$ are two different outermost positions in $p_1, \ldots, p_n$ then all the rewrites in $\pi|_{p_i}$ and $\pi|_{p_j}$ can be performed parallely. Thus, $|\mu(\pi)|$ is the number of sequential steps that would remain when $\pi$ is parallelized to the extent possible, and further, if the $i^{th}$ number in the sequence $\mu(\pi)$ is $n_i$ then the $i^{th}$ step in the parallelized version of $\pi$ would contain $n_i$ parallel reductions.

*Example 9.* Consider the TRS of Example 1 and the following sequence $\pi$:

$$\begin{aligned}
&\mathtt{s((}\underline{\mathtt{0 - 0}}\mathtt{) - 0) - s(0 - 0)}\\
&\quad\to \mathtt{s(}\underline{\mathtt{0 - 0}}\mathtt{) - s(0 - 0)}\\
&\quad\to \mathtt{s(0) - s(}\underline{\mathtt{0 - 0}}\mathtt{)}\\
&\quad\to \underline{\mathtt{s(0) - s(0)}} \to \underline{\mathtt{0 - 0}} \to \mathtt{0}
\end{aligned}$$

The metric for this sequence is $\mu(\pi) = \mu(\pi').1$, where $\pi'$ is the sequence containing the first four steps of $\pi$. Further, $\mu(\pi') = \mu(\pi'').1$, where $\pi''$ is the sequence containing the first three steps of $\pi$. Now, the outermost positions of $\pi''$ are namely 1.1 and 2.1, and hence we have $\mu(\pi'') = \mu(\pi'|_{1.1}) + \mu(\pi'|_{2.1})$. Further, $\pi'|_{1.1} = \mathtt{(}\underline{\mathtt{0 - 0}}\mathtt{) - 0} \to \underline{\mathtt{0 - 0}} \to \mathtt{0}$, and $\pi'|_{2.1} = \underline{\mathtt{0 - 0}} \to \mathtt{0}$. Now, $\mu(\pi'|_{2.1}) = 1$, and the reader can check that $\mu(\pi'|_{1.1}) = 1.1$. So finally, $\mu(\pi) = \mu(\pi').1 = \mu(\pi'').1.1 = (\mu(\pi'|_{1.1}) + \mu(\pi'|_{2.1})).1.1 = (1.1 + 1).1.1 = 2.1.1.1$, that indicates that there are two steps at the beginning that can be performed parallely, followed by three other steps that cannot be performed parallely.

The following is a useful property of the metric, which can be easily proved.

**Lemma 3.** $|\mu(\pi_1 \ ; \ \pi_2)| \leq |\mu(\pi_1)| + |\mu(\pi_2)|$.

For a term $s$, a position $q$, and a rewrite sequence $\pi = t_0 \xrightarrow{p_1} t_1 \cdots \xrightarrow{p_n} t_n$, we define the rewrite sequence $s[\pi]_q$ as $s[t_0]_q \xrightarrow{q.p_1} s[t_1]_q \cdots \xrightarrow{q.p_n} s[t_n]_q$. For a term $t$, two disjoint positions $p_1, p_2$, and two rewrite sequences $\pi_1, \pi_2$, we define the rewrite sequence $\pi = t[\pi_1]_{p_1} \diamond t[\pi_2]_{p_2}$ as $t[\pi_1]_{p_1}; s[\pi_2]_{p_2}$ where $s$ is the target of $t[\pi_1]_{p_1}$. Note that $\diamond$ is an associative operator, and $\mu(t[\pi_1]_{p_1} \diamond t[\pi_2]_{p_2}) = \mu(\pi_1) + \mu(\pi_2)$. For a set of mutually disjoint positions $Q = \{q_1, \ldots, q_k\}$, we write $t[\pi]_Q$ as shorthand for $t[\pi]_{q_1} \diamond \cdots \diamond t[\pi]_{q_k}$.

We are now ready to formalize the intuition behind the definition of our natural rewriting strategy.

**Definition 6 (Descendants of a position).** *Let $\pi : t \xrightarrow{p}_{l\to r} s$ be a rewrite step and $q \in \mathcal{P}os(t)$. The set $q\backslash\pi$ of* descendants *of $q$ in $s$ is defined as follows:*

$$q\backslash\pi = \begin{cases} \{q\} & \text{if } q < p \text{ or } q \parallel p \\ \{p.p_3.p_2 \mid r|_{p_3} = l|_{p_1}\} & \text{if } q = p.p_1.p_2,\ p_1 \in \mathcal{P}os_{\mathcal{X}}(l) \\ \varnothing & \text{otherwise} \end{cases}$$

If $Q \subseteq \mathcal{P}os(t)$ then $Q\backslash\pi$ denotes the set $\bigcup_{q\in Q} q\backslash\pi$. The notion of descendant extends to rewrite sequences in the obvious way. Note that, if $Q$ is a set of pairwise disjoint positions in $t$, and $\pi : t \to^* s$, then the positions in $Q\backslash\pi$ are pairwise disjoint.

**Lemma 4.** *Let $\pi = t_1 \to^* t_2 \xrightarrow{\Lambda}_{l\to r} t_3 \to^* t_4$ where $\langle \Lambda, l\to r\rangle \in NR_{\mathcal{R}}(t_1)$ and none of the redexes in $t_1 \to^* t_2$ is in $NR_{\mathcal{R}}(t_1)$. Then, there is $\pi' = t_1 \xrightarrow{\Lambda}_{l\to r} t_2' \to^* t_4$ such that $\mu(t_2' \to^* t_4) < \mu(\pi)$.*

*Proof.* Let the set of outermost positions at which reductions occur in the sequence $\rho = t_1 \to^* t_2$ be $Q$. Then, by Lemma 2, we have that each $q \in Q$ is either

(i) under a variable position in $l$, i.e., $q \notin \mathcal{P}os_{\mathcal{F}}(l)$, or (ii) is not above any variable position in $l$, i.e. $q \not\leq p$ for any $p \in \mathcal{P}os_{\mathcal{X}}(l)$. Further, since $\langle \Lambda, l \to r \rangle \in NR_{\mathcal{R}}(t_1)$ we have that $\langle \Lambda, l \to r \rangle$ is a redex in $t_1$, and hence $\eta : t_1 \xrightarrow{\Lambda}_{l \to r} t_2'$ for some $t_2'$. The fact that $\langle \Lambda, l \to r \rangle$ is also a redex in $t_2$ implies that

- rewrites in $t_1 \to^* t_2$ that occur under a $q \in Q$ of the kind (ii) above are inconsequential, and
- for any two positions $p, p' \in \mathcal{P}os_x(l)$, the rewrite sequences $\rho|_p$ and $\rho|_{p'}$ have the same target.

This motivates us to consider a maximal set $\{q_1, \ldots, q_n\} \subseteq Q$ such that, (i) each $q_i$ is under a variable position in $l$, and (ii) whenever there are $p, p' \in \mathcal{P}os_x(l)$ and $q_i, q_j$ such that $p < q_i$ and $p' < q_j$, then $p = p'$. More concretely, for each variable $x \in \mathcal{V}ar(l)$, we pick one occurrence of $x$ in $l$, and then pick only those $q_i \in Q$ that are below the position of that occurrence of $x$. Now, for $\rho_i = \rho|_{q_i}$ for $1 \leq i \leq n$, we have the sequence $\pi' = \eta \ ; \ \delta \ ; \ t_3 \to^* t_4$ where $\delta = (t_2'[\rho_1]_{q_1 \setminus \pi} \diamond \cdots \diamond t_2'[\rho_n]_{q_n \setminus \pi})$ whose target is $t_3$. The rewrite sequence $\pi'$ simply shuffles all the relevant rewrites in $t_1 \to^* t_2$ to after the rewrite step $\eta$. Now, we have

$$|\mu(\rho)| \ = \ |\sum_{q \in Q} \mu(\rho|_q)| \ \geq \ |\sum_{i=1}^{n} \mu(\rho_i)| \ = \ |\mu(\delta)|$$

From the above observation and the fact that $\mu(\pi) = \mu(\rho).1.\mu(t_3 \to^* t_4)$, Lemma 3 implies $|\mu(\delta \ ; \ t_3 \to^* t_4)| < |\mu(\pi)|$. This in turn implies $\mu(\delta \ ; \ t_3 \to^* t_4) < \mu(\pi)$. □

**Lemma 5.** *Let $\pi = t_1 \to^* t_2 \xrightarrow{q}_{l \to r} t_3 \to^* t_4$ where $\langle q, l \to r \rangle \in NR_{\mathcal{R}}(t_1)$ and none of the redexes in $t_1 \to^* t_2$ is in $NR_{\mathcal{R}}(t_1)$. Then, there is $\pi' = t_1 \xrightarrow{q}_{l \to r} t_2' \to^* t_4$ such that $\mu(t_2' \to^* t_4) < \mu(\pi)$.*

*Proof.* We prove the lemma by induction on $|\pi|$. The base case $|\pi| = 1$ is obvious. For the induction step, let $p_1, \ldots, p_n$ be the outermost redexes in $\pi$ and $\pi_i = \pi|_{p_i}$. Let $k$ be such that $p_k \leq q$. By Lemma 2, we know that there is no reduction at $p_k$ in $t_1 \to^* t_2$. Then consider the rewrite sequence

$$\pi_k = t_1|_{p_k} \to^* t_2|_{p_k} \xrightarrow{q/p_k}_{l \to r} t_3|_{p_k} \to^* t_4|_{p_k}$$

We have the following two cases:

- $(q = p_k)$ Then, $q/p_k = \Lambda$ and by Lemma 4, there is $\pi_k' = t_1|_{p_k} \xrightarrow{\Lambda}_{l \to r} s \ ; \ \delta$ for some $s$ and a rewrite sequence $\delta$ with the same target $t_4|_{p_k}$, and $\mu(\delta) < \mu(\pi_k)$.

- $(p_k < q)$ Then, $\pi_k = \eta \ ; \ (u \xrightarrow{\Lambda} u' \to^* t_4|_{p_k})$ for some $u, u'$ and $\eta = t_1|_{p_k} \to^* t_2|_{p_k} \xrightarrow{q/p_k} t_3|_{p_k} \to^* u$. Since $|\eta| < |\pi_k| \leq |\pi|$ by the induction hypothesis there is $\eta' = t_1|_{p_k} \xrightarrow{q/p_k}_{l \to r} s \ ; \ \delta'$ for some $s$ and a rewrite sequence $\delta$ with

the same target $u$. Further, $\mu(\delta') < \mu(\eta)$. Let $\pi'_k = t_1|_{p_k} \xrightarrow{q/p_k}_{l\to r} s \ ; \ \delta$, where $\delta = \delta' \ ; \ (u \xrightarrow{\Lambda} u' \to^* t_4|_{p_k})$. Now,

$$\mu(\delta) \ = \ \mu(\delta').1.\mu(u' \to^* t_4|_{p_k}) \ < \ \mu(\eta).1.\mu(u' \to^* t_4|_{p_k}) \ = \ \mu(\pi_k)$$

Now, consider the sequence $\pi' = t_1[\pi'_k]_q \diamond (\diamond_{i\neq k} t_1[\pi_i]_{p_i})$. Note that for some $t'_2$, we have

$$\pi' = t_1 \xrightarrow{q}_{l\to r} t'_2 \ ; \ \rho \quad \text{where } \rho = t'_2[\delta]_q \diamond (\diamond_{i\neq k} t'_2[\pi_i]_{p_i})$$

Finally, we have $\mu(\rho) \ = \ \mu(\delta) + \sum_{i\neq k} \mu(\pi_i) \ < \ \sum_{i=1}^{n} \mu(\pi_i) \ = \ \mu(\pi)$. □

**Theorem 2 (Completeness).** *If $t \to^* s$, then $\exists s'$ s.t. $t \xrightarrow{\mathtt{m}}{}^* s'$, $root(s') = root(s)$ and $s' \xrightarrow{>\Lambda}{}^* s$.*

*Proof.* Let $\pi = t \xrightarrow{p_1}_{l_1\to r_1} t_1 \cdots \xrightarrow{p_n}_{l_n\to r_n} s$. We prove the theorem by noetherian induction on $\mu(\pi)$. The base case $\mu(\pi) = \epsilon$ is obvious, since $|\pi| = 0$. For the induction step there are two cases:

- Suppose there is no $i$ such that $1 \leq i \leq n$ and $\langle p_i, l_i \to r_i\rangle \in NR_{\mathcal{R}}(t)$. Then, by Lemma 1, $p_i > \Lambda$ for all $i$, and thus the statement holds by taking $s' = t$.
- Now, we consider the least $k$ such that $\langle p_k, l_k \to r_k\rangle \in NR_{\mathcal{R}}(t)$. Then, by Lemma 5, we have $\pi' = t \xrightarrow{p_k}_{l_k\to r_k} t'_1 \ ; \ \delta$ for some $t'_1$ and a rewrite sequence $\delta$ with target $s$. Furthermore, $\mu(\delta) < \mu(\pi)$. By induction hypothesis, we have $t'_1 \xrightarrow{\mathtt{m}}{}^* s'$ for some $s'$ such that $root(s') = root(s)$ and $s' \xrightarrow{>\Lambda}{}^* s$. Then, the statement follows from the observation that $t \xrightarrow{\mathtt{m}}{}^* s'$. □

## 4 Refinements of the Strategy

In this section, we further refine the natural rewriting strategy defined in Section 3. We use the notions of failing terms and most frequently demanded positions, both of which were originally introduced, although for the special case of left-linear and constructor-based systems, in [7].

### 4.1 Failing terms

For a position $p$ and a term $t$, we define the set $R_t(p)$ of reflections of $p$ w.r.t. $t$ as follows: if $p$ is under a variable position in $t$, i.e., $p = q.q'$ for some $q$ such that $t|_q = x$ then $R_t(p) = \mathcal{P}os_x(t).q'$, else $R_t(p) = \{p\}$. We say that the path to $p$ in $t$ has defined symbols if there is $q \leq p$ and $q \neq \Lambda$ such that $root(t|_q) \in \mathcal{D}$.

**Definition 7 (Failing term).** *Given terms $l, t$, we say $t$ is* failing *w.r.t. $l$, denoted by $l \blacktriangleleft t$, if there is $p \in DP_l(t)$ such that the path to $p$ in $t$ does not have defined symbols, and either of the following holds:*

- *for every $q \in R_l(p) \cap DP_l(t)$, the path to $q$ in $t$ does not have defined symbols,*

– *there is* $q \in R_l(p) \cap DP_l(t)$ *with* $root(t|_p) \neq root(t|_q)$*, and the path to* $q$ *in* $t$ *does not have defined symbols.*

The idea behind the definition above is that if $l \blacktriangleleft t$, then it is the case that in any rewrite sequence starting from $t$ the first reduction at the root does not use the rule $l \to r$.

*Example 10.* Consider the terms $t = \texttt{10!} \ \% \ \texttt{0}$ and $l = \texttt{M} \ \% \ \texttt{s(N)}$ from Example 1. We have that $l \blacktriangleleft t$ because the position $2 \in DP_l(t)$, $R_l(2) = \{2\}$, and the path to position 2 in $t$ has no defined symbol except the root symbol. Now, consider the terms $t' = \texttt{s(Z)} \approx \texttt{0}$ and $l' = \texttt{X} \approx \texttt{X}$, again from Example 1. We have $l' \blacktriangleleft t'$, since the position $1 \in DP_{l'}(t')$, $R_{l'}(1) = \{1, 2\}$, the path to position 1 has no defined symbols, $root(t'|_1) = \texttt{s} \neq \texttt{0} = root(t'|_2)$, and the path to position 2 in $t'$ has no defined symbol.

Now, note that if $l \blacktriangleleft t$, then we need not include the set $D_t^{\uparrow}(DP_l(t))$ in the set $SP_{\mathcal{R}}(t)$ in Definition 3, because we know that $l \to r$ cannot be the first rule to be applied at the root in any rewrite sequence starting from $t$.

**Definition 8.** *We refine the set* $NR_{\mathcal{R}}(t)$ *in Definition 3 by refining the set* $SP_{\mathcal{R}}(t)$ *as follows*

$$SP_{\mathcal{R}}(t) = \bigcup_{l \in L(\mathcal{R}) \wedge l \leq t} D_t^{\uparrow}(\mathcal{P}os_{\mathcal{X}}(l)) \ \cup \ D_t^{\uparrow}( \bigcup_{l \in L(\mathcal{R}) \wedge l \not\leq t \wedge l \not\blacktriangleleft t} DP_l(t) \ )$$

The reader can check that Lemmas 1 and 2, and hence all the propositions so far, hold with this refined Definition 8 instead of Definition 3.

*Example 11.* Let us continue Example 7 with the term

$$t_2 = \texttt{10!} \ \% \ \texttt{(1-1)} \approx \texttt{10!} \ \% \ \texttt{0}$$

With the refined Definition 8 we have $NR_{\mathcal{R}}(t_2) = \{\langle 1.1 \rangle, \langle 1.2 \rangle\}$ and the redex at position 2.1 is not considered anymore. The reason is that according to Example 10, the subterm $\texttt{10!} \ \% \ \texttt{0}$ is failing w.r.t. every rule, and hence $SP_{\mathcal{R}}(t_2|_2) = \varnothing$. Hence, we have only two possible rewriting steps from the term $t_2$: (i) a rewriting step reducing the subterm $\texttt{1-1}$ at position 1.2, and (ii) a rewriting step reducing the subterm $\texttt{10!}$ at position 1.1. The second rewrite step is still undesirable and its removal motivates the further refinement introduced below.

### 4.2 Most frequently demanded positions

Suppose $l_n \not\leq t$, $l \not\blacktriangleleft t$, and we have a rewrite sequence $t \xrightarrow{p_1}_{l_1 \to r_1} \dots \xrightarrow{p_n}_{l_n \to r_n} t_n$ where $p_n = \Lambda$. Then note that for every $q \in DP_l(t)$ there has to be a $k$ such that $p_k \in D_t^{\uparrow}(q)$. This is indeed the argument used in Lemma 1. But then this implies that while computing the set $SP_{\mathcal{R}}(t)$ in Definition 8, for each $l \in L(\mathcal{R})$ such that $l \not\leq t$ and $l \not\blacktriangleleft t$ it is sufficient to include $D_t^{\uparrow}(q)$ for *atleast one* $q \in DP_l(t)$. This motivates a further refinement of Definition 8.

**Definition 9.** *For sets of positions $P, Q_1, \ldots, Q_n$ we say $P$ covers $Q_1, \ldots, Q_n$ if for all $1 \leq i \leq n$ we have that $P \cap Q_i \neq \varnothing$. For sets of positions $Q_1, \ldots, Q_n$ in a term $t$, we define $MinCovD_t^{\uparrow}(\{Q_1, \ldots, Q_n\}) = D_t^{\uparrow}(P)$ for some cover $P$ of $Q_1, \ldots, Q_n$ such that $|D_t^{\uparrow}(P)|$ is minimum.*

*Example 12.* Consider the subterm $t = \texttt{10!} \ \% \ \texttt{(1-1)}$ of the term $t_2$ in Example 1. Consider also the left-hand sides of the rules (3) and (4): $l_3 = \texttt{M} \ \% \ \texttt{s(N)}$ and $l_4 = \texttt{(0 - s(M))} \ \% \ \texttt{s(N)}$. We have that $DP_{l_3}(t) = \{2\}$ and $DP_{l_4}(t) = \{1, 2\}$. Now, the set $P = \{2\}$ covers $DP_{l_3}(t)$ and $DP_{l_4}(t)$, and in fact

$$MinCovD_t^{\uparrow}(DP_{l_3}(t), DP_{l_4}(t)) = D_t^{\uparrow}(P) = \{\Lambda, 2\}$$

**Definition 10.** *We further refine the set $NR_{\mathcal{R}}(t)$ in Definition 8 by refining the set $SP_{\mathcal{R}}(t)$ as follows:*

$$SP_{\mathcal{R}}(t) = \bigcup_{l \in L(\mathcal{R}) \wedge l \leq t} D_t^{\uparrow}(\mathcal{P}os_{\mathcal{X}}(l)) \ \cup \ MinCovD_t^{\uparrow}\Big( \bigcup_{l \in L(\mathcal{R}) \wedge l \not\leq t \wedge l \blacktriangleleft t} \{DP_l(t)\} \Big)$$

The reader can check that Lemmas 1 and 2, and hence all the propositions so far, hold with this refined Definition 10 instead of Definition 8.

*Example 13.* Let us continue with term $t_2 = \texttt{10!} \ \% \ \texttt{(1-1)} \approx \texttt{10!} \ \% \ \texttt{0}$ from Example 11. With Definition 10, the redexes computed by the natural rewriting strategy become even more refined. Specifically, we have $NR_{\mathcal{R}}(t_2) = \{\langle 1.2 \rangle\}$ and the redex at position 1.1 is not considered anymore. The reason is that according to Example 12, the position 1 in the subterm $\texttt{10!} \ \% \ \texttt{(1-1)}$ is not considered as demanded, since position 2 is enough for obtaining a minimal set covering of all the positions demanded by rules (3) and (4), i.e. $SP_{\mathcal{R}}(t_2|_1) = \{2, \Lambda\}$. Finally, we have only the optimal possible rewriting step for position 1.2 from the term $t_2$ and the optimal rewrite sequence:

$$\texttt{10! \% (}\underline{\texttt{s(0)-s(0)}}\texttt{)} \approx \texttt{10! \% 0}$$
$$\rightarrow \texttt{10! \% (}\underline{\texttt{0-0}}\texttt{)} \approx \texttt{10! \% 0} \rightarrow \underline{\texttt{10! \% 0} \approx \texttt{10! \% 0}} \rightarrow \texttt{True}$$

Note that, for the terms

$$t_3 = \texttt{10! \% (0-0))} \approx \texttt{10! \% 0} \qquad \text{and} \qquad t_4 = \texttt{10! \% 0} \approx \texttt{10! \% 0}$$

that occur in the rewrite sequence above, we have $NR_{\mathcal{R}}(t_3) = \{\langle 1.2 \rangle\}$, and $NR_{\mathcal{R}}(t_4) = \{\langle \Lambda \rangle\}$.

## 5 Conclusion

We have extended natural rewriting to general rewriting systems. This makes this strategy available for both expressive equational languages and for rewriting logic languages. This work provides a basis for a subsequent generalization of natural rewriting first to narrowing, and second to even more expressive rewrite theories suited for concurrent system specifications [12] and supporting: (i) sorts and subsorts; (ii) rewriting *modulo* axioms such as associativity, commutativity, identity, and so on; and (iii) conditional rewriting.

## References


1. S. Antoy. Definitional trees. In *Proc. of the 3rd International Conference on Algebraic and Logic Programming ALP'92*, volume 632 of *Lecture Notes in Computer Science*, pages 143–157. Springer-Verlag, Berlin, 1992.
2. S. Antoy, R. Echahed, and M. Hanus. Parallel evaluation strategies for functional logic languages. In *Proc. of the Fourteenth International Conference on Logic Programming (ICLP'97)*, pages 138–152. MIT Press, 1997.
3. S. Antoy, R. Echahed, and M. Hanus. A needed narrowing strategy. In *Journal of the ACM*, volume 47(4), pages 776–822, 2000.
4. S. Antoy and S. Lucas. Demandness in rewriting and narrowing. In M. Comini and M. Falaschi, editors, *Proc. of the 11th Int'l Workshop on Functional and (Constraint) Logic Programming WFLP'02*, volume 76 of *Electronic Notes in Theoretical Computer Science*. Elsevier Sciences Publisher, 2002.
5. P. Borovanský, C. Kirchner, H. Kirchner, and P.-E. Moreau. ELAN from a rewriting logic point of view. *Theoretical Computer Science*, 285:155–185, 2002.
6. M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and J. Quesada. Maude: specification and programming in rewriting logic. *Theoretical Computer Science*, 285:187–243, 2002.
7. S. Escobar. Refining weakly outermost-needed rewriting and narrowing. In D. Miller, editor, *Proc. of 5th International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming, PPDP'03*, pages 113–123. ACM Press, New York, 2003.
8. S. Escobar. Implementing natural rewriting and narrowing efficiently. In Y. Kameyama and P. J. Stuckey, editors, *7th International Symposium on Functional and Logic Programming (FLOPS 2004)*, volume 2998 of *Lecture Notes in Computer Science*, pages 147–162. Springer-Verlag, Berlin, 2004.
9. K. Futatsugi and R. Diaconescu. *CafeOBJ Report*. World Scientific, AMAST Series, 1998.
10. J. Goguen, T. Winkler, J. Meseguer, K. Futatsugi, and J.-P. Jouannaud. Introducing OBJ. In *Software Engineering with OBJ: Algebraic Specification in Action*, pages 3–167. Kluwer, 2000.
11. G. Huet and J.-J. Lévy. Computations in Orthogonal Term Rewriting Systems, Part I + II. In *Computational logic: Essays in honour of J. Alan Robinson*, pages 395–414 and 415–443. The MIT Press, Cambridge, MA, 1992.
12. J. Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science*, 96(1):73–155, 1992.
13. A. Middeldorp. Call by need computations to root-stable form. In *Proceedings of the 24th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 94–105. ACM Press, New York, 1997.
14. S. Peyton-Jones. *The Implementation of Functional Programming Languages*. Prentice Hall International, London, 1987.
15. R. Sekar and I. Ramakrishnan. Programming in equational logic: Beyond strong sequentiality. *Information and Computation*, 104(1):78–109, 1993.
16. A. van Deursen, J. Heering, and P. Klint. *Language Prototyping: An Algebraic Specification Approach*. World Scientific, 1996.