Transformations for Efficient Evaluations in Functional Programming*

Salvador Lucas

Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Camino de Vera s/n, E-46071 Valencia, Spain.
e.mail: slucas@dsic.upv.es
URL: http://www.dsic.upv.es/users/elp/slucas.html

Abstract. Rewriting is a suitable mechanism to implement functional programming languages. The definition of efficient evaluation strategies is a natural concern in this setting. Recently we have shown that, by imposing some simple restrictions on the replacements which are allowed for the arguments of functions, we can obtain efficient head-evaluations of terms in a certain class of programs. We are also able to completely evaluate terms, but in this case, the efficiency of computation can be compromised. This paper concerns the definition of program transformations that enable efficient but still complete evaluation of function calls.

Keywords: evaluation, functional programming, program transformation, term rewriting.

1 Introduction

Functional languages which allow the definition of functions by means of patterns (e.g. Haskell, Hope or Miranda), are well suited to use term rewriting [4] as execution principle. Computing involves reduction of (input) terms to a normal form [9]. Any effective execution procedure must include a normalizing rewriting strategy able to compute the normal form of the input term if it exists. Such a reduction strategy is expected to be efficient both in performing only useful reductions and in its implementation [8]. For orthogonal term rewriting systems (TRSs), the Huet and Levy's notion of *needed reduction* [2] formalizes what we mean by a *useful reduction*. Needed reduction is any strategy which only selects needed redexes. A redex of a term t is needed if it is reduced (itself or some of its residuals) in every reduction sequence leading t to a normal form. Reduction of needed redexes is normalizing and, moreover, each step do actually contributes to normalization. Computing such needed redexes is not possible in the general case, but some approximations have been defined in the literature [2, 8].

Using *strongly replacing-independent* TRSs [7], it is possible to define a needed reduction strategy by only considering syntactic replacement restrictions: a mapping $\mu : \Sigma \to \mathcal{P}(\mathbb{N})$ (called the *replacement map*) indicates the fixed argument

* This work has been partially supported by CICYT under grant TIC 95-0433-C03-03.

positions $\mu(f) \subseteq \{1, \ldots, k\}$ of each k-ary function f for which we enable replacements. The restrictions on arguments are lifted to occurrences and the resulting restriction on rewriting is said context-sensitive rewriting (*csr*) [5, 6]. Given an orthogonal TRS $\mathcal{R}$, we are able to define a *canonical* replacement map $\mu_{\mathcal{R}}^{com}$ under which *csr* is able to compute head-normal forms [5]. An orthogonal, strongly replacing-independent TRS makes this by only reducing needed redexes [7]. The computation of such redexes is very easy, since we can take *any* replacing redex occurrence. This is to say that the replacement restrictions themselves are able to capture only needed redexes. Therefore, the implementation of a head-normalizing strategy for orthogonal strongly replacing-independent TRSs is nicely simple. For this reason, in this paper we analyze the problem of transforming an arbitrary TRS into an equivalent strongly replacing-independent TRS. This is always possible. However, orthogonality is not always preserved. Therefore, we have some limits in obtaining the desired benefits.

Computing head-normal forms is not completely satisfactory in functional programming. Our final goal is to *evaluate* a term t, i.e., to obtain the constructor term δ (the value of t) which expresses the meaning of t. *csr* is also able to compute values by using a less restrictive replacement map $\mu_{\mathcal{R}}^{\mathcal{B}}$ [5]. However, when dealing with orthogonal strongly replacing-independent TRSs, if we use $\mu_{\mathcal{R}}^{\mathcal{B}}$ instead of $\mu_{\mathcal{R}}^{com}$ (and $\mu_{\mathcal{R}}^{com} \neq \mu_{\mathcal{R}}^{\mathcal{B}}$) we loose the nice correspondence between replacing redexes and needed redexes. Now we have replacing redexes which are *not* needed. To solve this problem, we introduce a new transformation which is able to combine the requirement of achieving complete evaluations by only reducing the needed redexes which can be computed by means of replacement restrictions.

In Section 2, we give some preliminary definitions. In Section 3, we recall the main definitions concerning *csr*. Section 4 gives the transformations to obtain a strongly replacing-independent TRS from a given TRS. In Section 5, we define a program transformation which raises the good properties of *csr* concerning head-evaluations to complete evaluations. Section 6 compares with some related work and concludes.

2 Preliminaries

Let us first introduce the main notations used in the paper [1, 4]. V denotes a countable set of variables and Σ denotes a set of function symbols $\{\mathtt{f}, \mathtt{g}, \ldots\}$, each of which has a fixed arity given by a function $ar : \Sigma \to \mathbb{N}$. By $\mathcal{T}(\Sigma, V)$ we denote the set of terms. $\tilde{t}$ denotes a k-tuple $t_1, \ldots, t_k$ of terms, where k will be clarified from the context. $Var(t)$ is the set of variables in a term t. Terms are viewed as labelled trees in the usual way. Occurrences $u, v, \ldots$ are chains of positive natural numbers used to address subterms of t. If W is a set of occurrences, $u.W$ is the set $\{u.v \mid v \in W\}$. Occurrences are ordered by the standard prefix ordering: $u \leq v$ iff $\exists w \;\; v = u.w$. $O(t)$ denotes the set of occurrences of a term t. Given a subset $U \subseteq O(t)$, $U\!\downarrow$ is the set $U\!\downarrow = \{v \in O(t) \mid u \in U \wedge v \leq u\}$. $t|_u$ is the subterm at occurrence u of t. $t[s]_u$ is the term t with the subterm at the occurrence u replaced with s. $root(t)$ is the symbol labelling the root of t. $\Sigma(t) =$

$\{f \in \Sigma \mid \exists u \in O(t).root(t|_u) = f\}$ is the set of symbols from Σ appearing in t. Given a $T \subseteq \mathcal{T}(\Sigma, V)$, $\Sigma(T) = \cup_{t \in T}\Sigma(t)$. $O_\Sigma(t) = \{u \in O(t) \mid root(t|_u) \in \Sigma\}$ is the set of nonvariable occurrences in a term t. $O_V(t) = O(t)\backslash O_\Sigma(t)$ is the set of variable occurrences. We refer to any term C, which is the same as t everywhere except below u, i.e., $\exists s.\ C[s]_u = t$, as the *context* within the replacement occurs.

A rewrite rule is an ordered pair (l, r), written $l \to r$, with $l, r \in \mathcal{T}(\Sigma, V)$, $l \notin V$ and $Var(r) \subseteq Var(l)$. l is the left-hand side (*lhs*) of the rule and r is the right-hand side (*rhs*). A TRS is a pair $\mathcal{R} = (\Sigma, R)$ where R is a set of rewrite rules. $L(\mathcal{R})$ is the set of *lhs*'s of $\mathcal{R}$. An instance $\sigma(l)$ of a *lhs* $l \in L(R)$ is a redex. $O_\mathcal{R}(t) = \{u \in O(t) \mid \exists l \in L(\mathcal{R})\ :\ t|_u = \sigma(l)\}$ is the set of redex occurrences in a term t. Given a TRS $\mathcal{R} = (\Sigma, R)$, we consider Σ as the disjoint union $\Sigma = \mathcal{C} \uplus \mathcal{F}$ of symbols $c \in \mathcal{C}$, called *constructors*, having no associated rule and symbols $f \in \mathcal{F}$, called *defined functions* or *operations*, which are defined by some program rule $f(\tilde{l}) \to r \in R$: $\mathcal{F} = \{f \in \Sigma \mid f(\tilde{l}) \to r \in R\}$ and $\mathcal{C} = \Sigma\backslash\mathcal{F}$. Constructor terms (i.e., *values*) are denoted $\delta \in \mathcal{T}(\mathcal{C}, V)$.

For a given TRS $\mathcal{R} = (\Sigma, R)$, a term t rewrites to a term s (at the occurrence u), written $t \xrightarrow{u}_\mathcal{R} s$ (or just $t \to s$), if $t|_u = \sigma(l)$ and $s = t[\sigma(r)]_u$, for some rule $l \to r$ in R, occurrence u in t and substitution σ. $\to_\mathcal{R}$ is the one-step rewrite relation for $\mathcal{R}$. A term t is in head-normal form if there is no derivation $t = t_1 \to t_2 \to \cdots$ starting from t which reduces the root of a term t_i, $i \geq 1$.

$\mathbb{N}_k^+$ is an initial segment $\{1, 2, \ldots k\}$ of the set of positive natural numbers $\mathbb{N}^+$, where $\mathbb{N}_0^+ = \emptyset$. $\mathcal{P}(\mathbb{N})$ is the powerset of natural numbers.

3 Context-sensitive rewriting

A mapping $\mu : \Sigma \to \mathcal{P}(\mathbb{N})$ is a *replacement map* (or Σ-map) for the signature Σ iff for all $f \in \Sigma$. $\mu(f) \subseteq \mathbb{N}^+_{ar(f)}$. A Σ-map μ determines the *argument* positions which can be reduced for each symbol in Σ [5, 6]. If we assume an arbitrary ordering in the signature[2] $\Sigma = \{f_1, \ldots, f_n\}$, then we express a Σ-map as $\mu = \langle I_1, \ldots, I_n\rangle$, where $I_j = \mu(f_j)$ for $1 \leq j \leq n$. Given $\Delta \subseteq \Sigma$, $\mu{\downarrow_\Delta}$ is the Δ-map given by: $\mu{\downarrow_\Delta}(f) = \mu(f)$ for all $f \in \Delta$.

The ordering $\subseteq$ on $\mathcal{P}(\mathbb{N})$ extends pointwise to an ordering $\sqsubseteq$ on M_Σ, the set of all Σ-maps: $\mu \sqsubseteq \mu'$ if for all $f \in \Sigma$, $\mu(f) \subseteq \mu'(f)$. $(M_\Sigma, \sqsubseteq, \mu_\bot, \mu_\top, \sqcup, \sqcap)$ is a complete lattice: for all $f \in \Sigma$, $\mu_\bot(f) = \emptyset$, $\mu_\top(f) = \mathbb{N}^+_{ar(f)}$, $(\mu \sqcup \mu')(f) = \mu(f) \cup \mu'(f)$ and $(\mu \sqcap \mu')(f) = \mu(f) \cap \mu'(f)$. Thus, $\mu \sqsubseteq \mu'$ means that μ considers less positions than μ' for reduction.

The set of μ-*replacing* occurrences is given by: $O^\mu(x) = \{\epsilon\}$ if $x \in V$, $O^\mu(f(t_1, \ldots, t_k)) = \{\epsilon\} \cup \bigcup_{i \in \mu(f)} i.O^\mu(t_i)$. In *csr*, we only rewrite on *replacing* occurrences: t μ-rewrites to s, written $t \hookrightarrow_{\mathcal{R}(\mu)} s$, if $t \xrightarrow{u}_\mathcal{R} s$ and $u \in O^\mu(t)$. The set of *replacing redexes* is $O^\mu_\mathcal{R}(t) = O_\mathcal{R}(t) \cap O^\mu(t)$. The set of *non-replacing* occurrences is $\widetilde{O^\mu}(t) = O(t)\backslash O^\mu(t)$. We also have $O^\mu_\Sigma(t) = O_\Sigma(t) \cap O^\mu(t)$ and $O^\mu_V(t) = O_V(t) \cap O^\mu(t)$.

A term t is μ-*compatible* ($\mathsf{comp}_\mu(t)$) if its non-replacing occurrences are variables: $\mathsf{comp}_\mu(t)$ iff $\widetilde{O^\mu}(t) \subseteq O_V(t)$. Equivalently, $\mathsf{comp}_\mu(t)$ holds iff $O_\Sigma(t) \subseteq$

[2] Usually, the order in which the symbols are written.

$O^\mu(t)$. There exists a *minimum* $\Sigma(t)$-map μ_t ensuring compatibility: if $t = x$, then $\mu_t(f) = \emptyset$ for all $f \in \Sigma$. If $t = f(t_1, \ldots, t_k)$, then $\mu_t = \mu_t^\epsilon \sqcup \mu_{t_1} \sqcup \cdots \sqcup \mu_{t_k}$, where μ_t^ϵ is $\mu_t^\epsilon(f) = \{i \mid t_i \notin V\}$ and $\mu_t^\epsilon(g) = \emptyset$ for all $g \neq f$.

Example 3.1 *Given* $t = \mathtt{first}(\mathtt{0}, \mathtt{x})$, $\Sigma(t) = \{\mathtt{0}, \mathtt{first}\}$, *and we have:*

$$\mu_{\mathtt{first}(\mathtt{0},\mathtt{x})} = \mu^\epsilon_{\mathtt{first}(\mathtt{0},\mathtt{x})} \sqcup \mu_{\mathtt{0}} \sqcup \mu_{\mathtt{x}} = \langle \emptyset, \{1\}\rangle \sqcup \langle \emptyset, \emptyset\rangle \sqcup \langle \emptyset, \emptyset\rangle = \langle \emptyset, \{1\}\rangle.$$

For $t' = \mathtt{first}(\mathtt{s}(\mathtt{x}), \mathtt{y} :: \mathtt{z})$, $\Sigma(t') = \{\mathtt{s}, ::, \mathtt{first}\}$, *and we get* $\mu_{\mathtt{first}(\mathtt{s}(\mathtt{x}),\mathtt{y}::\mathtt{z})} = \mu^\epsilon_{\mathtt{first}(\mathtt{s}(\mathtt{x}),\mathtt{y}::\mathtt{z})} \sqcup \mu_{\mathtt{s}(\mathtt{x})} \sqcup \mu_{\mathtt{y}::\mathtt{z}} = \langle \emptyset, \emptyset, \{1,2\}\rangle \sqcup \langle \emptyset, \emptyset, \emptyset\rangle \sqcup \langle \emptyset, \emptyset, \emptyset\rangle = \langle \emptyset, \emptyset, \{1,2\}\rangle$.

Analogously, for a set of terms T, we define $\mu_T = \sqcup_{t \in T} \mu_t$. These replacement maps characterize compatibility: $\mathsf{comp}_\mu(t) \Leftrightarrow \mu_t \sqsubseteq \mu$ and $\mathsf{comp}_\mu(T) \Leftrightarrow \mu_T \sqsubseteq \mu$.

Given a TRS $\mathcal{R} = (\Sigma, R)$, its *canonical replacement map* $\mu^{com}_{\mathcal{R}}$ is $\mu^{com}_{\mathcal{R}} = \sqcup_{l \in L(\mathcal{R})} \mu_l$, where $L(\mathcal{R})$ is the set of *lhs*'s of the TRS. $\mu^{com}_{\mathcal{R}}$ is the minimum replacement map which makes the *lhs*'s of the TRS compatible terms.

Example 3.2 *Consider the following program* $\mathcal{R}$ *which defines some functions on lists of natural numbers:*

$\mathtt{dbl}(\mathtt{0}) \to \mathtt{0}$	$\mathtt{dbls}([]) \to []$
$\mathtt{dbl}(\mathtt{s}(\mathtt{x})) \to \mathtt{s}(\mathtt{s}(\mathtt{dbl}(\mathtt{x})))$	$\mathtt{dbls}(\mathtt{x} :: \mathtt{y}) \to \mathtt{dbl}(\mathtt{x}) :: \mathtt{dbls}(\mathtt{y})$
$\mathtt{sel}(\mathtt{0}, \mathtt{x} :: \mathtt{y}) \to \mathtt{x}$	$\mathtt{indx}([], \mathtt{x}) \to []$
$\mathtt{sel}(\mathtt{s}(\mathtt{x}), \mathtt{y} :: \mathtt{z}) \to \mathtt{sel}(\mathtt{x}, \mathtt{z})$	$\mathtt{indx}(\mathtt{x} :: \mathtt{y}, \mathtt{z}) \to \mathtt{sel}(\mathtt{x}, \mathtt{z}) :: \mathtt{indx}(\mathtt{y}, \mathtt{z})$
$\mathtt{from}(\mathtt{x}) \to \mathtt{x} :: \mathtt{from}(\mathtt{s}(\mathtt{x}))$	

The canonical replacement map for $\mathcal{R}$ *is:* $\mu^{com}_{\mathcal{R}}(f) = \emptyset$ *for all* $f \in \{\mathtt{0}, \mathtt{s}, [], ::, \mathtt{from}\}$, $\mu^{com}_{\mathcal{R}}(\mathtt{dbl}) = \mu^{com}_{\mathcal{R}}(\mathtt{dbls}) = \mu^{com}_{\mathcal{R}}(\mathtt{indx}) = \{1\}$ *and* $\mu^{com}_{\mathcal{R}}(\mathtt{sel}) = \{1, 2\}$.

Theorem 3.3 ([5]) *Let* $\mathcal{R} = (\Sigma, R) = (\mathcal{C} \uplus \mathcal{F}, R)$ *be a left-linear TRS and* μ *be a* Σ*-map such that* $\mu^{com}_{\mathcal{R}} \sqsubseteq \mu$. *Let* t *be a term and* $s = c(\tilde{s})$ *for some* $c \in \mathcal{C}$. *If* $t \to^* s$, *there exists* $s' = c(\widetilde{s'})$ *s.t.* $t \hookrightarrow^* s'$, *and* $s' \to^* s$.

Theorem 3.4 ([5]) *Let* $\mathcal{R} = (\Sigma, R)$ *be an orthogonal TRS and* μ *be a* Σ*-map such that* $\mu^{com}_{\mathcal{R}} \sqsubseteq \mu$. *Let* t *be a term and* $s = f(\tilde{s})$ *be a head-normal form. If* $t \to^* s$, *there exists a head-normal form* $s' = f(\widetilde{s'})$ *s.t.* $t \hookrightarrow^*_{\mathcal{R}(\mu)} s'$, *and* $s' \to^* s$.

Hence, from Theorem 3.3, for left-linear TRSs, we have completeness in *head-evaluations* (i.e., reductions leading to a constructor head-normal form). For orthogonal TRSs, we have completeness in head-normalization (Theorem 3.4).

Example 3.5 *For* $\mathcal{R}$ *in Example 3.2, we get the following head-evaluation using* $\mu^{com}_{\mathcal{R}}$ *(redexes underlined; we mark some steps for further discussion):*

$$
\begin{array}{lll}
\mathtt{sel}(\mathtt{s}(\mathtt{0}), \mathtt{dbls}(\underline{\mathtt{from}(\mathtt{0})})) & & \\
\quad \hookrightarrow \mathtt{sel}(\mathtt{s}(\mathtt{0}), \underline{\mathtt{dbls}(\mathtt{0} :: \mathtt{from}(\mathtt{s}(\mathtt{0})))}) & & (*) \\
\quad \hookrightarrow \underline{\mathtt{sel}(\mathtt{s}(\mathtt{0}), \mathtt{dbl}(\mathtt{0}) :: \mathtt{dbls}(\mathtt{from}(\mathtt{s}(\mathtt{0}))))} & & (**) \\
\quad \hookrightarrow \mathtt{sel}(\mathtt{0}, \mathtt{dbls}(\underline{\mathtt{from}(\mathtt{s}(\mathtt{0}))})) & & \\
\quad \hookrightarrow \mathtt{sel}(\mathtt{0}, \underline{\mathtt{dbls}(\mathtt{s}(\mathtt{0}) :: \mathtt{from}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))}) & & (*) \\
\quad \hookrightarrow \underline{\mathtt{sel}(\mathtt{0}, \mathtt{dbl}(\mathtt{s}(\mathtt{0})) :: \mathtt{dbls}(\mathtt{from}(\mathtt{s}(\mathtt{s}(\mathtt{0})))))} & & \\
\quad \hookrightarrow \underline{\mathtt{dbl}(\mathtt{s}(\mathtt{0}))} & & \\
\quad \hookrightarrow \mathtt{s}(\mathtt{s}(\mathtt{dbl}(\mathtt{0}))) & &
\end{array}
$$

4 Strongly replacing-independent TRSs

In this section, we introduce the strongly replacing-independent TRSs, for which we have proven that any context-sensitive reduction is needed [7]. A term t is strongly μ-compatible ($\mathsf{scomp}_\mu(t)$) if $\widetilde{O^\mu}(t) = O_V(t)$. Alternatively $\mathsf{scomp}_\mu(t)$ holds if $O_\Sigma(t) = O^\mu(t)$. Clearly $\mathsf{scomp}_\mu(t) \Rightarrow \mathsf{comp}_\mu(t)$. A set of terms $T \subseteq \mathcal{T}(\Sigma, V)$ is said to be *weakly compatible* ($\mathsf{wcomp}(T)$), if, for all $t \in T$, $\mathsf{scomp}_{\mu_t}(t)$. T is strongly μ-compatible ($\mathsf{scomp}_\mu(T)$), if, for all $t \in T$, $\mathsf{scomp}_\mu(t)$.

Some terms have *no* replacement map μ which satisfies $\mathsf{scomp}_\mu(t)$. For instance, there is no replacement map which makes the term $\mathtt{f(f(x,a),y)}$ strongly compatible. When such a $\Sigma(t)$-map exists, it only can be μ_t. The $\Sigma(T)$-map μ which makes a set T strongly μ-compatible must be $\mu = \mu_T$. However, the replacement restrictions for symbols in $\Sigma \backslash \Sigma(T)$ can be freely established without damaging the strong compatibility of T. We assume this fact in the remainder of our exposition with no explicit mention. A weakly compatible set of terms is not, in general, strongly compatible.

Example 4.1 *Both* $t = \mathtt{first(0,x)}$ *and* $t' = \mathtt{first(s(x),y :: z)}$ *terms are strongly compatible w.r.t.* μ_t *and* $\mu_{t'}$*, respectively. Thus, the set* $T = \{t, t'\}$ *is weakly compatible. However, the set* T *is not strongly compatible w.r.t.* $\mu_t \sqcup \mu_{t'}$*.*

Weak compatibility is a modular property (i.e., given $T, S \subseteq \mathcal{T}(\Sigma, V)$, $\mathsf{wcomp}(T \cup S) \Leftrightarrow \mathsf{wcomp}(T) \wedge \mathsf{wcomp}(T)$), but scomp is not. We also have the following:

Proposition 4.2 *Let* $T \subseteq \mathcal{T}(\Sigma, V)$ *and* $\mu_T = \sqcup_{t \in T} \mu_t$*. Then* $\mathsf{scomp}_{\mu_T}(T)$ *iff* $\mathsf{wcomp}(T)$ *and for all* $t, t' \in T$ $\mu_t\downarrow_\Delta = \mu_{t'}\downarrow_\Delta$ *where* $\Delta = \Sigma(t) \cap \Sigma(t')$*.*

This proposition gives us a criterion to know whether a weakly compatible set of terms is strongly compatible. Note that, in Example 4.1, $\mu_t\downarrow_\Delta \neq \mu_{t'}\downarrow_\Delta$, where $\Delta = \Sigma(t) \cap \Sigma(t') = \{\mathtt{first}\}$, because $\mu_t(\mathtt{first}) = \{1\}$ and $\mu_{t'}(\mathtt{first}) = \{1, 2\}$.

Definition 4.3 (Weakly and strongly replacing-independent TRSs) *Let* $\mathcal{R} = (\Sigma, R)$ *be a TRS.* $\mathcal{R}$ *is* weakly (strongly) replacing-independent *if* $L(\mathcal{R})$ *is a weakly (strongly) compatible set of terms.*

If $\mathcal{R}$ is strongly replacing-independent, it is so w.r.t. $\mu_\mathcal{R}^{com}$. This is the *intended* replacement map when dealing with strongly replacing-independent TRSs. The TRS in Example 3.2 is strongly replacing-independent.

Theorem 4.4 ([7]) *Let* $\mathcal{R}$ *be an orthogonal, strongly replacing-independent TRS. Let* t *be a term. Then* $O_\mathcal{R}^{\mu_\mathcal{R}^{com}}(t)$ *is a set of needed redexes.*

This theorem motivates our interest in strongly replacing-independent TRSs: we can *only* perform needed $\mu_\mathcal{R}^{com}$-reductions when we work with this class of TRSs.

To transfom a TRS into a strongly replacing-independent TRS, first we obtain a weakly replacing-independent TRS (transformation W). From this one, we obtain a strongly replacing-independent TRS (transformation S).

4.1 From TRSs to weakly replacing-independent TRSs

If t is not strongly compatible, there is a μ_t-replacing variable occurrence $v.i \in O_V^{\mu_t}(t)$ which is the i-th argument of a function symbol $f \in \Sigma(t)$. By definition of μ_t, there exists a replacing argument of f which is *not* a variable at an occurrence $u.i \in O_\Sigma^{\mu_t}(t)$. We say that $u.i$ is a *disagreeing* occurrence in t. Denote as D_t the set of disagreeing occurrences of subterms in a term t: $D_t = \{u.i \in O_\Sigma^{\mu_t}(t) \mid t = C[f(\tilde{t})]_u \wedge (\exists v.i \in O_V^{\mu_t}(t).\, t = C'[f(\tilde{s})]_v)\}$. To eliminate i from $\mu_t(f)$, we need to *remove* such disagreeing occurrences. Let C_D^t be the occurrences of strongly compatible subterms in t: $C_D^t = \{u \in O(t) \mid t|_u \text{ is strongly compatible}\}$. We remove strongly compatible (proper) subterms containing disagreeing occurrences: U_D^t is the set of occurrences of maximal strongly compatible subterms in a term t which contain the disagreeing occurrences. $U_D^t = (maximal(D_t)\!\downarrow \cap\, minimal(C_D^t))\backslash\{\epsilon\}$. Note that, if t is strongly compatible, then $minimal(C_D^t) = \{\epsilon\}$ and therefore $U_D^t = \emptyset$.

If t is a *lhs* of a rule of a TRS, this elimination will change the matchings in the rewriting process. In order to keep the semantics of the TRS, new function symbols *match* gather the removed subterms to complete the delayed matching.

Definition 4.5 (Transformation W) *Let $\mathcal{R} = (\Sigma, R)$ be a TRS. The TRS $\mathsf{W}(\mathcal{R}) = (\Sigma \uplus M, R')$ is: $R' = \cup_{l\to r\in R}\mathsf{w}(l \to r)$, where*

$\mathsf{w}(l \to r) =$
  `if` $U_D^l = \emptyset$ `then` $\{l \to r\}$ `else`
  `let` $u_D \in U_D^l$
    $X \notin Var(l)$ `be a fresh variable`
    $\{y_1, \ldots, y_p\} = Var(l[\,]_{u_D})$`, and`
    $match \in M$ `be a new` $p+1$`-ary function symbol`
  `in` $\{match(l|_{u_D}, y_1, \ldots, y_p) \to r\} \cup \mathsf{w}(l[X]_{u_D} \to match(X, y_1, \ldots, y_p))$

Example 4.6 *Let us consider the following TRS $\mathcal{R}$:*

$\mathtt{f(c(d(x)),d(c(y)))} \to \mathtt{x}$
$\mathtt{f(c(x),f(y,c(b)))} \to \mathtt{y}$

We apply w to each rule. For $l = \mathtt{f(c(d(x)),d(c(x)))}$, we have $D_l = \{1.1, 2.1\}$, $minimal(C_D^l) = \{1,2\}$ and $U_D^l = \{1,2\}$. We choose $u_D = 1$ and we get:

$\mathtt{f(X,d(c(y)))} \to \mathtt{match(X,y)} \qquad \mathtt{match(c(d(x)),y)} \to \mathtt{x}$

Each lhs is strongly compatible now. For $l' = \mathtt{f(c(x),f(y,c(b)))}$, we have: $D_{l'} = \{1, 2.2.1\}$, $minimal(C_D^{l'}) = \{1,2\}$ and $U_D^{l'} = \{1,2\}$. Choosing $u_D = 1$, we get

$\mathtt{f(X,f(y,c(b)))} \to \mathtt{match'(X,y)} \qquad \mathtt{match'(c(x),y)} \to \mathtt{y}$

Theorem 4.7 *Let $\mathcal{R}$ be a TRS. Then $\mathsf{W}(\mathcal{R})$ is a weakly replacing-independent TRS. If $\mathcal{R}$ is weakly replacing-independent, then $\mathsf{W}(\mathcal{R}) = \mathcal{R}$.*

Theorem 4.8 (Correctness and completeness) *Let $\mathcal{R} = (\Sigma, R)$ be a TRS, and $\mathsf{W}(\mathcal{R}) = (\Sigma \uplus M, R')$. Let $t, s \in \mathcal{T}(\Sigma, V)$. Then, $t \to_\mathcal{R}^* s$ iff $t \to_{\mathsf{W}(\mathcal{R})}^* s$.*

In general, W does *not* preserve orthogonality or confluence of the original TRS but only left-linearity. For instance, $\mathsf{W}(\mathcal{R})$ in Example 4.6 is orthogonal. However, in the second step of Example 4.6, we could choose $u_D = 2$ instead of $u_D = 1$. Thus we obtain the following *non-orthogonal* weakly replacing-independent TRS:

$\mathtt{f(X,d(c(y)))} \to \mathtt{match(X,y)}$ $\quad$ $\mathtt{match(c(d(x)),y)} \to \mathtt{x}$

$\mathtt{f(c(x),X)} \to \mathtt{match'(X,x)}$ $\quad$ $\mathtt{match'(f(y,c(b)),x)} \to \mathtt{y}$

Therefore, we must check for confluence of $\mathsf{W}(\mathcal{R})$ (by using the standard methods) in order to make a practic use of the transformation.

4.2 From weakly to strongly replacing independent TRSs

W solves disagreements *inside* each *lhs*. In order to make a TRS strongly replacing-independent, we need to eliminate disagreements among *lhs*'s of *different* rules.

Example 4.9 *Let us consider the weakly replacing-independent TRS $\mathcal{R}$:*

$\mathtt{first(0,x)} \to \mathtt{[]}$

$\mathtt{first(s(x),y::z)} \to \mathtt{y::first(x,z)}$

The lhs's $l = \mathtt{first(0,x)}$ and $l' = \mathtt{first(s(x),y::z)}$ of $\mathcal{R}$ do not conform a strongly compatible set of terms (see Example 4.1). By Proposition 4.2, this is because of $\mu_l(\mathtt{first}) = \{1\}$ and $\mu_{l'}(\mathtt{first}) = \{1,2\}$. In order to convert $\{l,l'\}$ into a strongly compatible set of terms, we must eliminate the subterm $\mathtt{y::z}$ from l' and to replace it by a fresh variable X. In this way, we obtain $\{\mathtt{first(0,x)}, \mathtt{first(s(x),X)}\}$ which is a strongly compatible set.

We need to keep the semantics of the original TRS unchanged. We add a new symbol $\mathtt{match}$ which is intended to perform the matching of the second argument with the pattern $\mathtt{x::y}$. We obtain the strongly replacing-independent TRS $\mathcal{R}'$:

$\mathtt{first(0,x)} \to \mathtt{[]}$ $\quad$ $\mathtt{match(y::z,x)} \to \mathtt{y::first(x,z)}$

$\mathtt{first(s(x),X)} \to \mathtt{match(X,x)}$

We generalize this procedure. Let us consider a TRS $\mathcal{R} = (\Sigma, R)$, and $l \in L(\mathcal{R})$ in the following definitions:

- $D_l = \{l' \in L(\mathcal{R}) \mid \mu_l{\downarrow_\Delta} \neq \mu_{l'}{\downarrow_\Delta}$ where $\Delta = \Sigma(l) \cap \Sigma(l')\}$
- $\Sigma_D(l) = \{f \in \Sigma(l) \mid \exists l' \in D_l . f \in \Sigma(l') \wedge \mu_l(f) \neq \mu_{l'}(f)\}$
- For all $f \in \Sigma(l)$, $D_l^f = \{l' \in D_l \mid f \in \Sigma(l')\}$
- For all $f \in \Sigma_D(l)$, $\mu_D^l(f) = \cup_{l' \in D_l^f} \mu_l(f) \backslash \mu_{l'}(f)$
- $U_D^l = \{u.i \in O_\Sigma^{\mu_l}(l) \mid l = C[f(\tilde{t})]_u \wedge f \in \Sigma_D(l) \wedge i \in \mu_D^l(f)\}$
- $U_D^R = \{U_D^l \mid l \to r \in R\}$

Roughly speaking, U_D^l is the set of occurrences of subterms in l which should be *removed* from l in order to eliminate all disagreements introduced by l in $L(\mathcal{R})$. They are said *positive* disagreements of l, because by removing subterms from l itself, we can eliminate them.

Definition 4.10 (Transformation S) *Let $\mathcal{R} = (\Sigma, R)$ be a weakly replacing-independent TRS. The TRS $\mathsf{S}(\mathcal{R}) = (\Sigma \cup M, R')$ is: $R' = \mathsf{S}_0(R, U_D^R)$ where $\mathsf{S}_0(\emptyset, X) = \emptyset$, $\mathsf{S}_0(\{l \to r\} \cup R, \{U_D^l\} \cup U_D^R) = \mathsf{s}(l \to r, U_D^l) \cup \mathsf{S}_0(R, U_D^R)$ and*

$\mathsf{s}(l \to r, U_D^l) =$
  if $U_D^l = \emptyset$ then $\{l \to r\}$ else
  let $W = maximal(U_D^l) = \{v_1, \ldots, v_m\}$
    $X_1, \ldots, X_m$ be fresh, distinct variables
    $Y = \{y_1, \ldots, y_p\} = Var(l[\,]_{v_1} \cdots [\,]_{v_m})$, and
    $match \in M$ be a new $m+p$-ary function symbol
  in $\{match(l|_{v_1}, \ldots, l|_{v_m}, y_1, \ldots, y_p) \to r\} \cup$
    $\mathsf{s}(l[X_1]_{v_1} \cdots [X_m]_{v_m} \to match(X_1, \ldots, X_m, y_1, \ldots, y_p), U_D^l \setminus W)$

Example 4.11 *Let us consider the following weakly replacing-independent TRS:*
$\mathtt{g}(\mathtt{a}, \mathtt{x}, \mathtt{f}(\mathtt{a}, \mathtt{b})) \to \mathtt{b}$
$\mathtt{f}(\mathtt{x}, \mathtt{g}(\mathtt{x}, \mathtt{c}, \mathtt{d})) \to \mathtt{a}$
As we can see, $l = \mathtt{g}(\mathtt{a}, \mathtt{x}, \mathtt{f}(\mathtt{a}, \mathtt{b}))$ *has a positive* ($\mathtt{g}$-)*disagreement and* ($\mathtt{f}$-)*disagreement with* $l' = \mathtt{f}(\mathtt{x}, \mathtt{g}(\mathtt{x}, \mathtt{c}, \mathtt{d}))$ *and* l' *has a positive* ($\mathtt{g}$)*-disagreement with* l*. We have:*
$D_l = \{l'\}$, $\Sigma_D(l) = \{\mathtt{f}, \mathtt{g}\}$, $\mu_D^l = \mu_l \backslash \mu_{l'} = \langle\{1,2\},\{1,3\}\rangle \backslash \langle\{2\},\{2,3\}\rangle = \langle\{1\},\{1\}\rangle$, $U_D^l = \{1, 3.1\}$.
Also $D_{l'} = \{l\}$, $\Sigma_D^{l'} = \{\mathtt{f}, \mathtt{g}\}$, $\mu_D^{l'} = \mu_{l'} \backslash \mu_l = \langle\{2\},\{2,3\}\rangle \backslash \langle\{1,2\},\{1,3\}\rangle = \langle\emptyset,\{2\}\rangle$, $U_D^{l'} = \{2.2\}$.
Thus for l*, we get* $W = U_D^l$, $Y = \{\mathtt{x}\}$, $\mathtt{match}$ *is a* 3*-ary symbol, and*
$\mathsf{s}(l, U_D^l) = \{\mathtt{g}(\mathtt{X}, \mathtt{x}, \mathtt{f}(\mathtt{Y}, \mathtt{b})) \to \mathtt{match}(\mathtt{X}, \mathtt{Y}, \mathtt{x}),\ \mathtt{match}(\mathtt{a}, \mathtt{a}, \mathtt{x}) \to \mathtt{b}\}$.
For l', $W = U_D^{l'}$, $Y = \{\mathtt{x}\}$, $\mathtt{match}'$ *is a* 2*-ary symbol, and*
$\mathsf{s}(l', U_D^{l'}) = \{\mathtt{f}(\mathtt{x}, \mathtt{g}(\mathtt{x}, \mathtt{X}, \mathtt{d})) \to \mathtt{match}'(\mathtt{X}, \mathtt{x}),\ \mathtt{match}'(\mathtt{c}, \mathtt{x}) \to \mathtt{a}\}$.
Thus, $\mathsf{S}(\mathcal{R})$ *is:*
$\mathtt{g}(\mathtt{X}, \mathtt{x}, \mathtt{f}(\mathtt{Y}, \mathtt{b})) \to \mathtt{match}(\mathtt{X}, \mathtt{Y}, \mathtt{x})$ $\mathtt{match}(\mathtt{a}, \mathtt{a}, \mathtt{x}) \to \mathtt{b}$
$\mathtt{f}(\mathtt{x}, \mathtt{g}(\mathtt{x}, \mathtt{X}, \mathtt{d})) \to \mathtt{match}'(\mathtt{X}, \mathtt{x})$ $\mathtt{match}'(\mathtt{c}, \mathtt{x}) \to \mathtt{a}$

Theorem 4.12 *If* $\mathcal{R}$ *is a weakly replacing-independent TRS, then* $\mathsf{S}(\mathcal{R})$ *is strongly replacing-independent. If* $\mathcal{R}$ *is strongly replacing-independent, then* $\mathsf{S}(\mathcal{R}) = \mathcal{R}$*.*

S can fail if the input TRS is not weakly replacing-independent.

Example 4.13 *Let us consider* $\mathcal{R}$ *in Example 4.6. Note that* $\Delta = \Sigma(l) \cap \Sigma(l') = \{\mathtt{c}, \mathtt{f}\}$ *and* $\mu_l\!\downarrow_\Delta = \langle\{1\},\{1,2\}\rangle = \mu_{l'}\!\downarrow_\Delta$*. Therefore,* $D_l = D_{l'} = \emptyset$*, hence* $U_D^l = U_D^{l'} = \emptyset$ *and* $\mathsf{S}(\mathcal{R}) = \mathcal{R}$*, but* $\mathcal{R}$ *is not strongly replacing-independent.*

Theorem 4.14 (Correctness and completeness) *Let* $\mathcal{R} = (\Sigma, R)$ *be a weakly replacing-independent TRS, and* $\mathsf{S}(\mathcal{R}) = (\Sigma \uplus M, R')$*. Let* $t, s \in \mathcal{T}(\Sigma, V)$*. Then,* $t \to_{\mathcal{R}}^* s$ *iff* $t \to_{\mathsf{S}(\mathcal{R})}^* s$*.*

S preserves left-linearity. S preserved non-overlaps in Examples 4.9 and 4.11. Unfortunately, in general, S does *not* preserve either orthogonality or confluence.

Example 4.15 *The orthogonal, weakly replacing-independent TRS* $\mathcal{R}$*:*
$\mathtt{h}(\mathtt{x}, \mathtt{b}, \mathtt{c}) \to \mathtt{x}$
$\mathtt{h}(\mathtt{a}, \mathtt{x}, \mathtt{b}) \to \mathtt{x}$
$\mathtt{h}(\mathtt{b}, \mathtt{c}, \mathtt{x}) \to \mathtt{x}$
is transformed into $\mathsf{S}(\mathcal{R})$*:*

$$\begin{array}{ll} \mathtt{h}(\mathtt{x},\mathtt{X},\mathtt{Y}) \to \mathtt{match}(\mathtt{X},\mathtt{Y},\mathtt{x}) & \mathtt{match}(\mathtt{b},\mathtt{c},\mathtt{x}) \to \mathtt{x} \\ \mathtt{h}(\mathtt{X},\mathtt{x},\mathtt{Y}) \to \mathtt{match}'(\mathtt{X},\mathtt{Y},\mathtt{x}) & \mathtt{match}'(\mathtt{a},\mathtt{b},\mathtt{x}) \to \mathtt{x} \\ \mathtt{h}(\mathtt{X},\mathtt{Y},\mathtt{x}) \to \mathtt{match}''(\mathtt{X},\mathtt{Y},\mathtt{x}) & \mathtt{match}''(\mathtt{b},\mathtt{c},\mathtt{x}) \to \mathtt{x} \end{array}$$

which is not orthogonal. It is not confluent, since $\mathtt{h}(\mathtt{a},\mathtt{b},\mathtt{c}) \to^*_{\mathsf{S}(\mathcal{R})} \mathtt{a}$*, but also* $\mathtt{h}(\mathtt{a},\mathtt{b},\mathtt{c}) \to^*_{\mathsf{S}(\mathcal{R})} \mathtt{match}'(\mathtt{a},\mathtt{c},\mathtt{b})$ *and* $\mathtt{h}(\mathtt{a},\mathtt{b},\mathtt{c}) \to^*_{\mathsf{S}(\mathcal{R})} \mathtt{match}''(\mathtt{a},\mathtt{b},\mathtt{c})$.

Therefore, we should also test confluence of $\mathsf{S}(\mathcal{R})$ (using any standard method) to safely use it. W and S could be combined: Let $\mathcal{R}$ be as in Example 4.6. $\mathsf{S}(\mathsf{W}(\mathcal{R}))$ is:

$$\begin{array}{ll} \mathtt{f}(\mathtt{X},\mathtt{d}(\mathtt{Y})) \to \mathtt{match}''(\mathtt{Y},\mathtt{X}) & \mathtt{f}(\mathtt{X},\mathtt{f}(\mathtt{y},\mathtt{c}(\mathtt{Y}))) \to \mathtt{match}'''(\mathtt{Y},\mathtt{X},\mathtt{y}) \\ \mathtt{match}(\mathtt{c}(\mathtt{X}),\mathtt{y}) \to \mathtt{match}^{\mathtt{iv}}(\mathtt{X},\mathtt{y}) & \mathtt{match}'(\mathtt{c}(\mathtt{x}),\mathtt{y}) \to \mathtt{y} \\ \mathtt{match}''(\mathtt{c}(\mathtt{y}),\mathtt{X}) \to \mathtt{match}(\mathtt{X},\mathtt{y}) & \mathtt{match}'''(\mathtt{b},\mathtt{X},\mathtt{y}) \to \mathtt{match}'(\mathtt{X},\mathtt{y}) \\ \mathtt{match}^{\mathtt{iv}}(\mathtt{d}(\mathtt{x}),\mathtt{y}) \to \mathtt{x} & \end{array}$$

which is an orthogonal TRS. Therefore, the previous results ensure that $\mathsf{S}(\mathsf{W}(\mathcal{R}))$ is equivalent to $\mathcal{R}$, but we can perform optimal reductions by using the canonical replacement map because it is a strongly replacing-independent TRS.

5 Program transformation for complete evaluations

Regarding normalization of terms, if we restrict ourselves to values, *csr* is powerful enough to compute them. Given a set $\mathcal{B} \subseteq \mathcal{C}$, the replacement map $\mu^{\mathcal{B}}_{\mathcal{R}}$ is $\mu^{\mathcal{B}}_{\mathcal{R}} = \mu^{com}_{\mathcal{R}} \sqcup \mu_{\mathcal{B}}$, where $\mu_{\mathcal{B}}(c) = \mathbb{N}^+_{ar(c)}$ for all $c \in \mathcal{B}$ and $\mu_{\mathcal{B}}(f) = \emptyset$ if $f \notin \mathcal{B}$.

Theorem 5.1 ([5]) *Let* $\mathcal{R} = (\Sigma, R) = (\mathcal{C} \uplus \mathcal{F}, R)$ *be a left-linear TRS,* $\mathcal{B} \subseteq \mathcal{C}$ *and* μ *be a* Σ*-map such that* $\mu^{\mathcal{B}}_{\mathcal{R}} \sqsubseteq \mu$*. Let* $t \in \mathcal{T}(\Sigma, V)$*, and* $\delta \in \mathcal{T}(\mathcal{B}, V)$*. Then* $t \to^* \delta$ *iff* $t \hookrightarrow^* \delta$*.*

By taking $\mu^{com}_{\mathcal{R}}$ as given in Example 3.2, and $\mathcal{B} = \{\mathtt{0}, \mathtt{s}\}$, then $\mu^{\mathcal{B}}_{\mathcal{R}}(\mathtt{s}) = \{1\}$ (recall that $\mu^{com}_{\mathcal{R}}(\mathtt{s}) = \emptyset$). We continue the head-evaluation of Example 3.5,

$$\mathtt{s}(\mathtt{s}(\underline{\mathtt{dbl}(\mathtt{0})})) \hookrightarrow \mathtt{s}(\mathtt{s}(\mathtt{0}))$$

to get the complete evaluation. The non-terminating or useless computations which could start in the steps $(*)$ and $(**)$ in Example 3.5 are still avoided, because $\mu^{\mathcal{B}}_{\mathcal{R}}(::) = \emptyset$. Moreover, by using the results in [12] we can prove that $\mathcal{R}$ is $\mu^{\mathcal{B}}_{\mathcal{R}}$-terminating. Unfortunately, we cannot ensure optimality of the $\mu^{\mathcal{B}}_{\mathcal{R}}$-derivation, since $\mu^{com}_{\mathcal{R}} \sqsubset \mu^{\mathcal{B}}_{\mathcal{R}}$ and Theorem 4.4 does not apply. More problematic situations arise.

Example 5.2 *Let us consider* $t = \mathtt{indx}(\mathtt{s}(\mathtt{0}) :: [], \mathtt{dbls}(\mathtt{from}(\mathtt{0})))$*. By using* $\mathcal{R}$ *in Example 3.2, we can head-evaluate* t *under* $\mu^{com}_{\mathcal{R}}$*:*

$$\begin{array}{l} \underline{\mathtt{indx}(\mathtt{s}(\mathtt{0}) :: [], \mathtt{dbls}(\mathtt{from}(\mathtt{0})))} \\ \quad \hookrightarrow \mathtt{sel}(\mathtt{s}(\mathtt{0}), \mathtt{dbls}(\mathtt{from}(\mathtt{0}))) :: \mathtt{indx}([], \mathtt{dbls}(\mathtt{from}(\mathtt{0}))) \end{array}$$

If we want to compute the value of t*, we take* $\mathcal{B} = \{\mathtt{0}, \mathtt{s}, [], ::\}$*. Hence,* $\mu^{\mathcal{B}}_{\mathcal{R}}(\mathtt{s}) = \{1\}$ *and* $\mu^{\mathcal{B}}_{\mathcal{R}}(::) = \{1, 2\}$*. This allows us to continue until the end:*

$$
\begin{array}{ll}
\mathtt{sel}(\mathtt{s}(\mathtt{0}),\mathtt{dbls}(\mathtt{from}(\mathtt{0}))) :: \underline{\mathtt{indx}([\,],\mathtt{dbls}(\mathtt{from}(\mathtt{0})))} & \\
\quad\hookrightarrow \mathtt{sel}(\mathtt{s}(\mathtt{0}),\mathtt{dbls}(\underline{\mathtt{from}(\mathtt{0})})) :: [\,] & \\
\quad\hookrightarrow \mathtt{sel}(\mathtt{s}(\mathtt{0}),\underline{\mathtt{dbls}(\mathtt{0} :: \mathtt{from}(\mathtt{s}(\mathtt{0})))}) :: [\,] & (*) \\
\quad\hookrightarrow \underline{\mathtt{sel}(\mathtt{s}(\mathtt{0}),\mathtt{dbl}(\mathtt{0}) :: \mathtt{dbls}(\mathtt{from}(\mathtt{s}(\mathtt{0}))))} :: [\,] & (**) \\
\quad\hookrightarrow \mathtt{sel}(\mathtt{0},\mathtt{dbls}(\underline{\mathtt{from}(\mathtt{s}(\mathtt{0}))})) :: [\,] & \\
\quad\hookrightarrow \mathtt{sel}(\mathtt{0},\underline{\mathtt{dbls}(\mathtt{s}(\mathtt{0}) :: \mathtt{from}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))}) :: [\,] & (*) \\
\quad\hookrightarrow \underline{\mathtt{sel}(\mathtt{0},\mathtt{dbl}(\mathtt{s}(\mathtt{0})) :: \mathtt{dbls}(\mathtt{from}(\mathtt{s}(\mathtt{s}(\mathtt{0})))))} :: [\,] & \\
\quad\hookrightarrow \underline{\mathtt{dbl}(\mathtt{s}(\mathtt{0}))} :: [\,] & \\
\quad\hookrightarrow \mathtt{s}(\mathtt{s}(\underline{\mathtt{dbl}(\mathtt{0})})) :: [\,] & \\
\quad\hookrightarrow \mathtt{s}(\mathtt{s}(\mathtt{0})) :: [\,] &
\end{array}
$$

However, $\mathcal{R}$ is not $\mu^{\mathcal{B}}_{\mathcal{R}}$-terminating now: Since $\mu^{\mathcal{B}}_{\mathcal{R}}(::) = \{1, 2\}$, the repeated application of a `from`*-rule in the $(*)$ terms goes into a loop. Optimality of derivations is also compromised: we could perform an unbound number of (useless) reductions for the redex* `double(0)` *in $(**)$ which are not needed.*

The unpleasant situation sketched in Example 5.2 is solved by giving a new program transformation.

5.1 The transformation V

If we deal with typed TRSs, Theorem 5.1 is very easy to use, since, given a term $t = f(\tilde{t})$, we are able to establish the maximal set of constructors $\mathcal{B} \subseteq \mathcal{C}$ which should be considered. In this section we assume the existence of some valid type system [9]. Basic types are denoted as nat, bool, etc. Symbols $f \in \Sigma$ are typed by: $f : \tau_1 \times \ldots \times \tau_k \to \tau$. The *type* of f is $type(f) = \tau$. Types of arguments of f are gathered in $typearg(f) = \{\tau_1, \ldots, \tau_k\}$. Variables $x \in V$ also have a type, $type(x)$. We also assume that all terms are well typed everywhere. The type of a term t is the type of its outermost symbol.

We need to isolate the replacement restrictions which are needed to perform the head-evaluation of an operation rooted term $t = f(\tilde{t})$ for some $f \in \mathcal{F}$ from the restrictions which are needed to allow them within a constructor context $C[\,] \in \mathcal{T}(\mathcal{B}, V)$ for some $\mathcal{B} \subseteq \mathcal{C}$. *Below* the outermost defined symbols, reductions should be performed only up to head-evaluation. To deal with this task, we introduce the following notations:

- $outer_{\mathcal{F}}(t) = \{root(t|_u) \in \mathcal{F} \mid \forall\, v < u.\ root(t|_v) \notin \mathcal{F}\}$ is the set of outermost defined symbols in t.
- $Var_{\mathcal{B}}(t) = \{x \in Var(t) \mid \exists u \in O(t).\ t|_u = x \wedge \forall\, v < u.\ root(t|_v) \in \mathcal{B}\}$ is the set of $\mathcal{B}$-*variables*, i.e., variables of t having a maximal proper prefix which only points to symbols in $\mathcal{B}$. We also define $\mathcal{B}_\tau = \{c \in \mathcal{B} \mid type(c) = \tau\}$.
- The set of possible types for outermost defined functions arising by instantiation of a variable x is $Type_{\mathcal{B}}(type(x))$ where
  $Type_{\mathcal{B}}(\tau) = \{\tau\} \cup \bigcup_{\substack{c \in \mathcal{B}_\tau \\ \tau' \in typearg(c)}} Type_{\mathcal{B}}(\tau')$
- $Vouter_{\mathcal{B}}(t) = \bigcup_{x \in Var_{\mathcal{B}}(t)} \{f \in \mathcal{F} \mid type(f) \in Type_{\mathcal{B}}(type(x))\}$ are the defined symbols which can become outermost by instantiating $\mathcal{B}$-variables of t.

Example 5.3 *Let us consider the TRS in Example 3.2. We calculate* $Vouter_{\mathcal{B}}(x)$*. Let* $type(x) = \mathsf{nat}$ *and* $\mathcal{B} = \{\mathsf{0}, \mathsf{s}\}$*. Since* $Type_{\mathcal{B}}(type(x)) = \{\mathsf{nat}\} \cup Type_{\mathcal{B}}(\mathsf{nat}) = \{\mathsf{nat}\}$*, we have* $Vouter_{\mathcal{B}}(x) = \{\mathtt{dbl}, \mathtt{sel}\}$*.*

Given a TRS $\mathcal{R} = (\Sigma, R) = (\mathcal{F} \uplus \mathcal{C}, R)$, $\mathcal{B} \subseteq \mathcal{C}$ and $f \in \mathcal{F}$, $outrhs_{\mathcal{R}}(f) \subseteq \mathcal{F}$ contains the outermost defined symbols in rhs's of the f-rules: $outrhs_{\mathcal{R}}(f) = \cup_{f(\tilde{l}) \to r \in R}\, outer_{\mathcal{F}}(r)$. Now $Vrhs(f) \subseteq \mathcal{F}$ is the set of outermost defined symbols which can appear by instantiation of $\mathcal{B}$-variables in rhs's of the f-rules: $Vrhs(f) = \cup_{f(\tilde{l}) \to r \in R}\, Vouter_{\mathcal{B}}(r)$. If $\mathcal{F}_{\mathcal{R}} : \mathcal{P}(\mathcal{F}) \to \mathcal{P}(\mathcal{F})$ is:

$$\mathcal{F}_{\mathcal{R}}(F) = F \;\cup\; \bigcup_{g \in F} \mathcal{F}_{\mathcal{R}}(outrhs_{\mathcal{R}}(g)) \cup \bigcup_{g \in F} \mathcal{F}_{\mathcal{R}}(\mathit{Vrhs}(g)).$$

the set $\mathcal{F}^{f}_{\mathcal{R}} = \mathcal{F}_{\mathcal{R}}(\{f\})$ contains the defined symbols which can turn outermost when a f-rooted (well typed) term $f(\tilde{t})$ is rewritten.

Example 5.4 *Given* $\mathcal{R}$ *in Example 3.2, let us obtain* $\mathcal{F}^{\mathtt{sel}}_{\mathcal{R}}$ *and* $\mathcal{F}^{\mathtt{indx}}_{\mathcal{R}}$*.*

- *We have that* $outrhs_{\mathcal{R}}(\mathtt{sel}) = \{\mathtt{sel}\}$ *and* $Vrhs(\mathtt{sel}) = \{\mathtt{sel}, \mathtt{dbl}\}$*. Thus,* $\mathcal{F}^{\mathtt{sel}}_{\mathcal{R}} = \mathcal{F}_{\mathcal{R}}(\{\mathtt{sel}\}) = \{\mathtt{sel}\} \cup \mathcal{F}_{\mathcal{R}}(\{\mathtt{sel}\}) \cup \mathcal{F}_{\mathcal{R}}(\{\mathtt{dbl}\}) = \{\mathtt{sel}, \mathtt{dbl}\}$*.*
- $outrhs_{\mathcal{R}}(\mathtt{indx}) = \{\mathtt{sel}, \mathtt{indx}\}$ *and* $Vrhs(\mathtt{indx}) = \emptyset$*. Thus,* $\mathcal{F}^{\mathtt{indx}}_{\mathcal{R}} = \mathcal{F}_{\mathcal{R}}(\{\mathtt{indx}\}) = \{\mathtt{indx}\} \;\cup\; \mathcal{F}_{\mathcal{R}}(\{\mathtt{sel}\}) = \{\mathtt{indx}, \mathtt{sel}, \mathtt{dbl}\}$*.*

The formal treatment of the transformation which we are going to introduce becomes simpler, if we do not look for the least increment of the number of rules of the TRS. For instance, by introducing new rules $f'(\tilde{l}) \to r'$ to deal with *each* defined symbol f' which stands for the outermost version of a symbol $f \in \mathcal{F}$, the definition of the transformation would be easier, but we also get undesirable consequences. For example, in the evaluation of $t = \mathtt{indx}(\mathtt{s}(\mathtt{0}) :: [\,], \mathtt{dbls}(\mathtt{from}(\mathtt{0})))$ as given in Example 5.2, the symbol $\mathtt{from}$ does *not* become outermost. Thus, we do not need the rule:

$\mathtt{from}'(\mathtt{x}) \to \bar{\mathtt{T}}(\mathtt{x}) ::' \mathtt{from}'(\mathtt{s}(\mathtt{x}))$

(the meaning of $\bar{\mathtt{T}}$ is explained below) which introduces *non-termination*, because we need that $::'$ (the 'outermost' version of $::$) be replacing in both its first and second arguments in order to obtain completeness in evaluations. However, this rule does *not* add computational power to the transformation, because $\mathtt{from}'$ will never arise in the evaluation. For this reason, we perform a more accurated analysis of the required additional rules. We define now the program transformation. First, we give the new signature.

Definition 5.5 *Given a TRS* $\mathcal{R} = (\Sigma, R) = (\mathcal{C} \uplus \mathcal{F}, R)$*,* $f \in \mathcal{F}$ *and* $\mathcal{B} \subseteq \mathcal{C}$*,* $\Sigma^{f}_{\mathcal{B}} = \Sigma \uplus \mathcal{F}' \uplus \mathcal{B}'$*, where:* $c' \in \mathcal{B}' \Leftrightarrow c \in \mathcal{B} \wedge ar(c') = ar(c)$ *and* $g' \in \mathcal{F}' \Leftrightarrow g \in \mathcal{F}^{f}_{\mathcal{R}} \wedge ar(g') = ar(g)$*.* c' *represents the constructors above the outermost applications of defined symbols.* g' *represents outermost applications of* $g \in \mathcal{F}^{f}_{\mathcal{R}}$*.*

The (partial) function $\tau^{f}_{\mathcal{B}}$ transforms terms in $\mathcal{T}(\Sigma, V)$ into terms in $\mathcal{T}(\Sigma^{f}_{\mathcal{B}}, V)$: $\tau^{f}_{\mathcal{B}}(x) = x$, $\tau^{f}_{\mathcal{B}}(g(\tilde{t})) = g'(\tilde{t})$ if $g \in \mathcal{F}^{f}_{\mathcal{R}}$. Finally, $\tau^{f}_{\mathcal{B}}(c(\tilde{t})) = c'(\tau^{f}_{\mathcal{B}}(\tilde{t}))$ if $c \in \mathcal{B}$. $\tau^{f}_{\mathcal{B}}$ is only defined on a subset $ev^{f}_{\mathcal{B}}(\Sigma, V)$ of terms which is given as follows:

$V \subseteq ev^f_{\mathcal{B}}(\Sigma, V)$, $g(\tilde{t}) \in ev^f_{\mathcal{B}}(\Sigma, V)$ if $g \in \mathcal{F}^f_{\mathcal{R}}$ and $c(\tilde{t}) \in ev^f_{\mathcal{B}}(\Sigma, V)$ if $c \in \mathcal{B}$ and $t_1, \ldots, t_{ar(c)} \in ev^f_{\mathcal{B}}(\Sigma, V)$. If we do not require $V \subseteq ev^f_{\mathcal{B}}(\Sigma, V)$, then we are defining the set $gev^f_{\mathcal{B}}(\Sigma, V)$. Note that, if t can be evaluated to a value $\delta \in \mathcal{T}(\mathcal{B}, V)$ ($\delta \in \mathcal{T}(\mathcal{B})$), then $t \in ev^f_{\mathcal{B}}(\Sigma, V)$ ($t \in gev^f_{\mathcal{B}}(\Sigma, V)$).

The transformation introduces rules to deal with our differentiation of symbols. Rules for outermost defined symbols $f' \in \mathcal{F}'$ introduce new outermost constructors $c' \in \mathcal{B}'$ and outermost defined symbols $f' \in \mathcal{F}'$. Rules for inner defined symbols $f \in \mathcal{F}$ are exactly the original ones of the TRS.

Definition 5.6 (Transformation V) *Let $\mathcal{R} = (\Sigma, R) = (\mathcal{C} \uplus \mathcal{F}, R)$ be a TRS, $f \in \mathcal{F}$ and $\mathcal{B} \subseteq \mathcal{C}$. Then $\mathsf{V}^f_{\mathcal{B}}(\mathcal{R}) = (\Sigma^f_{\mathcal{B}} \uplus \{\bar{\mathtt{T}}\}, R \cup S_{\mathcal{B}} \cup T_{\mathcal{B}})$, where:*

- *$\bar{\mathtt{T}}$ with $ar(\bar{\mathtt{T}}) = 1$ is a fresh, defined symbol whose rules perform the renaming of external constructors $c \in \mathcal{B}$ to constructors $c' \in \mathcal{B}'$ and outermost application of $g \in \mathcal{F}^f_{\mathcal{B}}$ to outermost applications of the corresponding $g' \in \mathcal{F}'$.*
- *$S_{\mathcal{B}} = \{g'(\tilde{l}) \to \kappa^f_{\mathcal{B}}(r) \mid g(\tilde{l}) \to r \in R \wedge g \in \mathcal{F}^f_{\mathcal{R}} \wedge r \in ev^f_{\mathcal{B}}(\Sigma, V)\}$, where $\kappa^f_{\mathcal{B}}(x) = \bar{\mathtt{T}}(x)$, $\kappa^f_{\mathcal{B}}(g(\tilde{t})) = g'(\tilde{t})$ if $g \in \mathcal{F}^f_{\mathcal{R}}$, and $\kappa^f_{\mathcal{B}}(c(\tilde{t})) = c'(\kappa^f_{\mathcal{B}}(\tilde{t}))$ if $c \in \mathcal{B}$.*
  $T_{\mathcal{B}} = \{\bar{\mathtt{T}}(c(x_1, \ldots, x_k)) \to c'(\bar{\mathtt{T}}(x_1), \ldots, \bar{\mathtt{T}}(x_k)) \mid c \in \mathcal{B}\} \cup$
  $\{\bar{\mathtt{T}}(g(x_1, \ldots, x_k)) \to g'(x_1, \ldots, x_k) \mid g \in \mathcal{F}^f_{\mathcal{R}}\}$

To simplify the notation, we usually write V instead of $\mathsf{V}^f_{\mathcal{B}}$, when f and $\mathcal{B}$ are clear from the context. The following examples illustrate our transformation.

Example 5.7 *If we take $f = \mathtt{sel}$ and $\mathcal{B} = \{\mathtt{0}, \mathtt{s}\} \subseteq \{\mathtt{0}, \mathtt{s}, [], ::\} = \mathcal{C}$, the* additional *rules for the evaluation of $t = \mathtt{sel}(Num, ListNum)$ are:*

$\mathtt{dbl}'(\mathtt{0}) \to \mathtt{0}'$ | $\mathtt{sel}'(\mathtt{0}, \mathtt{x} :: \mathtt{y}) \to \mathtt{x}$
$\mathtt{dbl}'(\mathtt{s}(\mathtt{x})) \to \mathtt{s}'(\mathtt{s}'(\mathtt{dbl}'(\mathtt{x})))$ | $\mathtt{sel}'(\mathtt{s}(\mathtt{x}), \mathtt{y} :: \mathtt{z}) \to \mathtt{sel}'(\mathtt{x}, \mathtt{z})$

$\bar{\mathtt{T}}(\mathtt{0}) \to \mathtt{0}'$ | $\bar{\mathtt{T}}(\mathtt{dbl}(\mathtt{x})) \to \mathtt{dbl}'(\mathtt{x})$
$\bar{\mathtt{T}}(\mathtt{s}(\mathtt{x})) \to \mathtt{s}'(\bar{\mathtt{T}}(\mathtt{x}))$ | $\bar{\mathtt{T}}(\mathtt{sel}(\mathtt{x}, \mathtt{y})) \to \mathtt{sel}'(\mathtt{x}, \mathtt{y})$

Example 5.8 *If we take $f = \mathtt{indx}$ and $\mathcal{B} = \{\mathtt{0}, \mathtt{s}, [], ::\} = \mathcal{C}$, the* additional *rules for the evaluation of $t = \mathtt{indx}(ListNum, ListNum)$ are:*

$\mathtt{dbl}'(\mathtt{0}) \to \mathtt{0}'$ | $\mathtt{indx}'([], \mathtt{x}) \to []'$
$\mathtt{dbl}'(\mathtt{s}(\mathtt{x})) \to \mathtt{s}'(\mathtt{s}'(\mathtt{dbl}'(\mathtt{x})))$ | $\mathtt{indx}'(\mathtt{x} :: \mathtt{y}, \mathtt{z}) \to \mathtt{sel}'(\mathtt{x}, \mathtt{z}) ::' \mathtt{indx}'(\mathtt{y}, \mathtt{z})$
$\mathtt{sel}'(\mathtt{0}, \mathtt{x} :: \mathtt{y}) \to \mathtt{x}$
$\mathtt{sel}'(\mathtt{s}(\mathtt{x}), \mathtt{y} :: \mathtt{z}) \to \mathtt{sel}'(\mathtt{x}, \mathtt{z})$

$\bar{\mathtt{T}}(\mathtt{0}) \to \mathtt{0}'$ | $\bar{\mathtt{T}}(\mathtt{dbl}(\mathtt{x})) \to \mathtt{dbl}'(\mathtt{x})$
$\bar{\mathtt{T}}(\mathtt{s}(\mathtt{x})) \to \mathtt{s}'(\bar{\mathtt{T}}(\mathtt{x}))$ | $\bar{\mathtt{T}}(\mathtt{indx}(\mathtt{x}, \mathtt{y})) \to \mathtt{indx}'(\mathtt{x}, \mathtt{y})$
$\bar{\mathtt{T}}([]) \to []'$ | $\bar{\mathtt{T}}(\mathtt{sel}(\mathtt{x}, \mathtt{y})) \to \mathtt{sel}'(\mathtt{x}, \mathtt{y})$
$\bar{\mathtt{T}}(\mathtt{x} :: \mathtt{y}) \to \bar{\mathtt{T}}(\mathtt{x}) ::' \bar{\mathtt{T}}(\mathtt{y})$

Given a Σ-map μ, we define the replacement map μ' which is used in the transformed TRS $\mathsf{V}(\mathcal{R})$ to obtain *complete* evaluations (leading to values) provided that μ is complete for head-evaluations. To obtain it we use a function.

Definition 5.9 *Let $\mathcal{R} = (\Sigma, R) = (\mathcal{C} \uplus \mathcal{F}, R)$ be a TRS, $f \in \mathcal{F}$, and $\mathcal{B} \subseteq \mathcal{C}$. Let μ be a Σ-map. The function $rmap^f_{\mathcal{B}} : M_\Sigma \to M_{\Sigma^f_{\mathcal{B}} \uplus \{\bar{\mathtt{T}}\}}$ is given by: $rmap^f_{\mathcal{B}}(\mu) = \mu \sqcup \mu' \sqcup \mu_{\mathcal{B}'}$, where $\mu'(g') = \mu(g)$ for all $g \in \mathcal{F}^f_{\mathcal{R}}$ and $\mu'(g) = \emptyset$ for any other symbol. $\mu_{\mathcal{B}'}(c') = \mathbb{N}^+_{ar(c')}$ for all $c' \in \mathcal{B}'$, and $\mu_{\mathcal{B}'}(g) = \emptyset$ for any other symbol.*

Note that, if $\mu' = rmap^f_{\mathcal{B}}(\mu)$, then $\mu'(\bar{\mathtt{T}}) = \emptyset$.

Example 5.10 *Let us consider the TRS $\mathcal{R}$ in Example 3.2. If $f = \mathtt{indx}$, $\mathcal{B} = \{\mathtt{0}, \mathtt{s}, [\,], ::\}$, and $\mu' = rmap^f_{\mathcal{B}}(\mu^{com}_{\mathcal{R}})$, then $\mu'(\mathtt{s}) = \mu'(::) = \mu'(from) = \mu'(\bar{\mathtt{T}}) = \emptyset$, $\mu'(\mathtt{s'}) = \mu'(\mathtt{dbl}) = \mu'(\mathtt{dbl'}) = \mu'(\mathtt{dbls}) = \mu'(\mathtt{indx}) = \mu'(\mathtt{indx'}) = \{1\}$, $\mu'(::') = \mu'(\mathtt{sel}) = \mu'(\mathtt{sel'}) = \{1, 2\}$.*

We have that μ-replacing redex occurrences of t are exactly the same than $rmap^f_{\mathcal{B}}(\mu)$-replacing redex occurrences (w.r.t. $\mathsf{V}(\mathcal{R})$) of $\tau^f_{\mathcal{B}}(t)$. Thus, $rmap^f_{\mathcal{B}}(\mu)$ computes below an outermost defined symbol exactly in the same degree than μ does. Thus, we avoid useless computations.

Proposition 5.11 *Let $\mathcal{R} = (\Sigma, R) = (\mathcal{F} \uplus \mathcal{C}, R)$ be a TRS, $f \in \mathcal{F}$ and $\mathcal{B} \subseteq \mathcal{C}$. Let $t = C[t_1, \ldots, t_n] \in \mathcal{T}(\Sigma, V)$, $u_1, \ldots, u_n \in O(t)$ be such that $C[x_1, \ldots, x_n] \in \mathcal{T}(\mathcal{B}, V)$ and $root(t_i) \in \mathcal{F}^f_{\mathcal{R}}$, $t_i = t|_{u_i}$ for $1 \leq i \leq n$. Let μ be a Σ-map and $\mu' = rmap^f_{\mathcal{B}}(\mu)$. Then, $O^{\mu'}_{\mathsf{V}(\mathcal{R})}(\tau^f_{\mathcal{B}}(t)) = \cup_{1 \leq i \leq n} u_i.O^{\mu}_{\mathcal{R}}(t_i)$.*

If $\mathcal{R}$ is an orthogonal strongly replacing-independent TRS, computations in $\mathsf{V}(\mathcal{R})$ using $rmap^f_{\mathcal{B}}(\mu^{com}_{\mathcal{R}})$ are needed as long as computations in $\mathcal{R}$ using $\mu^{com}_{\mathcal{R}}$ are.

To utilize $\mathsf{V}(\mathcal{R})$ in evaluating a term $f(\tilde{t})$, we can directly evaluate $\bar{\mathtt{T}}(f(\tilde{t}))$. V preserves confluence of $\mathcal{R}$ (but not orthogonality) when evaluating a term $t = f(\tilde{t})$ for which V was defined. Thus, we can apply the rules careless: no undesired reduction sequence will arise.

Example 5.12 *Let us consider the TRS $\mathcal{R}$ in Example 3.2. Let $f = \mathtt{indx}$ and $\mathcal{B} = \{\mathtt{0}, \mathtt{s}, [\,], ::\}$. Let $\mu' = rmap^f_{\mathcal{B}}(\mu^{com}_{\mathcal{R}})$ (see Example 5.10). By using $\mathsf{V}(\mathcal{R})$ (see Example 5.8) and μ', we have:*

$$
\begin{array}{lr}
\bar{\mathtt{T}}(\mathtt{indx}(\mathtt{s}(\mathtt{0}) :: [\,], \mathtt{dbls}(\mathtt{from}(\mathtt{0})))) & \\
\quad \hookrightarrow \mathtt{indx'}(\mathtt{s}(\mathtt{0}) :: [\,], \mathtt{dbls}(\mathtt{from}(\mathtt{0}))) & \\
\quad \hookrightarrow \mathtt{sel'}(\mathtt{s}(\mathtt{0}), \mathtt{dbls}(\mathtt{from}(\mathtt{0}))) ::' \mathtt{indx'}([\,], \mathtt{dbls}(\mathtt{from}(\mathtt{0}))) & \\
\quad \hookrightarrow \mathtt{sel'}(\mathtt{s}(\mathtt{0}), \mathtt{dbls}(\mathtt{from}(\mathtt{0}))) ::' [\,]' & \\
\quad \hookrightarrow \mathtt{sel'}(\mathtt{s}(\mathtt{0}), \mathtt{dbls}(\mathtt{0} :: \mathtt{from}(\mathtt{s}(\mathtt{0})))) ::' [\,]' & (*) \\
\quad \hookrightarrow \mathtt{sel'}(\mathtt{s}(\mathtt{0}), \mathtt{dbl}(\mathtt{0}) :: \mathtt{dbls}(\mathtt{from}(\mathtt{s}(\mathtt{0})))) ::' [\,]' & (**) \\
\quad \hookrightarrow \mathtt{sel'}(\mathtt{0}, \mathtt{dbls}(\mathtt{from}(\mathtt{s}(\mathtt{0})))) ::' [\,]' & \\
\quad \hookrightarrow \mathtt{sel'}(\mathtt{0}, \mathtt{dbls}(\mathtt{s}(\mathtt{0}) :: \mathtt{from}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))) ::' [\,]' & (*) \\
\quad \hookrightarrow \mathtt{sel'}(\mathtt{0}, \mathtt{dbl}(\mathtt{s}(\mathtt{0})) :: \mathtt{dbls}(\mathtt{from}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))) ::' [\,]' & \\
\quad \hookrightarrow \bar{\mathtt{T}}(\mathtt{dbl}(\mathtt{s}(\mathtt{0}))) ::' [\,]' & \\
\quad \hookrightarrow \mathtt{dbl'}(\mathtt{s}(\mathtt{0})) ::' [\,]' & \\
\quad \hookrightarrow \mathtt{s'}(\mathtt{s'}(\mathtt{dbl'}(\mathtt{0}))) ::' [\,]' & \\
\quad \hookrightarrow \mathtt{s'}(\mathtt{s'}(\mathtt{0'})) ::' [\,]' &
\end{array}
$$

Note that, from ($*$) and ($**$), no harmful or wasteful computations can be initiated. The *cs*-derivation does really behave as being essentially guided by $\mu_{\mathcal{R}}^{com}$ (Proposition 5.11). Thus, neededness of computations is preserved.

Theorem 5.13 (Correctness and completeness) *Let $\mathcal{R} = (\Sigma, R) = (\mathcal{C} \uplus \mathcal{F}, R)$ be a left-linear TRS. Let $f \in \mathcal{F}$, $\mathcal{B} \subseteq \mathcal{C}$, $t = f(\tilde{t}) \in \mathcal{T}(\Sigma, V)$, and $\delta \in \mathcal{T}(\mathcal{B})$. Let $\mathcal{R}' = \mathsf{V}(\mathcal{R})$. Let μ be a Σ-map such that $\mu_{\mathcal{R}}^{com} \sqsubseteq \mu$, and $\mu' = rmap_{\mathcal{B}}^{f}(\mu)$. Then $t \to_{\mathcal{R}}^{*} \delta$ iff $\tau_{\mathcal{B}}^{f}(t) \hookrightarrow_{\mathcal{R}'(\mu')}^{*} \tau_{\mathcal{B}}^{f}(\delta)$.*

6 Conclusions and future work

We have introduced the class of *weakly replacing-independent* TRSs, a superclass of strongly replacing-independent TRSs which covers many usual functional programs. The transformation W yields a weakly replacing-independent TRS from an arbitrary TRS. S produces a strongly replacing-independent TRS from a weakly replacing-independent TRS. This allows us to eventually obtain benefit from the nice properties of strongly replacing-independent TRSs. In particular, neededness of computations when using orthogonal strongly replacing-independent TRSs. The transformations have been proven both correct and complete, but orthogonality is not preserved, in general.

We have also defined a program transformation V which allows us to raise the good properties of *csr* concerning head-evaluations to complete evaluations. In [3] we found a transformation to achieve lazy evaluations in a transformed TRS by using *eager* rewriting. This transformation is more complex than our V transformation. In the worst case for that transformation, it adds $3+n\ r+n\ 2^l + 2\ s$ rules to the original TRS $\mathcal{R} = (\Sigma, R)$ (see [3]), where n is the number of rules in R, s is the number of symbols in Σ, r is the maximal number of nonvariable nonreplacing occurrences in the *rhs*'s of the TRS, and l is the maximal number of nonvariable nonreplacing occurrences[3] in the *lhs*'s in R. The transformation V only introduces $n + s$ new rules in the worst case. This is because we obtain advantages in compiling to *csr* instead of rewriting as in [3]. However, a more detailed comparison of time complexity (to obtain the transformed TRS) would be required for a fair comparison.

Strongly replacing-independent TRSs have some links with the *path sequential* TRSs of Sekar et al. in [10]. The strict prefix $\mathit{sprefix}_t(u)$ of an occurrence u in a term t is the chain of symbols laying in occurrences $v < u$. Roughly speaking, a constructor-based TRS $\mathcal{R}$ is path sequential if, for all $l, l' \in L(\mathcal{R})$, and $u \in O(l) \cap O(l')$ such that $\mathit{sprefix}_l(u) = \mathit{sprefix}_{l'}(u)$, $u \in O_\Sigma(l)$ iff $u \in O_\Sigma(l')$. Path sequential TRSs are a particular class of strongly sequential TRSs [2, 10]. It is not difficult to see that constructor-based strongly replacing-independent TRSs are path sequential. However, computations with strongly replacing-independent TRSs are simpler. In [10], a method to transform a constructor-based strongly sequential TRS into a path sequential TRS is given. Our transformations W and

[3] There is an easy correspondence between the predicate Λ which is used in [3] to specify replacement restrictions and our replacement map μ. Nevertheless, *lazy rewriting* which is defined from Λ is quite different from *csr*.

S are simpler and they apply to any TRS. An interesting topic is using W and S to connect strongly sequential and strongly replacing-independent TRSs. Since our transformations do not preserve orthogonality, we are interested in giving conditions to ensure its preservation, or even to modify them to obtain this behavior. Nevertheless, after [11], orthogonality is not required to have strong sequentiality, hence normalizing strategies. Thus, the topic is also an interesting subject for further research, even for the current version of the transformations.

References

1. N. Dershowitz and J.P. Jouannaud. Rewrite Systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, pages 243-320. Elsevier, Amsterdam and The MIT Press, Cambridge, MA, 1990.
2. G. Huet and J.J. Lévy. Computations in orthogonal term rewriting systems. In J.L. Lassez and G. Plotkin, editors, *Computational logic: essays in honour of J. Alan Robinson*, pages 395-414 and 415-443. The MIT Press, Cambridge, MA, 1991.
3. J.F.Th. Kamperman and H.R. Walters. Lazy Rewriting and Eager Machinery. In J. Hsiang, editor, *Proc. of 6th International Conference on Rewriting Techniques and Applications, RTA'95*, LNCS 914:147-162, Springer-Verlag, Berlin, 1995.
4. J.W. Klop. Term Rewriting Systems. In S. Abramsky, D.M. Gabbay and T.S.E. Maibaum. *Handbook of Logic in Computer Science*, volume 3, pages 1-116. Oxford University Press, 1992.
5. S. Lucas. Context-sensitive computations in functional and functional logic programs. *Journal of Functional and Logic Programming*, 1997, *to appear*.
6. S. Lucas. Fundamentals of context-sensitive rewriting. In M. Bartŏsek, J. Staudek and J. Wiedermann, editors, *Proc. of XXII Seminar on Current Trends in Theory and Practice of Informatics, SOFSEM'95*, LNCS 1012:405-412, Springer-Verlag, Berlin, 1995.
7. S. Lucas. Needed Reductions with Context-Sensitive Rewriting. In M. Hanus and K. Meinke, editors, *Proc. of 6th International Conference on Algebraic and Logic Programming, ALP'97*, LNCS *to appear*.
8. M.J. O'Donnell. Equational Logic as a Programming Language. The MIT Press, Cambridge, Massachusetts, 1985.
9. C. Reade. Elements of Functional Programming. Addison-Wesley Publishing Company, 1993.
10. R.C. Sekar, S. Pawagi and I.V. Ramakrishnan. Transforming Strongly Sequential Rewrite Systems with Constructors for Efficient Parallel Execution. In N. Dershowitz, editor, *Proc. of 3rd International Conference on Rewriting Techniques and Applications, RTA'89*, LNCS 355:404-418, Springer-Verlag, Berlin, 1989.
11. Y. Toyama. Strong Sequentiality of Left-Linear Overlapping Rewrite Systems. In *Proc. of 7th IEEE Symposium on Logic in Computer Science*, Santa Cruz, CA, pages 274-284, 1992.
12. H. Zantema. Termination of Context-Sensitive Rewriting. In H. Comon, editor, *Proc. of 8th International Conference on Rewriting Techniques and Applications, RTA'97*, LNCS 1232:172-186, Springer-Verlag, Berlin, 1997.

This article was processed using the LaTeX macro package with LLNCS style