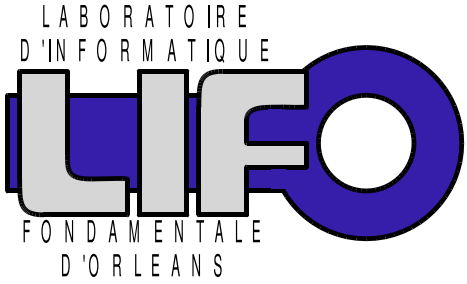

4 rue Léonard de Vinci
BP 6759
F-45067 Orléans Cedex 2
FRANCE
http://www.univ-orleans.fr/lifo

Rapport de Recherche

Transforming Prefix-Constrained or Controlled Rewrite Systems

Nirina Andrianarivelo, Vivien Pelletier, Pierre Réty
LIFO, Université d'Orléans

Rapport n^o **RR-2017-01**

Transforming Prefix-Constrained or Controlled Rewrite Systems

Nirina Andrianarivelo, Vivien Pelletier, Pierre Réty

LIFO - Université d'Orléans, B.P. 6759, 45067 Orléans cedex 2, France
{Nirina.Andrianarivelo, Vivien.Pelletier, Pierre.Rety}@univ-orleans.fr

Abstract

We present two techniques for transforming any prefix-constrained and any controlled term rewrite system into an ordinary rewrite system. We prove that both transformations preserve the rewrite computations, and preserve termination. In this way, prefix-constrained rewriting and controlled rewriting can be run, and termination can be checked, using the usual tools for ordinary rewriting.

1 Introduction

Term rewriting is a rule-based formalism that can be used to study properties of functional programs, security protocols, musical rhythmics and so on. More generally, it provides a finite abstraction of a system whose configurations are represented by ranked terms. In this framework, to ensure the termination of rewrite computations, it is often necessary to restrict the possible rewrite positions, using strategies, or by allowing only some redex positions. In context-sensitive rewriting [12], some arguments of a function symbol may be defined as being non-rewritable. Prefix-constrained rewriting [10] is an extension of context-sensitive rewriting, where rewritable positions are defined by a finite string automaton that indicates the allowed prefixes. Controlled rewriting [9] is an extension of prefix-constrained rewriting, where rewritable positions are defined by a finite tree automaton that considers the entire term (i.e. not only prefixes).

A possible way to check termination and to work with these restricted rewrite techniques, consists in transforming them into ordinary rewriting having the same computations, and use well-known results and tools over ordinary rewriting. However, it has already been done only for context-sensitive rewriting [5, 6]. On the other hand, a general approach for translating so-called programmable strategies into ordinary rewriting has been presented in [3]. However the combinators considered in that paper to define strategies can encode neither prefix-constrained nor controlled rewriting.

In this paper, we transform any prefix constrained term rewrite system (pCTRS) and any controlled term rewrite system (cntTRS) R into an ordinary term rewrite system R' that computes the same descendants. By introducing sorts and thanks to the persistence of termination when adding/removing sorts [13, 8], we prove that this transformation preserves termination, i.e. R is terminating if and only if R' is terminating. However, in the case of cntTRSs, we assume that R is non-duplicating for proving that the termination of R implies the termination of R'. The converse holds without any assumption. The transformed rewrite system R' can be implemented using usual tools providing rewrite rules, like Tom [2], and termination can be studied using well-known termination analysis techniques and the corresponding tools, like AProVE [7] and TTT2 [11].

The paper is organized as follows. The preliminaries are introduced in Section 2. Section 3 presents a transformation of any pCTRS into an ordinary TRS. In the particular case of context-sensitive rewriting, a comparison with [6] is given in Section 3.3. Section 4 presents a transformation of any cntTRS into an ordinary TRS. The relationship with [9] is discussed in Section 4.3.

2 Preliminaries

Let us recall general notions of rewriting. For details see [1].

Term and Substitution. Consider a *finite ranked alphabet* Σ and a set of variables Var. Each symbol $f \in \Sigma$ has a unique arity, denoted by $ar(f)$. The notions of *first-order term*, *position* and *substitution* are defined as usual. Given σ and σ' two substitutions, $\sigma \circ \sigma'$ denotes the substitution such that for any variable x, $\sigma \circ \sigma'(x) = \sigma(\sigma'(x))$. $T(\Sigma)$ denotes the set of ground terms (without variables) over Σ. For a term t, $Var(t)$ is the set of variables of t, $Pos(t)$ is the set of positions of t, and ϵ is the root position. For $p \in Pos(t)$, $t(p)$ is the symbol of $\Sigma \cup Var$ occurring at position p in t, and $t|_p$ is the subterm of t at position p. For $p, p' \in Pos(t)$, $p < p'$ means that p occurs in t strictly above p'. The term t is *linear* if each variable of t occurs only once in t. The term $t[t']_p$ is obtained from t by replacing the subterm at position p by t'.

Term Rewrite System (TRS). A *rewrite rule* is an oriented pair of terms, written $l \to r$. We always assume that l is not a variable, and $Var(r) \subseteq Var(l)$. A *rewrite system* R is a finite set of rewrite rules. *lhs* stands for left-hand-side, *rhs* for right-hand-side. The *rewrite relation* $\to_R$ is defined as follows: $t \to_R t'$ if there exist a non-variable position $p \in Pos(t)$, a rule $l \to r \in R$, and a substitution θ s.t. $t|_p = \theta(l)$ and $t' = t[\theta(r)]_p$ (also denoted $t \to_R^p t'$). $\to_R^+$ denotes the transitive closure of $\to_R$, and $\to_R^*$ denotes the reflexive-transitive closure of $\to_R$. t' is a *descendant* of t if $t \to_R^* t'$. If I is a set of ground terms, $R^*(I)$ denotes the set of descendants of elements of I. The rewrite rule $l \to r$ is *left (resp. right) linear* if l (resp. r) is linear. R is *left (resp. right) linear* if all its rewrite rules are left (resp. right) linear. R is *linear* if R is both left and right linear. $l \to r$ is said *collapsing* if r is a variable.

String Automaton. Given an alphabet Σ, the set of strings is denoted by Σ^*, and ϵ denotes the *empty string*. Symbol '.' denotes the concatenation. A finite *string automaton* is a 5-tuple $\mathcal{A} = (\Sigma, Q, Q_I, Q_f, \Delta)$ where Q is a set of states, $Q_I \subseteq Q$ is the set of initial states, $Q_f \subseteq Q$ is the set of final states, and $\Delta \subseteq Q \times \Sigma \times Q$ is the set of *transitions*. The *transition relation* $\mapsto_\Delta$ between elements of $Q \times \Sigma^*$ is defined as follows: for $q, q' \in Q$, $a \in \Sigma$, $w \in \Sigma^*$, $(q, a.w) \mapsto_\Delta (q', w)$ iff $(q, a, q') \in \Delta$. The reflexive-transitive closure of $\mapsto_\Delta$ is written $\mapsto_\Delta^*$. The language recognized by $\mathcal{A}$ is $L(\mathcal{A}) = \{w \in \Sigma^* \mid \exists q_I \in Q_I, \exists q_f \in Q_f, (q_I, w) \mapsto_\Delta^* (q_f, \epsilon)\}$.

Tree Automaton [4]. A (bottom-up) finite *tree automaton* over a signature Σ is a 4-tuple $\mathcal{A} = (\Sigma, Q, Q_f, \Delta)$ where Q is a set of states, $Q_f \subseteq Q$ is the set of final states, and Δ is a set of *transitions* of the form $f(q_1, \ldots, q_n) \to q$ where $f \in \Sigma$ and $q_1, \ldots, q_n, q \in Q$. A *run* of $\mathcal{A}$ on a term $t \in T(\Sigma)$ is a mapping α from $Pos(t)$ into Q s.t. for all $p \in Pos(t), t(p)(\alpha(p.1), ..., \alpha(p.n)) \to \alpha(p)$ is in Δ, where the arity of the symbol $t(p)$ is n. The run α is successful (or accepting) if $\alpha(\epsilon) \in Q_f$. The set of successful runs of $\mathcal{A}$ on t is denoted $sruns(\mathcal{A}, t)$. The language recognized by $\mathcal{A}$ is $L(\mathcal{A}) = \{t \in T(\Sigma) \mid sruns(\mathcal{A}, t) \neq \emptyset\}$.

Selection Automaton [9]. A *selection automaton* $\mathcal{A}$ is a 5-tuple $(\Sigma, Q, Q_f, S, \Delta)$ where (Σ, Q, Q_f, Δ) is a tree automaton denoted $ta(\mathcal{A})$ and S is a set of states of Q called selection states. Given a term $t \in T(\Sigma)$, the set of positions of t selected by $\mathcal{A}$ is defined as

$$sel(\mathcal{A}, t) = \{p \in Pos(t) \mid \exists \alpha \in sruns(ta(\mathcal{A}), t), \alpha(p) \in S\}$$

Context-Sensitive Term Rewrite System (CS-TRS) [12]. A *context-sensitive rewrite relation* is a sub-relation of the ordinary rewrite relation in which rewritable positions are indicated by specifying arguments of function symbols. A mapping $\mu : \Sigma \to P(\mathbb{N})$ is said to be a *replacement map* (or Σ-map) if $\mu(f) \subseteq \{1, \ldots, ar(f)\}$ for all $f \in \Sigma$. A *context-sensitive* term rewriting system (CS-TRS) is a pair $\mathcal{R} = (R, \mu)$ composed of a TRS and a replacement map. The set of μ-replacing positions[1] $Pos^\mu(t)$ ($\subseteq Pos(t)$) is recursively defined: $Pos^\mu(t) = \{\epsilon\}$ if t is a constant or a variable, otherwise $Pos^\mu(f(t_1, \ldots, t_n)) = \{\epsilon\} \cup \{i.p \mid i \in \mu(f), p \in Pos^\mu(t_i)\}$. The rewrite relation induced by a CS-TRS $\mathcal{R}$ is defined: $t \hookrightarrow_\mathcal{R} t'$ if $t \to_R^p t'$ for some $p \in Pos^\mu(t)$.

[1] Also called positions allowed by μ.

Example 1. Let $\Sigma = \{f^{\setminus 2}, g^{\setminus 2}, a^{\setminus 0}, b^{\setminus 0}\}$ and $R = \{a \to b\}$ with $\mu(f) = \{1\}$ and $\mu(g) = \{2\}$. The positions allowed by μ in the term $\boldsymbol{f}(\boldsymbol{a}, a)$ are written in bold. Then the only derivation issued from this term is $f(a,a) \hookrightarrow_{\mathcal{R}} f(b,a)$. On the other hand, consider $t = \boldsymbol{f}(\boldsymbol{g}(a, \boldsymbol{a}), a)$. Then the only derivation issued from this term is $f(g(a,a),a) \hookrightarrow_{\mathcal{R}} f(g(a,b),a)$.

Prefix Constrained Term Rewrite System (pCTRS) [10]. Prefix constrained rewriting allows rewrite steps only at the positions p of t s.t. the path from the root to t and p belongs to a given regular string language. More precisely, consider the set of directions $Dir(\Sigma) = \{\langle g, i\rangle \mid g \in \Sigma, 1 \leq i \leq ar(g)\}$. For a ground term $t = g(t_1, \ldots, t_{(ar(g))}) \in T(\Sigma)$ and a position p, $path(t,p) \in Dir(\Sigma)^*$ is defined recursively by:

$$path(g(t_1, \ldots, t_{(ar(g))}), \epsilon) = \epsilon$$
$$path(g(t_1, \ldots, t_{(ar(g))}), i.p) = \langle g, i\rangle.path(t_i, p) \;\; with\; 1 \leq i \leq ar(g)$$

A *prefix constrained rewrite system* is a finite set R of prefix constrained rewrite rules of the form $L : l \to r$ s.t. $L \subseteq Dir(\Sigma)^*$ is a regular string language over $Dir(\Sigma)$, $l \in T(\Sigma, \mathcal{X})\backslash\mathcal{X}$, and $r \in T(\Sigma, var(l))$. A term t is rewritten to t' in one step by a pCTRS R, denoted by $t \hookrightarrow_R t'$, if there exist a prefix-constrained rewrite rule $L : l \to r$ in R, a position $p \in Pos(t)$ s.t. $path(t,p) \in L$, and a substitution σ s.t. $t|_p = \sigma(l)$ and $t' = t[\sigma(r)]_p$. The reflexive-transitive closure of $\hookrightarrow_R$ is denoted by $\hookrightarrow_R^*$.

Example 2. Let $\Sigma = \{f^{\setminus 2}, g^{\setminus 2}, a^{\setminus 0}, b^{\setminus 0}\}$ and $R = \{(\langle f,1\rangle.\langle g,2\rangle)^* : a \to b\}$. Let $t = f(g(a, \boldsymbol{a}), a)$. Note that $t(1.2) = a$ (in bold in t) and $path(t, 1.2) = \langle f,1\rangle.\langle g,2\rangle \in (\langle f,1\rangle.\langle g,2\rangle)^*$. Then this position can be reduced by prefix constrained rewriting, i.e. $t = f(g(a,\boldsymbol{a}),a) \hookrightarrow_R f(g(a,\boldsymbol{b}),a)$, whereas the other occurrences of a are not reducible.

Note that the term $f(a,a)$ is not reducible by the pCTRS R, whereas it is reducible by the CS-TRS of Example 1. However the pCTRS $R_1 = \{(\langle f,1\rangle|\langle g,2\rangle)^* : a \to b\}$ is equivalent to the CS-TRS of Example 1.

Controlled Term Rewrite System (cntTRS) [9]. A controlled rewrite system R is a finite set of controlled rewrite rules of the form $\mathcal{A} : l \to r$, composed of a selection automaton $\mathcal{A}$ and a rewrite rule $l \to r$. A term t rewrites into t' in one step by a cntTRS R, denoted $t \hookrightarrow_R t'$, if there exist a controlled rewrite rule $\mathcal{A} : l \to r$ in R, a position $p \in sel(\mathcal{A}, t)$, and a substitution σ s.t. $t|_p = \sigma(l)$ and $t' = t[\sigma(r)]_p$.

Example 3. Let $\Sigma = \{f^{\setminus 2}, g^{\setminus 2}, a^{\setminus 0}, b^{\setminus 0}\}$, $R = \{\mathcal{A} : a \to b\}$, and $\mathcal{A} = (\Sigma, Q, Q_f, S, \Delta)$ s.t. $Q = \{q_1, q_2, q_f\}$, $Q_f = \{q_f\}$, $S = \{q_2\}$, and $\Delta = \{a \to q_1,\ a \to q_2,\ g(q_1,q_2) \to q_1, f(q_1,q_1) \to q_f\}$.

Let $t = f(g(a,a),a)$. Using the transitions of Δ, we get the derivation $f(g(a,a),a) \to^* f(g(q_1,q_2),q_1) \to f(q_1,q_1) \to q_f$. Since $q_f \in Q_f$, this derivation defines a successful run α on t. Note that $\alpha(1.2) = q_2 \in S$, then 1.2 is a reducible position. Then $t = f(g(a,a),a) \hookrightarrow_R f(g(a,b),a)$. Note that $t' = f(g(a,a),b)$ is not reducible by R since there is no successful run on t'. So, the position 1.2 is reducible in t but not in t'. In other words, with a cntTRS, the reducibility of a position depends on the entire term, and not only on the symbols located above the position (as with a pCTRS).

Remark. Context-sensitive rewriting is a particular case of prefix-constrained rewriting, which is a particular case of controlled rewriting [10].

3 Transforming a pCTRS into a TRS

Given a pCTRS $R = \{L_k : l_k \to r_k \mid 1 \leq k \leq n\}$, we assume that each language $L_k \subseteq Dir(\Sigma)^*$ is defined by a string automaton $\mathcal{A}^k = (\Sigma, Q^k, Q_I^k, Q_f^k, \Delta^k)$ s.t. $\forall k, k' \in \{1, \ldots, n\}$, $(k \neq k' \Longrightarrow Q^k \cap Q^{k'} = \emptyset)$. Consequently, if $k \neq k'$ then $Q_I^k \cap Q_I^{k'} = Q_f^k \cap Q_f^{k'} = \Delta^k \cap \Delta^{k'} = \emptyset$.

We write $Q = \cup_{1\leq k\leq n} Q^k$, $Q_I = \cup_{1\leq k\leq n} Q_I^k$, $Q_f = \cup_{1\leq k\leq n} Q_f^k$, $\Delta = \cup_{1\leq k\leq n} \Delta^k$. On the other hand, we also view each state of Q as a unary symbol, and each transition $\delta = (q, \langle f, k\rangle, q')$ of

Δ as a symbol having the same arity as f. We want to transform a pCTRS R into an ordinary TRS R' that simulates the behavior of R.

Definition 4. Let $R = \{L_k : l_k \to r_k \mid 1 \leq k \leq m\}$ be a pCTRS over a signature Σ. The corresponding TRS R' over the signature $\Sigma' = \Sigma \cup \{top^{\backslash 1}, j^{\backslash 1}\} \cup Q \cup \Delta$ is $R' = R'_1 \cup R'_2 \cup R'_3 \cup R'_4$, where $x, x_1, \ldots, x_n$ are variables:

$$
\begin{aligned}
R'_1 &= \{top(j(x)) \to top(q_I(x)) \mid q_I \in Q_I\} \\
R'_2 &= \{q(f(x_1, \ldots, x_n)) \to \delta(x_1, \ldots, q'(x_i), \ldots, x_n)) \mid \delta = (q, \langle f, i\rangle, q') \in \Delta\} \\
R'_3 &= \{q_f(l_k) \to j(r_k) \mid q_f \in Q_f^k, (L_k : l_k \to r_k) \in R\} \\
R'_4 &= \{\delta(x_1, \ldots, j(x_i), \ldots, x_n) \to j(f(x_1, \ldots, x_i, \ldots, x_n)) \mid \delta = (q, \langle f, i\rangle, q') \in \Delta\}
\end{aligned}
$$

To explain the transformation, let us consider a simplified version of R' obtained by replacing δ by f in R'_2 and R'_4 (δ is for preserving termination). Thus, if a state q appears at position p in a term $t'' \in T(\Sigma')$, this means that there is a path from the root of t'' to position p that is recognized by the automaton into the state q. Term t'' is obtained from $top(j(t))$ by applying one rule of R'_1 and some rules of R'_2 which move a state down using the automaton transitions. If $q \in Q_f$, a rewrite step of R can be applied, thanks to R'_3, and q is replaced by j, which is a kind of token. Then j goes up thanks to R'_4. Symbol *top* is for marking the root of a term.

Remark. The number of rules in R' is linear in the size of the global automaton, because $|R'| = |Q_I| + |\Delta| + |Q_f| + |\Delta| \leq 2 \times (|Q| + |\Delta|)$.

Theorem 5. *Let $t \in T(\Sigma)$. $t \hookrightarrow_R^* t'$ if and only if $top(j(t)) \to_{R'}^* top(j(t'))$.*

Example 6. Consider Example 2 again, where $R = \{(\langle f, 1\rangle.\langle g, 2\rangle)^* : a \to b\}$. The string automaton $\mathcal{A} = (Dir(\Sigma), Q, Q_I, Q_f, \Delta)$ is defined by $Dir(\Sigma) = \{\langle f, 1\rangle, \langle f, 2\rangle, \langle g, 1\rangle, \langle g, 2\rangle\}$, $Q = \{q, q'\}, Q_I = \{q\}, Q_f = \{q\}, \Delta = \{(q, \langle f, 1\rangle, q'), (q', \langle g, 2\rangle, q)\}$.
Let us write $\delta_1 = (q, \langle f, 1\rangle, q')$ and $\delta_2 = (q', \langle g, 2\rangle, q)\}$.
$R'_1 = \{top(j(x)) \to top(q(x))\}, R'_2 = \{q(f(x, y)) \to \delta_1(q'(x), y),\ q'(g(x, y)) \to \delta_2(x, q(y))\}, R'_3 = \{q(a) \to j(b)\}, R'_4 = \{\delta_1(j(x), y) \to j(f(x, y)),\ \delta_2(x, j(y)) \to j(g(x, y))\}$

Let us consider $t = f(g(a, a), a)$. The starting term is $top(j(f(g(a, a), a)))$.
$top(j(f(g(a, a), a))) \to_{R'_1} top(q(f(g(a, a), a))) \to_{R'_2} top(\delta_1(q'(g(a, a)), a)))$
$\to_{R'_2} top(\delta_1(\delta_2(a, q(a)), a)) \to_{R'_3} top(\delta_1(\delta_2(a, j(b)), a)) \to_{R'_4} top(\delta_1(j(g(a, b)), a))$
$\to_{R'_4} top(j(f(g(a, b), a)))$.
Thus $f(g(a, a), a) \hookrightarrow_R f(g(a, b), a)$.

Thanks to the following result, termination of a pCTRS could be proved using well-known termination techniques for ordinary rewriting.

Theorem 7. *R is terminating over Σ if and only if R' is terminating over Σ'.*

3.1 Proof of Theorem 5

Theorem 5 is obtained thanks to Corollary 11 and Lemma 13 below.

Notation. Given a term t, a position $p \in Pos(t)$ and a symbol h, let $t\{h\}_p$ denote the term obtained from t by replacing the symbol at position p by h, assuming that $t(p)$ and h have the same arity.

Lemma 8. *Let $t \in T(\Sigma)$, $p = u_1 \ldots u_n \in Pos(t)$, and $q_0, q_n \in Q^k$. For each $i \in \{1, \ldots, n\}$, let us write $v_i = u_1 \ldots u_i$, and $v_0 = \epsilon$. Then*

$$(q_0, path(t, p)) \mapsto^*_{\delta_1, \ldots, \delta_n \in \Delta^k} (q_n, \epsilon) \iff q_0(t) \to^*_{R'_2} t\{\delta_1\}_{v_0} \cdots \{\delta_n\}_{v_{n-1}}[q_n(t|_p)]_p$$

Proof. See Appendix A. □

Lemma 9. *Let $t \in T(\Sigma)$, $p = u_1 \dots u_n \in Pos(t)$, $q_I \in Q_I^k$ and $q_f \in Q_f^k$. For each $i \in \{1, \dots, n\}$, let us write $v_i = u_1 \dots u_i$, and $v_0 = \epsilon$.*
If $t \to_{[p, l_k \to r_k, \sigma]} t'$ and $(q_I, path(t,p)) \mapsto^*_{\delta_1, \dots, \delta_n \in \Delta^k} (q_f, \epsilon)$, then

$$\begin{aligned} & top(j(t)) \to_{R'_1} top(q_I(t)) \\ & \to^*_{R'_2} top(t\{\delta_1\}_{v_0} \cdots \{\delta_n\}_{v_{n-1}}[q_f(t|_p)]_p) = top(t\{\delta_1\}_{v_0} \cdots \{\delta_n\}_{v_{n-1}}[q_f(\sigma(l_k))]_p) \\ & \to_{R'_3} top(t\{\delta_1\}_{v_0} \cdots \{\delta_n\}_{v_{n-1}}[j(\sigma(r_k))]_p) \\ & \to^*_{R'_4} top(j(t[\sigma(r_k)]_p)) = top(j(t')) \end{aligned}$$

Proof. It comes from Lemma 8, and the form of the rules of R'_3 and R'_4. □

Corollary 10. *Let $t \in T(\Sigma)$. If $t \hookrightarrow_R t'$ then $top(j(t)) \to^+_{R'} top(j(t'))$.*

Corollary 11. *Let $t \in T(\Sigma)$. If $t \hookrightarrow^*_R t'$ then $top(j(t)) \to^*_{R'} top(j(t'))$.*

Lemma 12. *Let $t \in T(\Sigma)$ and $q_I \in Q_I^k$.*
If $top(q_I(t)) \to^+_{R'_2 \cup R'_3 \cup R'_4} top(j(t'))$, then $\exists p \in Pos(t)$, $t \hookrightarrow_{[p, l_k \to r_k]} t'$.

Proof. See Appendix B. □

Lemma 13. *Let $t \in T(\Sigma)$. If $top(j(t)) \to^+_{R'} top(j(t'))$, then $t \hookrightarrow^+_R t'$.*
Proof. The derivation is composed of steps of the form $top(j(t_i)) \to_{R'_1} top(q_i(t_i)) \to^*_{R'_2 \cup R'_3 \cup R'_4} top(j(t_{i+1}))$ where $q_i \in Q_I$. From Lemma 12, $t_i \hookrightarrow_R t_{i+1}$. □

3.2 Proof of Theorem 7

Theorem 7 is obtained thanks to Lemma 14 and Corollary 23 below.

Lemma 14. *If R' is terminating, then R is terminating.*

Proof. By contrapositive. Assume that R is not terminating, i.e. there is an infinite derivation $t_0 \hookrightarrow_R t_1 \hookrightarrow_R \cdots t_n \hookrightarrow_R \cdots$. From Corollary 10, $top(j(t_0)) \to^+_{R'} top(j(t_1)) \to^+_{R'} \cdots top(j(t_n)) \to^+_{R'} \cdots$, i.e. R' is not terminating. □

To prove the converse, we need to introduce sorts, and use the results of [13] about the persistence of termination when adding or removing sorts.

Definition 15. We consider the set of sorts $S = \{s_\Sigma, s_Q, s_\top\}$. Let us define a sort attachment of Σ' on S defined by:

- $\forall f \in \Sigma$, $f : s_\Sigma \times \cdots \times s_\Sigma \to s_\Sigma$
- $\forall q \in Q$, $q : s_\Sigma \to s_Q$
- $j : s_\Sigma \to s_Q$
- $\forall \delta = (q, \langle f, k \rangle, q') \in \Delta$, $\delta : s_\Sigma \times \cdots \times s_\Sigma \times s_Q \times s_\Sigma \times \cdots \times s_\Sigma \to s_Q$, where s_Q is the k-th argument of the left-hand-side.
- $top : s_Q \to s_\top$

Let R'_{st} be the TRS R' viewed as a many-sorted TRS, considering that all variables occurring in the rewrite rules of R'_{st} are of sort s_Σ. The sort attachment is consistent with R' since for each rule $l \to r$ of R'_{st}, terms l and r are well-sorted, and the sorts of l and r are equal.

Note that for all well-sorted terms $t, t' \in T(\Sigma')$, $t \to_{R'} t'$ if and only if $t \to_{R'_{st}} t'$. On the other hand, if $t \in T(\Sigma')$ is not well-sorted, then t is not reducible by R'_{st} whereas t may be reducible by R'. In other words, R'_{st} works only with well-sorted terms.

Remark. If $t \in T(\Sigma')$ is of sort s_Σ, then $t \in T(\Sigma)$.

Since R'_{st} does not include collapsing rules, Theorem 14 of [13] applies, i.e. termination is persistent. Therefore:

Lemma 16. *If R' is not terminating, then R'_{st} is not terminating.*

Lemma 17. *Let $t \in T(\Sigma')$ be a well-sorted term of sort $s_\top$. Then $t(\epsilon) = top$ and $\forall p \in Pos(t)\backslash\{\epsilon\}$, $t(p) \neq top$.*

Proof. Obvious. □

Lemma 18. *Let $t \in T(\Sigma')$ be a well-sorted term. If $t \to_{R'_1} t'$ then t is of sort $s_\top$.*
Proof. Assume that the step $t \to_{R'_1} t'$ is done at position $p \in Pos(t)$. Then $t(p) = top$ and $t|_p$ is of sort $s_\top$. If $p \neq \epsilon$, according to the sort attachment, t is not well-sorted. Then necessarily $p = \epsilon$. □

Lemma 19. *Let $t_0 \in T(\Sigma')$ be a well-sorted term. If $t_0 \to^+_{R'_{st}} t_n$ by a derivation containing at least one step by a rule of R'_1, then for all $i \in \{0, \ldots, n\}$, t_i is of sort $s_\top$ and $t_i(\epsilon) = top$.*
Proof. There is some $j \in \{0, \ldots, n-1\}$ s.t. $t_j \to_{R'_1} t_{j+1}$. From the previous lemma, t_j is of sort $s_\top$. Since every rule of R'_{st} is sort-preserving, for all i, t_i is of sort $s_\top$ and $t_i(\epsilon) = top$. □

Lemma 20. *$R'_2 \cup R'_3 \cup R'_4$ is terminating.*
Proof. Using the multiset path ordering (mpo) [1] with the precedence defined by:
$\forall q, q' \in Q, \forall \delta \in \Delta, \forall f \in \Sigma, q \sim q' > \delta \sim f > j$, and using that for each rule of R'_3, $Var(r_k) \subseteq Var(l_k)$. □

Lemma 21. *Let $t'_0 \in T(\Sigma')$ be a well-sorted term.*
If the derivation $t'_0 \to_{R'_{st}} t'_1 \to_{R'_{st}} t'_2 \to^*_{R'_{st}} \ldots$ is infinite, then

- *for all $i \in \mathbb{N}$, the sort of t'_i is $s_\top$,*
- *and there exists an infinite subset $I = \{i_1, i_2, \ldots\}$ of $\mathbb{N}$ s.t. $\forall i \in I, \exists q_i \in Q_I, \exists t_i \in T(\Sigma)$, $t'_i = top(q_i(t_i))$, and we have $t_{i_1} \hookrightarrow_R t_{i_2} \hookrightarrow_R \ldots$, which is an infinite derivation by R.*

Proof. See Appendix C. □

Corollary 22. *If R'_{st} is not terminating, then R is not terminating.*

Thanks to Corollary 22 and Lemma 16, we get:

Corollary 23. *If R is terminating, then R' is terminating.*

3.3 The context-sensitive case

A CS-TRS (R_0, μ) may be viewed as a particular pCTRS $R = \{L_k : l_k \to r_k \mid (l_k \to r_k) \in R_0\}$, where all languages L_k are the same (say L), and L is defined by the string automaton s.t. $Q = Q_I = Q_f = \{q\}$ and $\Delta = \{(q, \langle f, k\rangle, q) \mid f \in \Sigma, k \in \mu(f)\}$. Since q is the unique state, we write $\langle f, k\rangle$ instead $(q, \langle f, k\rangle, q)$, which denotes a transition $\delta \in \Delta$ considered as a symbol having the same arity as f. Then the rewrite system obtained by our transformation writes $R' = R'_1 \cup R'_2 \cup R'_3 \cup R'_4$ (where $x, x_1, \ldots, x_n$ are variables):

$$R'_1 = \{top(j(x)) \to top(q(x))\}$$
$$R'_2 = \{q(f(x_1, ..., x_n)) \to \langle f, k\rangle(x_1, ..., q(x_k), ..., x_n)) \mid f \in \Sigma, k \in \mu(f)\}$$
$$R'_3 = \{q(l_k) \to j(r_k) \mid (l_k \to r_k) \in R_0\}$$
$$R'_4 = \{\langle f, k\rangle(x_1, ..., j(x_k), ..., x_n) \to j(f(x_1, ..., x_k, ..., x_n) \mid f \in \Sigma, k \in \mu(f)\}$$

Now, if we replace q by *active*, and j by *mark*, and $\delta = \langle f, k\rangle$ by f, we get the second transformation of [6], except that the rewrite rules for the additional symbols *proper* and *ok* are missing. In other words, our transformation is simpler and we get fewer rules.

However, the transformation of [6], without the rules for *proper* and *ok*, does not preserve termination, as the authors has shown using a counter-example. Let us try to run the same counter-example, using our transformation.

Example 24. [6] Let $R_0 = \{f(x, g(x), y) \to f(y, y, y),\ g(b) \to c,\ b \to c\}$ with $\mu(f) = \emptyset$ and $\mu(g) = \{1\}$. This CS-TRS is clearly terminating. Using our transformation, we get
$R' = \{top(j(x)) \to top(q(x)),\ q(g(x)) \to \langle g, 1\rangle(q(x)),\ q(f(x, g(x), y)) \to j(f(y, y, y)),$
$q(g(b)) \to j(c),\ q(b) \to j(c), \langle g, 1\rangle(j(x)) \to j(g(x))\}$.
Let us consider the term $t = top(q(f(s, s, s)))$ with $s = q(g(b))$. Then:
$top(q(f(s, s, s))) \to_{R'} top(q(f(j(c), s, s))) \to_{R'} top(q(f(j(c), \langle g, 1\rangle(q(b)), s)))$
$\to_{R'} top(q(f(j(c), \langle g, 1\rangle(j(c)), s))) \not\to_{R'} top(j(f(s, s, s))) \to_{R'} top(q(f(s, s, s)))$
The step $\not\to_{R'}$ is not possible anymore thanks to our use of $\delta = \langle g, 1\rangle$ instead of g. Consequently this loop is not possible.

4 Transforming a cntTRS into a TRS

Given a cntTRS $R = \{\mathcal{A}_k : l_k \to r_k \mid 1 \le k \le m\}$, we assume that each selection automaton $\mathcal{A}_k$ writes $\mathcal{A}_k = (\Sigma, Q^k, Q^k_f, S^k, \Delta^k)$ s.t. $\forall k, k' \in \{1, \dots, m\}$, $(k \neq k' \implies Q^k \cap Q^{k'} = \emptyset)$. Consequently, if $k \neq k'$ then $Q^k_f \cap Q^{k'}_f = S^k \cap S^{k'} = \Delta^k \cap \Delta^{k'} = \emptyset$.
Let $Q = \cup_{1\le k\le m} Q^k$, $Q_f = \cup_{1\le k\le m} Q^k_f$, $S = \cup_{1\le k\le m} S^k$, $\Delta = \cup_{1\le k\le m} \Delta^k$, $\mathcal{A} = (\Sigma, Q, Q_f, S, \Delta)$.

We consider two additional sets of states $\bar{Q}$ and $\hat{Q}$ disjoint from Q s.t. $\bar{Q} \cap \hat{Q} = \emptyset$, and a bijection from Q to $\bar{Q}$ (resp. to $\hat{Q}$) that associates a state $\bar{q} \in \bar{Q}$ (resp. $\hat{q} \in \hat{Q}$) to each $q \in Q$. On the other hand, we also view each state of $Q \cup \bar{Q} \cup \hat{Q}$ as a unary symbol, and each transition $\delta = (f(q_1, \dots, q_n) \to q)$ of Δ as a symbol having the same arity as f.

We want to transform a cntCTRS R into an ordinary TRS R' that simulates the behavior of R.

Definition 25. Let $R = \{\mathcal{A}_k : l_k \to r_k \mid 1 \le k \le m\}$ be a cntCTRS over a signature Σ. The corresponding TRS R' over the signature $\Sigma' = \Sigma \cup Q \cup \bar{Q} \cup \hat{Q} \cup \Delta \cup \{top^{\setminus 1}\} \cup \{j_k^{\setminus 1} \mid 1 \le k \le m\}$ is $R' = R'_0 \cup \dots \cup R'_5$ (where $x, x_1, \dots, x_n$ are variables):

$$R'_0 = \{f(q_1(x_1), \dots, q_n(x_n)) \to q(\delta(q_1(x_1), \dots, q_n(x_n))) \mid \delta = (f(q_1, \dots, q_n) \to q) \in \Delta\}$$
$$R'_1 = \{top(q_f(x)) \to top(\hat{q_f}(q_f(x))) \mid q_f \in Q_f\}$$
$$R'_2 = \{\hat{q}(q(\delta(x_1, \dots, x_n))) \to f(\bar{q_1}(x_1), \dots, \hat{q_k}(x_k), \dots, \bar{q_n}(x_n)) \mid \begin{array}{l}\delta = (f(q_1, \dots, q_n) \to q) \in \Delta,\\ k \in \{1, \dots, n\},\ n \ge 1\end{array}\}$$
$$R'_3 = \{\bar{q}(q(\delta(x_1, \dots, x_n))) \to f(\bar{q_1}(x_1), \dots, \bar{q_n}(x_n)) \mid \delta = (f(q_1, \dots, q_n) \to q) \in \Delta\}$$
$$R'_4 = \{\hat{q}(q(\delta(x_1, \dots, x_n))) \to j_k(f(\bar{q_1}(x_1), \dots, \bar{q_n}(x_n))) \mid \delta = (f(q_1, \dots, q_n) \to q) \in \Delta,\ q \in S^k\}$$
$$R'_5 = \{j_k(l_k) \to r_k \mid (A_k : l_k \to r_k) \in R\}$$

To explain the transformation, let us consider a simplified and non-terminating[2] version of R', obtained by replacing δ by f in R'_0, R'_2, R'_3 and R'_4. The rules of R'_0 achieve a run of $\mathcal{A}$ on $t \in T(\Sigma)$, and we get a term $q(t')$ marked with states. If the run is successful, i.e. $q \in Q_f$, then $top(q(t')) \to_{R'_1} top(\hat{q}(q(t')))$. Symbols $\bar{q}$ and $\hat{q}$ are for removing the states in the whole term (thanks to R'_3 and R'_2), except that $\hat{q}$ will force a rewrite step by a rule of R (thanks to R'_4 and R'_5) at exactly one position marked by a state of S, i.e. a rewrite position allowed by the cntTRS R. Symbol *top* is for marking the root of a term.

Remark. For a given signature Σ, the number of rules in R' is linear in the size of the global automaton and the initial cntTRS, because $|R'| = |R'_0| + \dots + |R'_5| \le |\Delta| + |Q_f| + |\Delta| * ArMax(\Sigma) + |\Delta| + |\Delta| + |R| \le (3 + ArMax(\Sigma)) * |\Delta| + |Q| + |R|$.

Theorem 26. *Let $t, t' \in T(\Sigma)$.*
$t \hookrightarrow^*_R t'$ if and only if $top(t) \to^*_{R'} top(t')$.

[2] In this case, R'_0, and consequently R', are not terminating, even if R is. However, Theorem 26 still holds.

Example 27. Let $\Sigma = \{f^{\setminus 2}, a^{\setminus 0}, b^{\setminus 0}\}$, $R = \{\mathcal{A} : a \to b\}$, and $\mathcal{A} = (\Sigma, Q, Q_f, S, \Delta)$ s.t. $Q = \{q_1, q_2, q_f\}$, $Q_f = \{q_f\}$, $S = \{q_2\}$, $\Delta = \{\delta_1 : a \to q_1,\ \delta_2 : a \to q_2,\ \delta_3 : f(q_1, q_2) \to q_f\}$. Note that $L(\mathcal{A}) = \{f(a, a)\}$. We get:

$R'_0 = \{a \to q_1(\delta_1),\ a \to q_2(\delta_2),\ f(q_1(x_1), q_2(x_2)) \to q_f(\delta_3(q_1(x_1), q_2(x_2)))\}$
$R'_1 = \{top(q_f(x)) \to top(\hat{q_f}(q_f(x)))\}$
$R'_2 = \{\hat{q_f}(q_f(\delta_3(x_1, x_2))) \to f(\hat{q_1}(x_1), \hat{q_2}(x_2)),\ \hat{q_f}(q_f(\delta_3(x_1, x_2))) \to f(\bar{q_1}(x_1), \hat{q_2}(x_2))\}$
$R'_3 = \{\bar{q_1}(q_1(\delta_1)) \to a,\ \bar{q_2}(q_2(\delta_2)) \to a,\ \bar{q_f}(q_f(\delta_3(x_1, x_2))) \to f(\bar{q_1}(x_1), \bar{q_2}(x_2))\}$
$R'_4 = \{\hat{q_2}(q_2(\delta_2) \to j(a)\}$
$R'_5 = \{j(a) \to b\}$

The only successful run α on $f(a, a)$ satisfies $\alpha(1) = q_1$, $\alpha(2) = q_2$, $\alpha(\epsilon) = q_f$, and recall that $S = \{q_2\}$. Then $f(a, a) \hookrightarrow_R f(a, b)$. This R-step can be simulated by R' in this way:

- We start with the term $top(f(a, a))$.
- With $a \to q_1(\delta_1) \in R'_0$ we get $top(f(q_1(\delta_1), a))$.
- With $a \to q_2(\delta_2) \in R'_0$ we get $top(f(q_1(\delta_1), q_2(\delta_2)))$.
- With $f(q_1(x_1), q_2(x_2)) \to q_f(\delta_3(q_1(x_1), q_2(x_2))) \in R'_0$ we get $top(q_f(\delta_3(q_1(\delta_1), q_2(\delta_2))))$.
- With $top(q_f(x)) \to top(\hat{q_f}(q_f(x))) \in R'_1$ we get $top(\hat{q_f}(q_f(\delta_3(q_1(\delta_1), q_2(\delta_2)))))$.
- With $(\hat{q_f}(q_f(\delta_3(x_1, x_2))) \to f(\bar{q_1}(x_1), \hat{q_2}(x_2)) \in R'_2$ we get $top(f(\bar{q_1}(q_1(\delta_1)), \hat{q_2}(q_2(\delta_2))))$.
- With $\hat{q_2}(q_2(\delta_2)) \to j(a) \in R'_4$, we get $top(f(\bar{q_1}(q_1(\delta_1)), j(a)))$.
- With $j(a) \to b \in R'_5$ we get $top(f(\bar{q_1}(q_1(\delta_1)), b))$.
- With $\bar{q_1}(q_1(\delta_1)) \to a \in R'_3$ we get $top(f(a, b))$.

Thus $top(f(a, a)) \to^+_{R'_0} top(q_f(\delta_3(q_1(\delta_1), q_2(\delta_2)))) \to_{R'_1} top(\hat{q_f}(q_f(\delta_3(q_1(\delta_1), q_2(\delta_2))))$
$\to_{R'_2} top(f(\bar{q_1}(q_1(\delta_1)), \hat{q_2}(q_2(\delta_2)))) \to_{R'_4} top(f(\bar{q_1}(q_1(\delta_1)), j(a))) \to_{R'_5} top(f(\bar{q_1}(q_1(\delta_1)), b))$
$\to_{R'_3} top(f(a, b))$.

Thanks to the following results, termination of a cntTRS could be proved using well-known termination techniques for ordinary rewriting.

Theorem 28. *If R' is terminating over Σ', then R is terminating over Σ.*

To prove the equivalence, an additional restriction is needed. R is said *duplicating* if there is a rule $A_k : l_k \to r_k$ in R and a variable that has more occurrences in r_k than in l_k.

Theorem 29. *Assume that R is non-duplicating. Then:*
R is terminating over Σ if and only if R' is terminating over Σ'.

4.1 Proof of Theorem 26

Theorem 26 is obtained thanks to Corollary 33 and Lemma 30 below.

Lemma 30. *Let $t \in T(\Sigma)$ and $q \in Q$.*
$t \to^*_{R'_0} q(t_1)$ if and only if there exists a run α on t s.t. $\alpha(\epsilon) = q$.
Moreover
- if $\hat{q}(q(t_1)) \to^*_{R'_2} t_2 \wedge \exists p \in Pos(t_2), t_2(p) \in \hat{Q}$, then $p \in Pos(t)$ and $t_2(p) = \alpha\hat{(}p)$.
- conversely, for all $p \in Pos(t)$, there exists a derivation $\hat{q}(q(t_1)) \to^*_{R'_2} t_2$ s.t. $p \in Pos(t_2)$ and $t_2(p) = \alpha\hat{(}p)$.

Proof. See Appendix D. □

Lemma 31. *Let $t \in T(\Sigma)$, $p = u_1 \dots u_n \in Pos(t)$, $q_f \in Q_f^k$. For each $i \in \{1, \dots, n\}$, let us write $v_i = u_1 \dots u_i$, and $v_0 = \epsilon$.*
If $t \to_{[p, l_k \to r_k, \sigma]} t'$ and there is a successful run α on t s.t. $\alpha(p) \in S^k$, then (with $q_f \in Q_f$ and $q = \alpha(p)$)

$$\begin{aligned} &top(t) = top(t[t|_p]_p) \to^*_{R'_0} top(t[q(t'_1)]_p) \to^*_{R'_0} top(q_f(t_1)) \\ &\to_{R'_1} top(\hat{q_f}(q_f(t_1))) \\ &\to^*_{R'_2 \cup R'_3} top(t[\hat{q}(q(t'_1))]_p) \\ &\to_{R'_4} . \to^*_{R'_3} top(t[j_k(t|_p)]_p) = top(t[j_k(\sigma(l_k))]_p) \\ &\to_{R'_5} top(t[\sigma(r_k)]_p) = top(t') \end{aligned}$$

Proof. It comes from Lemma 30, and the form of the rules of R'_1, R'_4 and R'_5. □

Corollary 32. *Let $t \in T(\Sigma)$. If $t \hookrightarrow_R t'$ then $top(t) \to^+_{R'} top(t')$.*

Corollary 33. *Let $t \in T(\Sigma)$. If $t \hookrightarrow^*_R t'$ then $top(t) \to^*_{R'} top(t')$.*

Now, let us prove the converse.

Lemma 34. *Let $t \in T(\Sigma)$. If the derivation $top(t) \to^+_{R'} top(t')$ contains exactly one step by R'_1, and $t' \in T(\Sigma)$, then this derivation is of the form*

$$top(t) \to^*_{R'_0} t_0 \to_{R'_1} t_1 \to^*_{R'_2 \cup R'_3 \cup R'_4 \cup R'_5} top(t') \quad (A)$$

Moreover:
1) Each term t_i within the derivation $t_1 \to^* t_i \to^* top(t')$ contains at most one occurrence of symbols in $\hat{Q} \cup \{j_k\}$, and consequently $t_1 \to^* top(t')$ contains exactly one step by R'_5.
2) $t_1 \to^* top(t')$ can be commuted into $t_1 \to^*_{R'_2} t_2 \to_{R'_4} t_4 \to^*_{R'_3} t_3 \to_{R'_5} top(t')$.

Proof. See Appendix E. □

Lemma 35. *Let $t \in T(\Sigma)$. If the derivation $top(t) \to^+_{R'} top(t')$ contains exactly one step by R'_1, and $t' \in T(\Sigma)$, then there exists $p \in Pos(t)$ such that $t \hookrightarrow_{[p,R]} t'$.*

Proof. From Lemma 34, $top(t) \to^*_{R'_0} t_0 \to_{R'_1} t_1 \to^*_{R'_2} t_2 \to_{R'_4} t_4 \to^*_{R'_3} t_3 \to_{R'_5} top(t')$.

Note that the step by R'_4 and the step by R'_5 are achieved at the same position (say p), and $t_2(p) \in \hat{Q}$. From Lemma 30, there exists a run α on t s.t. $t_2(p) = \hat{\alpha(p)}$. Because of R'_4, $t_2(p) \in S^k$. Then the rewrite step by R'_5 is allowed by R, therefore $t \hookrightarrow_{[p,R]} t'$. □

Lemma 36. *Let $t \in T(\Sigma)$. If $top(t) \to^+_{R'} top(t')$ and $t' \in T(\Sigma)$, then $t \hookrightarrow^+_R t'$.*

Proof. See Appendix F. □

4.2 Proofs of Theorems 28 and 29

Theorem 28 comes from Lemma 37, whereas Theorem 29 is obtained thanks to Lemma 37 and Corollary 50 below.

Lemma 37. *If R' is terminating, then R is terminating.*

Proof. By contrapositive. Assume that R is not terminating, i.e. there is an infinite derivation $t_0 \hookrightarrow_R t_1 \hookrightarrow_R \dots t_n \hookrightarrow_R \dots$. From Corollary 32, $top(t_0) \to^+_{R'} top(t_1) \to^+_{R'} \dots top(t_n) \to^+_{R'} \dots$, i.e. R' is not terminating. □

To prove the converse, we need to introduce ordered sorts, and use the results of [8] about the persistence of termination when adding or removing sorts.

Definition 38. We consider the set of sorts $S = \{s_Q, s_\top\} \cup \{s_q \mid q \in Q\} \cup \{s'_q \mid q \in Q\}$ ordered by $\forall q \in Q,\ s_Q \succ s_q$. Note that $\succ$ is well-founded.
Let us define a sort attachment of Σ' on S defined by:

- $\forall f \in \Sigma,\ f : s_Q \times \cdots \times s_Q \to s_Q$
- $\forall q \in Q,\ q : s'_q \to s_q$
- $\forall \bar{q} \in \bar{Q},\ \bar{q} : s_q \to s_Q$
- $\forall \hat{q} \in \hat{Q},\ \hat{q} : s_q \to s_Q$
- $\forall k \in \{1, \ldots, m\},\ j_k : s_Q \to s_Q$
- $\forall \delta = (f(q_1, \ldots, q_n) \to q) \in \Delta,\ \delta : s_{q_1} \times \cdots \times s_{q_n} \to s'_q$
- $top : s_Q \to s_\top$

Let R'_{st} be the TRS R' viewed as an order-sorted TRS, considering that each variable x_i of R'_0 is of sort s'_{q_i}, the variable x of R'_1 is of sort s'_{q_f}, each variable x_i of $R'_2 \cup R'_3 \cup R'_4$ is of sort s_{q_i}, and the variables of R'_5 are of sort s_Q. Thus, the sort attachment is consistent with R', because for each rule $l \to r$ of R'_{st}, terms l and r are strictly well-sorted, and the sorts of l and r are equal, i.e. $s_\top$ for the rules of R'_1, s_Q for the rules of $R'_2 \cup \cdots \cup R'_5$; except for the rules of R'_0 where $l : s_Q$ and $r : s_q$, i.e. R'_0 is sort-decreasing, which is allowed in an order-sorted TRS.

Note that for all well-sorted terms $t, t' \in T(\Sigma')$, $t \to_{R'} t'$ if and only if $t \to_{R'_{st}} t'$. On the other hand, if $t \in T(\Sigma')$ is not well-sorted, then t is not reducible by R'_{st} whereas t may be reducible by R'. In other words, R'_{st} works only with well-sorted terms.

If R' does not include duplicating rules, Theorem 4.11 of [8] applies, i.e. termination is persistent. Therefore:

Lemma 39. *If R' is not terminating and does not include duplicating rules, then R'_{st} is not terminating.*

Lemma 40. *Let $t \in T(\Sigma')$ be a well-sorted term of sort $\top$. Then $t(\epsilon) = top$ and $\forall p \in Pos(t) \backslash \{\epsilon\}$, $t(p) \neq top$.*

Proof. Obvious. □

Lemma 41. *Let $t \in T(\Sigma')$ be a well-sorted term. If $t \to_{R'_1} t'$ then t is of sort $s_\top$.*

Proof. Assume that the step $t \to_{R'_1} t'$ is done at position $p \in Pos(t)$. Then $t(p) = top$ and $t|_p$ is of sort $s_\top$. If $p \neq \epsilon$, according to the sort attachment, t is not well-sorted. Then $p = \epsilon$. □

Lemma 42. *Let $t_0 \in T(\Sigma')$ be a well-sorted term. If $t_0 \to^+_{R'_{st}} t_n$ by a derivation containing at least one step by a rule of R'_1, then for all $i \in \{0, \ldots, n\}$, t_i is of sort $s_\top$ and $t_i(\epsilon) = top$.*

Proof. There is some $j \in \{0, \ldots, n-1\}$ s.t. $t_j \to_{R'_1} t_{j+1}$. From Lemma 41, t_j is of sort $s_\top$. Since every rule of R'_{st} is sort-preserving or sort-decreasing, and $s_\top$ is not comparable with any other sort, then for all i, t_i is of sort $s_\top$ and $t_i(\epsilon) = top$. □

Lemma 43. *$R'_0 \cup R'_2 \cup R'_3 \cup R'_4 \cup R'_5$ is terminating.*

Proof. Using the multiset path ordering (mpo) [1] with the precedence defined by $\forall q, q_1, q_2 \in Q$, $\forall \delta \in \Delta$, $\forall f \in \Sigma$, $\forall k$, $(\hat{q} \sim \bar{q_1} > j_k > f > q_2) \ \wedge \ (f > \delta)$, and using that for each rule of R'_5, $Var(r_k) \subseteq Var(l_k)$. □

Definition 44. We recursively define a mapping rm from $T(Q \cup \Delta)$ to $T(\Sigma)$ by

- $rm(q(t)) = rm(t)$ for $q \in Q$,
- $rm(\delta(t_1, \ldots, t_n)) = f(rm(t_1), \ldots, rm(t_n))$ for $\delta = (f(q_1, \ldots, q_n) \to q) \in \Delta$.

Remark. If t is of sort s_q or s'_q, then $t \in T(Q \cup \Delta)$, and $rm(t)$ is defined.

Lemma 45. *Let t of sort s'_q. Then $rm(t) \to^*_{R'_0} q(t)$.*
Proof. By structural induction on t.
- If $t = \delta$ with $\delta = (a \to q)$, then $rm(t) = a \to_{R'_0} q(\delta) = q(t)$.
- Otherwise $t = \delta(t_1, \ldots, t_n)$ with $\delta = (f(q_1, \ldots, q_n) \to q)$, and for each i, t_i is of sort s_{q_i}, i.e. $t_i = q_i(t'_i)$ and t'_i is of sort s'_{q_i}.
By the induction hypothesis, $rm(t_i) = rm(t'_i) \to^*_{R'_0} q_i(t'_i)$. Then $rm(t) = f(rm(t_1), \ldots, rm(t_n)) \to^*_{R'_0} f(q_1(t'_1), \ldots, q_n(t'_n)) \to_{R'_0} q(\delta(q_1(t'_1), \ldots, q_n(t'_n))) = q(\delta(t_1, \ldots, t_n)) = q(t)$. □

Lemma 46. *Let $t'' \in T(\Sigma)$. If $t'' \to^*_{R'_0} q'(t')$ and t' is of sort $s'_{q'}$, then $t'' = rm(t')$.*
Proof. By structural induction on t''. See Appendix G for details. □

Lemma 47. *Let $q, q' \in Q$ and $top(q(t))$ be a well-sorted term.*
If $top(q(t)) \to_{R'_1} . \to^*_{R'_0 \cup R'_2 \cup R'_3 \cup R'_4 \cup R'_5} top(q'(t'))$ then $\exists p \in Pos(rm(t))$, $rm(t) \hookrightarrow_{[p,R]} rm(t')$.
Proof. t is of sort s'_q. From Lemma 45 $top(rm(t)) \to^*_{R'_0} top(q(t)) \to_{R'_1} . \to^*_{R'_0 \cup R'_2 \cup R'_3 \cup R'_4 \cup R'_5} top(q'(t'))$. Since R'_0 is right-linear, one can commute $\to^*_{R'_0 \cup R'_2 \cup R'_3 \cup R'_4 \cup R'_5}$ into

$$\to^*_{R'_2 \cup R'_3 \cup R'_4 \cup R'_5} top(t'') \to^*_{R'_0} top(q'(t'))$$

and t' is of sort $s'_{q'}$ and $t'' \in T(\Sigma)$.
From Lemma 46, $t'' = rm(t')$. Then $top(rm(t)) \to^*_{R'_0} . \to_{R'_1} . \to^*_{R'_2 \cup R'_3 \cup R'_4 \cup R'_5} top(rm(t'))$.
However $rm(t) \in T(\Sigma)$ and $rm(t') \in T(\Sigma)$.
From Lemma 35 we get: $\exists p \in Pos(rm(t))$, $rm(t) \hookrightarrow_{[p,R]} rm(t')$. □

Lemma 48. *Let $t'_0 \in T(\Sigma')$ be a well-sorted term.*
If the derivation $t'_0 \to_{R'_{st}} t'_1 \to_{R'_{st}} t'_2 \to^*_{R'_{st}} \cdots$ is infinite, then
- *for all $i \in \mathbb{N}$, the sort of t'_i is $s_\top$,*
- *and there exists an infinite subset $I = \{i_1, i_2, \ldots\}$ of $\mathbb{N}$ s.t. $\forall i \in I$, $\exists q_i \in Q_f$, $\exists t_i : s'_{q_i}$, $t'_i = top(q_i(t_i))$, and $rm(t_{i_1}) \hookrightarrow_R rm(t_{i_2}) \hookrightarrow_R \cdots$, which is an infinite derivation by R.*

Proof. See Appendix H. □

Corollary 49. *If R'_{st} is not terminating, then R is not terminating.*

Thanks to Corollary 49 and Lemma 39, we get:

Corollary 50. *If R is terminating and non-duplicating, then R' is terminating.*

4.3 Comparison with [9]

Theorem 1 of [9] shows that starting from a context-sensitive tree grammar G and a monotonic cntTRS R, one can construct a context-sensitive tree grammar G_* that generates the closure of $L(G)$ by R. Roughly speaking, it amounts to consider the set Δ of production rules of G as a rewrite system, and to transform the cntTRS $\Delta \cup R$ into an ordinary TRS R', which will be considered as the production rules of G_*. This transformation of a cntTRS into a TRS has some similarities with ours, if we consider that λ corresponds to top, and fin to j_k. However, states are not considered as symbols. Instead, they are introduced in pairs of the form $\langle f, q \rangle$ (see Example 4 of [9]). Another difference is that R' is not terminating (see $P_{\mathcal{A}} \cup P_R$ in Example 4 of [9]) even if R is terminating and non-duplicating.

As we wanted to preserve termination, we have considered a more sophisticated transformation. Indeed, we view each transition δ as a function symbol, and use δ instead of $f \in \Sigma$.

On the other hand, instead of using δ, a simpler transformation could be considered, by using a new symbol f' for each $f \in \Sigma$. However, the proof technique of Theorem 29 based on sorts, does not work anymore because each symbol should have only one profile.

5 Conclusion and Further Work

Two techniques for transforming a pCTRS and a cntTRS into an ordinary TRS have been presented in this paper. Both preserve rewrite computations and termination. Thus, studying the termination of a pCTRS or a cntTRS amounts to study the termination of an ordinary TRS, which can be done using the well-known techniques and tools. Further work could consist in reducing the number of generated rewrite rules in particular cases, extending the methods for pCTRS or cntTRS with built-in algebras, dealing with pCTRS or cntTRS modulo usual theories (C,AC,....), dealing with conditional pCTRS or cntTRS.

References

[1] F. Baader and T. Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998.

[2] Emilie Balland, Paul Brauner, Radu Kopetz, Pierre-Etienne Moreau, and Antoine Reilles. Tom: Piggybacking Rewriting on Java. In Franz Baader, editor, *Term Rewriting and Applications, 18th International Conference, RTA 2007, Paris, France, June 26-28, 2007, Proceedings*, volume 4533 of *Lecture Notes in Computer Science*, pages 36–47. Springer, 2007.

[3] Horatiu Cirstea, Sergueï Lenglet, and Pierre-Etienne Moreau. A Faithful Encoding of programmable Strategies into Term Rewriting Systems. In *26th Conference RTA 2015, June 29 to July 1, 2015, Warsaw, Poland*, volume 36 of *LIPIcs*, pages 74–88, 2015.

[4] H. Comon, M. Dauchet, R. Gilleron, D. Lugiez, S. Tison, and M. Tommasi. *Tree Automata Techniques and Applications* (TATA).

[5] Jürgen Giesl and Aart Middeldorp. Transforming Context-Sensitive Rewrite Systems. In Paliath Narendran and Michaël Rusinowitch, editors, *Rewriting Techniques and Applications, 10th International Conference, RTA-99, Trento, Italy, July 2-4, 1999, Proceedings*, volume 1631 of *Lecture Notes in Computer Science*, pages 271–287. Springer, 1999.

[6] Jürgen Giesl and Aart Middeldorp. Transformation Techniques for Context-Sensitive Rewrite Systems. *J. Funct. Program.*, 14(4):379–427, 2004.

[7] Jürgen Giesl, René Thiemann, Peter Schneider-Kamp, and Stephan Falke. Automated Termination Proofs with AProVE. In Vincent van Oostrom, editor, *15th Conference, RTA 2004, Aachen, Germany, June 3-5, 2004, Proceedings*, volume 3091 of *LNCS*, pages 210–220. Springer, 2004.

[8] Munehiro Iwami. Persistence of Termination for Term Rewriting Systems with Ordered Sorts. *Journal of Mathematical, Computational, Physical, Electrical and Computer engineering*, 1(3):200–204, 2007.

[9] Florent Jacquemard, Yoshiharu Kojima, and Masahiko Sakai. Controlled Term Rewriting. In Cesare Tinelli and Viorica Sofronie-Stokkermans, editors, *Frontiers of Combining Systems, 8th International Symposium, FroCoS 2011, Saarbrücken, Germany, October 5-7, 2011. Proceedings*, volume 6989 of *Lecture Notes in Computer Science*, pages 179–194. Springer, 2011.

[10] Florent Jacquemard, Yoshiharu Kojima, and Masahiko Sakai. Term Rewriting with Prefix Context Constraints and Bottom-Up Strategies. In Amy P. Felty and Aart Middeldorp, editors, *Automated Deduction - CADE-25 - 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings*, volume 9195 of *LNCS*, pages 137–151. Springer, 2015.

[11] Martin Korp, Christian Sternagel, Harald Zankl, and Aart Middeldorp. Tyrolean Termination Tool 2. In Ralf Treinen, editor, *20th Conference RTA 2009, Brasília, Brazil, June 29 - July 1, 2009, Proceedings*, volume 5595 of *LNCS*, pages 295–304. Springer, 2009.

[12] S. Lucas. Context-Sensitive Computations in Functional and Functional logic Programs. *Journal of Functional and Logic Programming*, 1998(1), January 1998.

[13] Hans Zantema. Termination of Term Rewriting: Interpretation and Type Elimination. *Journal of Symbolic Computation*, 17(1):23–50, 1994.

A Proof of Lemma 8

$\Rightarrow$. By induction on the length n of the derivation $(q_0, path(t,p)) \mapsto^*_{\delta_1,\ldots,\delta_n \in \Delta^k} (q_n, \epsilon)$.
Base case: $n = 0$.
Thus $p = \epsilon$ and $(q_0, path(t,\epsilon)) = (q_n, \epsilon)$, i.e. $q_n = q_0$ and $path(t,\epsilon) = \epsilon$. We have to prove that $q_0(t) \to^*_{R'_2} t[q_n(t|_p)]_p$. Actually, this derivation can be achieved in 0 step because $t[q_n(t|_p)]_p = t[q_0(t|_\epsilon)]_\epsilon = t[q_0(t)]_\epsilon = q_0(t)$.
Induction step: suppose that $n \geq 1$.
Let us write $p' = u_2 \ldots u_n$ and for each $i \in \{2,\ldots,n\}$, let us write $v'_i = u_2 \ldots u_i$, and $v'_1 = \epsilon$. Thus $p = u_1.p'$ and $v_i = u_1.v'_i$. Note that $p \neq \epsilon$, therefore t is not a constant. Let us write $t = f(t_1,\ldots,t_{u_1},\ldots,t_{ar(f)})$. The derivation writes

$$(q_0, path(t,p)) \mapsto_{\delta_1 \in \Delta^k} (q_1, path(t_{u_1}, p')) \mapsto^*_{\delta_2,\ldots,\delta_n \in \Delta^k} (q_n, \epsilon)$$

Let us consider the first step. Note that $\delta_1 = (q_0, \langle f, u_1\rangle, q_1)$.
From Definition 4, $(q_0(f(x_1,\ldots,x_{ar(f)})) \to \delta_1(x_1,\ldots,q_1(x_{u_1}),\ldots,x_{ar(f)})) \in R'_2$. Thus

$$q_0(t) = q_0(f(t_1,\ldots,t_{u_1},\ldots,t_{ar(f)}) \to_{R'_2} \delta_1(t_1,\ldots,q_1(t_{u_1}),\ldots,t_{ar(f)})$$

On the other hand, the derivation $(q_1, path(t_{u_1}, p')) \mapsto^*_{\delta_2,\ldots,\delta_n \in \Delta^k} (q_n, \epsilon)$ includes $n-1$ steps. From induction hypothesis, we get $q_1(t_{u_1}) \to^*_{R'_2} t_{u_1}\{\delta_2\}_{v'_1} \cdots \{\delta_n\}_{v'_{n-1}}[q_n(t_{u_1}|_{p'})]_{p'}$.

Therefore

$$\begin{aligned}
&q_0(t) \to_{R'_2} \delta_1(t_1,\ldots,q_1(t_{u_1}),\ldots,t_{ar(f)})\\
&\to^*_{R'_2} \delta_1(t_1,\ldots,t_{u_1}\{\delta_2\}_{v'_1}\cdots\{\delta_n\}_{v'_{n-1}}[q_n(t_{u_1}|_{p'})]_{p'},\ldots,t_{ar(f)})\\
&= \delta_1(t_1,\ldots,t_{u_1},\ldots,t_{ar(f)})\{\delta_2\}_{u_1.v'_1}\cdots\{\delta_n\}_{u_1.v'_{n-1}}[q_n(t_{u_1}|_{p'})]_{u_1.p'}\\
&= \delta_1(t_1,\ldots,t_{u_1},\ldots,t_{ar(f)})\{\delta_2\}_{v_1}\cdots\{\delta_n\}_{v_{n-1}}[q_n(t|_p)]_p\\
&= f(t_1,\ldots,t_{u_1},\ldots,t_{ar(f)})\{\delta_1\}_\epsilon\{\delta_2\}_{v_1}\cdots\{\delta_n\}_{v_{n-1}}[q_n(t|_p)]_p\\
&= t\{\delta_1\}_{v_0}\{\delta_2\}_{v_1}\cdots\{\delta_n\}_{v_{n-1}}[q_n(t|_p)]_p
\end{aligned}$$

$\Leftarrow$. By induction on the length n of the derivation $q_0(t) \to^*_{R'_2} t\{\delta_1\}_{v_0}\cdots\{\delta_n\}_{v_{n-1}}[q_n(t|_p)]_p$.
Base case: $n = 0$.
Thus $p = \epsilon$ and $q_0(t) = t[q_n(t|_\epsilon)]_\epsilon = q_n(t)$, i.e. $q_n = q_0$. We have to prove that $(q_0, path(t,\epsilon)) \mapsto^* (q_0, \epsilon)$. This can be performed in 0 step, since $path(t,\epsilon) = \epsilon$ and then $(q_0, path(t,\epsilon)) = (q_0, \epsilon)$.
Induction step: suppose that $n \geq 1$.
Let us write $p' = u_2 \ldots u_n$ and for each $i \in \{2,\ldots,n\}$, let us write $v'_i = u_2 \ldots u_i$, and $v'_1 = \epsilon$. Thus $p = u_1.p'$ and $v_i = u_1.v'_i$. Note that $p \neq \epsilon$, therefore t is not a constant. Let us write $t = f(t_1,\ldots,t_{u_1},\ldots,t_{ar(f)})$. The derivation writes
$q_0(t) \to_{R'_2} t\{\delta_1\}_{v_0}[q_1(t|_{u_1})]_{u_1}$ and $q_1(t_{u_1}) \to^*_{R'_2} t_{u_1}\{\delta_2\}_{v'_1}\cdots\{\delta_n\}_{v'_{n-1}}[q_n(t_{u_1}|_{p'})]_{p'}$.
Note that $t\{\delta_1\}_{v_0}[q_1(t|_{u_1})]_{u_1} = f(t_1,\cdots,t_{u_1},\cdots,t_{ar(f)})\{\delta_1\}_{v_0}[q_1(t|_{u_1})]_{u_1}$
$= \delta_1(t_1,\cdots,t_{u_1},\cdots,t_{ar(f)})[q_1(t_{u_1})]_{u_1} = \delta_1(t_1,\cdots,q_1(t_{u_1}),\cdots,t_{ar(f)})$.
Therefore $q_0(t) \to_{R'_2} \delta_1(t_1,\cdots,q_1(t_{u_1}),\cdots,t_{ar(f)})$.

So, from Definition 4, we get $\delta_1 = (q_0, \langle f, u_1\rangle, q_1)$.
On the other hand, $path(t,p) = \langle f, u_1\rangle.path(t_{u_1}, p')$. Then $(q_0, path(t,p)) \mapsto_{\delta_1} (q_1, path(t_{u_1}, p'))$.

The derivation $q_1(t_{u_1}) \to^*_{R'_2} t_{u_1}\{\delta_2\}_{v'_1}\cdots\{\delta_n\}_{v'_{n-1}}[q_n(t_{u_1}|_{p'})]_{p'}$ includes $n-1$ steps. From the induction hypothesis, we get $(q_1, path(t_{u_1}, p')) \mapsto^*_{\delta_2,\cdots,\delta_n \in \Delta^k} (q_n, \epsilon)$. Finally $(q_0, path(t,p)) \mapsto_{\delta_1}$ $(q_1, path(t_{u_1}, p')) \mapsto^*_{\delta_2,\cdots,\delta_n \in \Delta^k} (q_n, \epsilon)$.

B Proof of Lemma 12

Since $t \in T(\Sigma)$, it is easy to see that each term in the derivation contains exactly one element in $Q \cup \{j\}$. To be applied, the rules of R'_2 need an occurrence of an element of Q, those of R'_3 replace an element of Q by j, and those of R'_4 need an occurrence of j. Then the derivation is of the form

$top(q_I(t)) \to^*_{R'_2} top(t\{\delta_1\}_{v_0} \cdots \{\delta_n\}_{v_{n-1}}[q_n(t|_p)]_p)$
$\to_{R'_3} top(t\{\delta_1\}_{v_0} \cdots \{\delta_n\}_{v_{n-1}}[j(t'|_p)]_p) \to^*_{R'_4} top(j(t[t'|_p])_p) = top(j(t'))$.
Note that $q_n \in Q^k_f$. From Lemma 8, $(q_I, path(t,p)) \mapsto^*_{\delta_1,\ldots,\delta_n \in \Delta^k} (q_n, \epsilon)$, in other words, $path(t,p) \in L_k$. Therefore $t \hookrightarrow_{[p, l_k \to r_k]} t'$.

C Proof of Lemma 21

Because of Lemma 20, the derivation contains necessarily infinitely many steps by R'_1. Let us write $R'' = R'_2 \cup R'_3 \cup R'_4$.
The derivation writes $t'_0 \to^*_{R''} t'_{i_1-1} \to_{R'_1} t'_{i_1} \to^*_{R''} t'_{i_2-1} \to_{R'_1} t'_{i_2} \cdots$. From Lemma 19, every term of the derivation is of sort $s_\top$. Therefore for each j, t'_{i_j-1} is of the form $t'_{i_j-1} = top(j(t_{i_j-1}))$ and t'_{i_j} is of the form $t'_{i_j} = top(q_{i_j}(t_{i_j}))$ where $q_{i_j} \in Q_I$ and $t_{i_j-1} = t_{i_j}$ are of sort s_Σ, i.e. $t_{i_j-1}, t_{i_j} \in T(\Sigma)$.
According to Lemma 12, for each j, $t_{i_j} \hookrightarrow_R t_{i_{j+1}-1} = t_{i_{j+1}}$. Therefore there is an infinite derivation $t_{i_1} \hookrightarrow_R t_{i_2} \hookrightarrow_R \ldots$.

D Proof of Lemma 30

1. Suppose $t \to^*_{R'_0} q(t_1)$ and let us prove that there exists a run α on t s.t. $\alpha(\epsilon) = q$. The proof is by induction on the length n of $t \to^*_{R'_0} q(t_1)$. Since $t \in T(\Sigma)$, necessarily $n > 0$.
If $n = 1$ then $t \to_{p'} q(t_1)$. Then $p' = \epsilon$, t is a constant (say a), and $t_1 = \delta$ where $\delta = (a \to q) \in \Delta$. Therefore there exists a run α on t s.t. $\alpha(\epsilon) = q$.
Otherwise $t \to^+ f(q_1(s_1), \ldots, q_n(s_n)) \to q(\delta(q_1(s_1), \ldots, q_n(s_n)))$ where $\delta = (f(q_1, \ldots, q_n) \to q) \in \Delta$.
Then for each i, $t|_i \to^*_{R'_0} q_i(s_i)$. From the induction hypothesis, there exists a run α_i on $t|_i$ s.t. $\alpha_i(\epsilon) = q_i$.
Now, we define α by $\forall i \in \{1, \ldots, n\}$, $\alpha(i.v) = \alpha_i(v)$ and $\alpha(\epsilon) = q$. Thus, α is a run on t because for each i, α_i is a run on $t|_i$, and $f(\alpha(1), \ldots, \alpha(n)) = f(\alpha_1(\epsilon), \ldots, \alpha_n(\epsilon)) = f(q_1, \ldots, q_n)$, $\alpha(\epsilon) = q$, and $f(q_1, \ldots, q_n) \to q \in \Delta$.

2. Suppose that there exists a run α on t s.t. $\alpha(\epsilon) = q$, and let us prove that $t \to^*_{R'_0} q(t_1)$. The proof is by structural induction on t.
If t is a constant, then $\delta = (t \to q) \in \Delta$, then $(t \to q(\delta)) \in R'_0$. Therefore $t \to_{R'_0} q(\delta)$.
Otherwise $t = f(\ldots)$. For each $i \in \{1, \ldots, n\}$ we define a run α_i on $t|_i$ by $\alpha_i(v) = \alpha(i.v)$. Then $\alpha_i(\epsilon) = \alpha(i)$. Let us write $q_i = \alpha_i(\epsilon) = \alpha(i)$. From the induction hypothesis, $t|_i \to^*_{R'_0} q_i(s_i)$.
On the other hand, α is a run on t, then $\delta = (f(q_1, \ldots, q_n) \to q) \in \Delta$, then the rule $f(q_1(x_1), \ldots, q_n(x_n)) \to q(\delta(q_1(x_1), \ldots, q_n(x_n)))$ is in R'_0.
Consequently $t \to^*_{R'_0} f(q_1(s_1), \ldots, q_n(s_n)) \to_{R'_0} q(\delta(q_1(s_1), \ldots, q_n(s_n)))$.

3. Suppose that $\hat{q}(q(t_1)) \to^*_{R'_2} t_2$ and $t_2(p) \in \hat{Q}$, and let us prove that $t_2(p) = \hat{\alpha(p)}$. The proof is by induction on the length n of $\hat{q}(q(t_1)) \to^*_{R'_2} t_2$.

If $n = 0$, then $t_2 = \hat{q}(q(t_1))$ and $p = \epsilon$. Then $p \in Pos(t)$ and $t_2(p) = t_2(\epsilon) = \hat{q} = \hat{\alpha(p)}$.
Otherwise $n \geq 1$. Then
$\hat{q}(q(t_1)) \to^*_{R'_2} t_2[\hat{q'}(q'(\delta(s_1, \ldots, s_n)))]_{p'} \to_{R'_2} t_2[f(\bar{q_1}(s_1), \ldots, \hat{q_k}(s_k), \ldots, \bar{q_n}(s_n))]_{p'} = t_2$.
Necessarily $p = p'.k$, $\delta \in \Delta$, and $\delta = (f(q_1, \ldots, q_n) \to q')$. From the induction hypothesis $p' \in Pos(t)$ and $q' = \alpha(p')$. Then $t_2(p) = t_2(p'.k) = \hat{q_k}$. On the other hand, $\alpha(p) = \alpha(p'.k) = q_k$ because $\alpha(p') = q'$ and $\delta = (f(q_1, \ldots, q_n) \to q') \in \Delta$. Consequently $t_2(p) = \hat{\alpha(p)}$.

4. For all $p \in Pos(t)$, let us prove that there exists a derivation $\hat{q}(q(t_1)) \to^*_{R'_2} t_2$ s.t. $p \in Pos(t_2)$ and $t_2(p) = \hat{\alpha(p)}$. The proof is by induction on the length of the position p.

If $p = \epsilon$, then $\hat{q}(q(t_1)) \to^0_{R'_2} t_2$, i.e. $t_2 = \hat{q}(q(t_1))$ and $t_2(\epsilon) = \hat{q} = \hat{\alpha(\epsilon)}$.
Otherwise $p = p'.k$. From the induction hypothesis $\hat{q}(q(t_1)) \to^*_{R'_2} t'_2$ and $t'_2(p') = \hat{\alpha(p')}$.
We write $t'_2(p') = \hat{q'}$. Thus $t'_2|_{p'} = \hat{q'}(q'(\delta(s_1, \ldots, s_n))) \to_{R'_2} t_2 = f(\bar{q_1}(s_1), \ldots, \hat{q_k}(s_k), \ldots, \bar{q_n}(s_n))$ and $\delta = (f(q_1, \ldots, q_n) \to q')$. Then $t_2(p) = t_2(p'.k) = \hat{q_k}$. On the other hand, $\alpha(p) = \alpha(p'.k) = q_k$ because $\alpha(p') = q'$ and $\delta = (f(q_1, \ldots, q_n) \to q') \in \Delta$. Consequently $t_2(p) = \hat{\alpha(p)}$.

E Proof of Lemma 34

As long as no rule of R'_1 has been applied, the terms do not contain symbols of $\hat{Q} \cup \bar{Q} \cup \{j_k\}$. Then the derivation is of the form (A).

1) t_1 contains exactly one symbol of $\hat{Q}$, and the rules of $R'_2 \cup R'_3 \cup R'_4 \cup R'_5$ transform one symbol of $\hat{Q} \cup \{j_k\}$ into 1 or 0 symbol of $\hat{Q} \cup \{j_k\}$. On the other hand, since R'_5 removes j_k, then R'_5 can be applied at most once. So, since $t' \in T(\Sigma)$ does not contain elements of $\{j_k\}$, R'_5 is applied exactly once.

2) The rules of $R'_2 \cup R'_3 \cup R'_4$ are right-linear. Then the rewrite steps by $R'_2 \cup R'_3 \cup R'_4$ can be commuted between them. Note that the steps by R'_2 are necessarily before the step by R'_4, which is necessarily before the step by R'_5. On the other hand, R'_5 is not necessarily right-linear. However, since $top(t')$ is irreducible by R'_3, the rewrite steps by R'_3 can be achieved before the step by R'_5.

F Proof of Lemma 36

Let $RR = R' \backslash R'_1$. The derivation writes

$$top(t) \to^*_{RR} .(\to_{R'_1} . \to^*_{RR}) \ldots (\to_{R'_1} . \to^*_{RR}) \, top(t')$$

and contains at least one step by R'_1.
Since R'_0 is right-linear, within each $\to^*_{RR}$ we can move the steps by $R'_2 \cup R'_3 \cup R'_4 \cup R'_5$ before the steps by R'_0. Thus, each $\to^*_{RR}$ can be commuted into $\to^*_{R'_2 \cup R'_3 \cup R'_4 \cup R'_5} top(t_i) \to^*_{R'_0}$, and then $t_i \in T(\Sigma)$. Therefore we get:

$$top(t) \to^*_{R'_2 \cup R'_3 \cup R'_4 \cup R'_5} top(t_0) \, (\to^*_{R'_0} . \to_{R'_1} . \to^*_{R'_2 \cup R'_3 \cup R'_4 \cup R'_5}) \, top(t_1)$$
$$(\to^*_{R'_0} . \to_{R'_1} . \to^*_{R'_2 \cup R'_3 \cup R'_4 \cup R'_5}) \ldots (\to^*_{R'_0} . \to_{R'_1} . \to^*_{R'_2 \cup R'_3 \cup R'_4 \cup R'_5}) \, top(t')$$

and $t_0, \ldots, t_n \in T(\Sigma)$. Actually $top(t_0) = top(t)$ since $t \in T(\Sigma)$. From Lemma 35, $t \hookrightarrow^+_R t'$.

G Proof of Lemma 46

- If $t'' = a$ is a constant, $a \to_{R'_0} q'(\delta')$ with $\delta' = (a \to q')$. Note that $t' = \delta'$ is of sort $s'_{q'}$. Then $rm(t') = rm(\delta') = a = t''$.
- Otherwise $t'' = f(t''_1, \ldots, t''_n)$. However
$t'' \to^*_{R'_0} f(q'_1(t'_1), \ldots, q'_n(t'_n)) \to_{[\epsilon, R'_0]} q'(\delta'(q'_1(t'_1), \ldots, q'_n(t'_n)))$ with $\delta' = (f(q'_1, \ldots, q'_n) \to q')$.
Let us write $t' = \delta'(q'_1(t'_1), \ldots, q'_n(t'_n))$. Note that t' is of sort $s'_{q'}$. For each i, from the previous derivation we can extract $t''_i \to^*_{R'_0} q'_i(t'_i)$ and t'_i is of sort $s'_{q'_i}$. From the induction hypothesis $t''_i = rm(t'_i)$. Then $rm(t') = f(rm(t'_1), \ldots, rm(t'_n)) = f(t''_1, \ldots, t''_n) = t''$.

H Proof of Lemma 48

Because of Lemma 43, the derivation contains necessarily infinitely many steps by R'_1. Let us write $R'' = R'_0 \cup R'_2 \cup R'_3 \cup R'_4 \cup R'_5$.
The derivation writes $t'_0 \to^*_{R''} t'_{i_1} (\to_{R'_1} . \to^*_{R''}) t'_{i_2} (\to_{R'_1} . \to^*_{R''}) \cdots$. From Lemma 42, every term of the derivation is of sort $s_\top$. Therefore for each j, t'_{i_j} is of the form $t'_{i_j} = top(q_{i_j}(t_{i_j}))$, where $q_{i_j} \in Q_f$.
According to Lemma 47, for each j, $rm(t_{i_j}) \hookrightarrow_R rm(t_{i_{j+1}})$. Therefore there is an infinite derivation $rm(t_{i_1}) \hookrightarrow_R rm(t_{i_2}) \hookrightarrow_R \cdots$.