On Tree Automata that Certify Termination of Left-Linear Term Rewriting Systems

Alfons Geser[1]*, Dieter Hofbauer[2], Johannes Waldmann[3], and Hans Zantema[4]

[1] National Institute of Aerospace, 144 Research Drive,
Hampton, Virginia 23666, USA. Email: geser@nianet.org
[2] Mühlengasse 16, D-34125 Kassel, Germany.
Email: dieter@theory.informatik.uni-kassel.de
[3] Hochschule für Technik, Wirtschaft und Kultur (FH) Leipzig
Fb IMN, PF 30 11 66, D-04251 Leipzig, Germany. Email:
waldmann@imn.htwk-leipzig.de
[4] Faculteit Wiskunde en Informatica, Technische Universiteit Eindhoven
Postbus 513, 5600 MB Eindhoven, The Netherlands. Email: h.zantema@tue.nl

Abstract. We present a new method for proving termination of term rewriting systems automatically. It is a generalization of the match bound method for string rewriting. To prove that a term rewriting system terminates on a given regular language of terms, we first construct an enriched system over a new signature that simulates the original derivations. The enriched system is an infinite system over an infinite signature, but it is locally terminating: every restriction of the enriched system to a finite signature is terminating. We then construct iteratively a finite tree automaton that accepts the enriched given regular language and is closed under rewriting modulo the enriched system. If this procedure stops, then the enriched system is compact: every enriched derivation involves only a finite signature. Therefore, the original system terminates. We present three methods to construct the enrichment: top heights, roof heights, and match heights. Top and roof heights work for left-linear systems, while match heights give a powerful method for linear systems. For linear systems, the method is strengthened further by a forward closure construction. Using these methods, we give examples for automated termination proofs that cannot be obtained by standard methods.

1 Introduction

We present a new method for proving automatically that a term rewriting system (TRS) terminates on each term from a given regular term language. Our method consists of two steps. In the first step, we switch to an *enrichment* of the given TRS, i.e., a rewriting system over a different signature that simulates the original derivations. We consider enriched systems over infinite signatures that are *locally terminating*: every restriction to a finite signature is terminating. In the second

* Partly supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-97046 while this author was in residence at the NIA.

step, we compute a *compatible* finite tree automaton for this enrichment, i.e., a tree automaton that contains the enriched given regular tree language and is closed under rewriting modulo the enriched system. The existence of such a compatible automaton ensures that the enriched TRS is *compact*, i.e., every infinite derivation involves only a finite signature. By local termination of the enrichment, the automaton certifies termination of the original system.

We have previously applied this method to string rewriting [7]. The string rewriting version is implemented in the tools TORPA [18], Matchbox [17] and AProVE [11]. In the present paper, we describe how to extend it to term rewriting. Non-linearities in the TRS complicate both the termination arguments and the automata constructions. The algorithms we present are implemented in Matchbox.

The enrichments that we consider are variants of the original TRS in which the symbols are labelled by natural numbers. An enrichment is more powerful than another if the labels in the right-hand sides are smaller. We introduce three enrichments with increasing power: top heights, roof heights and match heights. For match heights, linearity of the TRS is required for the desired theorem to hold. So for linear TRSs the best results are obtained by choosing the enrichment based on match heights, and for non-right-linear TRSs the best results are obtained by choosing the enrichment based on roof heights.

For linear systems, uniform termination can be concluded from termination on a restricted set of initial terms: the set of right-hand sides of forward closures. We use our method both to compute this set, and to prove termination on it, at the same time.This turns out to be more powerful than applying the method directly for the original system and the set of all terms. To our knowledge, this is the first method that computes finite representations of infinite sets of right-hand sides of forward closures on TRSs.

The paper is organized as follows. In Section 3 we define enrichments, give three instances, and compare them. In Section 4 we define compatible tree automata and in Section 5 we discuss how to construct them. Section 6 presents the simulation of forward closures by rewriting, while Section 7 shows how to implement this with an automata construction.

A preliminary version of this paper has been presented at the 7th International Workshop on Termination, Aachen 2004 [8].

2 Preliminaries

For a relation ρ on a set T and $t \in T$ write $\infty(t, \rho)$ if there is an infinite sequence $t_0, t_1, \ldots$ over T where $t = t_0$ and $t_i \ \rho \ t_{i+1}$ for every $i \geq 0$, and abbreviate $\neg\infty(t, \rho)$ by $\mathrm{SN}(t, \rho)$. Define $\mathrm{SN}(S, \rho)$ for $S \subseteq T$ by $\forall s \in S : \mathrm{SN}(s, \rho)$; then ρ is *terminating (or: strongly normalizing) on* S. Let $\mathrm{SN}(\rho)$ stand for $\mathrm{SN}(T, \rho)$. Analogously write $\infty(\rho)$ for $\neg(\mathrm{SN}(\rho))$. The reflexive closure of ρ is $\rho^{=}$, the composition of two relations $\rho \subseteq A \times B$ and $\sigma \subseteq B \times C$ is $\rho \circ \sigma = \{(a, c) \mid \exists b \in B : (a, b) \in \rho, (b, c) \in \sigma\}$.

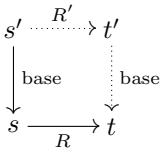
For standard notations on term rewriting see [1, 19], for instance. Throughout we fix a signature Σ, a set of variables X, and consider term rewriting systems $R \subseteq \mathcal{T}_\Sigma(X) \times \mathcal{T}_\Sigma(X)$. Unless otherwise stated, signatures and rewriting systems are finite. The set of left- and right-hand sides of R are denoted by $\mathrm{lhs}(R)$ and $\mathrm{rhs}(R)$ respectively. Since our topic is termination, we assume $\mathrm{lhs}(R) \cap X = \emptyset$, and $X(r) \subseteq X(\ell)$ for rules $\ell \to r$. Let $X(t) \subseteq X$ denote the set of variables that occur in $t \in \mathcal{T}_\Sigma(X)$, and let $X(T) = \bigcup_{t \in T} X(t)$ for $T \subseteq \mathcal{T}_\Sigma(X)$. For a mapping $h : \mathcal{T}_\Sigma(X) \to \mathcal{T}_\Gamma(X)$ define the term rewriting system $h(R) = \{h(\ell) \to h(r) \mid (\ell \to r) \in R\}$ over signature Γ. For the symbol at position p in term t we write $t(p)$. For $Y \subseteq \Sigma \cup X$ let $\mathrm{Pos}_Y(t)$ be the set of positions p such that $t(p) \in Y$. We use $<$ for the prefix ordering on positions. The set of *descendants* modulo R of a tree language $L \subseteq \mathcal{T}_\Sigma$ is $\to_R^*(L) = \{s \in \mathcal{T}_\Sigma \mid \exists t \in L : t \to_R^* s\}$.

The domain and the range of a substitution $\alpha : X \to \mathcal{T}_\Sigma(X)$ are $\mathrm{dom}(\alpha) = \{x \in X \mid x\alpha \neq x\}$ and $\mathrm{ran}(\alpha) = \{x\alpha \mid x \in \mathrm{dom}(\alpha)\}$. For $Y \subseteq \mathrm{dom}(\alpha)$ let $\alpha|_Y$ be the substitution with domain Y where $\alpha|_Y : x \mapsto x\alpha$ for $x \in Y$, $\alpha|_Y : x \mapsto x$ otherwise. For substitutions α and α' we write $\alpha \to_R \alpha'$ if $\mathrm{dom}(\alpha) = \mathrm{dom}(\alpha')$, and $x\alpha \to_R x\alpha'$ for some $x \in \mathrm{dom}(\alpha)$ and $y\alpha = y\alpha'$ for every $y \neq x$.

A *tree automaton* $A = (Q, \Sigma, F, T)$ over a signature Σ consists of a set Q of constant symbols, disjoint from Σ, called *states*; a set $F \subseteq Q$ of *final states*; and a ground rewriting system T over $\Sigma \cup Q$ with rules (*transitions*) of the form $q_0 \to q$ or $f(q_1, \ldots, q_n) \to q$ for n-ary $f \in \Sigma$, $n \geq 0$, and $q_0, \ldots, q_n, q \in Q$. The automaton is *finite* if T is finite, and it is *deterministic* if T is non-overlapping. The *language accepted* by A is $\mathcal{L}(A) = \{t \in \mathcal{T}_\Sigma \mid \exists q \in F : t \to_T^* q\}$. For more on tree languages we refer to [5, 2].

3 Enrichments of Rewriting Systems

Definition 1. *A TRS R' over a signature Σ' is an* enrichment *of a TRS R over a signature Σ if there is a mapping* base : $\mathcal{T}_{\Sigma'} \to \mathcal{T}_\Sigma$ *such that every R-derivation step can be lifted to an R'-derivation step: for each step $s \to_R t$ and each $s' \in \mathrm{base}^{-1}(s)$ there is some $t' \in \mathrm{base}^{-1}(t)$ with $s' \to_{R'} t'$.*



We use enrichments to propagate termination properties:

Proposition 1. *Let R and R' be TRSs over Σ and Σ' resp., let $L \subseteq \mathcal{T}_\Sigma$ and $L' \subseteq \mathcal{T}_{\Sigma'}$. If R' is an enrichment of R via* base : $\mathcal{T}_{\Sigma'} \to \mathcal{T}_\Sigma$, *and $L \subseteq \mathrm{base}(L')$, then termination of R' on L' implies termination of R on L.*

Suitable enrichments will satisfy the following property:

Definition 2. *A finite or infinite TRS R over a finite or infinite signature Σ is called* locally terminating *if every restriction of R to a finite signature $\Gamma \subseteq \Sigma$ is terminating: $R \cap (\mathcal{T}_\Gamma(X) \times \mathcal{T}_\Gamma(X))$ is terminating on $\mathcal{T}_\Gamma$.*

In the following, we will present three enrichments that are locally terminating, one of them under a suitable linearity restriction. We choose the enriched

signature $\Sigma' = \Sigma \times \mathbb{N}$, and call the numbers *heights*. We often write f_h for (f, h). Define the mappings base : $\Sigma' \to \Sigma$, height : $\Sigma' \to \mathbb{N}$, and $\mathrm{lift}_h : \Sigma \to \Sigma'$ by

$$\mathrm{base} : (f, h) \mapsto f, \qquad \mathrm{height} : (f, h) \mapsto h, \qquad \mathrm{lift}_h : f \mapsto (f, h),$$

which are extended pointwise to term morphisms. E.g., $\mathrm{lift}_2(f(x, a)) = f_2(x, a_2)$ where a is a constant symbol, and x is a variable. We will use one fixed ordering on Σ', called the *height ordering*, given by $(f, h) < (f', h')$ iff $h > h'$. This ordering is well-founded when restricted to finite sets.

The enrichments label symbols in the right-hand side of a rule with the successor of the minimum of the heights of all symbols at a specified subset of positions in the left-hand side:

Definition 3. *For a term rewriting system R over Σ, and a function f that maps a rewriting rule $(\ell \to r)$ to a nonempty subset of* $\mathrm{Pos}_\Sigma(\ell)$*, we define the* f-cover *of R to be the term rewriting system over $\Sigma \times \mathbb{N}$ given by*

$$\mathrm{cover}_f(R) = \{\ell' \to \mathrm{lift}_h(r) \mid\ (\ell \to r) \in R,\ \mathrm{base}(\ell') = \ell, \\ h = 1 + \min\{\mathrm{height}(\ell'(p)) \mid p \in f(\ell, r)\}\}.$$

Note that $\mathrm{cover}_f(R)$ is indeed an enrichment of R.

To present the enrichments, we need one auxiliary definition:

Definition 4. *A position $p \in \mathrm{Pos}_\Sigma(t)$ is a* roof position *in $t \in \mathcal{T}_\Sigma(X)$ for a set of variables $Y \subseteq X$ if for each $y \in Y$ there is $q \in \mathrm{Pos}_{\{y\}}(t)$ such that $p < q$. Let* $\mathrm{RPos}_Y(t)$ *denote the set of all roof positions in t for Y.*

E.g., term $t = f(f(x, g(y)), a)$ has $\mathrm{RPos}_{\{y\}}(t) = \{\epsilon, 1, 12\}$ and $\mathrm{RPos}_{\{x\}}(t) = \mathrm{RPos}_{\{x,y\}}(t) = \{\epsilon, 1\}$, so position 12 of g is not a roof position for $\{x\}$ or $\{x, y\}$. Also for $s = f(f(x, g(y)), x)$ we get $\mathrm{RPos}_{\{x\}}(s) = \mathrm{RPos}_{\{x,y\}}(s) = \{\epsilon, 1\}$.

Now we define the enrichments that we will use in the rest of this paper.

Definition 5.
- *The* top enrichment $\mathrm{top}(R)$ *is* $\mathrm{cover}_f(R)$ *for* $f(\ell, r) = \{\epsilon\}$.
- *The* roof enrichment $\mathrm{roof}(R)$ *is* $\mathrm{cover}_f(R)$ *for* $f(\ell, r) = \mathrm{RPos}_{X(r)}(\ell)$.
- *The* match enrichment $\mathrm{match}(R)$ *is* $\mathrm{cover}_f(R)$ *for* $f(\ell, r) = \mathrm{Pos}_\Sigma(\ell)$.

Example 1. Take $R = \{s(x) + 0 \to s(x)\}$. Then $\mathrm{top}(R)$ contains, among others, the rule $s_1(x) +_2 0_0 \to s_3(x)$, since 2 is the height of the top symbol $+_2$. The system $\mathrm{roof}(R)$ contains the rule $s_1(x) +_2 0_0 \to s_2(x)$, since 1 is the minimal height of a roof symbol (and 0_0 is not in roof position). Finally, $\mathrm{match}(R)$ contains the rule $s_1(x) +_2 0_0 \to s_1(x)$, since 0_0 has minimal height.

Lemma 1. *For a term rewriting system R, both the systems* $\mathrm{top}(R)$ *and* $\mathrm{roof}(R)$ *are locally terminating.*

Proof. $\mathrm{top}(R)$ and $\mathrm{roof}(R)$ are ordered by the recursive path ordering induced by the height ordering on Σ', which is well-founded for finite signatures. □

Lemma 2. *For a right-linear term rewriting system R, the system* $\mathrm{match}(R)$ *is locally terminating.*

Proof. To each term in a ground match(R)-derivation assign the multiset of its symbols. By right-linearity, this sequence of multisets is decreasing with respect to the height ordering on Σ'. □

Remark 1. Right-linearity is essential, as shown by the non-terminating system $\{f_1(a_0, x) \to f_1(x, x)\} \subseteq \mathrm{match}(\{f(a, x) \to f(x, x)\})$.

Definition 6. *For $e \in \{\mathrm{top}, \mathrm{roof}, \mathrm{match}\}$, a term rewriting system R over Σ is called* e-bounded *by $c \in \mathbb{N}$ for a language L over Σ if the maximal height occurring in $\to^*_{e(R)}(\mathrm{lift}_0(L))$ is at most c.*

Definition 7. *A finite or infinite term rewriting system R over a finite or infinite signature Σ is said to be* compact *for a language $L \subseteq \mathcal{T}_\Sigma$ if there exists a finite subset $\Gamma \subseteq \Sigma$ such that $\to^*_R(L) \subseteq \mathcal{T}_\Gamma$.*

Lemma 3. *If a finite or infinite term rewriting system R is locally terminating and compact for $L \subseteq \mathcal{T}_\Sigma$, then R is terminating on L.*

Obviously $e(R)$ is compact for every e-bounded TRS R. Together with Lemmas 1 and 2 we get:

Proposition 2.
- *If R is top-bounded for L, then R is terminating on L.*
- *If R is roof-bounded for L, then R is terminating on L.*
- *If R is right-linear and match-bounded for L, then R is terminating on L.*

Remark 2. All the enrichments discussed here are obtained as covers (Definition 3). This has two implications: since we take the minimum, each enrichment is monotonic (pointwise domination of heights is preserved by parallel derivations), and since the respective sets of positions are comparable by set inclusion the enrichments are comparable as well: for corresponding derivations, match heights are lower or equal to roof heights, and these are lower or equal to top heights. So we prefer roof-heights to top-heights in general, and we will use match-heights for right-linear systems.

Remark 3. Results on derivation lengths carry over from $\mathrm{cover}_f(R)$ to R. For instance, for right-linear systems R, every restriction of match(R) to a finite signature has linear derivational complexity, so the same complexity holds for every match-bounded right-linear system R. In contrast, for top-bounded R we can have (single) exponential complexity, as for the system $\{f(x) \to g(x, x)\}$ which is top-bounded by 1.

Remark 4. The correspondence between R and its enrichment R' is a *rewrite labelling* (with lift_0 as the initial labelling function) as defined by van Oostrom and de Vrijer [16], Section 8.4. They mention an earlier example of a labelling with the property that "bounded reductions are finite": the *Hyland-Wadsworth labelling* of a rewriting system R is defined just like match(R), with the only difference of taking max instead of min in Definition 3.

Remark 5. In the string rewriting case, which can be seen as a particular form of linear term rewriting, all non-variable positions are roof positions, therefore match(R) and roof(R) coincide. Match-boundedness and top-boundedness differ, as the example $\{ab \to a\}$ shows, which is match-bounded by 1, but not top-bounded.

4 Compatible Tree Automata

Definition 8. *We call a tree automaton $A = (Q, \Sigma, F, T)$* compatible *with a term rewriting system R over Σ and a language L over Σ if $L \subseteq \mathcal{L}(A)$, and for each rule $(\ell \to r) \in R$, for each state $q \in Q$, and for each substitution $\sigma : X(\ell) \to Q$, we have that $\ell\sigma \to_T^* q$ implies $r\sigma \to_T^* q$.*

Remark 6. We can decide compatibility of A with R and L in case A and R are finite, and L is given by a finite tree automaton, by just enumerating all cases.

A compatible automaton is closed under left-linear rewriting.

Lemma 4. *If A is compatible with R and L, and R is left-linear, then $\to_R^*(L) \subseteq \mathcal{L}(A)$.*

Proof. We show that R-derivations are covered "step by step" in A: if $t_1 \in \mathcal{L}(A)$ and $t_1 \to_R t_2$, then $t_2 \in \mathcal{L}(A)$. Let $t_1 = t_1[\ell\sigma]_p \to_R t_1[r\sigma]_p = t_2$ for some rule $\ell \to r$, position p, and substitution $\sigma : X(\ell) \to \mathcal{T}_\Sigma$. Since $t_1 \in \mathcal{L}(A)$, there is a state q, a final state $\bar{q}$, and a substitution $\rho : X(\ell) \to Q$ such that $t_1 = t_1[\ell\sigma]_p \to_T^* t_1[\ell\rho]_p \to_T^* t_1[q]_p \to_T^* \bar{q}$. Note that ρ exists as R is left-linear. From $\ell\rho \to_T^* q$ and compatibility of A with R we get $r\rho \to_T^* q$. This implies $t_2 = t_1[r\sigma]_p \to_T^* t_1[r\rho]_p \to_T^* t_1[q]_p \to_T^* \bar{q}$, thus $t_2 \in \mathcal{L}(A)$. □

The requirement of left-linearity in Lemma 4 cannot be dropped, as the following example shows.

Example 2. We take an automaton A with states $Q = \{1, 2, 3\}$ and transitions $a \to 1$, $a \to 2$, $f(1, 2) \to 3 \in F$. Then $\mathcal{L}(A) = \{f(a, a)\}$. This automaton is compatible with the rewriting system $R = \{f(x, x) \to b\}$ since there are *no* rule $(\ell \to r) \in R$, state q and substitution $\sigma : X(\ell) \to Q$ with $\ell\sigma \to_T^* q$. On the other hand, A is not closed under rewriting, as $\to_R^*(\mathcal{L}(A)) = \{f(a, a), b\}$.

The premise "R is left-linear" in Lemma 4 may be exchanged with "A is deterministic". We don't follow on this branch in the present paper.

By Lemma 4 we get

Lemma 5. *If R is left-linear, and there is some finite automaton A that is compatible with R and L, then R is compact for L.*

5 Constructing Compatible Automata

The following obvious procedure yields an automaton $A = (\Sigma, Q, F, T)$ that is compatible with a rewriting system R and a regular tree language L whenever the procedure terminates:

Start with an automaton A_0 that accepts L;
$A := A_0$;
while A is not compatible
    choose $q \in Q$, $(\ell \to r) \in R$, $\sigma : X(\ell) \to Q$
        such that $\ell\sigma \to_T^* q$ and $r\sigma \not\to_T^* q$;
    add new states and transitions to A
        yielding a new automaton A' with transitions T'
        such that $r\sigma \to_{T'}^* q$;
    $A := A'$;

The interesting issue is the *strategy*: exactly how new states and transitions are chosen. The straightforward strategy is to add a new state for each proper subterm of $r\sigma$ that is not in Q, and fill in the corresponding transitions.

Example 3. For the automaton A with transitions $\{a \to 0, b \to 1, f(0,1) \to 1\}$, and the rewriting system $R = \{\ell \to r\} = \{f(x,y) \to g(h(y),x)\}$ we have $\ell\sigma \to_T^* q$ for $q = 1$ and $\sigma = \{x \mapsto 0, y \mapsto 1\}$. Transitions and states have to be added such that $r\sigma = g(h(1),0) \to_T^* 1 = q$. We add one new state 2, corresponding to the subterm $h(1)$ of $r\sigma$, and transitions $\{h(1) \to 2, g(2,0) \to 1\}$.

The straightforward strategy is the basic idea behind automata closure constructions for various syntactically restricted classes of rewriting systems, e.g., ground, (generalized) (semi)-monadic, finite path overlapping systems. In each case, the syntactic restriction ensures that only finitely many states and transitions will be added.

We cannot generally avoid the addition of states. Therefore the completion procedure for tree automata need not stop. Indeed there are rewriting systems R, as in Example 5, for which the set of descendants is not regular. In such a case, we try to over-approximate the set of descendants by a compatible tree automaton. Genet [6] gets such an approximation by limiting the number of states that are added to the automaton during completion.

We follow a more simplistic approach here that works well for match-bounded string rewriting [9, 18]. We avoid generating some of the additional states as follows. If $r\sigma \not\to_T^* q$, we look for a context $D[\Box]$, a context $C[\Box, \ldots, \Box]$, and terms $t_1, \ldots, t_n \in \mathcal{T}_\Sigma(Q)$ such that $D[C[t_1, \ldots, t_n]] = r\sigma$. Suppose that $D[q_0] \to_T^* q$ for some state q_0, and $t_i \to_T^* q_i$ for states q_i, $1 \le i \le n$. Then we add a fresh state for each non-leaf, non-root subterm of $C[\Box, \ldots, \Box]$, and transitions such that $C[q_1, \ldots, q_n] \to_{T'}^* q_0$. In this way, we re-use states that occur in the derivations $D[q_0] \to_T^* q$ and $t_i \to_T^* q_i$.

This is a non-deterministic procedure. Our implementation chooses in each step one such context $C[\Box, \ldots, \Box]$ that requires the least number of new states.

For instance, take $R = \{\ell \to r\} = \{b(a(x)) \to c(b(x))\}$, and let A be a two-state automaton with transitions $T = \{e \to 0, a(0) \to 0, b(0) \to 1\}$, state 1 being final. Here, $\mathcal{L}(A) = b(a^*(e))$. Now we have $\ell\{x \mapsto 0\} = b(a(0)) \to_T 1$, but $r\{x \mapsto 0\} = c(b(0)) \not\to_T 1$. Here, $c(b(0)) = D[C[t_1]]$ for $D[\Box] = \Box$, $C[\Box] = c(\Box)$, and $t_1 = b(0)$. We have $t_1 \to_T^* 1$, so we add no new state (as $C[\Box]$ has no

non-trivial subterms), but the transition $c(1) \to 1$. The new automaton is now compatible with R and $\mathcal{L}(A)$, and it accepts $\to_R^*(\mathcal{L}(A)) = c^*(b(a^*(e)))$.

Note that a compatible automaton obtained this way may be an over-approximation of the set of descendants:

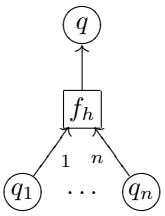
Example 4. Let $R = \{a(c) \to b(c)\}$, and $L = \{a(c), a(d)\}$ accepted by the automaton with states $\{0, 1\}$, state 1 being final, and transitions $\{c \to 0,\ d \to 0,\ a(0) \to 1\}$. The rewrite rule matches in state 1, so we have to ensure $b(c) \to^* 1$. This could be done by adding a new state 2 and transitions $c \to 2$, $b(2) \to 1$. As $c \to^* 0$, we might want to avoid state 2 and instead add the single transition $b(0) \to 1$. But then $b(d) \to^* 1$ as well, so the automaton now accepts $\{a(c), a(d), b(c), b(d)\}$, which is a proper superset of $\to_R^*(L) = \{a(c), a(d), b(c)\}$.

For string rewriting, match-boundedness implies preservation of regularity of languages [7]. As the following example shows, the corresponding property does not hold for term rewriting.

Example 5. The system $R = \{g(f(x, y)) \to f(h(x), h(y))\}$ is top-bounded by 1. However, the language $\to_R^*(L) \cap f(h^*(a), h^*(a)) = \{f(h^n(a), h^n(a)) \mid n \geq 0\}$ is not regular for the regular language $L = g^*(f(a, a))$, so $\to_R^*(L)$ is not regular either.

So contrary to the string rewriting case, there is no *exact* construction for the sets of descendants of a regular language modulo top-bounded term rewriting. Note that the same holds for roof- and match-bounded rewriting, since these heights are majorized by top heights.

We conclude this section with a few examples that illustrate our approach. In order to visualize tree automata, a transition $f_h(q_1, \ldots, q_n) \to q$ is graphically represented as the hyperedge in the illustration at the right. Squares contain function symbols with height annotations as subscripts, where the argument ordering is indicated by numbers at the incoming arrows. Circles denote states, and double circles denote final states.



Example 6. For $R = \{f(x, f(a, a)) \to f(f(x, a), x)\}$ over $\{f, a\}$ we present the construction that proves that R is top-bounded by 3. We have to find an automaton A that is compatible with $\mathrm{top}(R)$ and $\mathrm{lift}_0(L)$. We start with the automaton $A_0 = (\Sigma \times \mathbb{N}, \{0\}, \{0\}, T_0)$ where $T_0 = \{a_0 \to 0,\ f_0(0, 0) \to 0\}$, which accepts $\mathrm{lift}_0(\mathcal{T}_\Sigma)$. Now we have a derivation

$$f_0(0, f_0(a_0, a_0)) \to_{T_0}^* 0,$$

starting with a redex of the rule $f_0(x, f_0(a_0, a_0)) \to f_1(f_1(x, a_1), x)$ from $\mathrm{top}(R)$. The automaton A_0 is not compatible, for $f_1(f_1(0, a_1), 0) \not\to_{T_0}^* 0$. There are no states we could re-use, so our first step follows the straightforward strategy: to add the new states 1 and 2, corresponding to the subterms $f_1(0, a_1)$ and a_1, respectively, and the rules

$$a_1 \to 2,\ f_1(0, 2) \to 1,\ f_1(1, 0) \to 0.$$

This way we get another automaton, $A_1 = (\Sigma \times \mathbb{N}, \{0,1,2\}, \{0\}, T_1)$, where $T_1 = T_0 \cup \{a_1 \to 2,\ f_1(0,2) \to 1,\ f_1(1,0) \to 0\}$, such that $f_1(f_1(0,a_1),0) \to_{T_1} f_1(f_1(0,2),0) \to_{T_1} f_1(1,0) \to_{T_1} 0$. For the new automaton, we again look for violations of compatibility: We have the redex match

$$f_1(1, f_0(a_0, a_0)) \to^*_{T_1} 0$$

with the rule $f_1(x, f_0(a_0, a_0)) \to f_2(f_2(x, a_2), x)$ in $\mathrm{top}(R)$. So A_1 is not compatible, for $f_2(f_2(1,a_2),1) \not\to^*_{T_1} 0$. According to the straightforward strategy, we add states 3 and 4, corresponding to the subterms $f_2(1,a_2)$ and a_2, respectively, and transitions to A_1. We get $A_2 = (\Sigma \times \mathbb{N}, \{0,\dots,4\}, \{0\}, T_2)$ where

$$T_2 = T_1 \cup \{a_2 \to 4,\ f_2(1,4) \to 3,\ f_2(3,1) \to 0\},$$

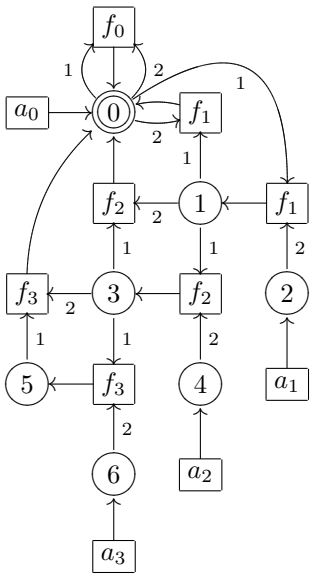
and $f_2(f_2(1,a_2),1) \to^*_{T_2} 0$ as wanted. For A_2 again there is a redex

$$f_2(3, f_1(a_0,a_1)) \to^*_{T_2} 0,$$

but $f_3(f_3(3,a_3),3) \not\to^*_{T_2} 0$. We add states 5 and 6 for $f_3(3,a_3)$ and a_3, and transitions

$$a_3 \to 6,\ f_3(3,6) \to 5,\ f_3(5,3) \to 0.$$

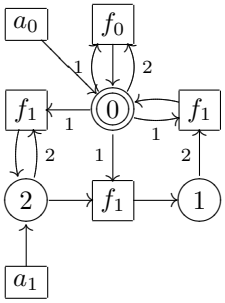
The resulting automaton is displayed at the right; it is compatible with $\mathrm{top}(R)$ and $\mathrm{lift}_0(\mathcal{T}_\Sigma)$. By Remark 6 we can check compatibility with $\mathrm{top}(R)$ restricted to the signature $\Sigma \times \{0,1,2,3,4\}$, being a finite system. Since heights ≥ 4 do not occur, it is even compatible with the infinite system $\mathrm{top}(R)$. So R is top-bounded by 3 as claimed, and thus terminating.



Example 7. For $R = \{f(f(x,a),a) \to f(x,f(x,a))\}$ we can easily show that it is not top-bounded, as we have the derivation

$$t_{n+2} = f(f(t_n,a),a) \to_R f(t_n, f(t_n,a)) = f(t_n, t_{n+1})$$

where $t_0 = a$ and $t_{n+1} = f(t_n,a)$. Thus by induction, $t_{n+2} \to^*_R f(t_n, f(t_{n-1}, \dots f(a,a)\dots))$. This derivation reaches top height $n+1$. However, R is roof-bounded by 1, as the compatible automaton to the right reveals.



Example 8. Let $R = \{f(a, f(x,a)) \to f(a, f(f(a,a),x))\}$. As before, we start with the one-state automaton with transitions $\{a_0 \to 0, f_0(0,0) \to 0\}$, accepting $\mathrm{lift}_0(\mathcal{T}_\Sigma)$. There is a redex match $f_0(a_0, f(0,a_0)) \to f_0(0,0) \to 0$ so we have to ensure that $f_1(a_1, f_1(f_1(a_1,a_1),0)) \to^* 0$. This is done by adding states

$\{1,2,3,4,5\}$ and transitions $\{a_1 \to 1, a_1 \to 4, a_1 \to 5, f_1(4,5) \to 3, f_1(3,0) \to 2, f_1(1,2) \to 0\}$. This produces another redex $f_1(a_1, f_1(3,a_0)) \to^* f_1(1,2) \to 0$ which requires $f_1(a_1, f_1(f_1(a_1,a_1),3)) \to^* 0$. Note that a_0 in the redex has minimal height, and thus the labels in the contractum are 1. This requirement can be fulfilled by adding the transition $f_1(3,3) \to 2$, since then

$$f_1(a_1, f_1(f_1(a_1,a_1),3)) \to^* f_1(1, f_1(f_1(4,5),3)) \to f_1(1, f_1(3,3)) \to f_1(1,2) \to 0.$$

Note that this is a state re-use corresponding to the choice $r\sigma = D[C[t_1,t_2]]$ with $t_1 = f_1(a_1,a_1) \to^* 3 = q_1$, $t_2 = 3 = q_2$, $C = f_1(\Box,\Box)$, $D = f_1(a_1,\Box)$, $q_0 = 2$, $D[q_0] \to^* 0 = q$. The resulting automaton is compatible with match(R) and lift$_0(\mathcal{T}_\Sigma)$, thus R is match-bounded by 1, and therefore terminating.

In general, if we want a compatible automaton for $L = \mathcal{T}_\Sigma$, then we may simply start with the automaton for L that has just one state q, which is also final, and for each symbol $f \in \Sigma$, a transition $f(q,\dots,q) \to q$. Doing so fails to distinguish between symbols, and this might cause non-termination of completion. In such cases it is better to "split" the automaton. We then take $Q = F = \{q_f \mid f \in \Sigma\}$, and transitions $\{f(q_1,\dots,q_n) \to q_f \mid f \in \Sigma, q_i \in Q\}$.

For the following example a termination proof via top heights can be obtained by completion starting with the split automaton, but not starting with the one-state automaton.

Example 9. (`AProVE-forward_instantiation2`, [14]) Consider the system

$$R = \{f(x,y,z) \to g(x,y,z),\ g(d,e,x) \to f(x,x,x),\ a \to b,\ a \to c\}.$$

over signature $\Sigma = \{a,b,c,d,e,f,g\}$. Completion does not stop if we start with an automaton for lift$_0(\mathcal{T}_\Sigma)$ with only one state q: We find a redex $f_0(q,q,q) \to q$, so we have to add the transition $g_1(q,q,q) \to q$. Since $g_1(d,e,q) \to^2_R g_1(q,q,q) \to q$, we have to add $f_2(q,q,q) \to q$, and so forth, creating symbols g_3, f_4, g_5, ...

Completion does succeed if we start with a split automaton. It has 7 states in $Q = \{q_a,q_b,q_c,q_d,q_e,q_f,q_g\}$, and all symbols are labelled by 0. Because of the rules $\{a \to b, a \to c\}$, we add transitions $b_1 \to q_a, c_1 \to q_a$. Due to the rule $f(x,y,z) \to g(x,y,z)$ we add 7^3 transitions $\{g_1(q_x,q_y,q_z) \to q_f \mid q_x,q_y,q_z \in Q\}$, and due to rule $g(d,e,x) \to f(x,x,x)$ we add $\{f_2(q,q,q) \to q_g \mid q \in Q\}$. Rule $f(x,y,z) \to g(x,y,z)$ entails the transitions $\{g_3(q,q,q) \to q_f \mid q \in Q\}$. The result is a compatible automaton, so 3 is a top bound, and R is terminating.

6 Simulating Forward Closures by Rewriting

Forward closures [13] can be used to characterize uniform termination by termination on a restricted set of terms: For a right-linear [4] or non-overlapping [10] term rewriting system R, termination is equivalent to termination on the set RFC(R) of right-hand sides of forward closures.

Following [10], we inductively define the set RFC(R) as the least subset of $\mathcal{T}_\Sigma(X)$ that contains rhs(R), is closed under renaming of variables, and satisfies the condition:

– if $t \in \mathrm{RFC}(R)$, $p \in \mathrm{Pos}_\Sigma(t)$, $(\ell \to r) \in R$, ℓ variable-disjoint with t, μ a most general unifier of $t|p$ and ℓ, then $(t[r]_p)\mu \in \mathrm{RFC}(R)$.

Next, we will show how to simulate the construction of $\mathrm{RFC}(R)$ by ordinary rewriting. Note that since we simulate unification by matching, we cannot cope with non-linearity in left- or right-hand sides. So for the rest of this section we consider *linear* rewrite rules only, i.e., rules with linear left- and right-hand sides.

Let $C \in \mathcal{T}_\Sigma(X)$ be linear, and let $\alpha : X \to \mathcal{T}_\Sigma(X)$ be a substitution with $\mathrm{dom}(\alpha) \subseteq X(C)$. We say that (C, α) is a *factorization* of $t \in \mathcal{T}_\Sigma(X)$ if $t = C\alpha$, and we call C the *context* of the factorization. The factorization is *non-trivial* if $C \notin X$, $\mathrm{dom}(\alpha) \neq \emptyset$, and $x\alpha \notin X$ for every $x \in \mathrm{dom}(\alpha)$. In order to mark the border between the context and the substitution of a factorization, we use the constant $\sharp$, not contained in Σ. Let the substitution $\sigma_\sharp : X \to \mathcal{T}_{\Sigma\cup\{\sharp\}}$ be defined by $\sigma_\sharp : x \mapsto \sharp$ for $x \in X$.

Definition 9. *For a linear TRS R define the TRS $R_\sharp$ over $\Sigma \cup \{\sharp\}$ by*

$$R_\sharp = R \cup \{C\sigma'\alpha \to r\sigma'' \mid \ell \to r \in R,\ (C,\alpha) \text{ a non-trivial factorization of } \ell,\ \sigma' = \sigma_\sharp|_{\mathrm{dom}(\alpha)} \text{ and } \sigma'' = \sigma_\sharp|_{X(\mathrm{ran}(\alpha))}\}.$$

That is, $R_\sharp$ consists of all rules obtained in the following way. If $\ell \to r$ is a rule in R and (C, α) is a non-trivial factorization of ℓ, then $\ell' \to r'$ is a rule in $R_\sharp$, where ℓ' is obtained from C by replacing every variable in $\mathrm{dom}(\alpha)$ by $\sharp$, and r' is obtained from r by replacing every variable that occurs in $\mathrm{ran}(\alpha)$ by $\sharp$. Note that $R_\sharp$ is linear, and that $X(r) \subseteq X(\ell)$ for each rule $\ell \to r$ in $R_\sharp$.

Example 10. For $R = \{g(f(h(x), h(y))) \to f(y, x)\}$, the system $R_\sharp$ consists of R together with the rules

$$g(\sharp) \to f(\sharp,\sharp), \qquad g(f(\sharp,\sharp)) \to f(\sharp,\sharp),$$
$$g(f(\sharp, h(y))) \to f(y,\sharp), \qquad g(f(h(x),\sharp)) \to f(\sharp, x).$$

For $R = \{g(x) \to x\}$ we get $R_\sharp = R$, for $R = \{g(g(x)) \to x\}$ we have $R_\sharp = R \cup \{g(\sharp) \to \sharp\}$, and for $R = \{g(g(x)) \to a\}$ we obtain $R_\sharp = R \cup \{g(\sharp) \to a\}$.

Replacing variables by $\sharp$, we can now characterize $\mathrm{RFC}(R)$ as the set of descendants modulo $R_\sharp$ of $\mathrm{rhs}(R)$, provided R is linear:

Lemma 6. *If R is linear then* $\mathrm{RFC}(R)\sigma_\sharp = \to^*_{R_\sharp}(\mathrm{rhs}(R)\sigma_\sharp)$.

Abbreviate $\mathrm{RFC}(R)\sigma_\sharp$ by $\mathrm{RFC}_\sharp(R)$. A self-contained proof of the following theorem can be found in the appendix.

Theorem 1. *Let R be a linear term rewriting system. Then*

$$\mathrm{SN}(\to_R) \text{ if and only if } \mathrm{SN}(\mathrm{RFC}_\sharp(R), \to_R).$$

Corollary 1. *If a linear term rewriting system R is match-bounded for* $\mathrm{RFC}_\sharp(R)$*, then R is terminating.*

Proof. If R is match-bounded for $\mathrm{RFC}_\sharp(R)$ then R is terminating on $\mathrm{RFC}_\sharp(R)$ by Proposition 2, thus terminating by Theorem 1. □

Remark 7. In general we do not have $\mathrm{SN}(\to_R)$ iff $\mathrm{SN}(\mathrm{rhs}(R)\sigma_\sharp, \to_{R_\sharp})$. As a counter-example consider the terminating system $R = \{g(g(x)) \to g(x)\}$. Here, $R_\sharp = R \cup \{g(\sharp) \to g(\sharp)\}$ is non-terminating on $\mathrm{rhs}(R)\sigma_\sharp = \{g(\sharp)\}$.

Theorem 1 cannot be generalized to left-linear and non-overlapping systems:

Example 11. For $R = \{f(a, x) \to f(x, x)\}$ we get $R_\sharp = R \cup \{f(\sharp, x) \to f(x, x)\}$. Obviously, R is terminating on $\mathrm{RFC}_\sharp(R) = \to^*_{R_\sharp}(\{f(\sharp, \sharp)\}) = \{f(\sharp, \sharp)\}$, but not terminating.

7 Compatible Automata and Forward Closures

According to Corollary 1, termination of R can be shown by verifying that R is match-bounded for $\mathrm{RFC}_\sharp(R)$. Literally following the definition, we would first construct an automaton A with $\mathrm{RFC}_\sharp(R) \subseteq \mathcal{L}(A)$, that is, A should be compatible with $R_\sharp$ and $\mathrm{rhs}(R)\sigma_\sharp$. Then we find an automaton A' that is compatible with some enrichment R' of R, and a suitable language L' with $\mathcal{L}(A) \subseteq \mathrm{base}(L')$. Since R is linear, we want to use $R' = \mathrm{match}(R)$ and $L' = \mathrm{lift}_0(\mathcal{L}(A))$.

We can merge these two automata constructions into one. To do so, we need an additional, trivial enrichment that completely disregards heights in left-hand sides and assigns height 0 everywhere in the right-hand sides.

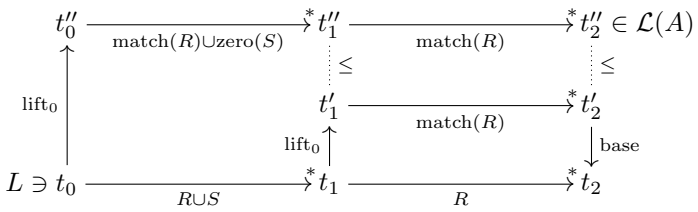
Definition 10. *For a term rewriting system R over Σ, the enrichment* $\mathrm{zero}(R)$ *over $\Sigma \times \mathbb{N}$ is defined by*

$$\mathrm{zero}(R) = \{\ell' \to \mathrm{lift}_0(r) \mid (\ell \to r) \in R,\ \mathrm{base}(\ell') = \ell\}.$$

Lemma 7. *For TRSs R and S and a language L over Σ, if a finite tree automaton A over $\Sigma \times \mathbb{N}$ is compatible with* $\mathrm{match}(R) \cup \mathrm{zero}(S)$ *and* $\mathrm{lift}_0(L)$*, then R is match-bounded for $\to^*_{R \cup S}(L)$.*

Proof. We will show that R is match-bounded for $\to^*_{R \cup S}(L)$ by c, where c is the maximal height occurring in $\mathcal{L}(A)$, which exists since A is assumed to be finite. Consider a derivation $t_1 \to^*_R t_2$ with $t_1 \in \to^*_{R \cup S}(L)$, and its canonical lifting $t'_1 \to^*_{\mathrm{match}(R)} t'_2$. We have to show that the maximal height in t'_2 is $\leq c$.

Define the relation $\leq$ on $\mathcal{T}_{\Sigma \times \mathbb{N}}$ by $s \leq t$ if $\mathrm{base}(s) = \mathrm{base}(t)$ and for each position p in s, $\mathrm{height}(s(p)) \leq \mathrm{height}(t(p))$.



Since $t_1 \in \to^*_{R\cup S}(L)$, there is a term $t_0 \in L$ such that $t_0 \to^*_{R\cup S} t_1$. For the canonical lifting $t_0'' \to^*_{\mathrm{match}(R)\cup\mathrm{zero}(S)} t_1''$ of the derivation $t_0 \to^*_{R\cup S} t_1$ we have $\mathrm{base}(t_1'') = t_1 = \mathrm{base}(t_1')$, therefore $t_1' \le t_1''$. Now there are two canonical liftings of the derivation $t_1 \to^*_R t_2$, both starting in terms with the same base: $t_1' \to^*_{\mathrm{match}(R)} t_2'$ and $t_1'' \to^*_{\mathrm{match}(R)} t_2''$. From $t_1' \le t_1''$ we obtain $t_2' \le t_2''$ by monotonicity, cf. Remark 2. We have $t_2'' \in \mathcal{L}(A)$ by compatibility, since $t_0'' \in \mathrm{lift}_0(L)$ and $t_0'' \to^*_{\mathrm{match}(R)\cup\mathrm{zero}(S)} t_2''$. Therefore the maximal height in t_2'' is $\le c$, and by $t_2' \le t_2''$ the same is true for t_2'. □

Choosing $S = R_\sharp \setminus R$ and $L = \mathrm{rhs}(R)\sigma_\sharp$, in combination with Corollary 1 and Lemma 6 this will be used as follows.

Corollary 2. *For a linear TRS R, if some finite tree automaton is compatible with* $\mathrm{match}(R) \cup \mathrm{zero}(R_\sharp \setminus R)$ *and* $\mathrm{lift}_0(\mathrm{rhs}(R)\sigma_\sharp)$*, then R is terminating.*

Example 12. The system $R = \{f(f(a,x),a) \to f(a,f(x,a))\}$ is not match-bounded. This can be seen as follows. Writing $R_h x$ for $f_h(a_h, x)$, and $L_h x$ for $f_h(x, a_h)$, we have $\mathrm{match}(R)$-derivations

$$L_0^n R_0^n a_0 \to^* R_1 R_2 \dots R_n L_n \dots L_2 L_1 a_0$$

for each $n \ge 0$, exceeding any given bound. On the other hand, the system *is* match-bounded for $\mathrm{RFC}_\sharp(R)$: We have

$$R_\sharp = \begin{cases} f(\sharp,\sharp) \to f(a,f(\sharp,a)), & f(f(\sharp,x),\sharp) \to f(a,f(x,a)), \\ f(f(a,x),\sharp) \to f(a,f(x,a)), & f(\sharp,a) \to f(a,f(\sharp,a)), \\ f(f(\sharp,x),a) \to f(a,f(x,a)) & \end{cases}$$

and $\mathrm{rhs}(R)\sigma_\sharp = \{f(a,f(\sharp,a))\}$. This can be accepted by an automaton with states $Q = \{0,1,2,3,4\}$, $F = \{1\}$, and transitions $\{a_0 \to 2, \sharp_0 \to 0, a_0 \to 4, f_0(0,4) \to 3, f_0(2,3) \to 1\}$. There is only one redex match for $\mathrm{match}(R) \cup \mathrm{zero}(R_\sharp \setminus R)$, namely $f_0(\sharp_0, a_0) \to f_0(0,4) \to 3$. So we need to ensure that $f_0(a_0, f_0(\sharp_0, a_0)) \to^* 3$, This can be achieved by adding the single transition $f_0(4,3) \to 3$, because then $f_0(a_0, f_0(\sharp_0, a_0)) \to^* f_0(4, f_0(0,4)) \to f_0(4,3) \to 3$. The resulting automaton is compatible with $\mathrm{match}(R) \cup \mathrm{zero}(R_\sharp \setminus R)$. In particular, no rule of $\mathrm{match}(R)$ matches in the automaton. So the automaton certifies that R is match-bounded for $\mathrm{RFC}_\sharp(R)$ by 0, and thus the system is terminating. Termination of R can also be proved by standard methods.

Example 13. Take $R = \{f(a,f(a,x)) \to f(a,f(x,f(f(a,a),a)))\}$. Here, the set of descendants of $\mathrm{rhs}(R)\sigma_\sharp$ modulo $R_\sharp$ is actually finite:

$$\{f(a,f(\sharp,f(f(a,a),a))),\ f(a,f(f(f(a,a),a),f(f(a,a),a)))\}.$$

Since $\mathrm{match}(R)$ does not match at all, it is match-bounded for $\mathrm{RFC}_\sharp(R)$ by 0.

8 Conclusion

In this paper, we presented a new automated method for termination proofs in term rewriting: constructing compatible tree automata for systems enriched by height annotations. We offered three enrichment schemes – top, roof, and match – which are increasingly more powerful. We demonstrated that match-bounds on the set of right-hand sides of forward closures can be even more powerful for linear TRSs. In contrast to string rewriting, match-bounded systems do not preserve regular languages for term rewriting.

The power of standard methods, like path orderings and interpretations, markedly decreases for small signatures. The fewer symbols there are, the fewer orderings and statuses there are to choose from. To improve this situation, people develop methods that encode additional information into the signature. This can be semantic information (as in semantic labelling, e.g.), or syntactic information (as in dependency pairs, e.g.). Our method belongs to the latter category, for the construction of compatible automata can be seen as a detailed analysis of overlap patterns. An earlier use of tree automata for analyzing rewrite patterns is Middeldorp's estimation of dependency graphs [15].

The algorithms described in this paper have been implemented in the program `Matchbox`. It is a highly configurable testbed for string and term rewriting with height annotations. The string rewriting version has been described in [17]. Our program is freely available (Haskell source, GNU/Linux executable, CGI interface) via `http://141.57.11.163/matchbox/`.

An earlier version of `Matchbox` took part in the Termination Competition during WST04. It solved part of the problems from the contest data base. Since `Matchbox` does not implement any of the standard methods for automated termination, this illustrates the power of our approach.

The present paper provides known (Example 9) and new (Examples 8, 13) termination problem instances that `Matchbox` can solve but that other available automated provers cannot. We checked with CiME [3], AProVE [11], TTT [12].

Acknowledgements. We thank the anonymous referees for carefully reading the paper and suggesting improvements. Thanks to Vincent van Oostrom for pointing out the relation to bisimulation and rewriting labellings.

References

1. F. Baader and T. Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998.
2. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. *Tree automata techniques and applications.* 1997–2001. Available at `http://www.grappa.univ-lille3.fr/tata/`.
3. E. Contejean, C. Marché, B. Monate, and X. Urbain. Proving Termination of Rewriting with C*i*ME, In A. Rubio (Ed.), *Proc. 6th Int. Workshop on Termination WST-03*, Technical Report DSIC II/15/03, pp. 71–73, Universidad Politécnica de Valencia, Spain

4. N. Dershowitz. Termination of linear rewriting systems. In S. Even and O. Kariv (Eds.), *Proc. 8th Int. Coll. Automata, Languages and Programming ICALP-81*, *Lecture Notes in Comput. Sci.* Vol. 115, pp. 448–458. Springer-Verlag, 1981.
5. F. Gécseg and M. Steinby. *Tree Languages*. In G. Rozenberg and A. Salomaa (Eds.), *Handbook of Formal Languages*, Vol. 3, pp. 1–68. Springer-Verlag, 1997.
6. T. Genet. Decidable approximations of sets of descendants and sets of normal forms. In T. Nipkow (Ed.), *Proc. 9th Int. Conf. Rewriting Techniques and Applications RTA-98*, *Lecture Notes in Comput. Sci.* Vol. 1379, pp. 151–165. Springer-Verlag, 1998.
7. A. Geser, D. Hofbauer and J. Waldmann. Match-bounded string rewriting systems. *Appl. Algebra Engrg. Comm. Comput.* 15(3-4):149–171, 2004.
8. A. Geser, D. Hofbauer, J. Waldmann, and H. Zantema. Tree automata that certify termination of term rewriting systems. In M. Codish and A. Middeldorp (Eds.), *Proc. 7th Int. Workshop on Termination WST-04*, Aachener Informatik Berichte AIB-2004-07, RWTH Aachen, Gemany, pp. 14–17, 2004.
9. A. Geser, D. Hofbauer, J. Waldmann, and H. Zantema. Finding finite automata that certify termination of string rewriting. In M. Domaratzki, A. Okhotin, K. Salomaa, and S. Yu (Eds.), *Proc. 9th Int. Conf. Implementation and Application of Automata CIAA-04*, *Lecture Notes in Comput. Sci.* Vol. 3317, pp. 134–145. Springer-Verlag, 2004.
10. O. Geupel. Overlap closures and termination of term rewriting systems. Technical Report MIP-8922, Universität Passau, Germany, 1989.
11. J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke. Automated termination proofs with AProVE. In V. van Oostrom (Ed.), *Proc. 15th Int. Conf. Rewriting Techniques and Applications RTA-04*, *Lecture Notes in Comp. Sci.* Vol. 3091, pp. 210–220. Springer-Verlag, 2004.
12. N. Hirokawa and A. Middeldorp. Tsukuba Termination Tool. In R. Nieuwenhuis (Ed.), *Proc. 14th Int. Conf. Rewriting Techniques and Applications RTA-03*, *Lecture Notes in Comp. Sci.* Vol. 2706, pp. 311–320. Springer-Verlag, 2003.
13. D.S. Lankford and D.R. Musser. A finite termination criterion. Unpublished draft, Information Sciences Institute, University of Southern California, Marina-del-Rey, CA, 1978.
14. C. Marché and A. Rubio (Eds.) Termination Problems Data Base. 2004. `http://www.lri.fr/~marche/wst2004-competition/tpdb.html`
15. A. Middeldorp. Approximations for strategies and termination. In *Proc. 2nd Int. Workshop on Reduction Strategies in Rewriting and Programming*, *Electron. Notes Theor. Comput. Sci.* 70(6), 2002.
16. V. van Oostrom and R. de Vrijer. Equivalences of Reductions. In Terese, *Term Rewriting Systems*, pp. 301–474. Cambridge Univ. Press, 2003.
17. J. Waldmann. Matchbox: a tool for match-bounded string rewriting, In V. van Oostrom (Ed.), *Proc. 15th Int. Conf. Rewriting Techniques and Applications RTA-04*, *Lecture Notes in Comp. Sci.* Vol. 3091, pp. 85–94. Springer-Verlag, 2004.
18. H. Zantema. TORPA: Termination of rewriting proved automatically. In V. van Oostrom (Ed.), *Proc. 15th Int. Conf. Rewriting Techniques and Applications RTA-04*, *Lecture Notes in Comp. Sci.* Vol. 3091, pp. 95–104. Springer-Verlag, 2004. Updated version accepted for *J. Automat. Reason.*.
19. H. Zantema. Termination. In Terese, *Term Rewriting Systems*, pp. 181–259. Cambridge Univ. Press, 2003.
20. H. Zantema. Termination of string rewriting proved automatically. Accepted for *J. Automat. Reason.*, 2005.

Appendix

Here we give a self-contained proof of Theorem 1. It follows the lines of the sketch from [4]. A similar proof can be found in [20] for the case of string rewriting, but with a slightly different definition of $R_\sharp$. Throughout, we consider linear (i.e. left- and right-linear) rules only.

Let $\Box$ and $\sharp$ be a unary and a constant symbol respectively not contained in Σ. Both symbols will be used to mark the border between the context and the substitution of a factorization. The first symbol keeps the substitution part, whereas the second symbol truncates it. Define the substitutions $\sigma_\Box : X \to \mathcal{T}_{\Sigma\cup\{\Box\}}(X)$ and $\sigma_\sharp : X \to \mathcal{T}_{\Sigma\cup\{\sharp\}}$ by $\sigma_\Box : x \mapsto \Box(x)$ and $\sigma_\sharp : x \mapsto \sharp$ for $x \in X$. Let $h_\sharp : \mathcal{T}_{\Sigma\cup\{\Box\}}(X) \to \mathcal{T}_{\Sigma\cup\{\sharp\}}(X)$ denote the term morphism that replaces every subterm with top-symbol $\Box$ by the constant $\sharp$, i.e., where $h_\sharp(\Box(t)) = \sharp$ and $h_\sharp(f(t_1,\dots,t_n)) = f(h_\sharp(t_1),\dots,h_\sharp(t_n))$ for $f \neq \Box$. Now we can define the term rewriting system

$$R_{\mathrm{a}} = \{C\sigma'\alpha \to r\sigma'' \mid \ell \to r \in R,\ (C,\alpha) \text{ a non-trivial factorization of } \ell,\ \sigma' = \sigma_\Box|_{\mathrm{dom}(\alpha)} \text{ and } \sigma'' = \sigma_\Box|_{X(\mathrm{ran}(\alpha))}\}$$

over signature $\Sigma \cup \{\Box\}$ and the term rewriting system

$$R_\sharp = R \cup h_\sharp(R_{\mathrm{a}})$$

over signature $\Sigma \cup \{\sharp\}$. Note that both R_{a} and $R_\sharp$ are linear, and that $X(r) \subseteq X(\ell)$ for each rule $\ell \to r$ in R_{a} or in $R_\sharp$.

Example 14. For $R = \{g(f(h(x),h(y))) \to f(y,x)\}$, the system R_{a} contains

$$g(\Box(f(h(x),h(y)))) \to f(\Box(y),\Box(x)),\quad g(f(\Box(h(x)),\Box(h(y)))) \to f(\Box(y),\Box(x)),$$
$$g(f(\Box(h(x)),h(y))) \to f(y,\Box(x)),\qquad g(f(h(x),\Box(h(y)))) \to f(\Box(y),x),$$

and the system $R_\sharp$ consists of R together with

$$g(\sharp) \to f(\sharp,\sharp),\qquad g(f(\sharp,\sharp)) \to f(\sharp,\sharp),$$
$$g(f(\sharp,h(y))) \to f(y,\sharp),\qquad g(f(h(x),\sharp)) \to f(\sharp,x).$$

For $R = \{g(x) \to x\}$ we get $R_{\mathrm{a}} = \emptyset$ and $R_\sharp = R$. For $R = \{g(g(x)) \to x\}$ we have $R_{\mathrm{a}} = \{g(\Box(g(x))) \to \Box(x)\}$ and $R_\sharp = R \cup \{g(\sharp) \to \sharp\}$. Finally, for $R = \{g(g(x)) \to a\}$ we obtain $R_{\mathrm{a}} = \{g(\Box(g(x))) \to a\}$ and $R_\sharp = R \cup \{g(\sharp) \to a\}$.

Let S consist of all ground terms over $\Sigma \cup \{\Box\}$ where each path from a leaf to the root contains at most one occurrence of the symbol $\Box$, that is, $S = \{C\sigma_\Box\alpha \mid (C,\alpha) \in F\}$ where F denotes the set of all factorizations of ground terms over Σ. Note that C might be ground for $(C,\alpha) \in F$; in this case we have $C\sigma_\Box\alpha = C$. (We could also write $S = \mathcal{T}_\Sigma(\Box(\mathcal{T}_\Sigma))$, using the standard notation

$f(T) = \{f(t) \mid t \in T\}$ for $T \subseteq \mathcal{T}_\Sigma$.) On S we consider two relations that describe *active* and *non-active* rewriting respectively, defined by

$$\xrightarrow{\mathrm{a}}_R = \{(C\sigma_\Box\alpha, C'\sigma_\Box\alpha) \mid (C,\alpha) \in F,\ C \to_R C'\} \cup (\to_{R_\mathrm{a}} \cap\, S \times S),$$
$$\xrightarrow{\mathrm{na}}_R = \{(C\sigma_\Box\alpha, C\sigma_\Box\alpha') \mid (C,\alpha) \in F,\ \alpha \to_R \alpha'\}.$$

Note that non-active steps always occur below an occurrence of the symbol $\Box$, whereas for active steps this is impossible.

The first lemma describes commutation between active and non-active steps.

Lemma 8. $\xrightarrow{\mathrm{a}}_R \circ \xrightarrow{\mathrm{na}}_R \subseteq \xrightarrow{\mathrm{na}}_R \circ \xrightarrow{\mathrm{a}}_R$.

Proof. Let $C\sigma_\Box\alpha \xrightarrow{\mathrm{a}}_R C'\sigma_\Box\alpha' \xrightarrow{\mathrm{na}}_R C'\sigma_\Box\alpha''$ where $\alpha' \to_R \alpha''$. The first rewriting step is modulo $\to_R \cup \to_{R_\mathrm{a}}$, the second one modulo $\to_R$. Since the claim is trivially true if these steps occur at parallel positions, we may assume the second step to occur below the first one. There are two cases for the active step: If $C \to_R C'$ and $\alpha = \alpha'$ then $C\sigma_\Box\alpha \xrightarrow{\mathrm{na}}_R C\sigma_\Box\alpha'' \xrightarrow{\mathrm{a}}_R C'\sigma_\Box\alpha''$. Otherwise, stripping off the context of the first step, the given derivation has the form

$$\bar{C}\sigma_\Box\bar{\alpha}\beta \to_{R_\mathrm{a}} r\sigma_\Box|_{X(\mathrm{ran}(\bar{\alpha}))}\beta \xrightarrow{\mathrm{na}}_R r\sigma_\Box|_{X(\mathrm{ran}(\bar{\alpha}))}\beta'$$

where $(\bar{C}, \bar{\alpha})$ is an non-trivial factorization of ℓ for some rule $\ell \to r$ in R, and where $\beta \to_R \beta'$ for substitutions β, β' with domain $X(\mathrm{ran}(\bar{\alpha}))$. Then we have

$$\bar{C}\sigma_\Box\bar{\alpha}\beta \xrightarrow{\mathrm{na}}_R \bar{C}\sigma_\Box\bar{\alpha}\beta' \to_{R_\mathrm{a}} r\sigma_\Box|_{X(\mathrm{ran}(\bar{\alpha}))}\beta'$$

by left-linearity of R_a. □

In the setting of abstract reduction systems, the following lemma describes how an infinite reduction modulo $\xrightarrow{\mathrm{a}}_R \cup \xrightarrow{\mathrm{na}}_R$ can be transformed into an infinite active reduction.

Lemma 9. *Let ρ and σ be relations on a set S such that $\sigma \circ \rho \subseteq \rho \circ \sigma$, and let $s \in S$ satisfy* $\mathrm{SN}(s, \rho)$ *and* $\infty(s, \rho \cup \sigma)$. *Then there exists $t \in S$ satisfying $s\ \rho^*\ t$ and $\infty(t, \sigma)$.*

Proof. Write $P(s)$ for the property to be proven. Since $\mathrm{SN}(s, \rho)$ we may prove this by Noetherian induction, i.e. for proving $P(s)$ we may assume the induction hypothesis $P(s')$ for $s\ \rho\ s'$. Consider an infinite chain modulo $\rho \cup \sigma$ starting in s. In case it does not contain any ρ step we may choose $t = s$ and we are done. Otherwise, the chain starts by $s\ \sigma^*\ \overline{s}\ \rho\ \tilde{s}$ where $\infty(\tilde{s}, \rho \cup \sigma)$. Using $\sigma \circ \rho \subseteq \rho \circ \sigma$ one easily proves $\sigma^* \circ \rho \subseteq \rho \circ \sigma^*$ by induction, hence we obtain s' satisfying $s\ \rho\ s'\ \sigma^*\ \tilde{s}$. Now clearly $\mathrm{SN}(s', \rho)$ and $\infty(s', \rho \cup \sigma)$, hence by the induction hypothesis $P(s')$ we obtain $t \in S$ satisfying $s\ \rho\ s'\ \rho^*\ t$ and $\infty(t, \sigma)$. □

The next lemma relates $\to_R$ and $\xrightarrow{\mathrm{a}}_R \cup \xrightarrow{\mathrm{na}}_R$.

Lemma 10. *If* $\infty(t, \to_R)$ *for* $t \in \mathcal{T}_\Sigma$ *then* $\infty(C\sigma_\Box\alpha, \xrightarrow{\mathrm{a}}_R \cup \xrightarrow{\mathrm{na}}_R)$ *for any factorization* (C, α) *of* t.

Proof. From the definition of $\xrightarrow{\mathrm{a}}_R \cup \xrightarrow{\mathrm{na}}_R$ it follows that if $C\alpha \to_R t'$ then there exist a factorization (C', α') of t' such that $C\sigma_\Box\alpha \; (\xrightarrow{\mathrm{a}}_R \cup \xrightarrow{\mathrm{na}}_R) \; C'\sigma_\Box\alpha'$. Repeating this, an infinite reduction modulo $\to_R$ starting in $C\alpha$ is transformed into an infinite reduction modulo $\xrightarrow{\mathrm{a}}_R \cup \xrightarrow{\mathrm{na}}_R$ starting in $C\sigma_\Box\alpha$. □

Lemma 11. *If* $\infty(\to_R)$ *then* $\infty(t, \xrightarrow{\mathrm{a}}_R)$ *for some ground instance* t *of a term in* $\mathrm{rhs}(R)\sigma_\Box$.

Proof. The claim is trivially true if $X \cap \mathrm{lhs}(R) \neq \emptyset$, so we may assume the contrary. Choose $g \in \mathcal{T}_\Sigma$ of minimal size satisfying $\infty(g, \to_R)$, and let $g = f(g_1, \ldots, g_n)$ for $f \in \Sigma_n$, $n \geq 0$. Then $\mathrm{SN}(g_i, \to_R)$ for $1 \leq i \leq n$, thus an infinite reduction starting in g is of the shape

$$g = f(g_1, \ldots, g_n) \to_R^* f(g'_1, \ldots, g'_n) = \ell\gamma \to_R r\gamma \to_R \cdots$$

for some rule $\ell \to r$ in R, a ground substitution γ, and ground terms g'_i satisfying $g_i \to_R^* g'_i$. We have $\infty(r\gamma, \to_R)$, and from $\mathrm{SN}(g_i, \to_R)$ and $g_i \to_R^* g'_i$ we get $\mathrm{SN}(g'_i, \to_R)$. Since each term in $X(r)\gamma$ is a subterm of some term g'_i we have $\mathrm{SN}(X(r)\gamma, \to_R)$. Now we apply Lemma 9 to $s = r\sigma_\Box\gamma$, $\rho = \xrightarrow{\mathrm{na}}_R$, and $\sigma = \xrightarrow{\mathrm{a}}_R$. The conditions are fulfilled due to Lemma 8, Lemma 10, and the observation that $\mathrm{SN}(s, \xrightarrow{\mathrm{na}}_R)$ follows from $\mathrm{SN}(X(r)\gamma, \to_R)$. Hence there exists $t \in S$ satisfying $s \; \xrightarrow{\mathrm{na}}_R{}^* \; t$ and $\infty(t, \xrightarrow{\mathrm{a}}_R)$. Since $s = r\sigma_\Box\gamma \; \xrightarrow{\mathrm{na}}_R{}^* \; t$ we can write $t = r\sigma_\Box\gamma'$ for some ground substitution γ', concluding the proof. □

As an immediate consequence of the respective definitions, we can relate $R_\sharp$ to $\xrightarrow{\mathrm{a}}_R$.

Lemma 12. *If* $C\sigma_\Box\alpha \xrightarrow{\mathrm{a}}_R C'\sigma_\Box\alpha'$ *for* $C\sigma_\Box\alpha, C'\sigma_\Box\alpha' \in S$ *then* $C\sigma_\sharp \to_{R_\sharp} C'\sigma_\sharp$. *In fact then we have either* $C \to_R C'$ *and* $\alpha' = \alpha|_{\mathrm{dom}(\alpha')}$ *with* $\mathrm{dom}(\alpha') \subseteq \mathrm{dom}(\alpha)$, *or* $C\sigma_\sharp \to_{R_\sharp} C'\sigma_\sharp$ *and* $\sum_{x \in \mathrm{dom}(\alpha)} \mathrm{size}(x\alpha) > \sum_{x \in \mathrm{dom}(\alpha')} \mathrm{size}(x\alpha')$.

Theorem. *Let* R *be a linear term rewriting system. Then*

$$\mathrm{SN}(\to_R) \text{ if and only if } \mathrm{SN}(\to_{R_\sharp}^*(\mathrm{rhs}(R)\sigma_\sharp), \to_R).$$

Proof. For the non-trivial 'if'-part assume R is non-terminating. By Lemma 11 we have $\infty(t, \xrightarrow{\mathrm{a}}_R)$ for some term $t = r\sigma_\Box\gamma$ with $r \in \mathrm{rhs}(R)$ and γ a ground substitution with domain $X(r)$. Write the corresponding infinite reduction as

$$t = C_0\sigma_\Box\gamma_0 \xrightarrow{\mathrm{a}}_R C_1\sigma_\Box\gamma_1 \xrightarrow{\mathrm{a}}_R C_2\sigma_\Box\gamma_2 \xrightarrow{\mathrm{a}}_R \cdots .$$

By Lemma 12, for every i we have $C_i \to_R C_{i+1}$, or we know that $C_i\sigma_\sharp \to_{R_\sharp} C_{i+1}\sigma_\sharp$ and $\sum_{x \in \mathrm{dom}(\gamma_i)} \mathrm{size}(x\gamma_i) > \sum_{x \in \mathrm{dom}(\gamma_{i+1})} \mathrm{size}(x\gamma_{i+1})$. This latter case occurs only finitely often (at most $\sum_{x \in \mathrm{dom}(\gamma_0)} \mathrm{size}(x\gamma_0)$ times), say not after the k-th step. By Lemma 12 we have $C_0\sigma_\sharp \to_{R_\sharp}^* C_k\sigma_\sharp$, i.e., $C_k\sigma_\sharp \in \to_{R_\sharp}^*(\mathrm{rhs}(R)\sigma_\sharp)$. Therefore this gives rise to an infinite reduction modulo R starting from $C_k\sigma_\sharp$, contradicting the assumption $\mathrm{SN}(\to_{R_\sharp}^*(\mathrm{rhs}(R)\sigma_\sharp), \to_R)$. □