The Termination Hierarchy for Term Rewriting

H. Zantema
Utrecht University, Department of Computer Science
P.O. box 80.089, 3508 TB Utrecht, The Netherlands
e-mail: hansz@cs.uu.nl

Abstract

A number of properties of term rewriting systems related to termination is discussed. It is examined how these properties are affected by modifications in the definitions like weakening the requirement of strict monotonicity and adding embedding rules. All counterexamples to prove non-equivalence of properties are string rewriting systems, in most cases even single string rewrite rules.

1 Introduction

The most basic idea of proving termination of a term rewriting system is the following:

> Define a weight function on terms such that by every rewrite step the weight strictly decreases.

Often there are infinitely many possible rewrite steps, but only finitely many rewrite rules. In order to reduce the set of proof obligations the basic idea can be refined to:

> Define a compositional and monotone weight function on terms such that for every rewrite rule the left hand side is greater than the right hand side

The set of possible weights is equipped with an order; to be able to conclude termination this order has to be well-founded. By compositionality of weights this set is an algebra, since the weight function has to be monotone it is called a well-founded monotone algebra. In [13] a hierarchy of different properties related to termination was proposed based on the kind of algebra involved. In this paper this hierarchy is extended by a number of other notions as they appear in the literature.

For all properties X and Y occurring in the hierarchy that satisfy $X \Rightarrow Y$ we give an example of a term rewriting system satisfying Y but not satisfying X, hence proving that the converse of the implication does not hold. Except for one all of these examples are single string rewrite rules. These single string rewrite rules are of particular interest since they look very simple, but proving termination can be very hard ([15, 10]).

Next we study some possible modifications of the definitions given so far, and study whether they affect the meaning of the definitions. In most cases indeed they do, hence we can say that the definitions of the properties occurring in the hierarchy are not robust. One modification of particular interest is the weakening of the monotonicity requirement. In the standard definition the operations in the algebra have to be strictly monotone in all arguments. If this requirement is weakened to weak monotonicity no termination can be concluded any more. However, if the operations are weakly monotone and satisfy the subterm property, then we prove that we can conclude termination. By this observation the set of operations that can be used for proving termination drastically increases. For instance, in the set of natural numbers the operation taking the maximum of both arguments is not stricly monotone, but it is weakly monotone and satisfies the subterm property.

In Section 2 we give some preliminaries including the hierarchy as given in [13]. In Section 3 we extend this hierarchy, and we prove that for none of the implications in the hierarchy the converse holds. In Section 4 we prove that algebras with weakly monotone operations satisfying the subterm property serve for proving termination, and discuss how this would affect the properties in the hierarchy. In Section 5 we study the subtle difference between the subterm property and the strict subterm property. This difference can also be described as the effect of adding embedding rules. In Section 6 we study the effect of reversing strings and show that our notion of total termination is not equivalent to the same notion introduced by Ursula Martin for string rewriting. Finally in Section 7 we give some concluding remarks.

2 Preliminaries

Let $\mathcal{F}$ be a set of operation symbols each having a fixed arity. Let $\mathcal{X}$ be a set of variable symbols. The set of terms over $\mathcal{F}$ and $\mathcal{X}$ is denoted by $\mathcal{T}(\mathcal{F}, \mathcal{X})$.

Definition 1 *A* well-founded monotone algebra *over $\mathcal{F}$ consists of*

- *A well-founded poset $(A, >)$*
- *for every operation symbol $f \in \mathcal{F}$ of arity n an operation $f_A : A^n \to A$ that is* strictly monotone *in all arguments, i.e., if $a_1, \ldots, a_n, b_1, \ldots, b_n \in A$ for*

which $a_i > b_i$ for some i and $a_j = b_j$ for all $j \neq i$ then

$$f_A(a_1, \ldots, a_n) > f_A(b_1, \ldots, b_n).$$

A map $\sigma : \mathcal{X} \to A$ extends to terms by

- *$[x, \sigma] = \sigma(x)$ for $x \in \mathcal{X}$*
- *$[f(t_1, \ldots, t_n), \sigma] = f_A([t_1, \sigma], \ldots, [t_n, \sigma])$ for $f \in \mathcal{F}$ and $t_1, \ldots, t_n \in \mathcal{T}(\mathcal{F}, \mathcal{X})$.*

A well-founded monotone algebra and a TRS are compatible *if*

$$[l, \sigma] > [r, \sigma]$$

for every rule $l \to r$ and every $\sigma : \mathcal{X} \to A$.

This definition is motivated by the following basic theorem.

Theorem 2 *A term rewriting system is terminating if and only if it is compatible with a well-founded monotone algebra.*

The proof of this theorem is not difficult. For the 'if'-part one shows that an infinite reduction $t_1 \to t_2 \to t_3 \to \cdots$ transforms to an infinite decreasing sequence

$$[t_1, \sigma] > [t_2, \sigma] > [t_3, \sigma] > \cdots,$$

using strict monotonicity, for the 'only if'-part one shows that $(\mathcal{T}(\mathcal{F}, \mathcal{X}), \to^+)$ is a compatible well-founded monotone algebra if the term rewriting system is terminating. For details of the proof we refer to [13].

The way of proving termination of a term rewriting system by means of Theorem 2 is now as follows: choose a well-founded poset $(A, >)$, define for each operation symbol f a corresponding operation f_A that is strictly monotone in all of its coordinates, and for which $[l, \sigma] > [r, \sigma]$ for all rewrite rules $l \to r$ and all $\sigma : \mathcal{X} \to A$. Then according to Theorem 2 the term rewriting system is terminating.

In this setting it is a natural question to ask which kind of posets $(A, >)$ and which kind of operations are useful or necessary to prove termination by this approach. By giving the following restrictions we define the following restricted kinds of termination.

Definition 3 *A term rewriting system is* simply terminating *if it is compatible with a well-founded monotone algebra $(A, >)$ such that $f_A(a_1, \ldots, a_n) \geq a_i$ for every $f \in \mathcal{F}$, every $a_1, \ldots, a_n \in A$ and every $i = 1, \ldots, n$.*

A term rewriting system is totally terminating *if it is compatible with a well-founded monotone algebra $(A, >)$ such that $>$ is a total order on A.*

A term rewriting system is ω-terminating *if it is compatible with a well-founded monotone algebra* $(A, >)$ *such that* $A = \mathbb{N}$ *and* $>$ *is the usual order on* $\mathbb{N}$.

A term rewriting system is polynomially terminating *if it is compatible with a well-founded monotone algebra* $(A, >)$ *such that* $A = \mathbb{N}$ *and* $>$ *is the usual order on* $\mathbb{N}$ *and* f_A *is a polynomial for every* $f \in \mathcal{F}$.

Summarizing:

A term rewriting system is	if it is compatible with a well-founded monotone algebra $(A, >)$ such that
terminating	—
simply terminating	$f_A(\ldots, a, \ldots) \geq a$
totally terminating	$>$ is a total order on A
ω-terminating	$(A, >) = (\mathbb{N}, >)$
polynomially terminating	$(A, >) = (\mathbb{N}, >)$ and f_A are polynomials

For these kinds of termination in [13] the following basic hierarchy was given:

polynomial termination
$\Longrightarrow$ ω-termination
$\Longrightarrow$ total termination
$\Longrightarrow$ simple termination
$\Longrightarrow$ termination.

Validity of all implications is by definition except for the implication of simple termination from total termination which is immediate from the following proposition.

Proposition 4 *Let* $(A, >)$ *be a well-founded monotone* $\mathcal{F}$*-algebra for which the order* $>$ *is total on* A. *Then* $f_A(a_1, \ldots, a_n) \geq a_i$ *for every* $f \in \mathcal{F}$, *every* $a_1, \ldots, a_n \in A$ *and every* $i = 1, \ldots, n$.

Proof: Assume not. Then there exist $f \in \mathcal{F}, a_1, \ldots, a_n \in A$ and $i \in \{1, \ldots, n\}$ such that $f_A(a_1, \ldots, a_n) \geq a_i$ does not hold. From totality we conclude:

$$a_i > f_A(a_1, \ldots, a_n).$$

Define $g : A \to A$ by $g(x) = f_A(a_1, \ldots, a_{i-1}, x, a_{i+1}, \ldots, a_n)$, then g is strictly monotone. We obtain an infinite chain

$$a_i > g(a_i) > g(g(a_i)) > g(g(g(a_i))) > \cdots,$$

contradicting the well-foundedness of $(A, >)$. $\Box$

Example:

$$g(f(x)) \to f(f(g(x)))$$

is polynomially terminating by $A = \mathbb{N}$ and

$$\begin{array}{rcl} f_A(x) & = & x+1 \\ g_A(x) & = & 3x \end{array}$$

since

$$\begin{array}{rcl} g_A(f_A(x)) & = & 3(x+3) \\ & > & 3x+1+1 \\ & = & f_A(f_A(g_A(x))) \end{array}$$

Example:

$$f(g(x)) \to g(f(f(x)))$$

is totally terminating by $A = \mathbb{N} \times \mathbb{N}$ and

$$\begin{array}{rcl} f_A((x,y)) & = & (x, x+y) \\ g_A((x,y)) & = & (2x+1, y) \end{array}$$

with lexicographic order, since

$$\begin{array}{rcl} f_A(g_A((x,y))) & = & (2x+1, 2x+y+1) \\ & > & (2x+1, 2x+y) \\ & = & g_A(f_A(f_A(x))) \end{array}$$

Here it is essential to take the lexicographic order from left to right, otherwise f_A is not strictly monotone. In [13] it was shown that this term rewriting system is not ω-terminating. Through this paper it will serve as an example a few more times.

In these examples only unary symbols occur, only one variable symbol occurs, and all terms consist of a string of unary symbols applied on this variable. Without loosing information we can omit the variable symbol and all parentheses, for instance writing $gf \to ffg$ and $fg \to gff$ for the above two examples. Term rewriting systems in which only unary symbols occur are called *string rewriting systems*; for string rewriting systems we will use this shorthand notation of omitting variables and parentheses. In the literature string rewriting systems are also called *semi-Thue systems*.

In [13] it was shown that for none of the implications in the basic hierarchy the reverse implication holds, even for single rule string rewriting.

For a signature $\mathcal{F}$ we define $Emb(\mathcal{F})$ or shortly Emb to be the term rewriting system consisting of the rules

$$f(x_1, \ldots, x_n) \to x_i,$$

for all $f \in \mathcal{F}$ with arity > 0 and all $i = 1, \ldots, n$, where n is the arity of f. The rules of *Emb* are called *embedding rules.* We conclude this section by two propositions for which the proofs can be found in [13] and [2] respectively.

Proposition 5 *A term rewriting system R is simply terminating if and only if $R \cup Emb$ is terminating.*

Proposition 6 *A term rewriting system R is totally terminating if and only if $l^\sigma > r^\sigma$ for all rules $l \to r$ in R and all ground substitutions σ, for some total well-founded order $>$ on ground terms that is closed under contexts. In case $\mathcal{F}$ does not contain constants a fresh constant is assumed here to be added in order to obtain ground terms.*

3 The extended hierarchy

There are numerous ways to refine the termination hierarchy. For instance, for polynomial termination one can distinguish between degrees of polynomials that are allowed. Another way is to refine total termination by the ordinal type of the total order used. In [2] it was proved that for every positive natural number n the string rewriting system consisting of the rules

$$f_i f_{i+1} \to f_{i+1} f_i f_i$$

for $i = 1, \ldots, n$ is compatible with a total well-founded monotone algebra of order type ω^{n+1}, but not with a total well-founded monotone algebra of a smaller order type. Hence the level of total termination can be subdivided into infinitely many distinct levels.

In this section however we will compare the properties of the basic hierarchy with other properties that are related to termination, but not to monotone algebras.

Definition 7 *A term rewriting system R is called* self-embedding *if terms t and u exist satisfying $t \to_R^+ u \to_{Emb}^* t$.*

A term rewriting system R is called looping *if a term t, a context C and a substitution σ exist satisfying $t \to_R^+ C[t^\sigma]$.*

A term rewriting system R is called cyclic *if a term t exists satisfying $t \to_R^+ t$.*

A term rewriting system R is called rpo-terminating *if a well-founded precedence $>$ on the signature exists such that $l >_{rpo} r$ for all rewrite rules $l \to r$ of R. Here $>_{rpo}$ is the recursive path order as described in [1].*

Using these properties we arrive at the following extended hierarchy:

$$\begin{tikzcd}
\text{polynomially terminating} \arrow[d, Rightarrow] & \\
\omega\text{-terminating} \arrow[d, Rightarrow] & \text{rpo-terminating} \arrow[l, Rightarrow] \\
\text{totally terminating} \arrow[d, Rightarrow] & \\
\text{simply terminating} \arrow[d, Rightarrow] & \\
\text{non-self-embedding} \arrow[d, Rightarrow] & \\
\text{terminating} \arrow[d, Rightarrow] \arrow[r, Rightarrow] & \text{weakly normalizing} \\
\text{non-looping} \arrow[d, Rightarrow] & \\
\text{non-cyclic} &
\end{tikzcd}$$

Validity of the implication 'rpo-terminating $\Rightarrow$ ω-terminating' is a corollary of the main result of [8]; this result only holds for finite signatures. For instance, if we have rules $b \to a_i$ and $a_{i+1} \to a_i$ for all $i \in \mathbb{N}$, where all symbols are constants, then this infinite system is rpo-terminating but not ω-terminating.

Validity of the implication 'non-self-embedding $\Rightarrow$ terminating' immediately follows from Kruskal's theorem in case of finite signatures as discussed in [1]; for infinite signatures it is not true: the system consisting of the rules $a_i \to a_{i+1}$ for all $i \in \mathbb{N}$ is clearly non-self-embedding but not terminating.

All other implications hold both for finite and infinite signatures; for 'totally terminating $\Rightarrow$ simply terminating' this follows from Proposition 4, for 'simply terminating $\Rightarrow$ non-self-embedding' this follows from Proposition 5, and validity of the remaining implications is immediate by definition.

For none of the implications the reversed implication holds for term rewriting. It even holds for single rule string rewriting except for the implication 'terminating $\Rightarrow$ non-looping', for which this is an open problem. For all other implications

$X \Rightarrow Y$ we give a single string rewrite rule that satisfies Y but not X:

satisfying	but not	example
ω-terminating	polynomially terminating	$fgh \to gfhg$
ω-terminating	rpo-terminating	$fgf \to gffh$
totally terminating	ω-terminating	$fg \to gff$
simply terminating	totally terminating	$fgh \to fhhgg$
non-self-embedding	simply terminating	$fg \to hggffh$
terminating	non-self-embedding	$ff \to fgf$
weakly normalizing	terminating	$fgfg \to gfgffg$
non-looping	terminating	open problem
non-cyclic	non-looping	$f \to fg$

The proof that $fgfg \to gfgffg$ is weakly normalizing but not terminating, and even that this is the smallest string rewrite rule having this property, is due to Alfons Geser ([3]).

The system $fgf \to gffh$ is not rpo-terminating since both $f > g$ and $g > f$ will not yield $f(g(f(x))) >_{rpo} g(f(f(h(x))))$. It is ω-terminating by $f_A(x) = x+1, g_A(x) = 2x, h_A(x) = x$.

The system $fg \to hggffh$ is not simply terminating due to Proposition 5 and the reduction

$$fgg \to hggffhg \to^*_{Emb} ffg \to fhggffh \to^*_{Emb} fgg.$$

It is non-self-embedding due to the next proposition.

The proofs of all other claims in the table are obvious or can be found in [13].

Proposition 8 *The string rewrite rule $fg \to hggffh$ is non-self-embedding.*

Proof: Assume we have a reduction

$$t \to^+ u \to^*_{Emb} t.$$

Assume the number n of steps in $t \to^+ u$ is minimal. Since there is no overlap between the left hand side and the right hand side we have $t = w_0 fgw_1 \cdots fgw_n$ and $u = w_0 hggffhw_1 \cdots hggffhw_n$. We will apply the following claim a number of times:

Claim:

If $av \to^*_{Emb} av'$, then $v \to^*_{Emb} v'$.

If $vv' \to^*_{Emb} av''$ for a string v in which the symbol a does not occur, then $v' \to^+_{Emb} v''$.

The proof of this claim is straightforward. Write $t = w_0 fgt'$ and $u = w_0 hggffhu'$. Applying the first part of the claim $\#w_0$ times yields $hggffhu' \rightarrow^*_{Emb} fgt'$. Applying the second part of the claim yields $ffhu' \rightarrow^+_{Emb} gt'$. Again applying the second part of the claim yields $u' \rightarrow^+_{Emb} t'$. Combined with $t' \rightarrow^* u'$ we obtain $t' \rightarrow^+ u' \rightarrow^*_{Emb} t'$, contradicting minimality of n. $\Box$

We state the existence of a single non-looping non-terminating string rewriting rule as an open problem. Partial results include the single non-looping non-terminating term rewriting rule

$$f(c, a(x), y) \ \rightarrow \ g(f(c, x, a(y)), f(x, y, a(a(c))))$$

from [16] and the two rule non-looping non-terminating string rewriting system

$$\begin{array}{rcl} fgh & \rightarrow & hghfgff \\ fh & \rightarrow & hf \end{array}$$

from [7], being a simplification of a similar two rule system from [16] constructed from four symbols. In [4] it was shown that for single string rewrite rules $l \rightarrow r$ in which a symbol a occurs that occurs in r not more often than in l, the notions termination and non-loopingness coincide.

One motivation for considering this open problem is its relation with the open problem of decidability of termination of single rule string rewriting. In case termination and non-loopingness coincide for single rule string rewriting this indicates that single rule string rewriting is essentially less powerful than both two rule string rewriting and single rule term rewriting and one may hope for decidability of termination of single rule string rewriting.

The extended termination hierarchy as presented here has been the basis of studying *relative undecidability*: for most of the implications $X \Rightarrow Y$ it has been proven in [5] that for term rewriting systems satisfying Y it is undecidable whether X holds. In [6] it has been proven that this even holds for single rule term rewriting.

4 Weak monotone algebras

In the monotone algebras we considered until now all operation were required to be strictly monotone in all arguments. This is a quite strong requirement. For instance, if $A = \mathbb{N}$ for a binary symbol a we are not allowed to choose $a_A(x, y) = \max(x, y)$ since a_A is not strictly monotone in its first argument as can be seen from $3 > 2$ and $\max(3, 4) \not> \max(2, 4)$. In this section we will see how we can weaken the restriction of strict monotonicity to the combination of weak monotonicity and the subterm property in such a way that we still can conclude termination from compatibility with a corresponding monotone algebra. In the setting of orderings instead of monotone algebras a similar replacement of strict monotonicity by the combination of weak monotonicity and the subterm property was called the *second termination theorem* in [1].

Definition 9 *A* weak monotone algebra *over $\mathcal{F}$ consists of*

- *A well-founded poset $(A, >)$*
- *for every operation symbol $f \in \mathcal{F}$ of arity n an operation $f_A : A^n \to A$ that is* weakly monotone *in all arguments, i.e., if $a_1, \dots, a_n, b_1, \dots, b_n \in A$ for which $a_i > b_i$ for some i and $a_j = b_j$ for all $j \neq i$ then*

  $$f_A(a_1, \dots, a_n) \geq f_A(b_1, \dots, b_n),$$

  and that satisfies the subterm property*, i.e.,*

  $$f_A(a_1, \dots, a_n) \geq a_i$$

  for every $a_1, \dots, a_n \in A$ and every $i = 1, \dots, n$.

A weak monotone algebra and a TRS are compatible *if*

$$[l, \sigma] > [r, \sigma]$$

for every rule $l \to r$ and every $\sigma : \mathcal{X} \to A$.

As usual here the relation $\geq$ is defined by

$$a \geq b \iff a > b \vee a = b.$$

In order to be able to prove the corresponding we start with a lemma.

Lemma 10 *Let a term rewriting system R be compatible with a weak monotone algebra $(A, >)$, and let $\sigma : \mathcal{X} \to A$. Then*

- *$[l^\alpha, \sigma] > [r^\alpha, \sigma]$ for $\alpha : \mathcal{X} \to \mathcal{T}(\mathcal{F}, \mathcal{X})$, and*
- *$[t, \sigma] \geq [u, \sigma]$ for every t, u satisfying $t \to_R^* u$.*

Proof: The first assertion follows from compatibility and the fact that $[t^\alpha, \sigma] = [t, \tau]$ for $\tau : \mathcal{X} \to A$ defined by $\tau(x) = [x^\alpha, \sigma]$, for every term t. This fact is easily proved by induction on the structure of t.

In case of $t \to_R u$ the second assertion follows from weak monotonicity and the first assertion, by induction on the context. The general version of the second assertion follows from this particular case by induction on the length of the reduction. $\Box$

Theorem 11 *A term rewriting system is terminating if it is compatible with a weak monotone algebra.*

Proof: Assume that a term rewriting system R admits an infinite reduction and is compatible with a weak monotone algebra $(A, >)$. Fix an arbitrary map $\sigma : \mathcal{X} \to A$. Since $>$ is well-founded there exists a term t having an infinite reduction that is minimal in the following sense: every term u satisfying $[t, \sigma] > [u, \sigma]$ has no infinite reduction. Next take a subterm t' of t such that t' admits an infinite reduction but no proper subterm of t' admits an infinite reduction. Due to the subterm property we have $[t, \sigma] \geq [t', \sigma]$. Take an infinite reduction of t'. Since no proper subterm of t' admits an infinite reduction, this infinite reduction has to be of the shape

$$t' \to_R^* l^\alpha \to_R r^\alpha \to_R \cdots$$

for some rule $l \to r$ of R and some $\alpha : \mathcal{X} \to \mathcal{T}(\mathcal{F}, \mathcal{X})$, where r^α admits an infinite reduction. From Lemma 10 we conclude

$$[t, \sigma] \geq [t', \sigma] \geq [l^\alpha, \sigma] > [r^\alpha, \sigma],$$

contradicting minimality of t. □

For validity of Theorem 11 it is essential to require the subterm property in a weak monotone algebra. For example, choose R to consist of the string rewrite rule $f \to gf$, and $(A, >) = (\mathbb{N}, >)$, $f_A(x) = x + 1$ and $g_A(x) = 0$. Then all requirements are fulfilled except the subterm property of g_A, and R is not terminating.

To illustrate the power of Theorem 11 we give two examples.

Example: Consider R to consist of the rule

$$f(g(x)) \to g(a(f(x), f(x))).$$

It is possible to prove that R is ω-terminating, but for doing so one for f_A one has to choose an exponential function. This is due to the fact that $a_A(x, y) \geq x + y$ for every $a_A : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ that is strictly monotone in both arguments.

The following weak monotone operations on $A = \mathbb{N}$ are much simpler:

$$\begin{array}{rcll} f_A(x) & = & 2x & \text{for } x \in \mathbb{N} \\ g_A(x) & = & x + 1 & \text{for } x \in \mathbb{N} \\ a_A(x, y) & = & \max(x, y) & \text{for } x, y \in \mathbb{N}; \end{array}$$

from

$$f_A(g_A(x)) = 2x + 2 > 2x + 1 = g(a(f(x), f(x)))$$

we conclude termination of R by Theorem 11.

The converse of Theorem 11 does not hold, more precisely, we have the following proposition.

Proposition 12 *A term rewriting system is simply terminating if and only if it is compatible with a weak monotone algebra.*

Proof: If the term rewriting system R is simply terminating then the corresponding well-founded monotone algebra satisfies all requirements of a weak monotone algebra.

Conversely let $(A, >)$ be a weak monotone algebra compatible with R. Let $>'$ be the lexicographic order on $A' = A \times \mathbb{N}$ from left to right. Define

$$f_{A'}((a_1, k_1), \ldots, (a_n, k_n)) = (f_A(a_1, \ldots, a_n), 1 + \sum_1^n k_i)$$

for every $f \in \mathcal{F}$. Then $(A', >')$ is a weak monotone algebra compatible with $R \cup Emb$. Then $R \cup Emb$ is terminating by Theorem 11, and R is simply terminating by Proposition 5. $\Box$

The following proposition states that the notion of total termination is not affected by replacing the strict monotonicity requirement by the combination of weak monotonicity and the subterm property.

Proposition 13 *A term rewriting system is totally terminating if and only if it is compatible with a weak monotone algebra $(A, >)$ in which the order $>$ is total on A.*

Proof: The 'only if'-part is immediate from Proposition 4. The proof of the 'if'-part is given in [11]. $\Box$

The next natural question is whether the notion of ω-termination is affected by replacing the strict monotonicity requirement by the combination of weak monotonicity and the subterm property. The following example shows that it is.

Example: In [13] it was proved that the string rewrite rule

$$f(g(x)) \to g(f(f(x)))$$

is not ω-terminating. However, by choosing

$$f_A(x) = 1 + 2 * \lfloor x/2 \rfloor$$

$$g_A(x) = 2 + 2 * \lfloor x/2 \rfloor$$

we have

$$f_A(g_A(x)) \ = \ 3 + 2 * \lfloor x/2 \rfloor \ > \ 2 + 2 * \lfloor x/2 \rfloor \ = \ g_A(f_A(f_A(x))).$$

Both f_A and g_A are not strictly monotone but they are weakly monotone and satisfy the subterm property, hence $f(g(x)) \to g(f(f(x)))$ is compatible with a weak monotone algebra $(A, >)$ for which $(A, >) = (\mathbb{N}, >)$.

5 Adding embedding rules

We have seen that for total termination and simple termination the requirement of strict monotonicity is equivalent to the combination of weak monotonicity and the subterm property. Here for the subterm property one can imagine two distinct kinds: a weak subterm property

$$f_A(\ldots, a, \ldots) \geq a$$

as we considerd until now, and a strict subterm property

$$f_A(\ldots, a, \ldots) > a.$$

Is there an essential difference between these two requirements? It is easy to see that the strict subterm property is equivalent to compatibility with Emb. Hence the question about the difference between weak and strict subterm property can be restated without referring to monotone algebras: for properties X of term rewriting systems related to termination we wonder whether $X(R)$ is equivalent to $X(R \cup Emb)$. In this section we will show that this holds for X being rpo-termination, total termination and simple termination, and not for all other properties in the extended hierarchy.

The string rewrite rule $ff \to fgf$ is terminating while adding embedding rules yields a cycle $ff \to fgf \to ff$, proving that the properties termination, being non-looping and being non-cyclic all three are affected by adding embedding rules.

The string rewrite rule $fg \to hggffh$ is non-self-embedding as we proved in Proposition 8, while by adding embedding rules we obtain the self-embedding

$$fgg \to hggffhg \to^*_{Emb} ffg \to fhggffh \to^*_{Emb} fgg.$$

Hence non-self-embeddingness is affected by adding embedding rules.

The string rewrite rule $f \to f$ is not weakly normalizing while it is after adding adding embedding rules, hence weak normalization is affected by adding embedding rules.

Proposition 14 *Let R be a term rewriting system.*

- *R is rpo-terminating if and only $R \cup Emb$ is rpo-terminating;*
- *R is totally terminating if and only $R \cup Emb$ is totally terminating;*
- *R is simply terminating if and only $R \cup Emb$ is simply terminating.*

Proof: For all three assertions the 'if'-part is evident.

The 'only if'-part of the first assertion follows since $f(x_1, \ldots, x_n) >_{rpo} x_i$ by definition.

The 'only if'-part of the second and third assertion follows by replacing the corresponding monotone algebra A by the lexicographic product $A \times \mathbb{N}$ as we did in the proof of Proposition 12. $\Box$

The remaining two properties in the hierarchy are polynomial termination and ω-termination. In the next two proposition we prove that for both properties X a string rewriting system R exists satisfying X while $R \cup Emb$ does not satisfy X.

Proposition 15 *Let R consist of the string rewrite rules*

$$\begin{array}{rcl} ff & \to & gf \\ gg & \to & fgh. \end{array}$$

Then R is ω-terminating and $R \cup \{h(x) \to x\}$ is not ω-terminating.

Proof: Let $(A, >) = (\mathbb{N}, >)$ and let $f_A, g_A : A \to A$ be defined by

	$f_A(x)$	$g_A(x)$
x even	$3x+2$	$3x+1$
x odd	$3x+1$	$3x+2$

Note that both f_A and g_A are strictly monotone. Since $f_A(x)$ is even for every $x \in A$ we have $f_A(f_A(x)) > g_A(f_A(x)$ for every $x \in A$. Since $g_A(x)$ is odd for every $x \in A$ we have $g_A(g_A(x)) > f_A(g_A(x)$ for every $x \in A$. By choosing $h_A(x) = x$ for every $x \in A$ we have proved that R is ω-terminating.

Next assume that $R \cup \{h(x) \to x\}$ is ω-terminating. Then there are strictly monotone functions $f, g, h : \mathbb{N} \to \mathbb{N}$ satisfying

$$\begin{array}{rcl} f(f(x)) & > & g(f(x)) \\ g(g(x)) & > & f(g(h(x))) \\ h(x) & > & x \end{array}$$

for every $x \in \mathbb{N}$. From the last line we conclude $h(x) \geq x+1$; applying monotonicity yields

$$\begin{array}{rcll} f(f(x)) & > & g(f(x)) & (1) \\ g(g(x)) & > & f(g(x+1)) & (2) \end{array}$$

for every $x \in \mathbb{N}$. From (1) we conclude that $f(x) > g(x)$ for $x = f(1)$. From (2) and monotonicity we conclude $g(g(f(1))) > f(g(f(1)+1)) > f(g(f(1)))$, hence $f(y) < g(y)$ for $y = g(f(1))$. Since g is monotone we have $x \leq y$. From $f(x) > g(x)$, $x \leq y$ and $f(y) \leq g(y)$ we conclude that there exists $x_0 \in \mathbb{N}$ satisfying $f(x_0) > g(x_0)$ and $f(x_0+1) \leq g(x_0+1)$. We conclude

$$\begin{array}{rcll} g(f(x_0)) & > & g(g(x_0)) & \text{(by monotonicity and } f(x_0) > g(x_0)) \\ & > & f(g(x_0+1)) & \text{(by (2))} \\ & \geq & f(f(x_0+1)) & \text{(by monotonicity and } f(x_0+1) \leq g(x_0+1)) \\ & > & g(f(x_0+1)) & \text{(by (1))} \\ & > & g(f(x_0)) & \text{(by monotonicity),} \end{array}$$

contradiction. □

Proposition 16 *Let R consist of the string rewrite rules*

$$\begin{array}{rcl} fg & \to & ggf \\ hf & \to & ffh \\ gh & \to & hi. \end{array}$$

Then R is polynomially terminating and $R \cup \{i(x) \to x\}$ is not polynomially terminating.

Proof: For the polynomials f, g, h, i defined by

$$f(x) = 3x + 5,\ g(x) = x + 1,\ h(x) = x^2,\ i(x) = x$$

one easily checks $f(g(x)) > g(g(f(x)))$, $h(f(x)) > f(f(h(x)))$ and $g(h(x)) > h(i(x))$ for all $x \in \mathbb{N}$, proving that R is polynomially terminating.

Assume that $R \cup \{i(x) \to x\}$ is polynomially terminating. Then there are monotone polynomials f, g, h, i for which

$$\begin{array}{rcl} f(g(x)) & > & g(g(f(x))) \\ h(f(x)) & > & f(f(h(x))) \\ g(h(x)) & > & h(i(x)) \\ h(x) & > & x \end{array}$$

for all $x \in \mathbb{N}$. Write $deg(p)$ for the degree of a polynomial p. It is easy to prove that $deg(p) \geq deg(q)$ if $p(x) > q(x)$ for all $x \in \mathbb{N}$. From the first inequality we conclude

$$deg(f) * deg(g) = deg(f \circ g) \geq deg(g \circ g \circ f) = deg(g) * deg(g) * deg(f),$$

hence $deg(g) = 1$. Similarly we conclude from the second inequality that $deg(f) = 1$. Hence we can write $f(x) = ax + b$ and $g(x) = cx + d$. Since f and g are monotone we have $a, c > 0$; since $f, g : \mathbb{N} \to \mathbb{N}$ and $0 \in \mathbb{N}$ we have $b, d \geq 0$. Since

$$acx + ad + b = f(g(x)) > g(g(f(x))) = ac^2x + bc^2 + dc + d$$

for all $x \in \mathbb{N}$ we conclude $ac \geq ac^2$, hence $c = 1$, and $ad + b > bc^2 + dc + d$, hence $ad > 2d$. From $d \geq 0$ we now conclude $a \geq 3$. The second inequality combined with $a \geq 3$ yields a contradiction in case h is linear, hence $deg(h) > 1$. From the fourth inequality we obtain $i(x) \geq x + 1$ for all $x \in \mathbb{N}$, from the third inequality and $g(x) = x + d$ we conclude

$$h(x) + d = g(h(x)) > h(i(x)) \geq h(x + 1).$$

hence $h(x + 1) - h(x) < d$ for all $x \in \mathbb{N}$. This contradicts $deg(h) > 1$ and monotonicity of h. □

6 Reversing strings

Strings can be reversed. Write rev for string reversing, for instance we have $rev(fghhfh) = hfhhgf$. A corresponding transformation on string rewriting is reversing all left and right hand sides:

$$rev(R) = \{\, rev(l) \to rev(r) \mid l \to r \in R \,\}.$$

Now one may consider the effect of reversing on the properties in the termination hierarchy. For the properties X of being simply terminating, non-self-embedding, terminating, weakly normalizing, non-looping and non-cyclic, it is straightforward to check that X holds for a string rewriting system R if and only if X holds for $rev(R)$. In the next propositions we prove that for the four remaining properties in the hierarchy, being total termination and everything above, this does not hold.

Proposition 17 *Let R consist of the string rewrite rule $gf \to ffg$. Then R is polynomially terminating, rpo-terminating and ω-terminating, while $rev(R)$ satisfies none of these three properties.*

Proof: Polynomial termination and hence ω-termination of R follows from choosing $f_A(x) = x + 1$ and $g_A(x) = 3x$ for all $x \in \mathbb{N}$; rpo-termination follows from choosing $g > f$.

Conversely, $rev(R)$ consists of the string rewrite rule $fg \to gff$, for which it was proved in [13] that it is not ω-terminating, hence neither polynomially terminating nor rpo-terminating. $\Box$

Proposition 18 *Let R consist of the string rewrite rules*

$$\begin{array}{rcl} ff & \to & gf \\ gg & \to & fg \end{array}$$

Then R is totally terminating and ω-terminating, while $rev(R)$ satisfies neither of these properties.

Proof: In the proof of Proposition 15 we gave monotone functions $f_A, g_A : \mathbb{N} \to \mathbb{N}$ proving that R is ω-terminating and hence totally terminating.

Assume that $rev(R)$ consisting of the rules

$$\begin{array}{rcl} f(f(x)) & \to & f(g(x)) \\ g(g(x)) & \to & g(f(x)) \end{array}$$

is totally terminating with a corresponding well-founded monotone algebra $(A, >)$. Choose $a \in A$ arbitrarily. Assuming $g_A(a) > f_A(a)$ yields a contradiction with $f_A(f_A(a)) > f_A(g_A(a))$ and monotonicity of f_A. From totality we conclude $f_A(a)) \geq g_A(a)$. Monotonicity of g_A and compatibility with the second rule yields

$$g_A(f_A(a)) \geq g_A(g_A(a)) > g_A(f_A(a)),$$

contradiction. Hence $rev(R)$ is not totally terminating and hence not ω-terminating. □

Restating Proposition 6 for string rewriting we have:

> A string rewriting system is totally terminating if and only if $lq > rq$ for all rules $l \to r$ and all strings q for some total order $>$ on strings satisfying $u > v \Rightarrow pu > pv$ for all strings p, u, v.

In case only string rewriting is considered, one can prefer an alternative symmetric definition of total termination as is given by Martin in [9]:

> A string rewriting system is totally terminating if and only if $l > r$ for all rules $l \to r$ for some total order $>$ on strings satisfying $u > v \Rightarrow puq > pvq$ for all strings p, q, u, v.

Using this alternative definition a string rewriting system is totally terminating if and only if its reverse is totally terminating. It is clear that every string rewriting system that is totally terminating in the sense of Martin is totally terminating in our sense. The converse does not hold: by Proposition 18 the string rewriting system $\{ff \to gf, gg \to fg\}$ is totally terminating in our sense and its reverse is not, hence neither in the sense of Martin.

For R consisting of the rules

$$\begin{array}{rcl} fff & \to & fgf \\ ggg & \to & gfg \end{array}$$

we have that $R = rev(R)$ is ω-terminating and hence totally terminating by the same interpretation as we gave in the proof of Proposition 18. It is not difficult to see that R is not totally terminating in the sense of Martin.

7 Conclusions

We considered a termination hierarchy for first order term rewriting partly based on interpretation in monotone algebras. This kind of interpretation is conceptually easy, and allows a number of natural extensions: to rewriting modulo equations, to context-sensitive rewriting ([14]) and even to higher order rewriting ([12]).

However, we saw that the part of the hierarchy based on various kinds of interpretations is not robust in the following directions:

- weakening monotonicity requirements;

- adding embedding rules;
- symmetry in string rewriting.

More precisely, we saw that the notion of ω-termination is essentially weakened by replacing strict monotonicity by the combination of weak monotonicity and the subterm property, we saw that the meaning of both polynomial termination and ω-termination is affected by adding embedding rules, and we saw that rpo-termination, polynomial termination, ω-termination and total termination are not symmetric for string rewriting.

References

[1] Dershowitz, N. Orderings for term rewriting systems. *Theoretical Computer Science 17*, 3 (1982), 279–301.

[2] Ferreira, M. C. F., and Zantema, H. Total termination of term rewriting. *Applicable Algebra in Engineering, Communication and Computing 7*, 2 (1996), 133–162. (Preliminary version appeared in Proceedings of RTA93, Lecture Notes in Computer Science 690, Springer, 1993.).

[3] Geser, A. On normalizing, non-terminating one-rule string rewriting systems. Tech. Rep. WSI 96-34, Universität Tübingen, Oct. 1996. Accepted as a note in *Theoretical Computer Science.*

[4] Geser, A. Decidability of termination in a large class of one-rule string rewriting systems. Tech. Rep. WSI 97-11, Universität Tübingen, Aug. 1997. Submitted.

[5] Geser, A., Middeldorp, A., Ohlebusch, E., and Zantema, H. Relative undecidability in term rewriting. In *Proceedings of the Conference of the European Association of Computer Science Logic (CSL96)* (1997), D. van Dalen, Ed., vol. 1258 of *Lecture Notes in Computer Science*, Springer Verlag, pp. 150–166.

[6] Geser, A., Middeldorp, A., Ohlebusch, E., and Zantema, H. Relative undecidability in the termination hierarchy of single rewrite rules. In *Proceedings Theory and Practice of Software Development (TAPSOFT97, CAAP/FASE)* (1997), M. Bidoit and M. Dauchet, Eds., vol. 1214 of *Lecture Notes in Computer Science*, Springer Verlag, pp. 237–248.

[7] Geser, A., and Zantema, H. Non-looping string rewriting. *RAIRO - Theoretical Informatics and Applications (ITA)* (1999). To appear.

[8] Hofbauer, D. Termination proofs by multiset path orderings imply primitive recursive derivation lengths. *Theoretical Computer Science 105*, 1 (1992), 129–140.

[9] Martin, U. Theorem proving with group presentations: examples and questions. In *Proceedings of the 13th International Conference on Automated Deduction* (1996), vol. 1104 of *Lecture Notes in Computer Science*, Springer, pp. 358–372.

[10] Sénizergues, G. On the termination problem for one-rule semi-Thue systems. In *Proceedings of the 7th Conference on Rewriting Techniques and Applications* (1996), vol. 1103 of *Lecture Notes in Computer Science*, Springer, pp. 302–316.

[11] Touzet, H. Encoding the hydra battle as a rewrite system. In *Proceedings 23rd International Symposium on Mathematical Foundations of Computer Science 1998* (1998), vol. 1450 of *Lecture Notes in Computer Science*, Springer, pp. 267–276.

[12] van de Pol, J. Termination proofs for higher order rewrite systems. In *Proceedings of the First International Workshop on Higher-Order Algebra, Logic and Term Rewriting (HOA93)* (1994), vol. 816 of *Lecture Notes in Computer Science*, Springer, pp. 305–325.

[13] Zantema, H. Termination of term rewriting: interpretation and type elimination. *Journal of Symbolic Computation 17* (1994), 23–50.

[14] Zantema, H. Termination of context-sensitive rewriting. In *Proceedings of the 8th Conference on Rewriting Techniques and Applications* (1997), H. Comon, Ed., vol. 1232 of *Lecture Notes in Computer Science*, Springer, pp. 172–186.

[15] Zantema, H., and Geser, A. A complete characterization of termination of $0^p1^q \to 1^r0^s$. In *Proceedings of the 6th Conference on Rewriting Techniques and Applications* (1995), J. Hsiang, Ed., vol. 914 of *Lecture Notes in Computer Science*, Springer, pp. 41–55.

[16] Zantema, H., and Geser, A. Non-looping rewriting. Tech. Rep. UU-CS-1996-03, Utrecht University, January 1996. Available via `http://www.cs.ruu.nl/docs/research/publication/TechList2.html`.