Solution to the Problem of Zantema on a Persistent Property of Term Rewriting Systems*

Takahito Aoto

Department of Computer Science, Gunma University
Tenjincho 1-5-1, Kiryuu, 376-8515, Japan

E-mail: aoto@cs.gunma-u.ac.jp

Abstract

A property P of term rewriting systems is persistent if for any many-sorted term rewriting system R, R has the property P if and only if its underlying term rewriting system $\Theta(R)$, which results from R by omitting its sort information, has the property P. It is shown that termination is a persistent property of many-sorted term rewriting systems that contain only variables of the same sort.

1 Introduction

The technique of sort introduction in term rewriting has caught attention recently [1][9]. In this paper we prove a conjecture which opens up a possibility of new applications of this technique. The conjecture reads: for any terminating many-sorted term rewriting system $\mathcal{R}$, if $\mathcal{R}$ contains only variables of the same sort then $\Theta(\mathcal{R})$ is also terminating. Here, $\Theta(\mathcal{R})$—called

*This is a completely revised version of the paper *Solution to the Problem of Zantema on a Persistent Property of Term Rewriting Systems*, which appeared in *Proceedings of the Joint International Symposium PLILP/ALP'98*, Pisa, Italy, September 1998, pages 250–265, Lecture Notes in Computer Science 1490, Springer-Verlag, 1998.

the underlying term rewriting system of $\mathcal{R}$—is the term rewriting system obtained from $\mathcal{R}$ by omitting its sort information. The conjecture was raised by Zantema and adapted in [4] as Rewriting Open Problem 60. The conjecture is particularly useful to show that termination is preserved under suitable translations of term rewriting systems (*TRSs*, for short). More precisely, it follows that coding function symbols of higher arity in an arbitrary TRS by binary symbols does not affect termination. Hence the problem of termination of arbitrary TRSs is equivalent to the problem of TRSs having only symbols of arity at most 2.

A property ϕ of TRSs is said to be *persistent* if for any TRS $\mathcal{R}$, $\phi(\mathcal{R})$[1] if and only if $\phi(\Theta(\mathcal{R}))$; this notion has been introduced by Zantema in [13]. We say that ϕ *is persistent for a class* $\mathcal{K}$ *of many-sorted TRSs* if for any $\mathcal{R} \in \mathcal{K}$, $\phi(\mathcal{R})$ if and only if $\phi(\Theta(\mathcal{R}))$. Thus the conjecture above can be restated as follows: termination is persistent for the class of many-sorted TRSs that contain only variables of the same sort. It is known that confluence is persistent for the class of many-sorted TRSs [2] and that termination is persistent for the class of many-sorted TRSs that do not contain both duplicating and collapsing rules [13]. Persistent properties in equational rewriting have been studied in [10].

A property ϕ of TRSs is said to be *modular for the direct sum* if $\phi(\mathcal{R}_1)$ and $\phi(\mathcal{R}_2)$ imply $\phi(\mathcal{R}_1 \cup \mathcal{R}_2)$ for any two TRSs $\mathcal{R}_1$ and $\mathcal{R}_2$ sharing no function symbols. For component closed properties ϕ, persistency of ϕ for the class of many-sorted TRSs implies modularity of ϕ for the direct sum of TRSs [13]. For many useful properties ϕ of TRSs, not only modularity of ϕ for the direct sum but modular aspects of ϕ for various combinations of TRSs have been studied in a number of papers; see e.g. [8], [11], [12] for the modularity of termination. Modular aspects of a combination of TRSs under a suitable sort assignment condition have been studied in [1].

It is not always true but often so that persistency of property ϕ for a subclass $\mathcal{K}$ of many-sorted TRSs implies modularity of ϕ for the direct sum of TRSs from the class $\Theta(\mathcal{K}) = \{\Theta(\mathcal{R}) \mid \mathcal{R} \in \mathcal{K}\}$. Indeed, if we assign sorts $1 \times \cdots \times 1 \to 1$ to function symbols in $\mathcal{R}_1 \in \Theta(\mathcal{K})$ and $2 \times \cdots \times 2 \to 2$ to those in $\mathcal{R}_2 \in \Theta(\mathcal{K})$, the statement $\phi(\mathcal{R}_1)$ and $\phi(\mathcal{R}_2)$ imply $\phi(\mathcal{R}_1 \cup \mathcal{R}_2)$ follows from the persistency of ϕ provided that $\mathcal{R}_1 \cup \mathcal{R}_2$ under this sort assignment is also contained in $\mathcal{K}$. Thus, for example, persistency of termination for the class of many-sorted TRSs that do not contain both duplicating and collapsing rules

[1] This stands for $\mathcal{R}$ *has the property* ϕ.

implies the corresponding modularity result for the direct sum of TRSs. To the contrary, our requirement on the sorts of variables (*variable-the-same-sort* condition) does not carry over for TRSs and therefore it is hard to formalize the corresponding modularity result. This contrasts sharply with other persistency results obtained so far.

The rest of this paper is organized as follows. In Section 2, we fix notion and notations on many-sorted term rewriting used in this paper. In Section 3, we review how unsorted rewriting (or, more generally, rewriting over sorts where some sorts are identified) is characterized by sort information. We simplify the conjecture at the end of this section. Section 4 is devoted to the rest of the proof. In Section 5, applications of our theorem and related works are discussed. Our conclusions are presented in Section 6.

2 Preliminaries

We assume familiarity with basic notions in term rewriting. In what follows, we fix notations used in this paper and recall some less common definitions. We refer [3] and [6] for extensive surveys.Let $\mathcal{S}$ be a set of sorts (denoted by $\alpha, \beta, \gamma, \ldots$), $\mathcal{F}$ a set of $\mathcal{S}$-sorted function symbols (denoted by $f, g, h, \ldots$), $\mathcal{V}$ a set of $\mathcal{S}$-sorted variables (denoted by $x, y, z, \ldots$). We write $f : \alpha_1 \times \cdots \times \alpha_n \to \beta$ when $f \in \mathcal{F}$ has sort $\alpha_1 \times \cdots \times \alpha_n \to \beta$. We assume that there are countably infinite variables of sort α for each sort $\alpha \in \mathcal{S}$. We denote by $\mathcal{T}$ (and $\mathcal{T}^\alpha$), or $\mathcal{T}(\mathcal{F}, \mathcal{V})$ (and $\mathcal{T}(\mathcal{F}, \mathcal{V})^\alpha$) when necessary, the set of terms (of sort α, respectively). For each term t, $\mathcal{V}(t)$ is the set of variables that appear in t. Syntactical equality is denoted by $\equiv$.

For each sort α, the *hole* of sort α is written as $\Box^\alpha$. A *context* is a term possibly containing holes. We denote by $\mathcal{C}$ the set of contexts. We write $C : \alpha_1 \times \cdots \times \alpha_n \to \beta$when $C \in \mathcal{C}$ has sort β (as a term) and has n holes $\Box^{\alpha_1}, \ldots, \Box^{\alpha_n}$ from left to right in it. If $C : \alpha_1 \times \cdots \times \alpha_n \to \beta$ and $t_1 \in \mathcal{T}^{\alpha_1}, \ldots, t_n \in \mathcal{T}^{\alpha_n}$ then $C[t_1, \ldots, t_n]$ is the term obtained from C by replacing the holes with $t_1, \ldots, t_n$ from left to right. A context C is written as $C[\,]$ when C contains precisely one hole. A context is said to be *empty* when C is a hole, otherwise *non-empty*. A term t is a *(proper) subterm of a term* s, or equivalently s *(properly) contains* t when $s \equiv C[t]$ (with non-empty C, respectively).

We denote by $Pos(t)$ the set of *positions* of a term t, and by t/p the subterm of t at position $p \in \mathrm{Pos}(t)$. The *empty* (or *root*) position is denoted

by Λ. For $u, v \in \mathrm{Pos}(t)$, we write $u \leq v$ when u is a prefix of v. For a context C, we write $C[,\ldots,]_{p_1,\ldots,p_n}$ when $C/p_1, \ldots, C/p_n$ are all holes in C from left to right.

A *substitution* σ is a mapping from $\mathcal{V}$ to $\mathcal{T}$ such that x and $\sigma(x)$ have the same sort for any $x \in \mathcal{V}$. A substitution is extended to a homomorphism from $\mathcal{T}$ to $\mathcal{T}$ in the obvious way; $t\sigma$ stands for $\sigma(t)$ for substitutions σ and terms t.

A *(many-sorted) rewrite rule* is a pair $l \to r$ of terms such that (1) l and r have the same sort, (2) $l \notin \mathcal{V}$, (3) $\mathcal{V}(r) \subseteq \mathcal{V}(l)$. A *many-sorted term rewriting system* (*STRS*, for short) is a set of rewrite rules. A rewrite rule $l \to r$ is *collapsing* if $r \in \mathcal{V}$; it is *duplicating* if r contains more occurrences of some variable than l does. Given an STRS $\mathcal{R}$, a term s reduces to a term t (denoted by $s \to_{\mathcal{R}} t$) when $s \equiv C[l\sigma]_p$ and $t \equiv C[r\sigma]_p$ for some $C[\ ]_p \in \mathcal{C}$, $l \to r \in \mathcal{R}$ and substitution σ. When this is the case, we also write $s \to_p t$. We call $s \to_{\mathcal{R}} t$ a *rewrite step*, and the relation $\to_{\mathcal{R}}$ *reduction (relation)*. The *(rewrite) redex* of this rewrite step is $l\sigma$.

Suppose $C_l, C_l' \in \mathcal{C}$ contain no variables. Redexes according to rules $C_l[\vec{x}] \to r$, $C_l'[\vec{y}] \to r'$ *overlap* if there is a symbol f that is a simultaneous instance of f in C_l and C_l'. A redex according to a rule $C_l[\vec{x}] \to r$ *contains* a redex according to a rule $C_l'[\vec{y}] \to r'$ if each instance of a symbol in C_l' is also an instance in C_l. A redex in s according to a rule $C_l[\vec{x}] \to r$ contains a position q if the root symbol of s/q is an instance of a non-root function symbol in C_l.

A sequence $s_0 \to_{\mathcal{R}} \cdots \to_{\mathcal{R}} s_n \to_{\mathcal{R}} \cdots$ is called a reduction sequence. The *length of a finite reduction sequence* $s_0 \to_{\mathcal{R}} \cdots \to_{\mathcal{R}} s_n$ equals to n. Reduction sequences may be infinite.An STRS $\mathcal{R}$ is *terminating* if there are no infinite reduction sequences. A term s is terminating when there are no infinite reduction sequences starting from s.

The reflexive and transitive closure of $\to_{\mathcal{R}}$ is denoted by $\twoheadrightarrow_{\mathcal{R}}$. One can readily check that s and t have the same sort whenever $s \twoheadrightarrow_{\mathcal{R}} t$.

Henceforth, the subscript $_{\mathcal{R}}$ will be omitted when $\mathcal{R}$ is obvious from the context.

3 Sort Information in Reduction and Terms

Let Φ be an equivalence relation on $\mathcal{S}$. By identifying sorts according to Φ, we obtain the set $\Phi(\mathcal{T})$ of terms well-sorted w.r.t. $\mathcal{S}/\Phi$ ($\mathcal{S}$ modulo Φ).

Likewise, from an STRS $\mathcal{R}$ over sorts $\mathcal{S}$, we obtain an STRS $\Phi(\mathcal{R})$ over sorts $\mathcal{S}/\Phi$: an STRS that has the same rules as $\mathcal{R}$ and acts over $\Phi(\mathcal{T})$. Note that terms in $\Phi(\mathcal{T})$ are possibly non-well-sorted w.r.t. $\mathcal{S}$, but terms in $\mathcal{T}$ are well-sorted w.r.t. $\mathcal{S}/\Phi$.The equivalence relation that identifies all sorts is denoted by Θ. We call $\Theta(\mathcal{R})$ the *underlying* TRS of $\mathcal{R}$.

Below, we assume $\mathcal{S}$, $\mathcal{F}$, Φ, $\mathcal{R}$ are fixed. Terms in $\Phi(\mathcal{T})$ are referred to just terms, and $\to_{\Phi(\mathcal{R})}$ is abbreviated as $\to$. Otherwise mentioned, well-sorted terms (etc.) are meant to be well-sorted w.r.t. $\mathcal{S}$.

Definition 3.1 *1. The sort of a term t is defined by*

$$\mathrm{sort}(t) = \begin{cases} \alpha & \text{if } t \equiv x^{\alpha}, \\ \beta & \text{if } t \equiv f(t_1, \ldots, t_n) \text{ with } f : \alpha_1 \times \cdots \times \alpha_n \to \beta. \end{cases}$$

2. Let p be a non-empty position in a term t. We define the sort of p in t like this:

$$\begin{array}{lcl} \mathrm{sort}(f(t_1, \ldots, t_n), i) & = & \alpha_i \quad \text{if } f : \alpha_1 \times \cdots \times \alpha_n \to \beta, \\ \mathrm{sort}(f(t_1, \ldots, t_n), i.w) & = & \mathrm{sort}(t_i, w), \end{array}$$

where $i \in \{1, \ldots, n\}$ and $w \neq \Lambda$.

3. A non-empty position p in t is a disconnection *if* $\mathrm{sort}(t, p) \neq \mathrm{sort}(t/p)$. *When p is a disconnection, the proper subterm t/p is* disconnected, *otherwise* connected. *A subterm of t is* special *if either it is t or it is disconnected.*

We note that $\mathcal{R}$ is well-sorted w.r.t. $\mathcal{S}$ and hence rewrite redexes contain no disconnections.

Definition 3.2 *Let $t \equiv C[t_1, \ldots, t_n]$ be a term with non-empty well-sorted context C. We write $t \equiv C[\![t_1, \ldots, t_n]\!]$ when all t_i's are disconnected. Terms $t_1, \ldots, t_n$ are the* principal *subterms of t. C is the* top well-sorted component *of t, denoted by* $\mathrm{cap}(t)$. *A top well-sorted component of special subterms of t is a well-sorted component in t.*

Thus, each term is partitioned into well-sorted components by the sort information.

Example 3.3 *Let*

$$\begin{cases} f : 2 \times 1 \to 1 \\ g : 0 \to 1 \\ h : 0 \times 2 \to 0 \\ a, b : 2 \end{cases}$$

and $\Phi = \{\{0, 2\}, \{1\}\}$. *Let* $t \equiv f(h(a, b), g(b))$. *Then* $\mathrm{cap}(t) \equiv f(\Box, g(\Box))$[2], *and* t *has two principal subterms* $h(a, b)$ *and* b. *The term* t *has* 4 *well-sorted components.*

Definition 3.4 *A rewrite step*

$$s \equiv C[l\sigma] \to C[r\sigma] \equiv t$$

is destructive *if* $l\sigma$ *is a special subterm of* s *and* $\mathrm{sort}(l\sigma) \neq \mathrm{sort}(r\sigma)$.

The next lemma is shown in a straightforward way.

Lemma 3.5 *If a rewrite step* $s \equiv C[u]_p \to_p C[v]_p \equiv t$ *is destructive, then the applied rule is collapsing and* v *is a principal subterm of* u.

When this is the case, we say that $\mathrm{cap}(u)$ is *eliminated*, or the subterm u *collapses*. If p is non-empty and $\mathrm{sort}(s, p) = \mathrm{sort}(v)$, v becomes connected along this rewrite step, resulting in a new well-sorted component.

Example 3.6 *Let our signature,* Φ, *and the term* t *be the ones given in Example 3.3. Let*

$$\mathcal{R} \begin{cases} f(a, g(x)) \to g(x) \\ h(x, b) \to x \end{cases}$$

The following is a reduction sequence of $\Phi(\mathcal{R})$*:*

$$\begin{array}{rcl} t & \to & f(a, g(b)) \\ & \to & g(b) \end{array}$$

The first rewrite step is destructive and the second is not.

Definition 3.7 *The* rank *of a term* t *is defined by*

$$\mathrm{rank}(t) = \begin{cases} 1 & \text{if } t \in \mathcal{T}, \\ 1 + \max\{\mathrm{rank}(t_i) \mid 1 \leq i \leq n\} & \text{if } t \equiv C[\![t_1, \ldots, t_n]\!] \ (n > 0). \end{cases}$$

We note that $\mathrm{rank}(s) \geq \mathrm{rank}(t)$ whenever $s \to t$. The rank of a reduction sequence is the rank of its starting term.

[2]In the sequel, we assume that every hole (at a non-empty position) has sort equivalent to the sort of its position, and omit the superscripts of holes.

Simplification of the conjecture We prove in this paper that termination of $\mathcal{R}$ and $\Theta(\mathcal{R})$ coincide, for STRSs $\mathcal{R}$ that contain only variables of the same sort. Such STRSs may have an arbitrary number of sorts other than the (unique) sort of variables. It is, however, sufficient to restrict our attention to the two-sorted case.

Lemma 3.8 *Let $\mathcal{R}$ be an STRS whose collapsing rules are of the same sort, namely 0. Then $\mathcal{R}$ is terminating if and only if $\Phi_I(\mathcal{R})$ is terminating, where Φ_I identifies all sorts except 0.*

Proof. ($\Leftarrow$) Trivial. ($\Rightarrow$) Suppose that $\mathcal{S} = \{0, \ldots, n\}$ and Φ_I identifies sorts $1, \ldots, n$ with 1. Let $\alpha : s_0 \to s_1 \to \cdots$ be an infinite reduction sequence over terms well-sorted on sort 0 and 1 (1 for the identified sort). Each term can be partitioned into well-sorted components w.r.t. sorts $0, \ldots, n$. Since we only have identified sorts $1, \ldots, n$ with 1, for each disconnection p in s_i, $\mathrm{sort}(s_i, p)$ differs from 0. Thus, by our assumption and Lemma 3.5, there are no destructive rewrite steps in α. W.l.o.g, we may take α of a minimal rank, that is, any reduction sequence of smaller rank is terminating. Then all principal subterms in $s_0, s_1, \ldots$ are terminating, and since there are no destructive rewrite steps in α, an infinite number of rewrite steps is performed at the top well-sorted components. Finally, to obtain an infinite reduction sequence well-sorted w.r.t. sort $0, \ldots, n$, for each s_i, replace all principal subterms at positions of sort $j_1, j_2, \ldots$ with $x^{j_1}, x^{j_2}, \ldots$, where $x^1, \ldots, x^n$ are new variables of sort $1, \ldots, n$. ∎

As a special case of the lemma, we know termination of $\mathcal{R}$ and $\Phi_I(\mathcal{R})$ coincide for STRSs $\mathcal{R}$ that contain only variables of the same sort. Thus, we may assume that **our signature is two-sorted, namely 0 and 1, and rewrite rules contain only variables of the sort 0**.

As in Lemma 3.8, one might think that the variable-single-sorted condition of our theorem (to be proved) may be weakened to the condition that all collapsing rules are of the same sort. This is not true, as observed by the following well-known Toyama's counterexample to the modularity of termination for the direct sum:

$$\mathcal{R} \left\{ \begin{array}{l} f(a, b, x) \to f(x, x, x) \\ h(x, y) \to x \\ h(x, y) \to y \end{array} \right.$$

If we let

$$\begin{cases} f : 1 \times 1 \times 1 \to 1 \\ h : 0 \times 0 \to 0 \\ a, b : 1 \end{cases}$$

the STRS $\mathcal{R}$ is terminating, but $\Theta(\mathcal{R})$ is not.

4 Projecting Reduction Sequences

¿From now on, w.l.o.g., we assume that terms contain no variables. We also set $\mathrm{sort}(t, \Lambda) = 1$ for any term t and extend the notion of (dis)connection accordingly.

Definition 4.1 *1. The* type *of a position p in a term t is determined as follows:*

$\mathrm{sort}(t,p)$	$\mathrm{sort}(t/p)$	*type*	
0	*0*	*0*	*(zero)*
0	*1*	$\rrbracket$	*(right-bracket)*
1	*0*	$\llbracket$	*(left-bracket)*
1	*1*	*1*	*(one)*

2. A bracket *is a left- or a right-bracket.*

3. Let $p < q$ be positions of $\llbracket$ and $\rrbracket$. Then $[p, q]$ is called a (matching bracket-)pair *at p if all brackets between p and q are member of some matching bracket-pair $[p', q']$ with $p < p'$ and $q' < q$.*

4. If $t \equiv C'[C[t_1, \ldots, t_n]_{q_1,\ldots,q_n}]_q$ ($n \geq 0$), where q is the position of $\llbracket$ and the $[q, q.q_i]$, for $i \in \{1, \ldots, n\}$, are all pairs at q, then C is the layer *at q in t.*

Like for strings every right-bracket matches at most one left-bracket,but, due to the tree structure of terms, a left-bracket may be matched by any number of right-brackets.

Example 4.2 *Let*

$$\begin{cases} f : 0 \times 1 \to 0 \\ g : 1 \to 0 \\ h : 0 \times 1 \to 1 \\ a, b : 1 \end{cases}$$

and $\Phi = \Theta$.

1. *Let* $s \equiv g(h(a, f(a, b)))$. *The term* s *has matching bracket-pairs* $[\Lambda, 1.1]$ *and* $[1.2, 1.2.1]$*;* s *has a layer at position* Λ*, namely* $g(h(\Box, f(a, b)))$ *and a layer at position* 2.1*, namely* $f(\Box, b)$.

2. *Let* $t \equiv g(f(a, f(h(a, b), b)))$. *The term* t *has matching bracket-pairs* $[1, 1.1]$*,* $[1, 1.2.1.1]$*, and* $[1.2, 1.2.1]$*;* t *has a layer at position* 1*, namely* $f(\Box, f(h(\Box, b), b))$ *and a layer at position* 1.2*, namely* $f(\Box, b)$.

Lemma 4.3 *If* $s \equiv C'[C[u]_q]_p \to_p C'[u] \equiv t$ *is destructive, then* $[p, q]$ *is a matching bracket-pair in* s *and* u *is connected in* t.

Proof. By Lemma 3.5, the applied rule is collapsing. Thus, by our assumption that the rewrite rules contain only variables of sort 0, $\mathrm{sort}(s/p) = \mathrm{sort}(s, p.q) = 0$. Since u is a principal subterm of $C[u]$ by Lemma 3.5, and by our assumption that our signature is two-sorted (and $\mathrm{sort}(s, \Lambda) = \mathrm{sort}(t, \Lambda) = 1$), we know $\mathrm{sort}(s, p) = \mathrm{sort}(t, p) = \mathrm{sort}(u) = 1$. Thus, u is connected in t, and since rewrite redexes contain no disconnections, $[p, q]$ is a matching bracket-pair. ∎

Example 4.4 *Let our signature and the term* t *be the ones given in Example 4.2. Let*

$$\mathcal{R} \left\{ \begin{array}{l} f(x, b) \to x \\ h(y, b) \to b \end{array} \right.$$

The following is a reduction sequence of $\Theta(\mathcal{R})$*:*

$$\begin{array}{rcl} t & \to & g(f(a, h(a, b))) \\ & \to & g(f(a, b)) \\ & \to & g(a) \end{array}$$

In the first and third rewrite steps, matching bracket-pairs $[1.2, 1.2.1]$ *and* $[1, 1.1]$ *'collapse'.*

Definition 4.5 *Any layer* C *at a position* $q = pj$ *forms a* d(istribution)-redex *at* p*. The corresponding* d-step $(\to^d)$ *is an instance of the following (schematic) d-rule.*

$$f(\vec{x}, C[\overrightarrow{y_i}]_{\vec{q_i}}, \vec{z}) \to^d C[\overrightarrow{f(\vec{x}, y_i, \vec{z})}]_{\vec{q_i}}$$

f *is the* head *(symbol) of the d-redex, and* C *is the layer of the d-redex.*

Although we will use TRS-notions for d-rewriting, d-rules are not applied like ordinary rewrite rules: a d-rule applies to a term only when C is a d-redex in that term.

Example 4.6 *Let our signature and the terms s, t be the ones given in Example 4.2. One can perform d-steps from s, t like this:*

$$\begin{array}{rcl} s & \to^d & g(f(h(a,a),b)) \\ & \to^d & f(g(h(a,a)),b) \end{array}$$

$$\begin{array}{rcl} t & \to^d & g(f(f(a,h(a,b)),b)) \\ & \to^d & f(f(g(a),h(g(a),b)),b) \equiv u \end{array}$$

We have a reduction sequence starting from u that 'simulates' the one in Example 4.4:

$$\begin{array}{rcl} u & \to & f(g(a),h(g(a),b)) \\ & \to & f(g(a),b) \\ & \to & g(a) \end{array}$$

The idea is that brackets can be used to statically determine the collapse behavior a term will have in the future. The goal of distribution is to decrease the 'bracket depth' of a term, preserving infinite reduction sequences which are possible from it.

Example 4.7 *Let our signature be*

$$\begin{cases} f : 0 \times 0 \times 0 \to 0 \\ g : 1 \to 0 \\ h : 0 \times 0 \to 0 \\ a, b : 1 \end{cases}$$

Let

$$\mathcal{R} \begin{cases} f(g(a),g(b),x) \to f(x,x,x) \\ h(x,y) \to x \\ h(x,y) \to y \end{cases}$$

Then, $\Theta(\mathcal{R})$ has the following infinite reduction sequence:

$$\begin{array}{rcl} f(g(a),g(b),g(h(a,b))) & \to & f(g(h(a,b)),g(h(a,b)),g(h(a,b))) \\ & \to & f(g(a),g(h(a,b)),g(h(a,b))) \\ & \to & f(g(a),g(b),g(h(a,b))) \to \cdots \end{array}$$

By projecting this reduction sequence over d-steps, we obtain the following infinite reduction sequence of $\mathcal{R}$.

$$\begin{array}{rcl} f(g(a),g(b),h(g(a),g(b))) & \to & f(h(g(a),g(b)),h(g(a),g(b)),h(g(a),g(b))) \\ & \to & f(g(a),h(g(a),g(b)),h(g(a),g(b))) \\ & \to & f(g(a),g(b),h(g(a),g(b))) \to \cdots \end{array}$$

Lemma 4.8 *1. If a rewrite redex and a d-redex overlap, then the rewrite redex is contained in the layer of the d-redex.*

2. If d-redexes overlap then either their head positions are identical or one of the d-redexes is contained in the layer of the other.

Proof. Straightforward. ∎

Below, we let *objects* range over elements of any kind in { left-bracket, right-bracket, (matching bracket-)pair, layer, d-redex }.

Definition 4.9 *Let $s \equiv C[l\sigma]_p \to C[r\sigma]_p \equiv t$ be a rewrite step. A position o in s traces to*

1. the position o in t if $o \not> p$,

2. all positions pp_ro' in t such that $l/p_l \equiv x \equiv r/p_r$ if $o = pp_lo'$.

For d-steps the definition is as for rewrite steps modified as follows (notation from Definition 4.5):

1. any position po' traces to positions of the form pq_io' if $o' \not\geq j$.

2. any position pjo' traces to the position po' if $o' \geq q_i$ for no i,

3. any position pjq_io' traces to the position pq_ijo'.

Symbols trace via their positions; objects trace via their positions, with the constraint that the result is of the same kind again.

Something which does not trace to anything is said to be *eliminated*. *Tracing back* and *creation* are the inverse of tracing and elimination.

Lemma 4.10 *1. Any object traces back to some object of the same kind along rewrite steps, and the objects' $<$-order on positions is preserved.*

2. *Each symbol traces to the same symbol along d-steps.*

3. *Any d-redex other than the contracted one traces to (one or more) d-redexes along d-steps.*

Proof. 1. Use Lemma 4.3 and Lemma 4.8. 2. Straightforward. 3. Use Lemma 4.8. ∎

By (the first item of) the lemma and the fact that $p < q$ holds for any matching bracket-pair $[p, q]$, one can always insert a dummy symbol of sort $1 \to 0$ at positions of unmatched $]$'s without affecting rewriting, so we may assume: **All $]$'s in a term are matched.**

To show that projecting an infinite reduction sequence over d-steps yields an infinite reduction sequence again, first, a strategy for d-rewriting is defined which yields unique d-normal forms.

Termination of d-rewriting We show d-rewriting terminates, for any strategy.

Definition 4.11 *We let $\mathcal{E}xt$ a new alphabet extended with unary function symbols $[,], 0$, for which prefix notation is used.*

1. *A translation is defined from terms to terms over $\mathcal{E}xt$ ($\mathcal{E}$-terms) like this: Insert at every position in a term t except of type 1, its type as symbol resulting in $\tilde{t}$.*

2. *The* depth *of a symbol from $\mathcal{F}$ in an $\mathcal{E}$-term is the number of $[$-occurrences minus that of $]$-occurrences on the path from Λ to its occurrence.*

3. *The rewrite relation $\to^e$ on $\mathcal{E}$-terms is generated by:*

$$\begin{array}{rcl} f(\vec{x}, [C[\overrightarrow{]y_i}], \vec{z}) & \to^{\tilde{d}} & [C[\overrightarrow{]f(\vec{x}, y_i, \vec{z})}] \\ 0[C[\overrightarrow{]y_i}] & \to^{0} & 0C[\overrightarrow{0y_i}] \\ ][C[\overrightarrow{]y_i}] & \to^{]} & 0C[\overrightarrow{]y_i}] \\ [[C[\overrightarrow{]y_i}] & \to^{[} & [C[\overrightarrow{0y_i}] \end{array}$$

where $[C[\overrightarrow{]y_i}]$ lists all $]$'s that match with this $[$. U(pdate)-steps, $\to^u$ are generated by the last three rules.

We note that $\to^u$ is size-decreasing, hence terminating.

Tracing of symbols from $\mathcal{F}$ along $\tilde{d}$-steps is defined by the corresponding d-steps. It is easy to extend Lemma 4.10 as follows:

Lemma 4.12 *Each symbol from $\mathcal{F}$ traces to that of the same depth along $\tilde{d}$-steps.*

Lemma 4.13 *$\to^d$ is terminating.*

Proof. If $s \to^d t$, then $\tilde{s} \to^{\tilde{d}} \to^u \tilde{t}$, since $\to^d$ is simulated by $\to^{\tilde{d}}$ after which type information is updated by the $\to^u$ step. (The rule used at the updating step depends on the sort of $f(\ldots)$ in s and that of the position of $f(\ldots)$ in s.) Hence it suffices to show termination of $\to^e$.

Updating can be postponed in $\to^e$: $s \to^u \to^{\tilde{d}} t$ implies $s \twoheadrightarrow^{\tilde{d}} \twoheadrightarrow^u t$. There are two 'critical postponent pairs': (below, to simplify the illustration, we assume f is unary and C has only one matching right-bracket.)

$$f([\![[\![C[\!]D[\!]\overrightarrow{]\!]u_i}]\!]) \to^{[\![} f([\![C[0D[\!]\overrightarrow{]\!]u_i}]\!]) \to^{\tilde{d}} [\![C[0D[\overrightarrow{f(]\!]u_i})]\!]$$

and

$$f([\![C[\!][\![D[\!]\overrightarrow{]\!]u_i}]\!]) \to^{]\!]} f([\![C[0D[\!]\overrightarrow{]\!]u_i}]\!]) \to^{\tilde{d}} [\![C[0D[\!]\overrightarrow{f(u_i)}]\!]$$

Each can be resolved as

$$f([\![[\![C[\!]D[\!]\overrightarrow{]\!]u_i}]\!]) \to^{\tilde{d}} [\![[\![C[\!]D[\!]\overrightarrow{f(u_i)}]\!]]\!]) \to^{[\![} [\![C[0D[\!]\overrightarrow{f(u_i)}]\!]])$$

and

$$f([\![C[\!][\![D[\!]\overrightarrow{]\!]u_i}]\!]) \to^{\tilde{d}} [\![C[\!]f([\![D[\!]\overrightarrow{]\!]u_i}])] \to^{\tilde{d}} [\![C[\!][\![D[\!]\overrightarrow{f(u_i)}]\!]] \to^{]\!]} [\![C[0D[\!]\overrightarrow{f(u_i)}]\!]]$$

Hence u-steps can be postponed and since they are terminating, it remains to show termination of $\to^{\tilde{d}}$.

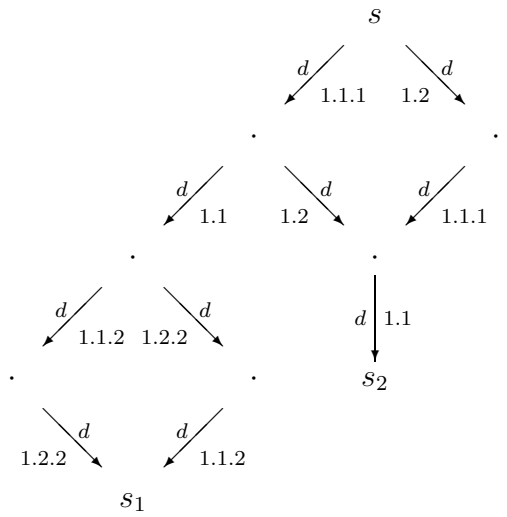
Let $s_0 \to^{\tilde{d}} s_1 \to^{\tilde{d}} \cdots$ be an infinite reduction sequence. Label each symbol from $\mathcal{F}$ by its depth. Let m be the maximal depth of s_0. By Lemma 4.12, s_i's are all contained in the set of terms over $\{f_i \mid f \in \mathcal{F}, 0 \le i \le m\} \cup \{0, 1, [\![,]\!]\}$. We set $f_i \ge g_j$ iff $i \le j$, and $f_i > 0, [\![,]\!]$, which results in a well-founded order on $\{f_i \mid f \in \mathcal{F}, 0 \le i \le m\} \cup \{0, [\![,]\!]\}$. Using Lemma 4.12, it is checked that $s_i \to^{\tilde{d}} s_{i+1}$ implies $s_i >_{rpo} s_{i+1}$, where $>_{rpo}$ is the recursive path-order generated by $>$.This is a contradiction. ∎

Confluence of the locally-leftmost strategy Due to overlap between heads of d-steps, d-rewriting is not confluent. A confluent strategy will have to avoid such overlaps (in the future).

Example 4.14 *Let our signature be*

$$\begin{cases} f : 1 \times 1 \to 0 \\ g : 1 \to 1 \\ h : 0 \times 0 \to 0 \\ k : 0 \to 0 \\ a, b : 1 \end{cases}$$

D-steps starting from a term $s \equiv k(f(g(h(a,b)),k(a)))$ *are illustrated in the Figure below. Note that* s *contains d-redexes at positions* 1.1.1 *and* 1.2 *having distinct heads. There are two d-normal forms of* s*, namely* $s_1 \equiv k(h(k(f(g(a),a)),k(f(g(b),a))))$ *and* $s_2 \equiv k(k(h(f(g(a),a),f(g(b),a))))$*.*



Definition 4.15 *1. The* virtual *head* $\hat{q}$ *of a position* q *in* t *is* $\hat{p}$ *if* $q = pj$ *and has type 1, and* q *otherwise. Virtual heads of layers are defined via their positions.*

2. A d-cluster *at a position* q *in* t *is the set of all d-redexes in* t *having* q *as virtual head.*

3. *A* locally-leftmost strategy $\rightarrow^{ll}$ *may only contract a d-redex which is leftmost among all d-redexes in the same d-cluster.*

Thus, in Example 4.14, we only obtain s_1 by the locally-leftmost strategy. One easily extend Lemma 4.10 as follows:

Lemma 4.16 *Any ll-redex other than the contracted one traces to ll-redexes (if any) along ll-steps.*

Lemma 4.17 $\rightarrow^{ll}$ *is confluent.*

Proof. Since $\rightarrow^{d}$ is terminating, so is $\rightarrow^{ll}$, and thus it suffices to show that $\rightarrow^{ll}$ is locally confluent. Since there are no two ll-redexes sharing a head, a common reduct of two ll-steps at p and $q \neq p$ can be found by contracting only redexes to which they trace. Using Lemma 4.16, one ensures they are again ll-redexes. ∎

Theorem 4.18 *Every term t has a unique d-normal form, denoted by $ll(t)$.*

Any deterministic strategy would trivially yield confluence, but such strategies generally do not commute with rewrite steps (needed in Lemma 4.22).

Projection over locally-leftmost steps First we show that ordinary rewrite steps commute with ll-reductions to d-normal form, so we can speak of about the projection of the former over the latter.

Definition 4.19 *The* depth *of a position p in a term t is the number of positions of d-clusters on the path from Λ to p. The depth of t is the maximal depth of its positions. A dead-step, $\rightarrow^{\dagger}$ is a rewrite step with redex at depth* 0.

Lemma 4.20 *The depth of a term does not increase by rewriting.*

Proof. Use Lemma 4.8. ∎

The depth of a reduction sequence is the depth of its starting term.

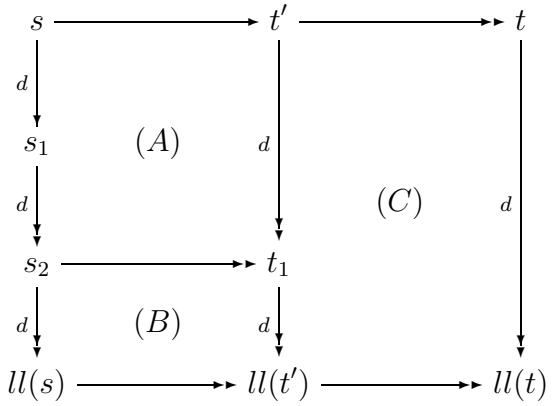
Lemma 4.21 *If* $s_1 \leftarrow^{ll} s \to t$, *then* $s_1 \twoheadrightarrow^{ll} s_2 \twoheadrightarrow t_1 \twoheadleftarrow^{ll} t$. *If* $s \to^{\dagger} t$, *then* $s_2 \to^{\dagger} t_1$.

Proof. Straightforward, using Lemma 4.8. Note that ll-redexes trace to ll-redexes (if any) along rewrite steps. ∎

Note that compensating ll-steps are needed in case of a non-left-linear rule in $\mathcal{R}$. This is the reason to project over ll-reductions to d-normal form instead of over single d-steps.

Lemma 4.22 *If* $s \to t$, *then* $ll(s) \twoheadrightarrow ll(t)$. *If* $s \to^{\dagger} t$, *then* $ll(s) \to^{\dagger} ll(t)$.

Proof. For any term u, let $\sharp u = \max\{n \mid n \text{ is the length of } u \twoheadrightarrow^{d} ll(u)\}$. Note $\sharp u$ is well-defined since $\to^{d}$ is terminating and every term contains only finitely many d-redexes. We show $s \twoheadrightarrow t$ implies $ll(s) \twoheadrightarrow ll(t)$ by double induction on $\sharp s$ and the length of $s \twoheadrightarrow t$. When $\sharp s = 0$, the statement follows from Lemma 4.10. For the induction step, suppose $\sharp s \neq 0$. When $s \equiv t$, we have nothing to do; so, suppose $s \to t' \twoheadrightarrow t$. Since s is not d-normal, by Lemma 4.21, $s \to^{d} s_1 \twoheadrightarrow^{d} s_2 \twoheadrightarrow t_1 \twoheadleftarrow^{d} t'$ for some terms s_1, s_2, t_1 (A). Since $\to^{ll}$ is confluent (Lemma 4.17), $\sharp s_1 < \sharp s$ holds, and thus $ll(s_1) \twoheadrightarrow ll(t_1)$ by induction hypothesis (B). Also, since the length of $t' \twoheadrightarrow t$ is shorter than that of $s \twoheadrightarrow t$, $ll(t') \twoheadrightarrow ll(t)$ by induction hypothesis (C). Finally, by Theorem 4.18, we have $ll(s) \equiv ll(s_2)$ and $ll(t') \equiv ll(t_1)$. Thus, $ll(s) \twoheadrightarrow ll(t)$. The statements of the lemma immediately follows from this and the latter statement of Lemma 4.21 ∎



Theorem 4.23 *Let $\mathcal{R}$ be an STRS that contains only variables of the same sort. Then $\mathcal{R}$ is terminating if and only if $\Theta(\mathcal{R})$ is terminating.*

Proof. By Lemma 4.22, one can associate to every infinite reduction sequence $\alpha : t_0 \to t_1 \to \cdots$ in $\Theta(\mathcal{R})$ a reduction sequence $ll(\alpha) : ll(t_0) \twoheadrightarrow ll(t_1) \twoheadrightarrow \cdots$. W.l.o.g. we may take α of a minimal depth, that is any reduction sequence of smaller depth is terminating. Then since steps below d-clusters are in subterms of smaller depth, α must contain infinitely many dead-steps. Thus, $ll(\alpha)$ is infinite.

By the definition of d-step, we know that the top positions of proper disconnected subterms of a d-normal term are of type $\rrbracket$. Thus, we may assume that $ll(\alpha)$ is of rank ≤ 2 by the reason exactly same as that appearing immediately below Lemma 4.10.

If either $ll(\alpha)$ is of rank 1 or there is a principal subterm of $ll(t_0)$ that is not terminating, we have done; so suppose otherwise. Then it follows that infinitely many number of rewrite steps are performed at the top well-sorted components. Thus, to obtain an infinite reduction sequence of $\mathcal{R}$, it suffices to replace, for each $ll(t_i)$, its all principal subterms at positions of sort $j_1, j_2, \ldots$ with $x^{j_1}, x^{j_2}, \ldots$, where $x^1, \ldots, x^n$ are new variables of sort $1, \ldots, n$. ∎

5 Applications and Related Results

The technique of sort introduction based on our result, as well as that based on other known persistency results, is useful to detect properties of TRSs.

Example 5.1 *Let $\mathcal{R} = \{f(g(a), g(b), x) \to f(x, x, x),\ g(x) \to x\}$. To show termination of $\mathcal{R}$, we assume the following sort assignment: $\{f : 0 \times 0 \times 0 \to 1,\ g : 0 \to 0,\ a : 0,\ b : 0\}$. By Theorem 4.23, it suffices to show termination of $\mathcal{R}$ under this sort assignment. Terms of sort 0 are terminating and confluent, since only applicable rule is $g(x) \to x$. Terms of sort 1 have form $f(t_1, t_2, t_3)$ where t_1, t_2, t_3 are terms of sort 0. Suppose contrary that there exists an infinite reduction sequence of terms of sort 1. Then since terms of sort 0 are terminating, it must contain a reduction at the root position, which has the form $f(g(a), g(b), t) \to f(t, t, t)$ for some term t of sort 0. Since $g(a)$ and $g(b)$ have distinct normal forms and terms of sort 0 are confluent, we never have $g(a) \twoheadleftarrow t \twoheadrightarrow g(b)$. Thus we know that every reduction sequence starting from $f(t, t, t)$ never has a reduction at the root position. Since t is terminating, this implies $f(t, t, t)$ is terminating, which is a contradiction.*

It is known that termination is persistent for the class of STRSs that do not contain both collapsing and duplicating rules [13]. However, the argument above does not work with this result, because $\mathcal{R}$ is collapsing and duplicating.

Another kind of applications of our result is to prove that termination is preserved under suitable translations of TRSs.

Example 5.2 ([4][3]) *For each n-ary function symbol $f \in \mathcal{F}$, let $f_1, \ldots, f_{n-1}$ be new binary function symbols, and let $\hat{\mathcal{F}}$ be the collection of such new function symbols. Define a transformation $\hat{}$ from terms over $\mathcal{F}$ to those over $\hat{\mathcal{F}}$ by*

$$\hat{t} = \begin{cases} t & \text{if } t \in \mathcal{V}, \\ f_1(\hat{t}_1, f_2(\hat{t}_2, f_3(\cdots, f_{n-1}(\hat{t}_{n-1}, \hat{t}_n)\cdots))) & \text{if } t \equiv f(t_1, \ldots, t_n). \end{cases}$$

And, finally let $\hat{\mathcal{R}} = \{\hat{l} \to \hat{r} \mid l \to r \in \mathcal{R}\}$. Using Theorem 4.23, one can show that $\mathcal{R}$ is terminating if and only if $\hat{\mathcal{R}}$ is terminating.

($\Leftarrow$) Trivial. ($\Rightarrow$) We introduce a set of sorts by $\mathcal{S} = \{0\} \cup \bigcup \{\delta_f^1, \ldots, \delta_f^{n-2} \mid f \in \mathcal{F},\ f \text{ is } n\text{-ary}\}$, and sort assignment on $\hat{\mathcal{F}}$ as: $f_1 : 0 \times \delta_f^1 \to 0$, $f_i : 0 \times \delta_f^i \to \delta_f^{i-1}$ $(i = 2, \ldots, n-2)$, and $f_{n-1} : 0 \times 0 \to \delta_f^{n-2}$ for each $f \in \mathcal{F}$. By Theorem 4.23, $\hat{\mathcal{R}}$ is terminating if and only if $\hat{\mathcal{R}}$ is terminating on this sort assignment. Suppose $\hat{\mathcal{R}}$ is not terminating. Then there also exists an infinite reduction sequence over terms well-sorted under this sort assignment. W.l.o.g. we assume terms in this reduction sequence have sort 0. It is clear from the sort assignment that if once a function symbol f_i occurs in a well-sorted term of sort 0, it occurs in a form of $f_1(s_1, f_2(s_2, \ldots, f_{n-2}(s_{n-2}, f_{n-1}(s_{n-1}, s_n))\ldots))$ for some terms $s_1, \ldots, s_n$. Thus, all these terms have form $\hat{t}$ for some t and so one can reversely translate this infinite reduction sequence to that of terms on $\mathcal{F}$. Thus, we know $\mathcal{R}$ is not terminating.

Note that, in the example above, function symbols in $\hat{\mathcal{F}}$ has arity of at most 2, and therefore terms in $\mathcal{T}(\hat{\mathcal{F}}, \mathcal{V})$ have simple structures. Instead, not all terms in $\mathcal{T}(\hat{\mathcal{F}}, \mathcal{V})$ are images of terms in $\mathcal{T}(\mathcal{F}, \mathcal{V})$. We believe, however, that this kind of coding that preserves termination behavior would be helpful for studying properties of complicated systems by interpreting them in simpler systems.

[3]This application is due to Zantema.

A well-known way of simulating a TRS by another TRS containing simpler function symbols is "currying": each function symbol is coded by a constant and a new binary function symbol for "application" is added. Again new terms have simple structures and not all new terms are the images of original terms. It is shown in [5], [7] that this transformation also does not affect termination behavior of TRSs.

6 Concluding Remarks

We have proved that for any terminating many-sorted TRS $\mathcal{R}$, if $\mathcal{R}$ contains only variables of the same sort then its underlying TRS is also terminating. This is the positive solution to the problem of Zantema that has been appeared as Rewriting Open Problem 60 in [4]. We have also presented some applications of this result.

Acknowledgments

The author is grateful to Hans Zantema for an account on his motivation of the conjecture. Deep appreciation goes to an anonymous referee of this journal version for many refinements of the proof.

References

[1] T. Aoto and Y. Toyama. On composable properties of term rewriting systems. In *Proceedings of the 6th International Joint Conference, ALP'97 – HOA'97, Southampton, UK*, volume 1298 of *Lecture Notes in Computer Science*, pages 114–128. Springer-Verlag, 1997.

[2] T. Aoto and Y. Toyama. Persistency of confluence. *Journal of Universal Computer Science*, 3(11):1134–1147, 1997.

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

[4] N. Dershowitz, J.-P. Jouannaud, and J. W. Klop. More problems in rewriting. In *Proceedings of the 5th International Conference on Rewriting Techniques and Applications (RTA-93), Montreal, Canada*, volume

690 of *Lecture Notes in Computer Science*, pages 468–487. Springer-Verlag, 1993.

[5] J. R. Kennaway, J. W. Klop, M. R. Sleep, and F. J. de Vries. Comparing curried and uncurried rewriting. *Journal of Symbolic Computation*, 21:15–39, 1996.

[6] J. W. Klop. Term Rewriting Systems. In S. Abramsky, D. Gabbay, and T. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2, pages 1–116. Oxford University Press, 1992.

[7] A. Middeldorp, H. Ohsaki, and H. Zantema. Transforming termination by self-labelling. In *Proceedings of the 13th International Conference on Automated Deduction (CADE-13), New Brunswick, NJ, USA*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 373–387. Springer-Verlag, 1996.

[8] E. Ohlebusch. On the modularity of termination of term rewriting systems. *Theoretical Computer Science*, 136:333–360, 1994.

[9] H. Ohsaki. *Termination of Term Rewriting Systems: Transformation and Persistence*. PhD thesis, University of Tsukuba, 1998.

[10] H. Ohsaki and A. Middeldorp. Type introduction for equational rewriting. In *Proceedings of the 4th International Symposium on Logical Foundations of Computer Science*, volume 1234 of *Lecture Notes in Computer Science*, pages 283–293. Springer-Verlag, 1997.

[11] M. Schmidt-Schauß, M. Marchiori, and S. E. Panitz. Modular termination of r-consistent and left-linear term rewriting systems. *Theoretical Computer Science*, 149:361–374, 1995.

[12] Y. Toyama, J. W. Klop, and H. P. Barendregt. Termination for direct sums of left-linear complete term rewriting systems. *Journal of the Association for Computing Machinery*, 42(6):1275–1304, 1995.

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