ソフトウェア基礎論 31－9
（1989. 9. 20）

# MODULARITY OF SIMPLE TERMINATION OF TERM REWRITING SYSTEMS

Masahito Kurihara　　Azuma Ohuchi

Hokkaido University

Kita-13 Nishi-8, Kita-ku, Sapporo, 060 Japan

A term rewriting system, which is a computer program written as a set of rewrite rules, is said to be simply-terminating if, intuitively, its termination is proved with the simplification ordering method of Dershowitz. The direct sum $R_0 \oplus R_1$ of term rewriting systems $R_0$ and $R_1$ is their disjoint union. A property of term rewriting systems is said to be modular if its validity for $R_0 \oplus R_1$ is implied by the validity for $R_0$ and $R_1$.

In this paper, we prove the modularity of simple termination, i.e., that $R_0 \oplus R_1$ is simply-terminating if and only if each of $R_0$ and $R_1$ is so. The result is novel in that it depends only upon how we proved both $R_0$ and $R_1$ terminating, rather than syntactic properties of the terminating systems. It is practically useful for the semi-mechanical termination proof of 'modular' computer programs written as the set of term rewriting systems.

# 項書き換えシステムの単純停止性のモジュラー性

栗原　正仁　　大内　東

北海道大学

項書き換えシステムの停止性が単純化順序を用いて証明できるとき、そのシステムは単純停止性をもつという。２つのＴＲＳ $R_0$と$R_1$の直和$R_0 \oplus R_1$は、$R_0$と$R_1$が関数記号を共有しないときのみ定義され、$R_0 \oplus R_1 = R_0 \cup R_1$ である。$R_0$と$R_1$がある性質をもつならば$R_0 \oplus R_1$もまたその性質をもつとき、その性質はモジュラーであるという。

本論文では、直和システムにおける単純停止性のモジュラー性、すなわち、関数記号を共有しない２つの項書き換えシステムがそれぞれ単純停止性をもつときに限り、その直和システムも単純停止性をもつことを示す。この結果は停止するシステムの構文論的な性質に依存するのではなく、$R_0$と$R_1$の停止性をいかに証明したかにのみ依存する点で新しい。これは項書き換えシステムの集合として書かれたモジュラーなプログラムの半機械的な停止性証明に有用である。

## 1 Introduction

A term rewriting system [3] $R$ is a program expressed as a finite set of rewrite rules, each of the form $\ell \longrightarrow r$, where $\ell$ and $r$ are terms constructed from function symbols and variables. The program is executed by rewriting; that is, by repeatedly applying rewrite rules to some given initial term, using single-directional pattern matching and subterm replacement. A rewrite rule $\ell \longrightarrow r$ may be applied to a term if the term contains a subterm which is an instance of the left-hand side $\ell$; then the subterm is replaced by the corresponding instance of the right-hand side $r$.

A term rewriting system is *terminating* if there is no infinite rewriting sequence. Termination is a crucial property not only for ensuring that a program eventually produce the expected result, but also for verifying the confluence because the termination is required for inferring the 'total confluence' from the 'local confluence'[3]. However, the problem is that the termination is undecidable. Thus it is unlikely that computers prove the termination automatically. The proof would require, more or less, the assistance of humans. But we should say that the intelligent power of us humans is too restrictive to assist the termination proof of large and complex systems.

A natural approach to the solution is the 'divide-and-conquer'; divide the problem into smaller ones whose solutions can be easily obtained and combined together to make the solution of the original problem. In the community of term rewriting systems, the related concept of 'modularity' [6] is defined and extensively investigated; a property of term rewriting systems is *modular* if its validity for the systems hierarchically composed of some smaller ones is implied by the validity of that property of the constituent systems.

Since this field is still young, many researchers assume as a start the most simple way of the composition — the direct sum. The *direct sum* $R_0 \oplus R_1$ of two term rewriting systems $R_0$ and $R_1$ is defined only when the set of function symbols contained in $R_0$ and $R_1$ is disjoint; then $R_0 \oplus R_1$ is just a union of the both set of rules.

On the modularity of confluence, Toyama [10] succeeded in proving the following useful result:

> $R_0 \oplus R_1$ is confluent iff both $R_0$ and $R_1$ are so.

On the termination, however, the analogous conjecture

> $R_0 \oplus R_1$ is terminating iff both $R_0$ and $R_1$ are so

was refuted by the counterexample by Toyama [11].

Then Toyama conjectured that

> $R_0 \oplus R_1$ is complete iff both $R_0$ and $R_1$ are so

where a system is *complete* if it is both terminating and confluent. However, this conjecture was also refuted by the counterexample by Klop and Barendregt [11]. Investigating this counterexample, Hsiang presented a revised conjecture but it was also refuted by Toyama [11].

It was in 1987 that the first positive results were discovered by Rusinowitch [7]:

> $R_0 \oplus R_1$ is terminating and non-collapsing iff both $R_0$ and $R_1$ are so.
>
> $R_0 \oplus R_1$ is terminating and non-duplicant iff both $R_0$ and $R_1$ are so.

where a system is *collapsing* if it contains a rule whose right-hand side is a variable, and *duplicant* if it contains a rule whose right-hand side has strictly more occurrences of one variable than its left-hand side. However, these results are too restrictive to be applied in practice, because most of the practical systems are collapsing and/or duplicant. (The results were further investigated by Middeldorp [5] in 1989, but his result is not modular, although it is a sufficient condition for the termination of a direct sum.)

In 1989, Toyama, Klop, and Barendregt [12] proved a reasonably practical result:

$R_0 \oplus R_1$ is complete and left-linear iff both $R_0$ and $R_1$ are so

where a system is *left-linear* if no variable occurs more than once on the left-hand side of a rule. This result is reasonably practical because, when we write a functional program as a term rewriting system, almost all systems can be written within the restriction of the completeness and the left-linearity.

However, term rewriting systems have important applications in automated theorem proving for first-order logic with equality, in which it is very common for a system to be non-left-linear.

To solve this problem, we take a completely different approach. Note that those three results explicitly depend upon the syntactic properties of the systems, such as non-collapsing, non-duplicant, and left-linear. In this paper, however, we present a new result discovered from another point of view:

$R_0 \oplus R_1$ is simply-terminating iff both $R_0$ and $R_1$ are so

where a system is *simply-terminating* if, intuitively, it's termination is proved with the simplification ordering method of Dershowitz [1]. The result is novel in that it depends only upon *how we proved* both $R_0$ and $R_1$ terminating, rather than the explicit syntactic properties of the terminating systems. The result is practical because many practical systems which appear in functional programming, theorem proving, and any other areas fall into the classes whose termination is semi-mechanically provable with the simplification ordering method.

## 2 Formal Preliminaries

### 2.1 Term Rewriting Systems

Let $\mathcal{V}$ be a set of *variables*, denoted by $x$, $y$, $z, \ldots$, and $\mathcal{F}$ be a set of *function symbols*, denoted by $f$, $g$, $h, \ldots$. Each function symbol may have variable arity, or may be restricted to a fixed arity. A function symbol with arity zero is called a *constant*.

A *term* is either a variable or a constant or is of the form $f(t_1, \ldots, t_n)$ for some terms $t_1, \ldots, t_n$ and a function symbol $f$ with variable arity or fixed arity $n > 0$. The *root* of a term $t$, notation $root(t)$, is $f$ if $t$ is of the form $f(t_1, \ldots, t_n)$; otherwise, it is $t$ itself. Every intermediate term we use in building up a term $t$ (including $t$ itself) is called a *subterm* of $t$. A *proper* subterm of $t$ is one that is distinct from $t$ itself. We denote terms by $s, t, u, \ldots$, the set of terms on $\mathcal{F}$ and $\mathcal{V}$ by $\mathcal{T}(\mathcal{F},\mathcal{V})$, and the set of terms on $\mathcal{F}$ by $\mathcal{T}(\mathcal{F})$. We use $\mathcal{T}$ for $\mathcal{T}(\mathcal{F},\mathcal{V})$ when the context makes it clear.

A substitution $\theta$ is a mapping from $\mathcal{V}$ to $\mathcal{T}$. As usual, it is naturally extended to a mapping from $\mathcal{T}$ to $\mathcal{T}$ — $\theta(c) = c$ if $c$ is a constant, and $\theta(t) = f(\theta(t_1), \ldots, \theta(t_n))$ if $t = f(t_1, \ldots, t_n)$, $n > 0$. We write $t\theta$ instead of $\theta(t)$.

Let $\Box$ be an extra constant called a *hole*. A term $C$ on $\mathcal{F} \cup \{\Box\}$ and $\mathcal{V}$ is called a *context* on $\mathcal{F}$. A context $C$ is *trivial* if it is the $\Box$. When $C$ is a non-trivial context with $n$ holes, $C[t_1, \ldots, t_n]$ denotes the result of replacing the holes by the terms $t_1, \ldots, t_n$ from left to right.

A *rewrite rule* on $\mathcal{T}$ is a pair $\ell \rightarrow r$ of terms in $\mathcal{T}$ such that $\ell$ is not a variable and any variable occurring in $r$ also occurs in $\ell$. A *term rewriting system* $R$ on $\mathcal{T}$ is a set of rewrite rules on $\mathcal{T}$. A term $s$ is *rewritten* to a term $t$ by $R$, notation $s \rightarrow_R t$, if there exists a rewrite rule $\ell \rightarrow r$ and a substitution $\theta$ such that $s \equiv C[\ell\theta]$ and $t \equiv C[r\theta]$ for some context $C$ (where the symbol $\equiv$ denotes the 'syntactic equality'). The term $\ell\theta$ is called a *redex*. The pair $s \rightarrow_R t$ of the terms $s$ and $t$ is called a *rewriting* or a *reduction*. The transitive closure of a relation, say, $\rightarrow_R$ is denoted by $\rightarrow_R^+$. The reflexive transitive closure of $\rightarrow_R$ is denoted by $\rightarrow_R^*$. In the rest of this paper, we restrict the relation $\rightarrow_R$ to be defined only on the ground terms $\mathcal{T}(\mathcal{F})$. This is just for clarifying the discussions, and our major results are easily shown to hold for the more general case.

## 2.2 Direct Sum Systems

Let $\mathcal{F}_0$ and $\mathcal{F}_1$ be disjoint sets of function symbols, and let $\mathcal{F} = \mathcal{F}_0 \cup \mathcal{F}_1$; then the term rewriting systems $R_0$ on $\mathcal{T}(\mathcal{F}_0, \mathcal{V})$ and $R_1$ on $\mathcal{T}(\mathcal{F}_1, \mathcal{V})$ are said to be *disjoint.* The union of the disjoint systems $R_0$ and $R_1$, which is a term rewriting system on $\mathcal{T}(\mathcal{F},\mathcal{V})$, is called the *direct sum* system, notation $R_0 \oplus R_1$. In the rest of this paper, we assume that $R = R_0 \oplus R_1$. For mnemotechnical reasons we will paint the function symbols — the function symbols of $R_0$ in *black*, and those in $R_1$ in *white.* We say that a non-variable term or a non-trivial context is *black-rooted* (or *white-rooted*), if its root symbol is black (or white). A non-variable term or a non-trivial context is (entirely) black (or white) if every function symbol in it is black (or white); otherwise, it is *mixed*. We also say that the rules in $R_0$ is black and those in $R_1$ is white. To distinguish in print between them, the black symbols are printed in upper case and the white ones in lower case.

**Definition 2.1** An *alien*[1] in a term $t$ is a non-variable proper subterm $u$ of $t$ which is maximal with respect to the 'subterm' relation, such that $root(t)$ and $root(u)$ have distinct colors.

We write $t \equiv C[\![t_1, \ldots, t_n]\!]$ if $t_1, \ldots, t_n$ are all the aliens in $t$ (from left to right) and $C$ is the context obtained by replacing each alien by a hole. For example, the term $t \equiv F(G(b,x), h(A,c))$ has the two aliens $b$ and $h(A,c)$; thus $t \equiv C[\![b, h(A,c)]\!]$, where $C \equiv F(G(\Box, x), \Box)$. Note that all the function symbols in $C$ have the same color, and actually, $C$ is the maximal of such contexts.

Since each alien $t_i$ in $t$ may have aliens in itself, we can identify a hierarchy of aliens defined below:

[1]also called a *principal subterm*

**Definition 2.2** The *alien tree* $AT(t)$ of a term $t$ is the tree each node of which is either a black or a white context such that:

1. if $t$ has no alien, then $AT(t)$ consists of a single node $t$, the root of the tree;
2. if $t \equiv C[\![t_1, \ldots, t_n]\!]$ $(n > 0)$, then $AT(t)$ consists of the root $C$ and the subtrees $AT(t_i)$, $1 \leq i \leq n$.

The *rank* of a term $t$, notation $rank(t)$, is the height of the alien tree of $t$.

As an example, the alien tree of $t \equiv F(G(b,x), h(A,c))$ with rank 2 is depicted below:

$$\begin{tikzcd} F(G(\Box, x), \Box) \arrow[d, dash] \arrow[dr, dash] & \\ b & h(\Box, c) \arrow[d, dash] \\ & A \end{tikzcd}$$

It is known that the rank of a term never increases after rewriting [10]:

**Lemma 2.3** *If* $s \rightarrow_R t$ *then*
$$rank(s) \geq rank(t).$$

**Proof.** Routine, by induction on $rank(s)$. □

**Lemma 2.4** *If* $s \rightarrow_R t$ *and* $rank(s) = rank(t)$, *then* $root(s)$ *and* $root(t)$ *are in the same color.*

**Proof.** Routine. □

When discussing reduction by a direct sum system, it is often useful to distinguish *inner* reductions and *outer* ones[10]:

**Definition 2.5** The reduction $s \rightarrow_R t$ is an *inner reduction* if the redex occurs in an alien in $s$; otherwise it is an *outer reduction.*

**Lemma 2.6** *Let* $s \equiv C[\![s_1, \ldots, s_n]\!] \rightarrow_R t$ *and* $rank(s) = rank(t)$. *Then*

*1. If* $s \rightarrow_R t$ *is an inner reduction, then*

$$t \equiv C[\![s_1, \ldots, s_{i-1}, t_i, s_{i+1}, \ldots, s_n]\!]$$

*for some $i$, where $s_i \rightarrow_R t_i$.*

2. *If $s \rightarrow_R t$ is an outer reduction, then*

$$t \equiv C'[\![s_{i_1}, \ldots, s_{i_m}]\!]$$

*for some context $C'$ in the same color as $C$, and $1 \leq i_1, \ldots, i_m \leq n$.*

**Proof.** Routine. □

Comment: The second part of the lemma says that, when $rank(s) = rank(t)$, an outer reduction never creates new aliens, although it might duplicate some existing aliens. To intuitively understand why, assume that the context $C$ is black. Then the rule which reduces $s$ is black. Since aliens are white-rooted, each alien in a redex must be entirely 'covered' by some variable of the black left-hand side of the rule, and 'embedded' in the black right-hand side of the rule. Thus, the alien is again an alien in the resultant term $t$, never 'collapsed' into a small alien nor 'surrounded' by any white context to become larger alien.

When dealing with non-left-linear rules, the following notion, introduced first by Toyama [10], is useful:

**Definition 2.7** A term $s$ is *proportional* to a term $t$, notation $s \propto t$, if $s \equiv C[\![s_1, \ldots, s_n]\!]$, $t \equiv C[\![t_1, \ldots, t_n]\!]$, and that $s_i \equiv s_j$ implies $t_i \equiv t_j$ $(1 \leq i, j \leq n)$.

**Lemma 2.8** *Let $s \rightarrow_R t$ be an outer reduction such that $rank(s) = rank(t)$, and let $s'$ be a term such that $s \propto s'$. Then there exists a term $t'$ such that $s' \rightarrow_R t'$ is an outer reduction, and $t \propto t'$.*

**Proof.** Apply to $s'$ the same rewrite rule that reduced $s$. □

## 3 Modularity of Simple Termination

### 3.1 Simple Termination

**Definition 3.1** A term rewriting system $R$ is *terminating* if there is no infinite rewrite sequence $t_0 \rightarrow_R t_1 \rightarrow_R \cdots$.

Consider a *partial ordering* (a transitive and irreflexive relation) $\succ$ on the set of terms $\mathcal{T}$. When the ordering is *monotonic*, a 'local' reduction always ensures the 'global' reduction in the ordering:

**Definition 3.2** A partial ordering $\succ$ on $\mathcal{T}$ is *monotonic* if it possesses the *replacement* property,

$$s \succ t \text{ implies } f(\ldots s \ldots) \succ f(\ldots t \ldots)$$

for all terms in $\mathcal{T}$.

Many orderings used in practice is in the class of *simplification orderings* which ensure that 'syntactically simpler' terms are always smaller in the orderings:

**Definition 3.3** A monotonic partial ordering $\succ$ is a *simplification ordering* for $\mathcal{T}$ if it possesses the *subterm* property,

$$f(\ldots t \ldots) \succ t,$$

and the *deletion* property,

$$f(\ldots t \ldots) \succ f(\ldots \quad \ldots),$$

for all terms in $\mathcal{T}$.

We relate simplification orderings to rewritings, by the following notion:

**Definition 3.4** A simplification ordering $\succ$ on $\mathcal{T}$ *supports* a rewriting $s \rightarrow_R t$, if $s \succ t$. It supports a system $R$ if it supports every rewriting $s \rightarrow_R t$ defined by $R$ on $\mathcal{T}$.

Now, we are ready to define the simple termination of term rewriting systems:

**Definition 3.5** A term rewriting system $R$ on $\mathcal{T}$ is *simply-terminating* if there exists a simplification ordering $\succ$ which supports $R$.

It is known that the simplification ordering $\succ$ is well-founded [1], so there is no infinite reduction sequence $t_0 \succ t_1 \succ \cdots$. Therefore:

**Theorem 3.6 (Dershowitz)**
*A simply-terminating system is, in fact, terminating.*

Note that such orderings may not exist even if the system is terminating.

As mentioned in the introduction, the purpose of this paper is to prove that $R_0 \oplus R_1$ is simply-terminating if and only if both $R_0$ and $R_1$ are so. The following definitions and lemmas will become useful for that purpose:

**Definition 3.7** The *minimal support* for $R_a$ ($a = 0, 1$) is the relation $\succ_a$ on $\mathcal{T}(\mathcal{F}_a, \mathcal{V})$ defined as:

$$\succ_a = (\rightarrow_{R_a} \cup \rightarrow_{sub} \cup \rightarrow_{del})^+$$

where

$$\begin{aligned} s \rightarrow_{sub} t \quad \text{iff} \quad & s \equiv C[f(\ldots u \ldots)] \text{ and} \\ & t \equiv C[u], \\ s \rightarrow_{del} t \quad \text{iff} \quad & s \equiv C[f(\ldots u \ldots)] \text{ and} \\ & t \equiv C[f(\ldots \quad \ldots)], \end{aligned}$$

for some $C$, $f$, and $u$; In both definitions, the occurrence $f(\ldots u \ldots)$ in $s$ is called a *redex,* and we use the terminology 'inner' reduction and 'outer' reduction in the same manner as in the definition 2.5.

We introduce two abbreviations:

$$\rightarrow_{asd} = \rightarrow_{R_a} \cup \rightarrow_{sub} \cup \rightarrow_{del}, \ (a = 0, 1)$$

$$\rightarrow_{01sd} = \rightarrow_{R_0} \cup \rightarrow_{R_1} \cup \rightarrow_{sub} \cup \rightarrow_{del}$$

Notice that $\succ_a$ is the transitive closure of $\rightarrow_{asd}$.

**Lemma 3.8** *If $R_a$ is simply-terminating, then the minimal support $\succ_a$ is a simplification ordering which supports $R_a$.*

**Proof.** The lemma is proved by showing that the relation $\succ_a$ has the following six properties: irreflexivity, transitivity, monotonicity, subterm, deletion, and support for $R_a$. The transitivity, the subterm property, the deletion property, and the support for $R_a$ are obvious, since $\succ_a$ is a transitive closure which includes $\rightarrow_{sub}$, $\rightarrow_{del}$, and $\rightarrow_{R_a}$.

To show the irreflexivity, let $\succ_a$ be a simplification ordering which supports every reduction $s \rightarrow_{R_a} t$ defined by $R_a$. Then $s \rightarrow_{asd} t$ implies $s \succ_a t$, because (1) if $s \rightarrow_{R_a} t$ then obviously $s \succ_a t$, (2) if $s \rightarrow_{sub} t$ then from the subterm property and the monotonicity of $\succ_a$ we see $s \succ_a t$, and (3) if $s \rightarrow_{del} t$ then $s \succ_a t$ from the deletion property and the monotonicity of $\succ_a$. Therefore, if there were a cyclic derivation $t \rightarrow_{asd} \cdots \rightarrow_{asd} t$, then we would have $t \succ_a \cdots \succ_a t$, thus $t \succ_a t$, which contradicts the irreflexivity of $\succ_a$.

To show the monotonicity, first note that $\rightarrow_{asd}$ is monotonic, because each of the $\rightarrow_{R_a}$, $\rightarrow_{sub}$, and $\rightarrow_{del}$ is monotonic. Hence, $s \rightarrow_{asd} \cdots \rightarrow_{asd} t$ implies $f(\ldots s \ldots) \rightarrow_{asd} \cdots \rightarrow_{asd} f(\ldots t \ldots)$. Therefore, $s \succ_a t$ implies $f(\ldots s \ldots) \succ_a f(\ldots t \ldots)$. $\Box$

## 3.2 Alien Replacement

**Definition 3.9** An *alien replacement* $\rho$ is a mapping from the range N of the *rank* function to the set of terms $\mathcal{T}(\mathcal{F}, \mathcal{V})$. It is extended to a mapping from $\mathcal{T}(\mathcal{F}, \mathcal{V}) \cup \mathrm{N}$ to $\mathcal{T}(\mathcal{F}, \mathcal{V})$ by

$$\begin{aligned} & \rho(C[\![t_1, \ldots, t_n]\!]) \\ & \equiv C[\rho(rank(t_1)), \ldots, \rho(rank(t_n))], \ n \geq 0. \end{aligned}$$

**Example** Let $\rho(0) \equiv E$, and $\rho(1) \equiv F(E)$. Then

$$\rho(F(A, \underline{g(A)}, \underline{b})) \equiv F(A, F(E), E).$$

Note that the aliens, which we have underlined, were replaced by the terms determined by their ranks.

**Definition 3.10** Let $E$ be the distinguished black function symbol, $E \in \mathcal{F}_0$, with variable arity. We assume that $E$ is not used in the rules in $R_0$ (nor, of course, in $R_1$). The *alien replacement $\rho$ determined by a 'black-rooted' finite sequence*

$$s_0 \rightarrow_{01sd} s_1 \rightarrow_{01sd} \cdots \rightarrow_{01sd} s_m$$

where $root(s_i) \in \mathcal{F}_0$ $(0 \leq i \leq m)$, is defined inductively by

$$\begin{aligned}\rho(0) &= E\\ \rho(r) &= E(\rho(0),\rho(1),\cdots,\rho(r-1),\\ &\qquad \rho(t'_1),\rho(t'_2),\cdots,\rho(t'_{n_r}))\\ &\qquad\qquad ,r \geq 1\end{aligned}$$

where $\{t'_1, t'_2, \cdots, t'_{n_r}\}$, which is determined for each $r$, is the set of terms $t'$ such that:

$(\exists k)\quad s_k \rightarrow_{01sd} s_{k+1}$ is an inner reduction, and
$\qquad(\exists t)\quad s_k \equiv C[\![\ldots t \ldots]\!]$, $rank(t) = r$
$\qquad\qquad s_{k+1} \equiv C[\ldots t' \ldots]$, $root(t') \in \mathcal{F}_0$
$\qquad\qquad t \rightarrow_{01sd} t'$.

In other words, $t'$ is a black-rooted term derived from a white-rooted alien $t$ with rank $r$.

Note:

- $rank(t') < rank(t) = r$, so $\rho(t')$ can be computed if $\rho(1), \ldots, \rho(r-1)$ are already determined.
- $\rho(r) \rightarrow_{sub} \rho(r')$, if $r > r'$.
- $\rho(r) \rightarrow_{sub} \rho(t')$, if $t'$ satisfies the above conditions.
- Since $E$ is black and $t'$ is black-rooted, we can verify, by induction on ranks, that $\rho(r)$ is entirely black ($r \geq 0$).
- Since $s_i$ is black-rooted, $\rho(s_i)$ is entirely black ($0 \leq i \leq m$).

**Example** Consider the systems

$$R_0 = \{F(x) \rightarrow G(x,x,x)\},$$
$$R_1 = \{h(x,h(x,y)) \rightarrow y\}.$$

and the sequence

$$\begin{aligned} s_0 &\equiv \underline{F(t)}\\ \rightarrow_{R_0}\ s_1 &\equiv G(\underline{t},t,t)\\ \rightarrow_{sub}\ s_2 &\equiv G(A,\underline{t},t)\\ \rightarrow_{R_1}\ s_3 &\equiv G(A,F(b),t)\end{aligned}$$

where $t \equiv h(A,h(A,F(b)))$ and the underlined terms are redexes. Then the alien replacement $\rho$ determined by this sequence is:

$$\begin{aligned}\rho(0) &\equiv E\\ \rho(1) &\equiv E(E)\\ \rho(2) &\equiv E(E,E(E),A,F(E)).\end{aligned}$$

Note that

$$\begin{aligned} &\rho(s_0) \equiv F(u)\\ &\rightarrow_{R_0} \rho(s_1) \equiv G(u,u,u)\\ &\rightarrow_{sub} \rho(s_2) \equiv G(A,u,u)\\ &\rightarrow_{sub} \rho(s_3) \equiv G(A,F(E),u)\end{aligned}$$

where $u \equiv \rho(2) \equiv E(E,E(E),A,F(E))$.

This definition might seem too complicated to understand. Actually, the definition was introduced just for technical reasons. The idea comes from the intention that, when there is a cyclic sequence

$$s \rightarrow_{01sd} \cdots \rightarrow_{01sd} s$$

of black-rooted *mixed-color* terms, we want to construct a cyclic sequence

$$\rho(s) \rightarrow_{0sd} \cdots \rightarrow_{0sd} \rho(s)$$

of *entirely black* terms, thus uncovering the contradiction to the irreflexivity of the minimal support $\succ_0$ for the black system $R_0$. The proof of the following lemma would show you more exactly how this definition is used:

**Lemma 3.11** *If $\rho$ is the alien replacement determined by the sequence*

$$s \equiv s_0 \rightarrow_{01sd} \cdots \rightarrow_{01sd} s_m \equiv s'$$

*of black-rooted terms with the same rank, then*

$$\rho(s) \succeq_0 \rho(s').$$

*where $\succeq_0$ is the union of the minimal support $\succ_0$ and the syntactical equality $\equiv$; that is, $\succeq_0$ is the reflexive transitive closure of $\rightarrow_{0sd} = \rightarrow_{R_0} \cup \rightarrow_{sub} \cup \rightarrow_{del}$.*

*In particular, if the sequence contains at least a single outer reduction, then*

$$\rho(s) \succ_0 \rho(s').$$

**Proof.** To prove the lemma, we show that $\rho(s_k) \succeq_0 \rho(s_{k+1})$ for $0 \leq k < m$. As stated in the section 2.1, we assume that the terms $s_k$ ($0 \leq k \leq m$) contain no variable.

CASE 1. $s_k \rightarrow_R s_{k+1}$ (outer reduction).
In this case, $s_k \rightarrow_{R_0} s_{k+1}$ since the outermost context of $s_k$ is black. Recall that $s_k$ and $s_{k+1}$ have the same rank. Since $s_k \propto \rho(s_k)$, $\rho(s_k)$ is reducible with the same rule that has reduced

$s_k$, and obviously $\rho(s_k) \rightarrow_{R_0} \rho(s_{k+1})$ (Lemma 2.8). Therefore, $\rho(s_k) \succ_0 \rho(s_{k+1})$.

CASE 2. $s_k \rightarrow_R s_{k+1}$ (inner reduction).
Assume, without loss of generality, that the first alien was reduced:

$$s_k \equiv C[\![t_1, t_2, \ldots, t_n]\!],$$
$$\rho(s_k) \equiv C[\rho(rank(t_1)), \rho(rank(t_2)), \ldots, \rho(rank(t_n))],$$
$$s_{k+1} \equiv C[t'_1, t_2, \ldots, t_n],$$
$$t_1 \rightarrow_R t'_1.$$

SUBCASE 2(A): $rank(t_1) = rank(t'_1)$. In this case, we see that $t'_1$ is white-rooted, so $t'_1$ is an alien in $s_{k+1}$, thus:

$$s_{k+1} \equiv C[\![t'_1, t_2, \ldots, t_n]\!],$$
$$\rho(s_{k+1}) \equiv C[\rho(rank(t'_1)), \rho(rank(t_2)), \ldots, \rho(rank(t_n))].$$

Therefore, $\rho(s_k) \equiv \rho(s_{k+1})$.

SUBCASE 2(B): $rank(t_1) > rank(t'_1)$ and $t'_1$ is white-rooted. Then, $t'_1$ is an alien in $s_{k+1}$:

$$s_{k+1} \equiv C[\![t'_1, t_2, \ldots, t_n]\!],$$
$$\rho(s_{k+1}) \equiv C[\rho(rank(t'_1)), \rho(rank(t_2)), \ldots, \rho(rank(t_n))].$$

Since $rank(t_1) > rank(t'_1)$, we see that
$$\rho(rank(t_1)) \rightarrow_{sub} \rho(rank(t'_1)).$$
Therefore, $\rho(s_k) \rightarrow_{sub} \rho(s_{k+1})$, thus $\rho(s_k) \succ_0 \rho(s_{k+1})$.

SUBCASE 2(C): $rank(t_1) > rank(t'_1)$ and $t'_1$ is black-rooted. In this case, we see that

$$\rho(s_{k+1}) \equiv C[\rho(t'_1), \rho(rank(t_2)), \ldots, \rho(rank(t_n))]$$

because $t'_1$ is not an alien in $s_{k+1}$ but the aliens in $t'_1$ are aliens in $s_{k+1}$. By the definition, $\rho$ is constructed such that $\rho(t'_1)$ is contained as an argument in $\rho(rank(t_1))$, so $\rho(rank(t_1)) \rightarrow_{sub} \rho(t'_1)$. Therefore, $\rho(s_k) \rightarrow_{sub} \rho(s_{k+1})$, thus $\rho(s_k) \succ_0 \rho(s_{k+1})$.

CASE 3. $s_k \rightarrow_{sub} s_{k+1}$ (outer reduction).
Recall that $s_k$ and $s_{k+1}$ have the same rank. Obviously, $\rho(s_k) \rightarrow_{sub} \rho(s_{k+1})$, so $\rho(s_k) \succ_0 \rho(s_{k+1})$.

CASE 4. $s_k \rightarrow_{sub} s_{k+1}$ (inner reduction).
Assume, without loss of generality, that the first alien was reduced:

$$s_k \equiv C[\![t_1, t_2, \ldots, t_n]\!],$$
$$s_{k+1} \equiv C[t'_1, t_2, \ldots, t_n],$$
$$t_1 \rightarrow_{sub} t'_1.$$

By the arguments similar to Case 2, we can verify that

- if $rank(t_1) = rank(t'_1)$, then
  $\rho(s_k) \equiv \rho(s_{k+1})$.
- if $rank(t_1) > rank(t'_1)$ and $t'_1$ is white-rooted, then
  $\rho(rank(t_1)) \rightarrow_{sub} \rho(rank(t'_1))$ and
  $\rho(s_k) \rightarrow_{sub} \rho(s_{k+1})$.
- if $rank(t_1) > rank(t'_1)$ and $t'_1$ is black-rooted, then
  $\rho(rank(t_1)) \rightarrow_{sub} \rho(t'_1)$ and
  $\rho(s_k) \rightarrow_{sub} \rho(s_{k+1})$.

CASE 5. $s_k \rightarrow_{del} s_{k+1}$ (outer reduction).
Obviously, $\rho(s_k) \rightarrow_{del} \rho(s_{k+1})$, so $\rho(s_k) \succ_0 \rho(s_{k+1})$.

CASE 6. $s_k \rightarrow_{del} s_{k+1}$ (inner reduction).
Similar to Case 2, except that we need not consider the subcase (c). □

## 3.3 Modularity of Simple Termination

Now, we can extend the Lemma 3.8:

**Lemma 3.12** *Let* $\succ = \rightarrow_{01sd}{}^+ = (\rightarrow_{R_0} \cup \rightarrow_{R_1} \cup \rightarrow_{sub} \cup \rightarrow_{del})^+$. *If* $R_0$ *and* $R_1$ *are simply-terminating, then* $\succ$ *is a simplification ordering which supports* $R_0 \oplus R_1$.

**Proof.** We have to show that the relation $\succ$ has the following six properties: irreflexivity, transitivity, monotonicity, subterm, deletion, and support for $R_0 \oplus R_1$. The most difficult part is the irreflexivity. The other five properties are easily verified by the arguments similar to Lemma 3.8.

To show the irreflexivity, we prove by induction on $rank(s)$, that there is no cyclic sequence $s \equiv s_0 \rightarrow_{01sd} s_1 \rightarrow_{01sd} \cdots \rightarrow_{01sd}$

$s_n \equiv s$, where we assume without loss of generality that $s$ is black-rooted. Note that all the terms have the same rank and, as a result, are black-rooted.

Supposing that there is such a cyclic sequence, we will derive a contradiction.

Base Case: $rank(s) = 0$. Since $rank(s_i) = 0$, all the terms are entirely black. Hence the cyclic sequence may be expressed as $s \rightarrow_{0sd} \cdots \rightarrow_{0sd} s$. This means that $s \succ_0 s$, which contradicts the irreflexivity of $\succ_0$(Lemma 3.8).

Induction Step: $rank(s) \geq 1$. From the induction hypothesis, it is impossible that all the reductions in the sequence be inner reductions. Thus at least a single reduction is an outer reduction. Let $\rho$ be the alien replacement determined by the cyclic sequence. Since we assumed that $E$ is black, $\rho(s_i)$ is entirely black. Then, from Lemma 3.11, we have that $\rho(s) \succ_0 \rho(s)$. This again contradicts the irreflexivity of $\succ_0$. □

**Theorem 3.13** *$R_0 \oplus R_1$ is simply-terminating iff both $R_0$ and $R_1$ are so.*

**Proof.** The *only-if* part is trivial. The *if* part is direct from Lemma 3.12. □

**Example** Consider the following system:

$$R_0 = \left\{ \begin{array}{l} x \cdot x \rightarrow x, \\ x \cdot (y + z) \rightarrow (x \cdot y) + (x \cdot z) \end{array} \right\}$$

$$R_1 = \{(x^{-1})^{-1} \rightarrow x\}$$

$R_0$ is shown to be simply-terminating by the recursive path ordering (which is a simplification ordering introduced by Dershowitz [1]). $R_1$ is also simply-terminating. Therefore, by our theorem, $R_0 \oplus R_1$ is simply-terminating. Note that the termination of $R_0 \oplus R_1$ cannot be proved by the three results by Rusinowitch, Toyama, et al. because $R_0$ contains collapsing, duplicant, and non-left-linear rules.

Many orderings suitable for semi-mechanical termination proof fall into the class of the simplification ordering. For example, the recursive path ordering (RPO) of Dershowitz, the path of subterm ordering (PSO) of Plaisted, and the recursive decomposition ordering (RDO) of Jouannaud are typical simplification orderings widely used [2,8]. It is known that there is a term rewriting system, say $R_0$, whose termination is proved with PSO but not with RDO nor RPO, and also there is a system, say $R_1$, whose termination is proved with RDO but not with PSO nor RPO [8]. Thus the termination of $R_0 \oplus R_1$ is proved with neither PSO nor RDO nor RPO. However, by our theorem, $R_0 \oplus R_1$ is simply-terminating, because both $R_0$ and $R_1$ are so.

## 4 Conclusion

We have presented a novel result on the modularity of the termination of term rewriting systems. The authors claim that not only the result itself is novel but also the class of the result is novel in that it focuses on the termination proof method, rather than explicitly restricted syntactic properties. Also, the result is independent of the confluence.

Proof with simplification ordering is one of the most powerful methods that are suitable for semi-mechanical termination proof. Therefore, our result is practically useful for the semi-mechanical termination proofs of 'modular' computer programs written as the set of term rewriting systems. You can load and merge several disjoint, simply-terminating systems together, without losing termination. Our result is also useful for other applications which require semi-mechanical termination procedures. Inductionless induction theorem proving (on 'direct sum' theories) based on Knuth-Bendix completion procedure [9] is an example.

We feel that the result by Toyama, et al. and our result in this paper are practically the final solutions to the problem on the termination of the direct sum of term rewriting systems, and will close the series of the discussions so far.

The results will promote the research and the development of modular languages [13] based on term rewriting systems.

Our interest in future works will be in *non-direct sums* — under what conditions and strategies may the function symbols be shared among systems without losing termination and confluence? The problem is worth challenging, and some restricted results are already obtained [2]. Also, a novel approach to the modularity is proposed [4]. We believe that the solution would greatly contribute to the development of large-scale programs written as term rewriting systems, and thus enhance the mechanization of software engineering in the future.

## Acknowledgement

The authors would like to thank Yoshihito Toyama for discussing the definitions and the proofs.

## References

[1] Dershowitz, N., Orderings for term-rewriting systems, *Theor. Comput. Sci.* **17**, (1982), 279–301.

[2] Dershowitz, N., Termination of rewriting, *J. Symbolic Computation* **3**, (1987), 69–116.

[3] Futatsugi, K. and Toyama, Y., Term rewriting systems and their applications: a survey, *J. IPS Japan* **24**, (1983), 133–146.

[4] Kurihara, M. and Kaji, I., Modular term rewriting systems: termination, confluence, and strategies, *IEICE Tech. Rep.* **COMP88**, (1988), 57–66.

[5] Middeldorp, A., A sufficient condition for the termination of the direct sum of term rewriting systems, Proc. 4th IEEE symp. Logic in Computer Science, Asilomar, 1989.

[6] Middeldorp, A., Modular aspects of properties of term rewriting systems related to normal forms, Proc. 3rd Conf. Rewriting Techniques and Applications, *Lecture Notes in Computer Science* **355**, (1989).

[7] Rusinowitch, M., On termination of the direct sum of term-rewriting systems, *Inf. Process. Lett.* **26**, (1987), 65–70.

[8] Rusinowitch, M., Path of subterms ordering and recursive decomposition ordering revisited, *J. Symbolic Computation* **3**, (1987), 117–131.

[9] Sakai, K., Knuth-Bendix algorithm and it's applications, *Computer Software* **4**, (1987), 2–22.

[10] Toyama, Y., On the Church-Rosser property for the direct sum of term rewriting systems, *J. ACM* **34**, (1987), 128–143.

[11] Toyama, Y., Counterexamples to termination for the direct sum of term rewriting systems, *Inf. Process. Lett.* **25**, (1987), 141–143.

[12] Toyama, Y., Klop, J.W. and Barendregt, H.P., Termination for the direct sum of left-linear term rewriting systems, Proc. 3rd Conf. Rewriting Techniques and Applications, *Lecture Notes in Computer Science* **355**, (1989).

[13] Yamanaka, H., Language for modular programming on term rewriting systems, *IEICE Tech. Rep.* **COMP88**, (1988), 67–76.