PAPER

# An Improved Recursive Decomposition Ordering for Higher-Order Rewrite Systems

**Munehiro IWAMI**[†], *Nonmember*, **Masahiko SAKAI**[††], *and* **Yoshihito TOYAMA**[†], *Member*

**SUMMARY** Simplification orderings, like the recursive path ordering and the improved recursive decomposition ordering, are widely used for proving the termination property of term rewriting systems. The improved recursive decomposition ordering is known as the most powerful simplification ordering. Recently Jouannaud and Rubio extended the recursive path ordering to higher-order rewrite systems by introducing an ordering on type structure. In this paper we extend the improved recursive decomposition ordering for proving termination of higher-order rewrite systems. The key idea of our ordering is a new concept of pseudo-terminal occurrences.

***key words:*** *higher-order rewrite system, term rewriting system, termination, improved recursive decomposition ordering, pseudo-terminal occurrence, type, simplification ordering*

## 1. Introduction

Term rewriting systems (TRSs) are regarded as a computation model that reduces terms by applying directed equations, called rewrite rules[3]. TRSs are widely used as a model of functional programming languages and as a basis of automated theorem proving. Since TRSs can't naturally handle higher-order terms, higher-order rewrite systems (HRSs) were suggested to handle directly higher-order terms[11].

The terminating property is fundamental notion of TRSs as computation models[4]. Since the terminating property of TRS is undecidable in general, several sufficient conditions for proving this property have been successfully developed in particular cases. These techniques can be classified into two approaches: semantic methods and syntactic methods.

Simplification orderings are representatives of syntactic methods[14], [17]. Many simplification orderings (for instance, the recursive path ordering (with status) (RPO(S))[1], the improved recursive decomposition ordering (with status) (IRD(S))[13], [15] and so on) have been defined on TRSs. IRDS is among the most powerful simplification orderings[15], [16].

In case of HRSs, the terminating property is also important. Loría-Sáenz and Steinbach[9] first extended RPOS for proving termination of HRSs. Their extension method is based on an interpretation that maps normalized terms to first-order terms. Lysne and Piris[10] also extended RPOS for proving termination of HRSs by introducing the notions of termination functions, critical positions and dominations. We gave an extension of IRDS to HRSs in [5]–[7] according to Lysne and Piris's method. Jouannaud and Rubio[8] gave a simple definition of RPOS for HRSs by introducing an ordering on type structure. Van de Pol extended semantic method to HRSs[12].

In this paper we propose IRDS for HRSs, called the higher-order improved recursive decomposition ordering (HIRDS). Our method is inspired by Jouannaud and Rubio's idea for RPOS[8] and particular properties of IRDS. Further we show that our ordering is more powerful ordering than their ordering.

In Sect. 2 we give the basic notations. Section
presents the definition of the higher-order improved re
cursive decomposition ordering (HIRDS) by introduc
ing a new concept of pseudo-terminal occurrences an
shows that HIRDS is the powerful tool for proving te
mination of HRSs.

## 2. Preliminaries

We mainly follow the basic notations[8]. An *abstra*
*reduction system* (ARS for short) is a pair $\langle A, \rightarrow \rangle$ co
sisting of a set $A$ and a binary relation $\rightarrow \subseteqq A \times$
The relation $\rightarrow^*$ is the reflexive and transitive closu
of $\rightarrow$, and the relation $\leftrightarrow^*$ is reflexive, symmetric ar
transitive closure of $\rightarrow$. If there is no element $b \in$
such that $a \rightarrow b$, then we say $a \in A$ is a *normal for*
(with respect to $\rightarrow$). If $b \in A$ is a normal form su
that $a \rightarrow^* b$ then we say that $b$ is a normal form of
We say that ARS $\langle A, \rightarrow \rangle$ is *terminating* if there is
infinite sequence $a_0 \rightarrow a_1 \rightarrow a_2 \rightarrow \ldots$ of elements in
ARS $\langle A, \rightarrow \rangle$ is *confluent* if for any $a, b, c \in A$, $a \rightarrow^*$
and $a \rightarrow^* c$ implies that there exists $d \in A$ such that
$\rightarrow^* d$ and $c \rightarrow^* d$. A binary relation on a set $A$ is call
a (*strict*) *partial ordering* over $A$ if it is a irreflexive a
transitive on $A$. The partial ordering is usually denot
by $>$. A partial ordering $>$ on a set $A$ is *well-found*
if $>$ has no infinite descending sequences, i.e., there
no sequence of the form $a_0 > a_1 > a_2 > \cdots$ of eleme
in $A$.

Let $S$ be a set of *basic types* (or *sorts*). The

Manuscript received August 5, 1997.
Manuscript revised January 19, 1998.

[†]The authors are with the Faculty of Information Science, Japan Advanced Institute of Science and Technology, Hokuriku, Ishikawa-ken, 923–1292 Japan.
[††]The author is with the Faculty of Information Science, Nagoya University, Nagoya-shi, 464–8603 Japan.

$\mathcal{T}$ of *types* is generated from the set of basic types by constructor $\rightarrow$ as follows: $\mathcal{T} := \mathcal{S} \mid \mathcal{T} \rightarrow \mathcal{T}$. We use $\sigma$, $\tau$ and $\rho$ to denote types. *Type declarations* are expressions of the form $\sigma_1 \times \cdots \times \sigma_n \rightarrow \sigma$, where $\sigma_1$ , $\cdots$ , $\sigma_n$ , $\sigma$ are types. Types occurring before the arrow in the expressions are called *input types*, while the type occurring after the arrow is called the *output type*. The latter is assumed to be a basic type. Let $\mathcal{F}_{\sigma_1 \times \cdots \times \sigma_n \rightarrow \sigma}$ be a set of *function symbols* that equipped with a fixed number $n$ (called the *arity*) of arguments of respective types $\sigma_1,\cdots,\sigma_n$, and an output type (assumed to be a basic type) $\sigma$. A *signature* $\mathcal{F}$ is a set of all function symbols as follows:

$$\mathcal{F} = \bigcup_{\sigma_1,\cdots,\sigma_n,\sigma} \mathcal{F}_{\sigma_1 \times \cdots \times \sigma_n \rightarrow \sigma}$$

We will assume that there are finitely many symbols of a given output type $\sigma$.

The set of *untyped terms* is generated from a denumerable set $\mathcal{X}$ of *variables* according to the grammar: $\mathcal{L} := \mathcal{X} \mid (\lambda\, \mathcal{X}.\, \mathcal{L}) \mid \mathcal{L}(\mathcal{L}) \mid \mathcal{F}\,(\mathcal{L},\cdots,\mathcal{L})$. The application of $s$ to $t$ is denoted by $s(t)$. We write $s(t_1,\cdots,t_n)$ for $s(t_1)\cdots(t_n)$. We use $FV(t)$ for the set of *free variables* of $t$. Also we use $BV(t)$ for the set of *bound variables* of $t$. We may assume for convenience that bound variables are all different, and are different from the free ones.

*Typing rules* restrict the set of terms constraining them to follow a precise discipline. *Environments* are sets of pairs of the form $x{:}\sigma$, where $x$ is a variable and $\sigma$ is a type. *Typing judgments* are written as $\Gamma \vdash s{:}\sigma$ if the term $s$ can be proved to have the type $\sigma$ in the environment $\Gamma$ :

$X : \sigma \in \Gamma$ implies $\Gamma \vdash X : \sigma$.
$F : \sigma_1 \times \cdots \times \sigma_n \rightarrow \sigma$ and $\Gamma \vdash t_1 : \sigma_1 \cdots \Gamma \vdash t_n : \sigma_n$ imply $\Gamma \vdash F(t_1,\cdots,t_n) : \sigma$.
$\Gamma \cup \{x : \sigma\} \vdash s : \tau$ implies $\Gamma \vdash (\lambda x : \sigma.s) : \sigma \rightarrow \tau$.
$\Gamma \vdash s : \sigma \rightarrow \tau$ and $\Gamma \vdash t : \sigma$ imply $\Gamma \vdash s(t) : \tau$.

A term $s$ has type $\sigma$ in the environment $\Gamma$ if $\Gamma \vdash s : \sigma$ is provable in the above typing rules. A term $s$ is typable in the environment $\Gamma$ if there exists a type $\sigma$ such that $s$ has type $\sigma$ in the environment $\Gamma$. A term $s$ is typable if it is typable in some environment $\Gamma$. We will only consider these typable terms in this paper. A term is *ground* if it contains no free variables.

*Substitutions* are written as in $\{x_1 \leftarrow t_1,\cdots,x_n \leftarrow t_n\}$ where term $t_i$ is assumed different from variable $x_i$ and $x_i$ and $t_i$ have the same type ($i = 1,\ldots,n$). We use the letter $\gamma$ for substitutions. Substitutions behave as endomorphisms defined on free variables. Letting $\gamma = \{x_1 \leftarrow t_1,\cdots,x_n \leftarrow t_n\}$, $dom(\gamma)$ denotes the set $\{x_1,\cdots,x_n\}$. A substitution $\gamma$ is a *ground* if $x\gamma$ is a ground term for all $x \in dom(\gamma)$.

Two rules originate from the *$\lambda$-calculus*, *$\beta$-reduction* and *$\eta$-expansion*:

$(\lambda\, x.s)(t) \rightarrow_\beta s\, \{x \leftarrow t\}$,

$s \rightarrow_\eta (\lambda\, x.s(x))$ if $s : \sigma \rightarrow \tau$, $x : \sigma \notin Var(s)$ and $s$ is not an abstraction.

Given a term $s$, we will denote by $s\!\downarrow$ its *$\eta$-long $\beta$-normal form*, defined as the $\beta$-normal form of its $\eta$-expansion. We say that $s$ is *normalized* when $s = s\!\downarrow$, and use $\leftrightarrow^*_{\beta\eta}$ for the congruence generated by these two rules. If $s \leftrightarrow^*_{\beta\eta} t$ then $s$ and $t$ are called *$\beta\eta$-equivalent*. We suppose that for every type $\sigma$ there is a function symbol of type $\sigma$ not occurring $\mathcal{F}$ that is denoted by $\Box$ (called hole). *Context* is a normalized term with at least one occurrence of $\Box$. Context with only one occurrence of $\Box$ is denoted by $C[\ \ ]$. Higher-order variable is a variable having a type such that is not basic. Normalized terms with no abstraction and no higher-order variables are called *algebraic*. An algebraic term is *ground* if it contains no variables. $|t|$ denotes the size of term $t$, i.e., the total number of function symbols and variables occurring in $t$. The *type of algebraic term* $t$ are denoted by *type*($t$). Normalized terms are either one of the two forms[8] : $(\lambda\, x.s)$ for some normalized term $s$, or $f(s_1,\cdots,s_n)$ for some $f \in \mathcal{F} \cup \mathcal{X}$ and normalized terms $s_1, \cdots, s_n$.

Algebraic terms and normalized terms are identified with finite trees. *Occurrences* in a term can be viewed as a finite sequence of natural numbers, pointing out a path from the root of this tree. $O(t)$ denotes the set of all occurrences of a term $t$. $Ot(t)$ denotes the set of all *terminal occurrences* (occurrences of all leaves) of the term $t$. The letter $\epsilon$ denotes *root occurrences*. We write $w \preceq z$ if $w$ is a prefix of $z$. The *subterms* of $t$ at position $p$ is denoted by $t|_p$, and we write $t \trianglerighteq t|_p$. If $t \trianglerighteq t|_p$ and $t \neq t|_p$ then $t|_p$ is called the *proper subterm* of $t$, denoted by $t \vartriangleright t|_p$. Function symbol positioned at root of a term $t$ is denoted by *top*($t$). The result of replacing $t|_p$ at position $p$ in $t$ by $u$ is denoted by $t[u]_p$.

A *higher-order rewrite system* (HRS) is a set of *rewrite rules* $R = \{\Gamma_i{:}l_i \rightarrow r_i\}_i$, where rewrite rule $l_i \rightarrow r_i$ satisfies the following constrains: $l_i$ and $r_i$ are normalized terms such that $l_i$ is not $\beta\eta$-equivalent to free variable, $FV(l_i) \supseteq FV(r_i)$ and $l_i$ and $r_i$ have the same type $\sigma_i$ in the environment $\Gamma_i$. Given a higher-order rewrite system $R$, a normalized term $s$ is rewritten to a term $t$ at position $p$ with the rewrite rule $l \rightarrow r$ and the substitution $\gamma$, written $s \rightarrow^p_{l\rightarrow r} t$, or simply $s \rightarrow_R t$, if $s|_p \leftrightarrow^*_{\beta\eta} l\gamma$ and $t = s[r\gamma\!\downarrow]_p$. Note that $t$ is normalized since $s$ is.

A partial ordering $>$ is *stable under substitutions* if and only if $u > v$ implies $u\gamma > v\gamma$, for any substitution $\gamma$. And a partial ordering $>$ is *stable under contexts* if and only if $u > v$ implies $C[u] > C[v]$, for any context $C$. *Rewrite ordering* is defined as partial ordering such that stable under substitutions and contexts. *Reduction ordering* is well-founded rewrite ordering. Reduction ordering is used to prove the termination of higher-order rewrite systems by comparing the left and right hand sides of rewrite rules. Given a binary relation $>$, *multiset extension* $\gg$ is defined as the transitive closure of following relation $\Rightarrow$ on mutlisets. $M \cup \{s\} \Rightarrow M \cup \{t_1,\cdots,t_n\}$ where $n \geq 0$ and $s > t_i$ for any $i \in \{1,\cdots,n\}$. Assume $>_1$ and $>_2$ are well-founded orderings on sets $A_1$ and $A_2$, respectively. Then the *lexicographic extension* $(>_1,>_2)_{lex}$ is well-founded ordering on $A_1 \times A_2$ [2]. Assume $>$ is a well-founded ordering on a set $A$. Then $\gg$ is a well-founded ordering on the multisets of elements of $A$ [2]. Finally, *simplification ordering* [1], [4] on ground terms is partial ordering that is stable under contexts and has the *subterm property*, i.e. any term is strictly bigger than any of its proper subterms.

## 3. Higher-Order Improved Recursive Decomposition Ordering

Higher-order improved recursive decomposition ordering is defined by two comparison stages. In the first stage, higher-order terms $s$ and $t$ interpret into algebraic terms $s'$ and $t'$ having the extended signature $\lambda\mathcal{F}$. In the second stage, $s'$ and $t'$ are compared by a typed improved recursive decomposition ordering on algebraic terms.

If we simply extended *improved recursive decomposition ordering* (IRDS) [14]–[17] to handle higher-order by using Jouannaud and Rubio's idea of type structure [8] for the second stage, the resulting ordering would not be stable under ground normalized substitutions. For instance, let $a{:}\sigma$, $b{:}\sigma \in \mathcal{F}$, $Y{:}\sigma \rightarrow \sigma \in \mathcal{X}$ and $a >_{\mathcal{F}} b$. Let $s = Y(a)$, $t = a$. Since improved recursive decomposition ordering $>_{IRDS}$ is simplification ordering [14], [15], it has a subterm property. Hence $s >_{IRDS} t$ holds. For the substitution $\gamma = \{Y \leftarrow \lambda\ x.b\}$, $s\,\gamma\downarrow = b <_{IRDS} a = t\,\gamma\downarrow$. In order to avoid this problem, we introduce the notion of pseudo-terminal occurrences.

First, we define typed improved recursive decomposition ordering on algebraic terms. It requires a partial ordering on types in addition to a partial ordering on function symbols.

**Definition 1** ([8]): *Quasi-ordering on types* $\geq_T$ is *compatible* with the term structure if the following conditions holds: for all ground normalized terms $s$, $s{:}\sigma \trianglerighteq t{:}\tau$ implies $\sigma \geq_T \tau$.

In the rest of paper, we assume $>_T$, strict part of $\geq_T$, is well-founded and $\geq_T$ is compatible with term structure.

Let $>_S$ be a partial ordering on basic types and constructor $\rightarrow$. The recursive path ordering (RPOS) [1], [14]–[17] based on precedence $>_S$ on $S \cup \{\rightarrow\}$ is well-founded and compatible [8]. Hence we can take the *recursive path ordering* (RPOS) as a partial ordering $>_T$ on types.

**Example 2:** Let *List* and *Nat* be basic types. Given the following precedence: $List >_S Nat$ and $List >_S \rightarrow$. Then $List >_T Nat \rightarrow Nat$ holds, since the ordering $>_T$ is the recursive path ordering on types and we consider the type $Nat \rightarrow Nat$ as the form $\rightarrow(Nat, Nat)$.

Partial ordering $>_{\mathcal{F}}$ on function symbols compares symbols of the same output type only and must be well-founded. A variable $x{:}\sigma$ is considered as a function symbol comparable only itself. *Status* is a function $ST$: $\mathcal{F} \rightarrow \{mult, left, right\}$. Therefore a function symbol have one of the following three statuses: $mult$ (the arguments will be compared as multiset), $left$ (lexicographical comparison from left to right), $right$ (lexicographical comparison from right to left). The results of an application of function $args$ to a term $t = f(t_1,\cdots,t_n)$ depend on the status of $f$ : If $ST(f) = mult$, then $args(t)$ is the multiset $\{t_1,\cdots,t_n\}$ and otherwise, $args(t)$ is the tuple $(t_1,\cdots,t_n)$. The notation

$$-s >_1 t$$
$$-s' >_2 t'$$

means $s >_1 t$ or ($s =_1 t$ and $s' >_2 t'$).

**Definition 3** ([8]): The *extended signature* $\lambda\mathcal{F}$ is defined by the following:

$$\widetilde{\mathcal{X}} = \bigcup_{\sigma \in T} \{X_\sigma \mid X : \sigma \in \mathcal{X}\},$$
$$\widetilde{\mathcal{F}} = \{F_\sigma \mid F \in \mathcal{F}_{\sigma_1 \times \cdots \times \sigma_n \rightarrow \sigma} (\subseteqq \mathcal{F})\}$$

and $\lambda\mathcal{F} = \widetilde{\mathcal{X}} \cup \widetilde{\mathcal{F}} \cup \bigcup_{\sigma \in T}\{c_\sigma\} \cup \bigcup_{\sigma,\tau \in T} \{\lambda_{\sigma \rightarrow \tau}\}$ where $T$ is a set of types which are equipped for each function symbols in $\mathcal{F}$.

Note that we have a single symbol $c_\sigma$ for each type $\sigma$ and unary symbol $\lambda_{\sigma \rightarrow \tau}$ for each type $\sigma \rightarrow \tau$. Hence if $\mathcal{F}$ has a finite set of function symbols for each type, then $\lambda\mathcal{F}$ has a finite set of function symbols for each type. The precedence $>_{\mathcal{F}}$ on $\mathcal{F}$ will be extended to the precedence $>_{\lambda\mathcal{F}}$ on $\lambda\mathcal{F}$. Each function symbol in $\widetilde{\mathcal{X}}$ is comparable by only to itself and does not have a status. The symbol $\lambda_{\sigma \rightarrow \tau}$ may have one of any status. From the compatibility with the term structure $(\lambda_{\sigma \rightarrow \tau}(t : \tau)){:}\sigma \rightarrow \tau$, we require that $\sigma \rightarrow \tau \geq_T \tau$. In the sequel, we consider algebraic terms on $\lambda\mathcal{F}$.

We generalize the definition of a set of terminal occurrences. We define pseudo-terminal occurrences of term $t$ as follows.

**Definition 4:** Let $t$ be an algebraic term. An occurrence $q \in O(t)$ is a *pseudo-terminal occurrence* if it satisfies both of the following conditions.

(1) $q \in Ot(t) \vee top(t\,|_q) \in \widetilde{\mathcal{X}}$.

(2) $\forall q' \prec q, [top(t\,|_{q'}) \notin \widetilde{\mathcal{X}}]$.

We write $q \in Op(t)$ if $q$ is a pseudo-terminal occurrence of $t$.

Path-decomposition and decomposition [15], [16] can be naturally extended by using pseudo-terminal occurrences. For $u \in Op(t)$, *path-decomposition* $dec_u(t)$ is defined as follows.

$$\begin{cases} dec_\epsilon(t) = \{t\} \\ dec_{i.v}(f(t_1,\cdots,t_n)) = \{f(t_1,\cdots,t_n)\} \cup dec_v(t_i) \end{cases}$$

Note that $i.v \in Op(f(t_1,\cdots,t_n))$ implies $v \in Op(t_i)$. We also define a *decomposition* $dec$ $(\{t_1,\cdots,t_n\}) = \{dec_u(t_i) \mid i \in \{1,\cdots,n\},\ u \in Op(t_i)\}$. For path-decomposition $dec_u(t)$, $sub(dec_u(t),s) = \{s' \in dec_u(t) \mid s \rhd s'\}$.

**Example 5:** Let $List$ and $Nat$ be basic types. We consider the term $s = map_{List}$ $(\lambda_{Nat\to Nat}$ $(X_{Nat}$ $(c_{Nat}))$, $nil_{List})$ where $X_{Nat} \in \tilde{\mathcal{X}}$. Figure 1 denotes the tree structure of $s$. We have $Op$ $(s)$ $=$ $\{11,\ 2\}$. Then $dec(\{s\}) = \{dec_{11}(s),\ dec_2(s)\}$ where $dec_{11}(s) = \{s,\ \lambda_{Nat\to Nat}(X_{Nat}(c_{Nat})),\ X_{Nat}(c_{Nat})\}$ and $dec_2(s) = \{s,\ nil_{List}\}$.

Further $sub(dec_{11}(s),\ \lambda_{Nat\to Nat}(X_{Nat}(c_{Nat}))) = \{X_{Nat}(c_{Nat})\}$ and $sub(dec_2(s),s) = \{nil_{List}\}$.

We define the typed improved recursive decomposition ordering (TIRDS) as following. This ordering is based on improved recursive decomposition ordering (IRDS) defined by Steinbach [15], [16] and extended by introducing the notion of pseudo-terminal occurrences and type ordering.

**Definition 6** (TIRDS): Let $s$ and $t$ be algebraic terms. The *typed improved recursive decomposition ordering* (TIRDS) on algebraic terms is defined as follows:
$s >_{TIRDS} t \iff dec(\{s\}) \ggg_{EL} dec(\{t\})$ where $\ggg_{EL}$ is the multiset extension of $\gg_{EL}$.
$dec_p(u) \ni u' >_{EL} v' \in dec_q(v)$ is defined by the following (1) and (2).

(1) $type(u') >_T type(v')$, or,

(2) $type(u') = type(v')$, and,

 a) $top(u') >_{\lambda\mathcal{F}} top(v')$,

 b) $top(u') = top(v')$, $ST(top(u')) = mult$,

  – $sub(dec_p(u),u')$ $\gg_{EL} sub(dec_q(v),v')$.

  – $dec(args(u')) \ggg_{EL} dec(args(v'))$.

 c) $top(u')$ $=$ $top(v')$, $ST(top(u'))$ $\neq$ $mult$, $args(u')$ $>_{TIRDS,ST(top(u'))}$ $args(v')$, $\{u'\}$ $\gg_{TIRDS}$ $args(v')$.

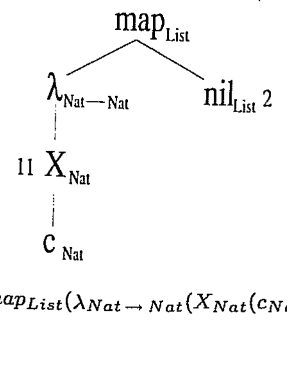


Fig. 1 $map_{List}(\lambda_{Nat\to Nat}(X_{Nat}(c_{Nat})), nil_{List})$.

**Example 7:** We compare the following algebraic terms with respect to TIRDS. Let $List$ and $Nat$ be basic types. Let $s$ $=$ $map_{List}(\lambda_{Nat\to Nat}(X_{Nat}(c_{Nat}))$, $cons_{List}(N_{Nat},L_{Nat}))$ and, $t = cons_{List}(X_{Nat}$ $(N_{Nat})$, $map_{List}(\lambda_{Nat\to Nat}(X_{Nat}(c_{Nat}))$, $L_{list}))$. Then we have $dec(\{s\})$ $=$ $\{dec_{11}(s),\ dec_{21}(s),\ dec_{22}(s)\}$ and $dec(\{t\}) = \{dec_1(t),\ dec_{211}(t),\ dec_{22}(t)\}$.

Given the following precedences: $map_{List} >_{\lambda\mathcal{F}} cons_{List}$, $List >_S Nat$, $List >_T \to$ and $ST(map_{List})$ $= mult$, we have $s >_{EL} t$ by $top(s) >_{\lambda\mathcal{F}} top(t)$ and $cons_{List}(N_{Nat},L_{List}) >_{EL} X_{Nat}(N_{Nat})$ by $List >_T Nat$. Hence $dec_{21}$ $(s)$ $=$ $\{s,\ cons_{List}$ $(N_{Nat},L_{List})\}$ $\gg_{EL}$ $\{t,\ X_{Nat}$ $(N_{Nat})\}$ $= dec_1$ $(t)$.

Next, we compare $dec_{21}(s)$ and $dec_{211}(t)$ $=$ $\{t,$ $t',$ $\lambda_{Nat\to Nat}$ $(X_{Nat}$ $(c_{Nat}))$,$X_{Nat}$ $(c_{Nat})\}$ where $t'$ $= map_{List}$ $(\lambda_{Nat\to Nat}$ $(X_{Nat}$ $(c_{Nat}))$, $L_{List})$. We have $s >_{EL} \lambda_{Nat\to Nat}(X_{Nat}(c_{Nat}))$ holds by $List >_T Nat \to Nat$. We also have $s >_{EL} t'$ since $top(s)$ $= top(t')$, $ST(top(s))$ $=$ $mult$ and $sub(dec_{21}(s),s)$ $=$ $\{cons_{List}(N_{Nat},L_{List}),\ L_{List}\}$ $\gg_{EL}$ $\{\lambda_{Nat\to Nat}$ $(X_{Nat}$ $(c_{Nat}))$, $X_{Nat}$ $(c_{Nat})\}$ $=$ $sub(dec_{211}(t),t')$. Hence $dec_{21}(s) \gg_{EL} dec_{211}(t)$.

Further $dec_{22}$ $(s)$ $=$ $\{s,$ $cons_{List}$ $(N_{Nat},L_{List})$, $L_{List}\}$ $\gg_{EL}$ $\{t,$ $map_{List}$ $(\lambda_{Nat\to Nat}(X_{Nat}(c_{Nat}))$, $L_{List})$, $L_{List}\}$ $= dec_{22}$ $(t)$.

Therefore we have $dec$ $(\{s\}) \ggg_{EL} dec$ $(\{t\})$, i.e., $s >_{TIRDS} t$.

Higher-order improved recursive decomposition ordering is defined as TIRDS comparing algebraic terms interpreted from given normalized terms. Thus, we define an interpretation function that maps normalized terms to algebraic terms as follows.

**Definition 8:** Let $V$ be a set of variables. An *interpretation function* $\|\ \ \|_V$ from normalized terms to algebraic terms over the signature $\lambda\mathcal{F}$ is defined by:
$\|(\lambda x.s){:}\sigma\to\tau\|_V = \lambda_{\sigma\to\tau}(\|s\|_V)$.
$\|F(s_1,\cdots,s_n){:}\sigma\|_V = F_\sigma(\|s_1\|_V,\cdots,\|s_n\|_V)$ if $F \in \mathcal{F}$.
$\|X(s_1,\cdots,s_n){:}\sigma\|_V = X_\sigma(\|s_1\|_V,\cdots,\|s_n\|_V)$ if $X \in \mathcal{X}\backslash V$.
$\|x(s_1,\cdots,s_n){:}\sigma\|_V = c_\sigma(\|s_1\|_V,\cdots,\|s_n\|_V)$ if $x \in V$.

Let $s$ be a normalized term and $V \supseteqq BV(s)$. We have $\|s\|_V|_p = \|s|_p\|_V$, for any $p \in O(s)$. Whenever we write $\|s\|_V$ for normalized terms $s$, we assume that $V \supseteqq BV(s)$.

**Example 9:** Given the following set of basic types and signature: $S = \{Nat,\ List\}$, $\mathcal{F} = \{nil{:}List,\ cons{:}Nat \times List \to List,\ map{:}(Nat \to Nat) \times List \to List\}$ and $\mathcal{X} = \{X{:}Nat \to Nat,\ N{:}Nat,\ L{:}List\}$.

Let $V = \{x\}$. Let $s$ be $map$ $(\lambda x.X$ $(x)$,$cons$ $(N,L))$. The followings are examples that explain the interpretation.

- $\|s\|_V$ $=$ $map_{List}(\lambda_{Nat\to Nat}$ $(X_{Nat}$ $(c_{Nat}))$, $cons_{List}$ $(N_{Nat},L_{List}))$. (See Fig. 2).

- $\|s|_{11}\|_V = \|X$ $(x)\|_V = X_{Nat}$ $(c_{Nat})$.

- $\|s\|_V|_{11} = map_{List}\ (\lambda_{Nat\to Nat}\ (X_{Nat}\ (c_{Nat})),\ cons_{List}\ (N_{Nat},\ L_{List}))\ |_{11} = X_{Nat}\ (c_{Nat})$.

We define the higher-order improved recursive decomposition ordering by using definitions 6 and 8.

Note that we assume bound variables may appear on only the leaf position of normalized terms in the rest of paper.

**Definition 10** (HIRDS): Let $s$ and $t$ be normalized terms. Then the *higher-order improved recursive decomposition ordering* (HIRDS) on normalized terms is defined as follows: $s >_{HIRDS} t \Longleftrightarrow \|s\|_V >_{TIRDS} \|t\|_V$ where $V = BV(s) \cup BV(t)$.

In the sequel, $\|s\|_V$ and $\|t\|_V$ are abbreviated just as $\|s\|$ and $\|t\|$, respectively.

We must show that HIRDS is a partial ordering that is stable under ground normalized substitutions and is stable under ground contexts and is well-founded on ground normalized terms. These properties are essential for applying HIRDS to termination proof of HRS.

**Lemma 11:** The HIRDS is partial ordering on normalized terms.

**Proof.** By definition 10, it is enough to show that $>_{TIRDS}$ is partial ordering on algebraic terms. To prove the transitivity of $\ggg_{EL}$, it is enough to show that $>_{EL}$ is transitive. Let $s' \in dec_p(s)$, $t' \in dec_q(t)$ and $u' \in dec_r(u)$. If $type(s') >_T type(t')$ or $type(t') >_T type(u')$, the claim is trivial. If $type(s') = type(t')$ and $type(t') = type(u')$, we can show that $dec_p(s) \ni s' >_{EL} u' \in dec_q(t)$ by induction on $|s'| + |t'| + |u'|$.

We next show that $s \not>_{TIRDS} s$ by proving the irreflexivity of $>_{EL}$. For any term $s$ and $s'$ in $dec_p(s)$, we can easily prove that $dec_p(s) \ni s' \not>_{EL} s' \in dec_p(s)$ by induction on $|s|$. □

**Lemma 12:** The HIRDS has the subterm property for ground normalized terms.

**Proof.** By definition 10, we must show that the TIRDS has the subterm property for algebraic terms. Let $s$ and $t$ be algebraic terms such that $s \rhd t$. It is shown by induction on $|s|$ that $s >_{TIRDS} t$. □

We prove that the stability under ground normalized substitution $\gamma$ where $dom(\gamma)$ is singlton set. Since any ground sbstitution $\gamma$ can be denoted by composition

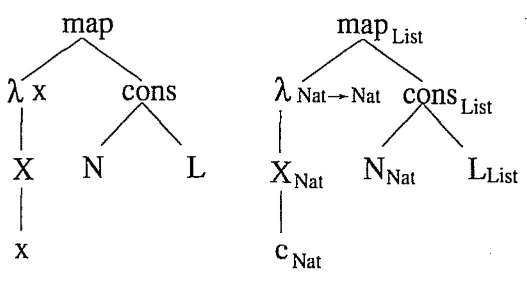


**Fig. 2** $map\ (\lambda x.X(x), cons\ (N,L))$ and $\|map\ (\lambda x.X\ (x), cons\ (N,L))\|_{\{x\}}$.

of ground substitutions $\gamma_i$ where $dom(\gamma_i)$ is a singleton set, the stability under ground normalized substitution is easily reduced from the above statement.

Figures 3, 4 and 5 exhibit the cases (1), (2) and (3) in the lemma 13, respectively.

**Lemma 13:** Let $dec_w(\|s\|) \gg_{EL} dec_z(\|t\|)$ where $s$ and $t$ be normalized terms and $w \in Op(s)$, $z \in Op(t)$. Then for any ground normalized substitution $\gamma$, the following three claims hold.

(1) If $\|s\||_w = \|t\||_z$ and $top\ (\|t\||_z) \in \widetilde{\mathcal{X}}$ then $dec_{w.j}(\|s\gamma\downarrow\|) \gg_{EL} dec_{z.j}(\|t\gamma\downarrow\|)$, for any $j \in N^*$ such that $z.j \in Op\ (\|t\gamma\downarrow\|)$.

(2) If $\|s\||_w \neq \|t\||_z$ and $top\ (\|t\||_z) \in \widetilde{\mathcal{X}}$ then $dec_{w.i}(\|s\gamma\downarrow\|) \gg_{EL} dec_{z.j}(\|t\gamma\downarrow\|)$, for any $j, i \in N^*$ such that $z.j \in Op\ (\|t\gamma\downarrow\|))$ and $w.i \in Op\ (\|s\gamma\downarrow\|)$.

(3) If $top\ (\|t\||_z) \notin \widetilde{\mathcal{X}}$ then $dec_{w.i}(\|s\gamma\downarrow\|) \gg_{EL} dec_z(\|t\gamma\downarrow\|)$, for any $i \in N^*$ such that $w.i \in Op(\|s\gamma\downarrow\|)$.

**Proof.** See Appendix A. □

**Lemma 14:** Let $s$ and $t$ be algebraic terms. Then $dec(\{s\}) \ggg_{EL} dec(\{t\})$ implies $dec(\{s\}) \cap dec(\{t\}) = \emptyset$.

**Proof.** Let $M \in dec(\{s\}) \cap dec(\{t\})$. Then $M = dec_w(s) = dec_z(t)$ for some $w \in Op(s)$ and $z \in Op(t)$. Then $s = t$ must hold by definition of path-decomposition. This contradicts the assumption from irreflexivity of $\gg_{EL}$. Hence $dec(\{s\}) \cap dec(\{t\}) = \emptyset$. □

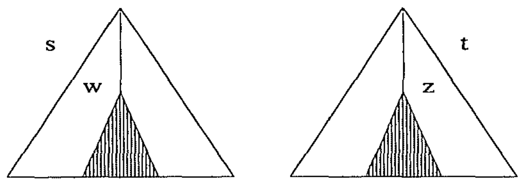


**Fig. 3** Case (1) in lemma 13.

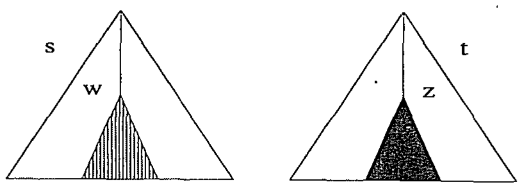


**Fig. 4** Case (2) in lemma 13.

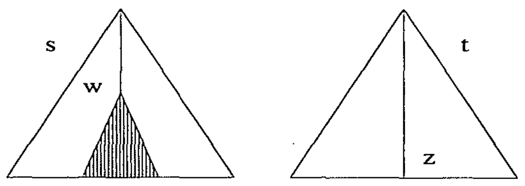


**Fig. 5** Case (3) in lemma 13.

**Lemma 15:** The HIRDS is stable under ground normalized substitutions $\gamma$, i.e., $s >_{HIRDS} t$ implies $s\gamma\downarrow >_{HIRDS} t\gamma\downarrow$.

**Proof.** Assume that $s >_{HIRDS} t$, i.e., $dec(\{\|s\|\}) \gg\gg_{EL} dec(\{\|t\|\})$ where $s$ and $t$ are normalized terms. We show that $dec(\{\|s\gamma\downarrow\|\}) \gg\gg_{EL} dec(\{\|t\gamma\downarrow\|\})$ holds for any ground normalized substitution $\gamma$. By lemma 14, we can assume that for any $z \in Op(\|t\|)$, there exists $w \in Op(\|s\|)$ such that $dec_w(\|s\|) \gg_{EL} dec_z(\|t\|)$. Then we can show the following claim by definition of multiset extension and lemma 13: for any $z' \in Op(\|t\gamma\downarrow\|)$, there exists $w' \in Op(\|s\gamma\downarrow\|)$ such that $dec_{w'}(\|s\gamma\downarrow\|) \gg_{EL} dec_{z'}(\|t\gamma\downarrow\|)$. □

In order to show that the HIRDS is stable under ground contexts in lemma 18, we need the following lemmas.

**Lemma 16:** Let $s$ and $t$ be normalized terms. Let $V = BV(s) \cup BV(t)$. For any set of variables $V'$ such that $V' \supseteqq V$, $\|s\|_V >_{TIRDS} \|t\|_V$ implies $\|s\|_{V'} >_{TIRDS} \|t\|_{V'}$.

**Proof.** Let the difference of the set $V' \setminus V = \{x_1,\cdots,x_m\}$ $(m \geq 0)$. Let $\gamma = \{x_1 \leftarrow c,\cdots,x_m \leftarrow c\}$. By definition 8, we can consider that $\|s\|_{V'}$ and $\|t\|_{V'}$ equal to $\|s\gamma\downarrow\|_V$ and $\|t\gamma\downarrow\|_V$ for $\gamma = \{x_1 \leftarrow c,\cdots,x_m \leftarrow c\}$, respectively. $\|s\gamma\downarrow\|_V >_{TIRDS} \|t\gamma\downarrow\|_V$ holds by lemma 15. Hence $\|s\|_{V'} >_{TIRDS} \|t\|_{V'}$ holds. □

**Lemma 17:** The TIRDS is stable under ground context.

**Proof.** Let $s$ and $t$ be algebraic terms with same type. We have to show that $s >_{TIRDS} t$ implies $C[s] >_{TIRDS} C[t]$ for any ground context $C[\ \ ]$. It can be proved by induction on $|C[\ \ ]|$ by the similar way of IRDS. □

**Lemma 18:** The HIRDS is stable under ground contexts, i.e., $s >_{HIRDS} t$ implies $C[s] >_{HIRDS} C[t]$ for any ground context $C$ and normalized terms $s$ and $t$ with the same type such that $C[s]$ and $C[t]$ are normalized terms.

**Proof.** By definition 10, we prove that $\|s\|_V >_{TIRDS} \|t\|_V$ implies $\|C[s]\|_{V'} >_{TIRDS} \|C[t]\|_{V'}$, where $V = BV(s) \cup BV(t)$ and $V' = BV(C[s]) \cup BV(C[t])$ by induction on $|C[\ \ ]|$. Then it is obvious that $V' \supseteqq V$ holds. It is enough to consider the only case that $C[\ \ ] = F(u_1,\cdots,\square,\cdots,u_n)$. Since lemma 16, $\|s\|_{V'} >_{TIRDS} \|t\|_{V'}$. We consider the following cases.

1. If $F : \sigma_1\times\cdots\times\sigma_n \to \sigma \in \mathcal{F}$ then $\|F\ (u_1,\cdots,s,\cdots,u_n)\|_{V'} = F_\sigma\ (\|u_1\|_{V'},\cdots,\|s\|_{V'},\cdots,\|u_n\|_{V'})$ and $\|F\ (u_1,\cdots,t,\cdots,u_n)\|_{V'} = F_\sigma\ (\|u_1\|_{V'},\cdots,\|t\|_{V'},\cdots,\|u_n\|_{V'})$. Hence $\|F\ (u_1,\cdots,s,\cdots,u_n)\|_{V'} >_{TIRDS} \|F\ (u_1,\cdots,t,\cdots,u_n)\|_{V'}$ by lemma 17.
2. $F = \lambda x$ for some variable $x$ of type $\sigma$. $\|\lambda x.s\|_{V'} = \lambda(\|s\|_{V'})$, $\|\lambda x.t\|_{V'} = \lambda(\|t\|_{V'})$. Hence $\|\lambda x.s\|_{V'} >_{TIRDS} \|\lambda x.t\|_{V'}$ by lemma 17.

□

In order to show the well-foundness of HIRDS on ground normalized terms, we need to show that HIRDS is simplification ordering on ground normalized terms since simplification ordering is well-founded.

**Lemma 19:** The HIRDS is a simplification ordering on ground normalized terms.

**Proof.** By lemmas 11, 12 and 18, the HIRDS is partial ordering on ground normalized terms that is stable under contexts and has the subterm property. □

**Lemma 20:** The HIRDS is well-founded on ground normalized terms.

**Proof.** Since simplification ordering is well-founded [1], [4], the HIRDS is well-founded on ground normalized terms by lemma 19. □

Next theorem guarantees that we can use HIRDS to prove the termination of higher-order rewrite systems.

**Theorem 21:** Let $R$ be a higher-order rewrite system. If $l >_{HIRDS} r$ for any rewrite rule $l \to r$ in $R$ then $R$ is terminating.

**Proof.** Since terms in a sequence can always be considered as ground, we can use the previous properties. Assume that $s \to^{p}_{l\to r} t$, where $s$ and $t$ are ground normalized terms. Then $s|_p =_{\beta\eta} l\gamma$ and $t = s[r\gamma\downarrow]_p$. Since $\to_{\beta\eta}$ is confluent and $s$ is normalized, we have $s|_p = l\gamma\downarrow$. By the assumption $l >_{HIRDS} r$ and lemma 15, hence $l\gamma\downarrow >_{HIRDS} r\gamma\downarrow$ holds. Thus, it follows that $s = s[l\gamma\downarrow]_p >_{HIRDS} s[r\gamma\downarrow]_p = t$ by lemma 18. Since the HIRDS is well-founded on ground normalized terms by lemma 20, $R$ is terminating. □

In the next theorem and example, we compare higher-order improved recursive decomposition ordering (HIRDS) with *higher-order recursive path ordering* (HRPOS) defined in [8]. Improved recursive decomposition ordering (IRDS) is more powerful than recursive path ordering (RPOS) in the frame of term rewriting systems [17]. The similar result holds in the frame of higher-order rewrite systems. We show that HIRDS is more powerful than HRPOS. We use our interpretation function $\|\ \ \|$ in definition 8 for HRPOS instead of it defined in [8].

**Theorem 22:** Let $s$ and $t$ be normalized terms. Then $s >_{HRPOS} t$ implies $s >_{HIRDS} t$.

**Proof.** See Appendix B. □

**Example 23** ([14]): Given the following set of basic type, signature and HRS $R$: $S = \{Bool\}$, $\mathcal{F} = \{\neg{:}Bool \to Bool,\ \supset{:}Bool \times Bool \to Bool,\ \vee{:}Bool \times Bool \to Bool\}$, $\mathcal{X} = \{X{:}Bool,\ Y{:}Bool,\ Z{:}Bool\}$ and

$$R = \{\ \neg X \supset (Y \supset Z) \longrightarrow Y \supset (X \vee Z).$$

Given the following precedence: $\neg >_{\mathcal{F}} \supset >_{\mathcal{F}} \vee$. Then $\neg X \supset (Y \supset Z) >_{HIRDS} Y \supset (X \vee Z)$ but $\neg X \supset (Y \supset Z) \not>_{HRPOS} Y \supset (X \vee Z)$.

**Example 24:** Given the following set of basic types, signature and HRS $R$: $S = \{Nat, List\}$, $\mathcal{F} = \{nil{:}List,\ cons{:}Nat \times List \to List,\ map{:}(Nat \to Nat) \times List \to List\}$, $\mathcal{X} = \{X{:}Nat \to Nat,\ N{:}Nat,\ L{:}List\}$ and

$$R = \begin{cases} map(\lambda x.X(x), nil) \rightarrow nil \\ map(\lambda x.X(x), cons(N, L)) \\ \qquad \rightarrow cons(X(N), map(\lambda x.X(x), L)). \end{cases}$$

The termination proof uses precedences: $List >_S Nat$, $List >_S \rightarrow$, $map >_{\mathcal{F}} cons$ and $ST(map) = mult$. Since it is obvious that $map(\lambda x.X(x),nil) >_{HIRDS} nil$, we consider the second rule. Let $s = map(\lambda x.X(x), cons(N,L))$, $t = cons(X(N),map(\lambda x.X(x),L))$. Then we have $dec(\{\|s\|\}) = \{dec_{11}(\|s\|), dec_{21}(\|s\|), dec_{22}(\|s\|)\}$ and $dec(\{\|t\|\}) = \{dec_1(\|t\|), dec_{211}(\|t\|), dec_{22}(\|t\|)\}$.

By example 7, we have $dec(\{\|s\|\}) \ggg_{EL} dec(\{\|t\|\})$, i.e., $\|s\| >_{TIRDS} \|t\|$. Hence $s >_{HIRDS} t$ holds. Therefore HRS $R$ is terminating.

## 4. Conclusion

We have extended the improved recursive decomposition ordering to higher-order rewrite systems for proving termination. Our extension method is inspired by Jouannaud and Rubio's idea [8] and the particular properties of improved recursive decomposition ordering. We have shown that our ordering is more powerful than the ordering defined by Jouannaud and Rubio. We believe that our ordering is the powerful tools to prove termination of higher-order rewrite systems.

## Acknowledgment

This work is partially supported by Grants from Ministry of Education, Science and Culture of Japan, ♯07680350, ♯07680347 and ♯09245212.

## References

[1] N. Dershowitz, "Orderings for term-rewriting systems," Theoretical Computer Science, vol.17, pp.279–301, 1982.
[2] N. Dershowitz and Z. Manna, "Proving termination with multiset orderings," Commun. ACM, vol.22, no.8, pp.465–476, 1979.
[3] N. Dershowitz and J.P. Jouannaud, "Rewrite systems," in Handbook of Theoretical Computer Science, vol.B, ed. J. van Leeuwen, pp.243–320, The MIT Press/Elsevier, 1990.
[4] N. Dershowitz, "Termination of rewriting," J. Symbolic Computation, vol.3, pp.69–116, 1987.
[5] M. Iwami, M. Sakai, and Y. Toyama, "Termination of higher-order rewrite systems," IEICE Technical Report, COMP95-85, 1996 (in Japanese).
[6] M. Iwami, "Termination of higher-order rewrite systems," Master thesis, JAIST, 1996 (in Japanese).
[7] M. Iwami, M. Sakai, and Y. Toyama, "An improved recursive decomposition ordering for higher-order rewrite systems," IEICE Technical Report, COMP96-73, 1997.
[8] J.P. Jouannaud and A. Rubio, "A recursive path ordering for higher-order terms in $\eta$-long $\beta$-normal form," Proc. 7th International Conf. on Rewriting Techniques and Applications, LNCS, vol.1103, pp.108–122, 1996.
[9] C. Loría-Sâenz and J. Steinbach, "Termination of combined (rewrite and $\lambda$-calculus) systems," Proc. 3th International Conf. on Rewriting Techniques and Applications, LNCS, vol.656, pp.143–147, 1992.
[10] O. Lysne and J. Piris, "A termination ordering for higher order rewrite systems," Proc. 6th International Conf. on Rewriting Techniques and Applications, LNCS, vol.914, pp.26–40, 1995.
[11] T. Nipkow, "Higher-order critical pairs," Proc. IEEE Symp. on Logic in Computer Science, pp.342–349, 1991.
[12] J. van de Pol, "Termination proofs of higher-order rewrite systems," Proc. 1th International Workshop on Higher-Order Algebra, Logic and Rewriting, LNCS, vol.816, pp.305–325, 1993.
[13] M. Rusinowitch, "Path of subterms ordering and recursive decomposition ordering revisited," J. Symbolic Computation, vol.3, pp.117–131, 1987.
[14] J. Steinbach, "Termination of rewriting-extension, comparison and automatic generation of simplification orderings," Ph.D. Thesis, University of Kaiserslautern, 1994.
[15] J. Steinbach, "Term orderings with status," SEKI Report SR-88-12, University of Kaiserslautern, 1988.
[16] J. Steinbach, "Extensions and comparison of simplification orderings," Proc. 3th International Conf. on Rewriting Techniques and Applications, LNCS, vol.335, pp.434–448, 1989.
[17] J. Steinbach, "Simplification ordering: History of results," Fundamenta Informaticae, vol.24, pp.44–87, 1995.

## Appendix A: Proof of Lemma 13

**Lemma 25:** Let $u$ and $v$ be normalized terms. Then, $\|u\|_V = \|v\|_V$ implies $\|u\gamma\downarrow\|_V = \|v\gamma\downarrow\|_V$, for any ground normalized substitution $\gamma$.

**Proof.** We define a restriction of substitution $\gamma$ with respect to set $V$.

$$\gamma/_V(x) = \begin{cases} \gamma(x) & \text{if} \quad x \notin V \\ x & \text{if} \quad x \in V. \end{cases}$$

We show $\|u\|_V = \|v\|_V$ implies $\|u(\gamma/_V)\downarrow\|_V = \|v(\gamma/_V)\downarrow\|_V$ by induction on structure of $u$. We abbreviate an index $V$.

In the case that $u = F(u_1,\cdots,u_m)$ for $F \in \mathcal{F}$ or $u = \lambda x.t$ for $x \in V$, the proof is straightforward. We consider the case that $u = F(u_1,\cdots,u_m)$ if $F \in \mathcal{X} \setminus V$.

By $u = F(u_1,\cdots,u_m)$ and $\|u\| = \|v\|$, we have $v = F(v_1,\cdots,v_m)$ such that $\|u_i\| = \|v_i\|$ $(1 \leq i \leq m)$.

Then $u\gamma = (F\gamma)(u_1\gamma,\cdots,u_m\gamma)$ and $v\gamma = (F\gamma)(v_1\gamma,\cdots,v_m\gamma)$. Letting $u' = u\gamma$ and $v' = v\gamma$, we show $\|u'(u_1\gamma,\cdots,u_m\gamma)\downarrow\| = \|v'(v_1\gamma,\cdots,v_m\gamma)\downarrow\|$ by induction on structure of $u'$.

If $u' = \lambda x_1\cdots x_m.x_i$ then $\|u_i\gamma\downarrow\| = \|v_i\gamma\downarrow\|$ by induction hypothesis. Let $u' = \lambda x_1\cdots x_m.f(t_1,\cdots,t_n)$ $(n \geqq 0$ and $f \in \mathcal{F})$. Let $\theta = \{x_1 \leftarrow u_1\gamma,\cdots,x_m \leftarrow u_m\gamma\}$.

$\|u'(u_1\gamma,\cdots,u_m\gamma)\downarrow\|$
$= f(\|t_1\theta\downarrow\|,\cdots,\|t_n\theta\downarrow\|)$
$= f(\ \|(\lambda x_1 \cdots x_m.t_1)\ (u_1\gamma, \cdots, u_m\gamma)\downarrow\|, \cdots, \|(\lambda x_1 \cdots x_m.t_n)\ (u_1\gamma, \cdots, u_m\gamma)\downarrow\|)$
$= f(\ \|(\lambda x_1 \cdots x_m.t_1)\ (v_1\gamma, \cdots, v_m\gamma)\downarrow\|, \cdots, \|(\lambda x_1 \cdots x_m.t_n)\ (v_1\gamma, \cdots, v_m\gamma)\downarrow\|)$ by induction hypothesis.
□

**finition 26:** Let $s$ be a algebraic term and $\gamma$ be a ›und normalized substitution. $s\gamma\downarrow$ denotes $\|t\gamma\downarrow\|$ some normalized term $t$ such that $\|t\| = s$. t $\{s_1,\cdots,s_n\}$ be a (multi)set of algebraic terms. $,\cdots,s_n\}\gamma\downarrow$ denotes $\{s_1\gamma\downarrow,\cdots,s_n\gamma\downarrow\}$.

**mma 13:** Let $dec_w(\|s\|) \gg_{EL} dec_z(\|t\|)$ where $s$ and e normalized terms and $w \in Op(s)$, $z \in Op(t)$. Then any ground normalized substitution $\gamma$, the following ee claims hold.

) If $\|s\||_w = \|t\||_z$ and $top\ (\|t\||_z) \in \widetilde{\mathcal{X}}$ then $dec_{w.j}(\|s\gamma\downarrow\|) \gg_{EL} dec_{z.j}(\|t\gamma\downarrow\|)$, for any $j \in N^*$ such that $z.j \in Op\ (\|t\gamma\downarrow\|)$.

) If $\|s\||_w \neq \|t\||_z$ and $top\ (\|t\||_z) \in \widetilde{\mathcal{X}}$ then $dec_{w.i}(\|s\gamma\downarrow\|) \gg_{EL} dec_{z.j}(\|t\gamma\downarrow\|)$, for any $j, i \in N^*$ such that $z.j \in Op\ (\|t\gamma\downarrow\|)$ and $w.i \in Op\ (\|s\gamma\downarrow\|)$.

) If $top\ (\|t\||_z) \notin \widetilde{\mathcal{X}}$ then $dec_{w.i}(\|s\gamma\downarrow\|) \gg_{EL} dec_z(\|t\gamma\downarrow\|)$, for any $i \in N^*$ such that $w.i \in Op(\|s\gamma\downarrow\|)$.

**oof.** Let $s' = \|s\|$ and $t' = \|t\|$. We show that the im (1) $\wedge$ (2) $\wedge$ (3) by induction on $|s'| + |t'|$. Assume t $dec_w(s') \gg_{EL} dec_z(t')$.

) Consider the case $s'|_w = t'|_z$ and $top(t'|_z) \in \widetilde{\mathcal{X}}$.

By the assumption $dec_w(s') \gg_{EL} dec_z(t')$ and definition of multiset extension, consider the case that $dec_w(s') = M \cup \{s_1,\cdots,s_m\}$, $dec_z(t') = M \cup \{t_1,\cdots,t_n\}$, and for any $k \in \{1,\cdots,n\}$, there exists $l \in \{1,\cdots,m\}$ such that $dec_w(s') \ni s_l >_{EL} t_k \in dec_z(t')$.

For any $j \in N^*$ $(z.j \in Op(t'\gamma\downarrow))$, we can show that $dec_{w.j}(s'\gamma\downarrow) = M\gamma\downarrow \cup \{s_1\gamma\downarrow,\ \cdots,\ s_m\gamma\downarrow\} \cup L$, $dec_{z.j}(t'\gamma\downarrow) = M\gamma\downarrow \cup \{t_1\gamma\downarrow\ ,\cdots,\ t_n\gamma\downarrow\} \cup L$ where $L = \{v \mid v \in sub(dec_j(s'|_w\gamma\downarrow), s'|_w\gamma\downarrow)\}$ by lemma 25. Hence we have to show that $dec_w(s') \ni s_l >_{EL} t_k \in dec_z(t')$ implies $dec_{w.j}(s'\gamma\downarrow) \ni s_l\gamma\downarrow >_{EL} t_k\gamma\downarrow \in dec_{z.j}(t'\gamma\downarrow)$. We distinguish the cases with respect to the definition of $>_{EL}$.

(1 − 1) $type(s_l) >_T type(t_k)$.

Then, $type(s_l\gamma\downarrow) >_T type(t_k\gamma\downarrow)$.

(1 − 2) $type(s_l) = type(t_k)$.

(a) If $top(s_l) >_{\lambda\mathcal{F}} top(t_k)$ then $top(s_l\gamma\downarrow) >_{\lambda\mathcal{F}} top(t_k\gamma\downarrow)$ holds.

(b) If $top(s_l) = top(t_k)$, $ST(top(s_l)) = mult$ and $sub(dec_w(s'),s_l) \gg_{EL} sub(dec_z(t'),t_k)$ then we can show $sub(dec_{w.j}(s'\gamma\downarrow), s_l\gamma\downarrow) \gg_{EL} sub(dec_{z.j}(t'\gamma\downarrow), t_k\gamma\downarrow)$ by induction hypothesis.

(c) In the case that $top(s_l) = top(t_k)$, $ST(top(s_l)) = mult$, $sub(dec_w(s'),s_l) = sub(dec_z(t'),t_k)$ and $dec(args(s_l)) \ggg_{EL} dec(args(t_k))$, it follows that $dec(args(s_l\gamma\downarrow)) \ggg_{EL} dec(args(t_k\gamma\downarrow))$ from types and induction hypothesis.

(d) Consider the case that $top(s_l) = top(t_k)$, $ST(top(s_l)) \neq mult$, $args(s_l) >_{TIRDS,ST(top(s_l))} args(t_k)$ and $\{s_l\} \gg_{TIRDS} args\ (t_k)$. We can show $args(s_l\gamma\downarrow) >_{TIRDS,ST(top(s_l\gamma\downarrow))} args(t_k\gamma\downarrow)$ and $\{s_l\gamma\downarrow\} \gg_{TIRDS} args(t_k\gamma\downarrow)$ by types, in case of $type(s_l) >_T type(t_k)$ and by induction hypothesis, in case of $type(s_l) = type(t_k)$.

(2) In case of $s'|_w \neq t'|_z$ and $top(t'|_z) \in \widetilde{\mathcal{X}}$, we have to consider only the case that $dec_w(s') = M \cup \{s_1,\cdots,s_m\}$, $dec_z(t') = M \cup \{t_1,\cdots,t_n\}$, and for any $k \in \{1,\cdots,n\}$, there exists $l \in \{1,\cdots,m\}$ such that $dec_w(s') \ni s_l >_{EL} t_k \in dec_z(t')$ by the assumption $dec_w(s') \gg_{EL} dec_z(t')$ and definition of multiset extension. For any $j \in N^*$ $(z.j \in Op(t'\gamma\downarrow))$ and any $i \in N^*$ $(w.i \in Op(s'\gamma\downarrow))$, we can show that $dec_{w.i}(s'\gamma\downarrow) = M\gamma\downarrow \cup \{s_1\gamma\downarrow,\cdots, s_m\gamma\downarrow\} \cup L$ where $L = \{v \mid v \in sub\ (dec_i(s'|_w\gamma\downarrow), s'|_w\gamma\downarrow)\}$, $dec_{z.j}(t'\gamma\downarrow) = M\gamma\downarrow \cup \{t_1\gamma\downarrow,\cdots, t_n\gamma\downarrow\} \cup L'$ where $L' = \{v' \mid v' \in sub(dec_j(t'|_z\gamma\downarrow), t'|_z\gamma\downarrow)\}$ by lemma 25. Since $t'|_z \notin dec_w(s')$, $t'|_z \in \{t_1, \cdots, t_n\}$. That is, $dec_w(s') \ni s_l >_{EL} t'|_z \in dec_z(t')$. $type(s_l) >_T type(t'|_z)$ holds, by $top(t'|_z) \in \widetilde{\mathcal{X}}$. Since $t'|_z\gamma\downarrow \rhd y'$ for any $y' \in L'$ and compatibility of $>_T$, $type(t'|_z\gamma\downarrow) \geqq_T type(y')$. Hence $type(s_l\gamma\downarrow) >_T type(y')$, i.e., $dec_{w.i}(s'\gamma\downarrow) \ni s_l\gamma\downarrow >_{EL} y' \in dec_{z.j}(t'\gamma\downarrow)$, for any $y' \in sub(dec_j(t'|_z\gamma\downarrow), t'|_z\gamma\downarrow)$. Further we can show that $dec_w(s') \ni s_l >_{EL} t_k \in dec_z(t')$ implies $dec_{w.i}(s'\gamma\downarrow) \ni s_l\gamma\downarrow >_{EL} t_k\gamma\downarrow \in dec_{z.j}(t'\gamma\downarrow)$, in similar to the proof of (1).

(3) In case of $top(t'|_z) \notin \widetilde{\mathcal{X}}$, for any $i \in N^*$ $(w.i \in Op(t'\gamma\downarrow))$, we can show that $dec_{w.i}(s'\gamma\downarrow) = M\gamma\downarrow \cup \{s_1\gamma\downarrow,\cdots, s_m\gamma\downarrow\} \cup L$ where $L = \{v \mid v \in sub(dec_i(s'|_w\gamma\downarrow), s'|_w\gamma\downarrow)\}$, $dec_z(t'\gamma\downarrow) = M\gamma\downarrow \cup \{t_1\gamma\downarrow, \cdots, t_n\gamma\downarrow\}$ by lemma 25. Hence we can show that $dec_w(s') \ni s_l >_{EL} t_k \in dec_z(t')$ implies $dec_{w.i}(s'\gamma\downarrow) \ni s_l\gamma\downarrow >_{EL} t_k\gamma\downarrow \in dec_z(t'\gamma\downarrow)$, in similar to the proof of (1). □

## Appendix B: Proof of Theorem 22

**Theorem 22:** Let $s$ and $t$ be normalized terms. Then $s >_{HRPOS} t$ implies $s >_{HIRDS} t$.

**Proof.** This proof is obtained by modification of the proof of the following claim. $u >_{RPOS} v$ implies $u >_{IRDS} v$, for any term $u$ and $v$[14],[15]. It holds that $u >_{HRPOS} v \Longleftrightarrow \|u\| >_{TRPOS} \|v\|$ by definition of the HRPOS, higher-order recursive path ordering[8] and $u >_{HIRDS} v \Longleftrightarrow \|u\| >_{TIRDS} \|v\|$ by definition 10. We have to show that $s >_{TRPOS} t$ implies $s >_{TIRDS} t$ for any algebraic terms $s$ and $t$. The proof is performed by using induction on $|s| + |t|$. Let $s = F_\sigma(t_1,\cdots,t_m)$, $t = G_\tau(s_1,\cdots,s_n)$. We distinguish the cases with respect to the definition of TRPOS, *typed recursive path ordering*[8].

(1) $\sigma >_T \tau$. For any $w \in Op(s)$ and any $z \in Op(t)$,

$dec_w(s) \ni s >_{EL} t \in dec_z(t)$. For any $t_j \in dec_z(t)$, $type(t) \geqq_T type(t_j)$ from compatibility of type ordering. Thus, $dec_w(s) \ni s >_{EL} t_j \in dec_z(t)$ holds by type ordering. Since for any $z \in Op(t)$, there exist $w \in Op(s)$ such that $dec_w(s) \gg_{EL} dec_z(t)$, we have $dec(\{s\}) \ggg_{EL} dec(\{t\})$.
(2) $\sigma = \tau$.

(a) $F_\sigma \notin \mathcal{X}_{T_\sigma}$ and $s_i \geqq_{TRPOS} t$ for some $i$.

By induction hypothesis, $s_i \geqq_{TIRDS} t$. Hence $dec(\{s_i\}) \underline{\ggg}_{EL} dec(\{t\})$ holds. Then $dec(\{s\}) \ggg_{EL} dec(\{t\})$ because for any $dec_w(s_i) \in dec(\{s_i\})$, there exists $dec_{i.w}(s) \in dec(\{s\})$ such that $dec_{i.w}(s) \gg_{EL} dec_w(s_i)$.

(b) $F_\sigma >_{\lambda\mathcal{F}} G_\tau$ and $s >_{TRPOS} t_i$ for all $i$.

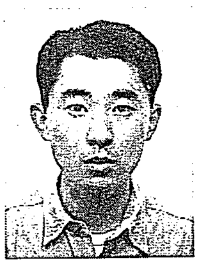
For all $i \in \{1, \cdots, n\}$, $s >_{TIRDS} t_i$ holds by induction hypothesis. Thus, $dec(\{s\}) \ggg_{EL} dec(\{t_i\})$ $(i = 1,\cdots,n)$ holds by definition 6. Note that $dec(\{t\}) = \{d \cup \{t\} \mid d \in dec(\{t_i\}), i \in \{1,\cdots,n\}\}$. For any $z \in Op(t_i)$, there exists $w \in Op(s)$ such that $dec_w(s) \gg_{EL} dec_z(t_i)$ $(i = 1,\cdots,n)$. $dec_w(s) \ni s >_{EL} t \in dec_{i.z}(t)$ since $top(s) = F_\sigma >_{\lambda\mathcal{F}} G_\tau = top(t)$. Since there exists no subterm of $t$ that is equal to $s$ (otherwise $s$ would not be greater than all $t_i$'s), for any $z' \in Op(t)$, there exists $w' \in Op(s)$ such that $dec_{w'}(s) \gg_{EL} dec_{z'}(t)$. Hence $dec(\{s\}) \ggg_{EL} dec(\{t\})$.

(c) $F_\sigma = G_\tau$, $ST(F_\sigma) = mult$ and $args(s) \gg_{TRPOS} args(t)$.

It is holds that $args(s) \gg_{TIRDS} args(t)$ by induction hypothesis. For any $t_j \in args(t)$, there exists $s_i \in args(s)$ such that $s_i \geqq_{TIRDS} t_j$ and $args(s) \neq args(t)$ by definition of multiset extension. Then for any $t_j \in args(t)$, there exists $s_i \in args(s)$ such that $dec(\{s_i\}) \underline{\ggg}_{EL} dec(\{t_j\}) \quad \cdots (*)$. Note that $dec(\{s\}) = \{d \cup \{s\} \mid d \in dec(\{s_i\}),\ i \in \{1,\cdots,m\}\}$ and $dec(\{t\}) = \{d \cup \{t\} \mid d \in dec(\{t_j\}),\ j \in \{1,\cdots,n\}\}$. We have to show that for any $z \in Op(t)$, there exists $w \in Op(s)$ such that $dec_w(s) \ni s >_{EL} t \in dec_z(t)$. Then we have to prove that either $sub(dec_w(s), s) \gg_{EL} sub(dec_z(t), t)$ or $sub(dec_w(s), s) = sub(dec_z(t), t)$ and $dec(args(s)) \ggg_{EL} dec(args(t))$. This can easily be shown with $(*)$.

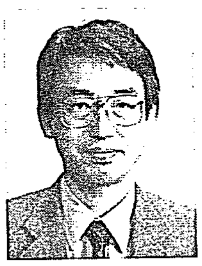
(d) $F_\sigma = G_\tau$, $ST(F_\sigma) \neq mult$, $args(s) >_{TRPOS, ST(F_\sigma)} args(t)$ and $\{s\} \gg_{TRPOS} args(t)$.

Then $args(s) >_{TIRDS, ST(F_\sigma)} args(t)$ and $\{s\} \gg_{TIRDS} args(t)$ by induction hypothesis. □



**Munehiro Iwami** was born on September 22 1970. He received the B.S. in mathematics from Tokai University in 1994. He received the M.S. in information science from JAIST, Hokuriku in 1996. He is now a doctoral student of the School of Information Science in JAIST, Hokuriku. He is interested in term rewriting systems. He is a member of JSSST.

**Masahiko Sakai** was born on June 10 1961. He completed graduate course of Nagoya University in 1989 and became Assistant Professor, where he obtained a D.E. degree in 1992. From April 1993 to March 1997, he was Associate Professor in JAIST, Hokuriku. In 1996 he stayed at SUNY at Stony Brook for six months as Research Professor. Since April 1997, he has been Associate Professor in Nagoya University. He is interested in term rewriting system, verification of specification and software generation. He was received the Best Paper Award from IEICE in 1992. He is member of IPSJ.



**Yoshihito Toyama** received B.E. from Niigata University in 1975, M.E. and D.E. from Tohoku University in 1977 and 1990. From 1977 to 1993 he worked at NTT Laboratories. Since 1993, he is a a professor of School of information Science, Japan Advanced Institute of Science and Technology (JAIST). His research interests include term rewriting systems, program theory, and automated theorem proving. He is a member of IPSJ, JSSST, ACM, EATCS.