Termination of S-Expression Rewriting Systems: Lexicographic Path Ordering for Higher-Order Terms*

Yoshihito Toyama

RIEC, Tohoku University
Katahira 2-1-1, Aoba-ku, Sendai 980-8577, Japan
email : toyama@nue.riec.tohoku.ac.jp

Abstract. This paper expands the termination proof techniques based on the lexicographic path ordering to term rewriting systems over varyadic terms, in which each function symbol may have more than one arity. By removing the deletion property from the usual notion of the embedding relation, we adapt Kruskal's tree theorem to the lexicographic comparison over varyadic terms. The result presented is that finite term rewriting systems over varyadic terms are terminating whenever they are compatible with the lexicographic path order. The ordering is simple, but powerful enough to handle most of higher-order rewriting systems without λ-abstraction, expressed as S-expression rewriting systems.

1 Introduction

A term rewriting system [2, 15] is said to be terminating if all reduction sequences are finite. An important syntactical method to prove termination of a first-order term rewriting system is the one using the lexicographic path order (or the recursive path order with lexicographic status) relying on Kruskal's tree theorem [2, 4, 5, 7, 13, 15].

A higher-order rewriting system is a rewriting system to accommodate higher-order functions. Several syntactical methods for proving termination of them are presented [1, 6, 10, 11, 12, 14]. In the framework of the algebraic-functional systems, Jouannaud and Rubio [6] generalize the recursive path order (with lexicographic status) to higher-order rewrite rules, by adapting the notion of computability of typed λ calculus due to Tait and Girald. Concerning the termination method relying on Kruskal's tree theorem [9], Linfantsev and Bachmair [11] present the lexicographic path ordering for higher-order rewriting systems without λ-abstraction, in which higher-order terms are expressed by application and pairing. Aoto and Yamada [1] present a transformation method for applying the lexicographic path ordering to simply typed term rewriting systems. These syntactical methods have to use the types to guarantee well-foundedness

* This work was partially supported by grants from Japan Society for the Promotion of Science, No. 14580357, and from Ministry of Education, Culture, Sports, Science and Technology, No. 15017203.

of the presented orderings [1, 6, 11]. From this observation, a natural question arises: whether the syntactical methods relying entirely on Kruskal's tree theorem, without using the types, are really useful for proving termination of higher-order rewriting systems.

The main purpose of this paper is to give a positive answer to the above question; the lexicographic path ordering relying only on Kruskal's tree theorem is powerful enough to handle most of higher-order rewriting systems in the literature [1, 6, 10, 11, 12, 14]. The crucial point is the notion of simplification order [2, 4, 13, 15] over varyadic signatures, in which each function symbol may have more than one arity [4, 5]. Over varyadic signatures, the usual definition of simplification order requires the deletion property [4, 5], since it is tightly related to the use of Kruskal's tree theorem [9]. However, the lexicographic comparison is not compatible with this property. Hence, in the literature [2, 7, 15] the lexicographic path order (or the lexicographic comparison of the recursive path order with status) is restricted to fixed-arity (or bounded-varyadic [5]) signatures. In this paper, by removing the deletion property from the usual definition, we adapt the notion of simplification order to the lexicographic path order over varyadic signature.

The key result presented here is that finite term rewriting systems over varyadic signatures are terminating whenever they are compatible with the lexicographic path order. The result is simple but by no means trivial because we do not restrict to fixed-arity (or bounded-varyadic) signatures. To develop the termination method based on this result, we propose a new framework of the S-expressions built from constants and a varyadic function symbol of infinitely many arities, in which higher-order rewriting systems are expressed as S-expression rewriting systems. Termination of various examples is shown by the lexicographic path order on S-expressions, without using types. If we use the type information of higher-order rewrite rules, the power of our termination method can be strengthened more. For demonstrating this, we propose a non-termination preserving transformation of higher-order rewriting systems based on *currying* technique.

The remainder of this paper is organized as follows. After a preliminary section, in Section 3 we discuss the simplification ordering over varyadic signatures. In Section 4 the lexicographic path ordering over varyadic signatures is studied. In sections 5 and 6 we introduce S-expression rewriting systems and discuss termination of them. Section 7 shows that our ordering over S-expressions is a conservative extension of the usual ordering. Section 8 strengthens our ordering through transformational method based on the type information.

2 Preliminaries

We mainly follow the notation of [2, 15]. A signature is a set $\mathcal{F}$ of *function symbols* denoted by $f, g, h, \ldots$. Function symbols in $\mathcal{F}$ may be *varyadic*, i.e., have more than one arity. If at least one function symbol $f \in \mathcal{F}$ is varyadic, we say that $\mathcal{F}$ is varyadic; otherwise, $\mathcal{F}$ is fixed-arity. Elements of $\mathcal{F}$ of arity 0 are called constants. Let $\mathcal{V}$ be a countably infinite set of *variables* denoted by $x, y, z, \ldots, \alpha, \beta, \gamma, \ldots$ where $\mathcal{F} \cap \mathcal{V} = \phi$. The set of all *terms* built from $\mathcal{F}$ and $\mathcal{V}$ is denoted by $\mathcal{T}(\mathcal{F}, \mathcal{V})$, and terms are usually denoted by $s, t, r, \ldots$. The set of function symbols in a term t is denoted by $\mathcal{F}(t)$, and the set of variables is denoted by $\mathcal{V}(t)$.

A *substitution* σ is a mapping from $\mathcal{V}$ into $\mathcal{T}(\mathcal{F}, \mathcal{V})$. Substitutions are extended into homomorphisms from $\mathcal{T}(\mathcal{F}, \mathcal{V})$ into $\mathcal{T}(\mathcal{F}, \mathcal{V})$. Following common usage, we write $t\theta$ instead of $\theta(t)$.

Consider an extra constant $\square$ called a hole. Then $C \in T(\mathcal{F} \cup \{\square\}, \mathcal{V})$ is called a context on $\mathcal{F}$. We use the notation $C[\,]$ for the context containing precisely one hole, and $C[t]$ denotes the result of placing a term t in the hole of $C[\,]$. A term s is called a subterm of $t = C[s]$. We denote $s \subseteq t$ if s is a subterm occurrence of t, and $s \subset t$ if $s \subseteq t$ and $s \neq t$.

A rewrite rule is a pair $\langle l, r\rangle$ of terms such that $l \notin \mathcal{V}$ and $\mathcal{V}(r) \subseteq \mathcal{V}(l)$. We write $l \to r$ for $\langle l, r\rangle$. A *term rewriting system* (TRS for short) $\mathcal{R}$ is a set of rewrite rules. The rewrite rules of a TRS $\mathcal{R}$ define a reduction relation $\to_{\mathcal{R}}$ on $\mathcal{T}(\mathcal{F}, \mathcal{V})$ as follow: $t \to_{\mathcal{R}} s$ iff there exist a rewrite rule $l \to r \in \mathcal{R}$, a context $C[\,]$ and a substitution θ such that $t = C[l\theta]$ and $s = C[r\theta]$. The transitive closure of $\to_{\mathcal{R}}$ is denoted by $\xrightarrow{+}_{\mathcal{R}}$. A TRS $\mathcal{R}$ is *terminating* if there exists no infinite reduction sequence $t_0 \to_{\mathcal{R}} t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} \cdots$.

A (strict) partial order $\succ$ is a transitive and irreflexive relation. The reflexive closure of $\succ$ is denoted by $\succeq$. A partial order $\succ$ is well-founded if there is no infinite decreasing sequence $t_1 \succ t_2 \succ t_3 \succ \cdots$. A partial order $\succ$ is a *partial-well-order* iff for every infinite sequence $t_1, t_2, t_3, \cdots$ there exist indices $i < j$ such that $t_i \preceq t_j$.

A partial order $\succ$ on $\mathcal{T}(\mathcal{F}, \mathcal{V})$ is called a rewrite order if it is closed under context and substitution. A TRS $\mathcal{R}$ is compatible with a partial order $\succ$ if $l \succ r$ for every rule $l \to r \in \mathcal{R}$.

3 Simplification Order over Varyadic Signature

We now define the simplification ordering over varyadic signatures. Note that our definition lacks the deletion property, though it is necessary to relate the ordering with Kruskal's tree theorem in a varyadic setting [4, 5, 13].

Let $\mathcal{F}$ be a varyadic or fixed-arity signature. We say that a partial order $\succ$ on $\mathcal{T}(\mathcal{F}, \mathcal{V})$ has the subterm property if $s \succ t$ whenever $t \subset s$, and $\succ$ has the deletion property if $f(\cdots t \cdots) \succ f(\cdots\cdots)$ for all terms t whenever the arities of f allow it.

The TRS $\mathcal{R}_{emb}$ consists of all rewrite rules $f(x_1, \cdots, x_m) \to x_i$ with $f \in \mathcal{F}$ when the arities of f allow it. Here $x_1, \cdots, x_m$ are pairwise different variables. The embedding relation $\rhd_{emb}$ is defined by $\rhd_{emb} = \xrightarrow{+}_{\mathcal{R}_{emb}}$. The embedding relation $\rhd_{emb}$ is a partial order which has the subterm property.

Proposition 1 (Kruskal's Tree Theorem (Finite Version)). Let $\mathcal{F}$ be a fixed-arity signature and $\mathcal{F} \cup \mathcal{V}$ be finite. Then the embedding relation $\rhd_{emb}$ is a partial-well-order on $\mathcal{T}(\mathcal{F}, \mathcal{V})$ [2, 13, 15].

It should be noted that in the literature [4, 5] the embedding relation requires not only the subterm property but also the deletion property, i.e., $f(\cdots, t, \cdots) \rhd_{emb} f(\cdots\cdots)$, when function symbols are varyadic. This requirement is necessary for Kruskal's tree theorem. To see why, consider the infinite sequence $f(c), f(c,c), f(c,c,c), \cdots$ for a varyadic function symbol f and a constant c. This sequence shows that Kruskal's tree theorem fails over varyadic signatures if the embedding relation $\rhd_{emb}$ does not have the deletion property. However, the deletion property is incompatible with the lexicographic path ordering $\succ_{lpo}$ (which is just the subject studied through this paper); for instance, $f(c, b, b, b) \succ_{delete} f(b, b, b)$ but $f(c, b, b, b) \prec_{lpo} f(b, b, b)$ for $c < b$. Thus, we do not suppose the deletion property of $\rhd_{emb}$, but in order for $\rhd_{emb}$ to be a partial-well-order over a varyadic signature, we impose alternative restriction, namely that varyadic function symbols occurring in an infinite sequence of terms have their arities bounded by a natural number.

The maximum degree $D(t)$ of a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ is defined by: (i) $D(x) = 0$ for a variable x, (ii) $D(c) = 0$ for a constant c, (iii) $D(f(t_1, \cdots, t_n)) = max\{n, D(t_1), \cdots, D(t_n)\}$ for $n > 0$. The set of terms having the maximum degree not more than m is defined by $\mathcal{T}_m(\mathcal{F}, \mathcal{V}) = \{t \mid D(t) \leq m\}$. Then, we have the following tree theorem over varyadic signatures as a corollary of Kruskal's tree theorem, even if the embedding relation $\rhd_{emb}$ does not have the deletion property.

Lemma 2 (Tree Theorem). Let $\mathcal{F}$ be a varyadic or fixed-arity signature and $\mathcal{F} \cup \mathcal{V}$ be finite. Then the embedding relation $\rhd_{emb}$ is a partial-well-order on $\mathcal{T}_m(\mathcal{F}, \mathcal{V})$ for any natural number m.

Proof. We code varyadic function symbols f into fixed-arity function symbols f^k by labeling f with its arity k as in Ferreira [5]. Since all function symbols occurring in terms over $\mathcal{T}_m(\mathcal{F}, \mathcal{V})$ have their arities bounded by m, we need to consider only finite coded function symbols f^k $(k \leq m)$ even if f has infinitely many arities. In this new setting, Kruskal's tree theorem remains valid for the embedding relation $\rhd_{emb}$ without the deletion property since the new signature is finite and fixed-arity. □

Definition 3. An order $\succ$ on $\mathcal{T}(\mathcal{F}, \mathcal{V})$ is called a *simplification order* if it is a rewrite order with the subterm property [2, 13, 15].

Lemma 4. A simplification order $\succ$ on $\mathcal{T}(\mathcal{F}, \mathcal{V})$ satisfies $\rhd_{emb} \subseteq \succ$ [2, 13, 15].

Proof. Trivial from the definitions of simplification order and of $\rhd_{emb}$. □

Definition 5. Let $\mathcal{F}$ be a varyadic or fixed-arity signature. A TRS $\mathcal{R}$ over $\mathcal{T}(\mathcal{F}, \mathcal{V})$ is *bounded* if for any infinite reduction sequence $t_0 \rightarrow_{\mathcal{R}} t_1 \rightarrow_{\mathcal{R}} t_2 \rightarrow_{\mathcal{R}} t_3 \rightarrow_{\mathcal{R}} \cdots$ there exist some natural number m and a finite set $\tilde{\mathcal{F}} \subseteq \mathcal{F}$ such that $D(t_i) \leq m$ and $\mathcal{F}(t_i) \subseteq \tilde{\mathcal{F}}$ for all i.

Lemma 6. Let $\mathcal{F}$ be a varyadic or fixed-arity signature and $\mathcal{R}$ be a finite TRS over $\mathcal{T}(\mathcal{F}, \mathcal{V})$. Then $\mathcal{R}$ is bounded.

Proof. Let $t_0 \to_{\mathcal{R}} t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} t_3 \to_{\mathcal{R}} \cdots$ be an infinite reduction sequence. Taking $\mathcal{F}(t_0) \cup \bigcup_{l \to r \in \mathcal{R}} \mathcal{F}(r)$ as a finite set $\tilde{\mathcal{F}}$, we have $\mathcal{F}(t_i) \subseteq \tilde{\mathcal{F}}$ for all i. Let $p = max\{D(r) \mid l \to r \in \mathcal{R}\}$. Consider a reduction $s \to_{\mathcal{R}} t$ where $s = C[l\theta]$, $t = C[r\theta]$ and $l \to r \in \mathcal{R}$. Then we have

$$
\begin{aligned}
&D(t) = D(C[r\theta]) = \\
&max\{D(C[\,]), D(r), D(x\theta) \mid x \in \mathcal{V}(r)\} \leq \\
&max\{D(C[\,]), D(l), D(r), D(x\theta) \mid x \in \mathcal{V}(r)\} \leq \\
&max\{D(C[l\theta]), D(r)\} = max\{D(s), D(r)\} \leq max\{D(s), p\}.
\end{aligned}
$$

Thus it holds that $D(t_{i+1}) \leq max\{D(t_i), p\}$ $(i = 0, 1, 2, \cdots)$. By taking $m = max\{D(t_0), p\}$ we conclude the claim. □

A TRS $\mathcal{R}$ is *right-bounded* if the sets $\bigcup_{l \to r \in \mathcal{R}} \mathcal{F}(r)$ and $\{D(r) \mid l \to r \in \mathcal{R}\}$ are finite. Note that $\mathcal{R}$ is right-bounded whenever $\mathcal{R}$ is finite. We can weaken a finite TRS to a right-bounded (infinite) TRS in Lemma 6 as follows.

Lemma 7. Let $\mathcal{F}$ be a varyadic or fixed-arity signature and $\mathcal{R}$ be a right-bounded TRS over $\mathcal{T}(\mathcal{F}, \mathcal{V})$. Then $\mathcal{R}$ is bounded.

Theorem 8. Let $\mathcal{F}$ be a varyadic or fixed-arity signature and $\mathcal{R}$ be a bounded TRS over $\mathcal{T}(\mathcal{F}, \mathcal{V})$ compatible with a simplification order. Then $\mathcal{R}$ is terminating.

Proof. Let $\mathcal{R}$ be compatible with a simplification order $\succ$. For a proof by contradiction, suppose an infinite reduction sequence $t_0 \to_{\mathcal{R}} t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} t_3 \to_{\mathcal{R}} \cdots$. Then we have $\mathcal{V}(t_i) \subseteq \mathcal{V}(t_0)$ for all i. Since $\mathcal{R}$ is bounded, $t_0, t_1, t_2, \cdots \in \mathcal{T}_m(\tilde{\mathcal{F}}, \mathcal{V}(t_0))$ for some natural number m and a finite set $\tilde{\mathcal{F}}$. According to Lemma 2 (Tree Theorem), there exit $i < j$ such that $t_i \trianglelefteq_{emb} t_j$. From Lemma 4, we have $t_i \preceq t_j$; contradiction to $t_i \succ t_j$. Hence we conclude the claim. □

Corollary 9. Let $\mathcal{F}$ be a varyadic or fixed-arity signature and $\mathcal{R}$ be a finite (or right-bounded) TRS over $\mathcal{T}(\mathcal{F}, \mathcal{V})$ compatible with a simplification order. Then $\mathcal{R}$ is terminating.

4 Lexicographic Path Order over Varyadic Signature

The lexicographic path order over a fixed-arity signature was first described in Kamin and Lévy [7]. The following definition gives the lexicographic path order over varyadic signature.

Definition 10 (lpo for varyadic signature). Let $\mathcal{F}$ be a varyadic or fixed-arity signature and $>$ a precedence (i.e., a partial order) on $\mathcal{F}$. The lexicographic path order $\succ_{lpo}$ on $\mathcal{T}(\mathcal{F}, \mathcal{V})$ is recursively defined as follows: $s \succ_{lpo} t$ iff $s = f(s_1, \cdots, s_m)$, $t = g(t_1, \cdots, t_n)$ or $t \in \mathcal{V}$, and

(L1) $\exists i.\ s_i \succeq_{lpo} t$, or
(L2) $f > g$ and $\forall i.\ s \succ_{lpo} t_i$, or
(L3) $f = g$, $m > n$, and $s_1 = t_1, \cdots, s_n = t_n$, or
(L4) $f = g$ and $\exists i.\ s_1 = t_1, \cdots, s_{i-1} = t_{i-1}$, $s_i \succ_{lpo} t_i$, $s \succ_{lpo} t_{i+1}, \cdots, s \succ_{lpo} t_n$.

Lemma 11. The lexicographic path order $\succ_{lpo}$ is a simplification order.

Proof. It can be shown by structural induction. $\square$

Note that the simplification order is not sufficient to guarantee well-foundedness of $\succ_{lpo}$; for example, we can have the infinite descending sequence $f(b) \succ_{lpo} f(c, b) \succ_{lpo} f(c, c, b) \succ_{lpo} f(c, c, c, b) \succ_{lpo} \cdots$ if f is varyadic and $b > c$. The problem arises from the fact that lexicographic sequences of unbounded size are not well-founded. Kamin and Lévy [7] proved that $\succ_{lpo}$ is well-founded when all function symbols are fixed-arity. Ferreira [5] weakened the fixed-arity restriction to the bounded varyadic restriction, i.e, the arity of every varyadic function is bounded by a natural number. In the following theorem we show that these restrictions can be weakened more.

Theorem 12. Let $\mathcal{F}$ be a varyadic or fixed-arity signature and $\mathcal{R}$ be a bounded TRS over $\mathcal{T}(\mathcal{F}, \mathcal{V})$ compatible with the lexicographic path order $\succ_{lpo}$. Then $\mathcal{R}$ is terminating.

Proof. From Theorem 8 and Lemma 11 it follows. $\square$

Corollary 13. Let $\mathcal{F}$ be a varyadic or fixed-arity signature and $\mathcal{R}$ be a finite (or right-bounded) TRS over $\mathcal{T}(\mathcal{F}, \mathcal{V})$ compatible with the lexicographic path order $\succ_{lpo}$. Then $\mathcal{R}$ is terminating.

5 S-Expression Rewriting Systems

An S-expression, which is short for *symbolic expression*, is used for representing an expression or data in Lisp-like programming languages. As an S-expression uses the prefix notation, the term $f(a, g(b, c), d, e)$ is written in the S-expression $(f\ a\ (g\ b\ c)\ d\ e)$. A variable α at the prefix, like $(\alpha\ a\ (g\ b\ c)\ d\ e)$, works as a higher-order variable. This feature allows us to present higher-order rewrite rules as simple first-order rewrite rules. For example, the higher-order function *map* is presented by the following rewrite rules over S-expressions:

$$R \begin{cases} (map\ \alpha\ nil) \to nil \\ (map\ \alpha\ (cons\ x\ y)) \to (cons\ (\alpha\ x)\ (map\ \alpha\ y)) \end{cases}$$

where *map*, *cons*, *nil* are constants and α, x, y variables.

The most common method to treat higher-order rewriting without λ-*notation* is to use applicative terms, like $ap(ap(ap(f, a), b), c)$ for the term $f(a, b, c)$ with the application symbol ap [8]. The notion of applicative term can give a simple way for presenting higher-order rewrite rules just like S-expressions, but the left-associated sequence of ap is not convenient for proving termination of rewriting systems by simplification orderings. Consider two terms $g(b)$ and $f(g(a), b)$ where $g > f > b > a$. Then $g(b) \succ_{lpo} f(g(a), b)$ by the lexicographic path order $\succ_{lpo}$, but we have the reverse direction between the corresponding applicative terms, i.e., $ap(g, b) \prec_{lpo} ap(ap(f, ap(g, a)), b)$, because of $ap(g, b) \lhd_{emb} ap(ap(f, ap(g, a)), b)$.

The same problem arises from the usual syntax of S-expressions, by which $(f\ a\ b\ c)$ is expressed as $cons(f, cons(a, cons(b, cons(c, nil))))$ based on the constructor symbols $cons$ and nil. Thus apart from this syntax, we introduce a new syntax of S-expression based on a varyadic function symbol having infinitely many arities, on which the lexicographic path order works well like usual simplification orderings over first-order fixed-arity terms.

Definition 14 (S-Expression). Let $\mathcal{C}$ be a set of constants, $\circ \notin \mathcal{C}$ a varyadic function symbol with any natural numbers as its arities, and $\mathcal{V}$ a set of variables. Then the set of *S-expressions* built from $\mathcal{C}$ and $\mathcal{V}$, denoted by $\mathcal{S}(\mathcal{C}, \mathcal{V})$, is recursively defined as follows:

(i) $\mathcal{C} \cup \mathcal{V} \subseteq \mathcal{S}(\mathcal{C}, \mathcal{V})$.
(ii) $\circ(s_1, \cdots, s_n) \in \mathcal{S}(\mathcal{C}, \mathcal{V})$ if $s_1, \cdots, s_n \in \mathcal{S}(\mathcal{C}, \mathcal{V})$ $(n \geq 0)$.

From the definition it is trivial that $\mathcal{S}(\mathcal{C}, \mathcal{V}) = \mathcal{T}(\mathcal{C} \cup \{\circ\}, \mathcal{V})$. S-expressions $\circ(s_1, s_2, \cdots, s_n)$ are usually denoted by $(s_1\ s_2\ \cdots\ s_n)$, for short. For example, the S-expression $\circ(f, a, \circ(g, x), b, \circ())$ is written as $(f\ a\ (g\ x)\ b\ ())$.

The lexicographic path order on S-expressions is defined as follows.

Definition 15 (lps on S-expressions). Let $\mathcal{C}$ be a set of constants and $>$ a precedence (i.e., a partial order) on $\mathcal{C}$. The lexicographic path order $\succ_{lps}$ on $\mathcal{S}(\mathcal{C}, \mathcal{V})$ is recursively defined as follows: $s \succ_{lps} t$ iff $s = (s_1\ \cdots\ s_m)$ or $s \in \mathcal{C}$, $t = (t_1\ \cdots\ t_n)$ or $t \in \mathcal{C} \cup \mathcal{V}$, and

(S0) $s, t \in \mathcal{C}$ and $s > t$, or
(S1) $s \in \mathcal{C}$ and $\forall i.\ s \succ_{lps} t_i$, or
(S2) $\exists i.\ s_i \succeq_{lps} t$, or
(S3) $m > n$ and $s_1 = t_1, \cdots, s_n = t_n$, or
(S4) $\exists i.\ s_1 = t_1, \cdots, s_{i-1} = t_{i-1},\ s_i \succ_{lps} t_i,\ s \succ_{lps} t_{i+1}, \cdots, s \succ_{lps} t_n$.

Lemma 16. The lexicographic path order $\succ_{lps}$ on $\mathcal{S}(\mathcal{C}, \mathcal{V})$ is a simplification order.

Proof. We extend the precedence $>$ on $\mathcal{C}$ to on $\mathcal{C} \cup \{\circ\}$ by $a > \circ$ for all $a \in \mathcal{C}$. This extension gives the lexicographic path order $\succ_{lpo}$ on $\mathcal{T}(\mathcal{C} \cup \{\circ\}, \mathcal{V})$, i.e., $\succ_{lpo}$ on $\mathcal{S}(\mathcal{C}, \mathcal{V})$ since $\mathcal{S}(\mathcal{C}, \mathcal{V}) = \mathcal{T}(\mathcal{C} \cup \{\circ\}, \mathcal{V})$. It is proven that $\succ_{lps}$ and $\succ_{lpo}$ coincide. Thus from Lemma 11 the claim follows. □

An *S-expression rewriting system* (SRS for short) $\mathcal{R}$ is a term rewriting system on $\mathcal{S}(\mathcal{C}, \mathcal{V})$, i.e., the rewrite rules $l \to r \in \mathcal{R}$ consist of S-expressions $l, r \in \mathcal{S}(\mathcal{C}, \mathcal{V})$ and they define a reduction relation $\to_{\mathcal{R}}$ on $\mathcal{S}(\mathcal{C}, \mathcal{V})$.

Theorem 17. Let $\mathcal{R}$ be a bounded SRS over $\mathcal{S}(\mathcal{C}, \mathcal{V})$ compatible with the lexicographic path order $\succ_{lps}$. Then $\mathcal{R}$ is terminating.

Proof. From Theorem 8 and Lemma 16 it follows. □

Corollary 18. Let $\mathcal{R}$ be a finite (or right-bounded) SRS over $\mathcal{S}(\mathcal{C}, \mathcal{V})$ compatible with the lexicographic path order $\succ_{lps}$. Then $\mathcal{R}$ is terminating.

6 Termination of SRS

We verify termination of S-expression rewriting systems by the lexicographic path order $\succ_{lps}$. Various higher-order rewriting systems in the literature [1, 6, 10, 11, 12, 14] are naturally expressed in SRSs and termination of them is easily proven as that of SRSs without using the notion of type.

In the following examples, we denote variables playing like higher-order variables by $\alpha, \beta, \gamma, \cdots$ in distinction from $x, y, z, \cdots$ for readability.

Example 1 (map). Let $\mathcal{C} = \{map, cons, nil\}$ with $map > cons > nil$ and consider the following SRS $\mathcal{R}$.

$$\mathcal{R} \begin{cases} ((map\ \alpha)\ nil) \to nil \\ ((map\ \alpha)\ (cons\ x\ y)) \to (cons\ (\alpha\ x)\ ((map\ \alpha)\ y)) \end{cases}$$

For the first rule $((map\ \alpha)\ nil) \succ_{lps} nil$ is trivial by the subterm property. For the second rule we have (i) $(map\ \alpha) \succ_{lps} cons$ because $map > cons$, (ii) $((map\ \alpha)\ (cons\ x\ y)) \succ_{lps} (\alpha\ x)$ because $(map\ \alpha) \succ_{lps} \alpha$ and $((map\ \alpha)\ (cons\ x\ y)) \succ_{lps} x$, and (iii) $((map\ \alpha)\ (cons\ x\ y)) \succ_{lps} ((map\ \alpha)\ y)$ because $(cons\ x\ y) \succ_{lps} y$. From this we have $((map\ \alpha)\ (cons\ x\ y)) \succ_{lps} (cons\ (\alpha\ x)\ ((map\ \alpha)\ y))$. Thus $\mathcal{R}$ is terminating since it is compatible with $\succ_{lps}$. □

Note that in the above example, *map* is represented as a crimped notion $((map\ s)\ t)$ by the extra parentheses [11], instead of the usual flat notion $(map\ s\ t)$. This crimping is necessary in our ordering because $(map\ \alpha\ (cons\ x\ y)) \not\succ_{lps} (cons\ (\alpha\ x)\ (map\ \alpha\ y))$.

Example 2 (maplist). Let $\mathcal{C} = \{fmap, cons, nil\}$ with $fmap > cons > nil$ and consider the following SRS $\mathcal{R}$.

$$\mathcal{R} \begin{cases} ((fmap\ nil)\ x) \to nil \\ ((fmap\ (cons\ \alpha\ \beta))\ \ x) \to (cons\ (\alpha\ x)\ ((fmap\ \beta)\ x)) \end{cases}$$

Then $\mathcal{R}$ is terminating since it can be shown to be compatible with $\succ_{lps}$. □

Example 3 (twice). Let $\mathcal{C} = \{twice, \bullet\}$ with $twice > \bullet$ and consider the following SRS $\mathcal{R}$.

$$\mathcal{R} \begin{cases} ((\bullet\ \alpha\ \beta)\ x) \to (\alpha\ (\beta\ x)) \\ (twice\ \alpha) \to (\bullet\ \alpha\ \alpha) \end{cases}$$

Then $\mathcal{R}$ is terminating since it can be shown to be compatible with $\succ_{lps}$. □

Example 4 (filter). Let $\mathcal{C} = \{filter, if, true, false, cons, nil\}$ with $filter > if > cons > nil, true, false$ and consider the following SRS $\mathcal{R}$.

$$\mathcal{R} \begin{cases} ((filter\ \alpha)\ nil) \to nil \\ ((filter\ \alpha)\ (cons\ x\ y)) \to (if\ (\alpha\ x)\ (cons\ x\ ((filter\ \alpha)\ y))\ ((filter\ \alpha)\ y)) \\ \\ (if\ true\ x\ y) \to x \\ (if\ false\ x\ y) \to y \end{cases}$$

Then $\mathcal{R}$ is terminating since it can be shown to be compatible with $\succ_{lps}$. □

Example 5 (folding). Let $\mathcal{C} = \{sum, prod, fold, *, +, s, 0, cons, nil\}$ with $sum, prod > fold > * > + > s > 0 > cons > nil$ and consider the following SRS $\mathcal{R}$.

$$\mathcal{R}\begin{cases}((fold\ \alpha\ x)\ nil) \to x\\ ((fold\ \alpha\ x)\ (cons\ y\ z)) \to (\alpha\ y\ ((fold\ \alpha\ x)\ z))\\ \\ sum \to (fold\ +\ 0)\\ prod \to (fold\ *\ (s\ 0))\\ \\ (+\ 0\ y) \to y\\ (+\ (s\ x)\ y) \to (s\ (+\ x\ y))\\ (*\ 0\ y) \to 0\\ (*\ (s\ x)\ y) \to (+\ (*\ x\ y)\ y)\end{cases}$$

Then $\mathcal{R}$ is terminating since it can be shown to be compatible with $\succ_{lps}$. □

Example 6 (recursor). Let $\mathcal{C} = \{rec, s, 0\}$ with $rec > s > 0$ and consider the following SRS $\mathcal{R}$.

$$\mathcal{R}\begin{cases}((rec\ \alpha\ x)\ 0) \to x\\ ((rec\ \alpha\ x)\ (s\ y)) \to (\alpha\ (s\ y)\ ((rec\ \alpha\ x)\ y))\end{cases}$$

Then $\mathcal{R}$ is terminating since it can be shown to be compatible with $\succ_{lps}$. □

Example 7 (sorting). Let $\mathcal{C} = \{asort, dsort, sort, ins, max, min, s, 0, cons, nil\}$ with $asort, dsort > sort > ins > max, min > s > 0 > cons > nil$ and consider the following SRS $\mathcal{R}$.

$$\mathcal{R}\begin{cases}((sort\ \alpha\ \beta)\ nil) \to nil\\ ((sort\ \alpha\ \beta)\ (cons\ x\ y)) \to ((ins\ \alpha\ \beta)\ ((sort\ \alpha\ \beta)\ y)\ x)\\ ((ins\ \alpha\ \beta)\ nil\ y) \to (cons\ y\ nil)\\ ((ins\ \alpha\ \beta)\ (cons\ x\ z)\ y) \to (cons\ (\alpha\ x\ y)\ ((ins\ \alpha\ \beta)\ z\ (\beta\ x\ y)))\\ \\ (max\ 0\ y) \to y\\ (max\ x\ 0) \to x\\ (max\ (s\ x)\ (s\ y)) \to (s\ (max\ x\ y))\\ (min\ 0\ y) \to 0\\ (min\ x\ 0) \to 0\\ (min\ (s\ x)\ (s\ y)) \to (s\ (min\ x\ y))\\ \\ (asort\ z) \to ((sort\ min\ max)\ z)\\ (dsort\ z) \to ((sort\ max\ min)\ z)\end{cases}$$

Then $\mathcal{R}$ is terminating since it can be shown to be compatible with $\succ_{lps}$. □

7 Relation between TRS and SRS

A set of terms over a signature $\mathcal{F}$ can be embedded in a set of S-expressions by regarding a function symbol as a constant. In this section we discuss the relation between two lexicographic path orders $\succ_{lpo}$ on terms and $\succ_{lps}$ on S-expressions. We show that under isomorphic embedding, $\succ_{lpo}$ and $\succ_{lps}$ coincide if they are induced from the same precedence $>$ over a signature $\mathcal{F}$. Thus, the lexicographic path orders $\succ_{lps}$ on S-expressions is a conservative extension of $\succ_{lpo}$ on usual first-order terms.

Let $\mathcal{T}(\mathcal{F},\mathcal{V})$ be a set of terms bult from a signature $\mathcal{F}$ and a set $\mathcal{V}$ of variables. The set $\mathcal{S}(\mathcal{F},\mathcal{V})$ of S-expressions induced from a signature $\mathcal{F}$ is recursively defined by: (i) $x \in \mathcal{S}(\mathcal{F},\mathcal{V})$ for $x \in \mathcal{V}$, (ii) $c \in \mathcal{S}(\mathcal{F},\mathcal{V})$ for a constant c, and (iii) $(f\ s_1\ \cdots\ s_m)$ if $s_1,\cdots,s_m \in \mathcal{S}(\mathcal{F},\mathcal{V})$ and $f \in \mathcal{F}$ has the arity m ($m \geq 1$). Then $\mathcal{T}(\mathcal{F},\mathcal{V})$ and $\mathcal{S}(\mathcal{F},\mathcal{V})$ are isomorphic thanks to an isomorphism $\Psi : \mathcal{T}(\mathcal{F},\mathcal{V}) \rightarrow \mathcal{S}(\mathcal{F},\mathcal{V})$ such that $\Psi(f(t_1,\cdots,t_m)) = (f\ \Psi(t_1)\ \cdots\ \Psi(t_m))$ ($m \geq 1$) and $\Psi(a) = a$ for a constant or variable a; for example, $\Psi(f(g(x,c),y)) = (f\ (g\ x\ c)\ y)$. Thus $\Psi(\mathcal{T}(\mathcal{F},\mathcal{V})) = \mathcal{S}(\mathcal{F},\mathcal{V})$.

Let $\mathcal{R}$ be a TRS over $\mathcal{T}(\mathcal{F},\mathcal{V})$. Then an SRS over $\Psi(\mathcal{T}(\mathcal{F},\mathcal{V}))$ induced from $\mathcal{R}$ is defined by $\Psi(\mathcal{R}) = \{\Psi(l) \rightarrow \Psi(r) \mid l \rightarrow r \in \mathcal{R}\}$. We remark that $s \rightarrow_{\mathcal{R}} t \Leftrightarrow \Psi(s) \rightarrow_{\Psi(\mathcal{R})} \Psi(t)$. Thus $\mathcal{R}$ is terminating if and only if $\Psi(\mathcal{R})$ is terminating.

A precedence $>$ on a signature $\mathcal{F}$ gives two lexicographic path orders $\succ_{lpo}$ on $\mathcal{T}(\mathcal{F},\mathcal{V})$ and $\succ_{lps}$ on $\Psi(\mathcal{T}(\mathcal{F},\mathcal{V}))$. The following result shows that $\succ_{lpo}$ and $\succ_{lps}$ are isomorphic.

Theorem 19. Two lexicographic path orders $\succ_{lpo}$ on $\mathcal{T}(\mathcal{F},\mathcal{V})$ and $\succ_{lps}$ on $\Psi(\mathcal{T}(\mathcal{F},\mathcal{V}))$ are isomorphic, i.e., $s \succ_{lpo} t \Leftrightarrow \Psi(s) \succ_{lps} \Psi(t)$.

Proof. By induction on the size of terms t and s, it can be proven. □

8 Typed S-Expressions and Currying

Transformation techniques [3] are widely used to prove termination of rewriting systems that cannot be proven by the classical orderings. In this section we propose a transformational method to strengthen the power of the lexicographic path order $\succ_{lps}$ based on the type information.

The set $\mathcal{T}$ of types built from a non-empty set of base types $\mathcal{B}$ is recursively defined by: (i) $\tau \in \mathcal{T}$ for $\tau \in \mathcal{B}$, and (ii) $\tau_1 \times \tau_2 \times \cdots \times \tau_n \rightarrow \tau_0$ ($n \geq 1$) for $\tau_0,\ldots,\tau_n \in \mathcal{T}$. A type not in $\mathcal{B}$ is called function type. We use the symbols τ, σ, ... to denote types. For every type we assume a countably infinite set of variables of that type, written as $x, y, z, \ldots$ for base types and as $\alpha, \beta, \gamma, \ldots$ for functional types.

Let $\mathcal{C}^\tau$ be a set of constants of type τ, denoted by $f, g, h, \ldots$, and $\mathcal{V}^\tau$ a set of variables of type τ, and $\mathcal{C} = \bigcup_\tau \mathcal{C}^\tau$ and $\mathcal{V} = \bigcup_\tau \mathcal{V}^\tau$. Here, $\mathcal{C}^\tau \cap \mathcal{C}^\sigma = \phi$ and $\mathcal{V}^\tau \cap \mathcal{V}^\sigma = \phi$ if $\tau \neq \sigma$. The set $\mathcal{S}(\mathcal{C},\mathcal{V})^\tau$ of terms (i.e., S-expressions) of type τ is recursively defined by: (i) $a \in \mathcal{S}(\mathcal{C},\mathcal{V})^\tau$ for $a \in \mathcal{C}^\tau \cup \mathcal{V}^\tau$, and (ii) $(f\ t_1\ \cdots\ t_n) \in \mathcal{S}(\mathcal{C},\mathcal{V})^\tau$ for $f \in \mathcal{C}^{\tau_1 \times \tau_2 \times \cdots \times \tau_n \rightarrow \tau}$ and $t_i \in \mathcal{S}(\mathcal{C},\mathcal{V})^{\tau_i}$ ($i = 1,\cdots,n$). We define $\mathcal{S}(\mathcal{C},\mathcal{V}) = \bigcup_\tau \mathcal{S}(\mathcal{C},\mathcal{V})^\tau$. The symbols $s, t, r, \ldots$ denote terms. The notation t^τ is sometimes used to indicate the type τ of a term t explicitly. In keeping well-typed structure we make use of the usual notions of substitution of a term t, denoted as $t\theta$, and replacement in a context, denoted as $C[t]$.

A rewrite rule, written as $l \to r$, is a pair of typed terms such that $l \notin \mathcal{V}$, $\mathcal{V}(r) \subseteq \mathcal{V}(l)$, and l and r have the same type. A *typed SRS* $\mathcal{R}$ is a set of rewrite rules. The rewrite rules of a typed SRS $\mathcal{R}$ define a reduction relation $\to_{\mathcal{R}}$ on $\mathcal{S}(\mathcal{C},\mathcal{V})$ in the usual manner.

Definition 20 (currying). Let $\mathcal{T}$ be a set of types generated from a set $\mathcal{B}$ of base types. A *currying* Γ is a pair of types $\langle \tau_1 \times \cdots \times \tau_p \times \tau_{p+1} \times \cdots \times \tau_m \to \tau_0,\ \tau_1 \times \cdots \times \tau_p \to (\tau_{p+1} \times \cdots \times \tau_m \to \tau_0)\rangle$ where $1 \leq p < m$. A currying Γ can be extended to an endomorphism over $\mathcal{T}$ as follows.

$$\Gamma(\tau) = \begin{cases} \sigma & \text{if } \Gamma = \langle \tau, \sigma\rangle \\ \tau & \text{if } \tau \in \mathcal{B} \\ \Gamma(\rho_1) \times \cdots \times \Gamma(\rho_n) \to \Gamma(\rho) & \text{otherwise} \quad (\tau = \rho_1 \times \cdots \times \rho_n \to \rho) \end{cases}$$

Note that $\Gamma(\rho)$ is a type obtained from a type ρ by replacing all occurrences τ in ρ with a type σ when $\Gamma = \langle \tau, \sigma\rangle$. Thus $\Gamma(\rho) = \rho$ if no τ occurs in ρ.

Let $\mathcal{C}$ and $\mathcal{V}$ be sets of typed constants and variables respectively. By replacing the type τ of elements a^τ in $\mathcal{C}$ and $\mathcal{V}$ with $\Gamma(\tau)$, we obtain a set of constants $\Gamma(\mathcal{C}) = \{c^{\Gamma(\tau)} \mid c^\tau \in \mathcal{C}\}$ and a set of variables $\Gamma(\mathcal{V}) = \{x^{\Gamma(\tau)} \mid x^\tau \in \mathcal{V}\}$. Then each term in $\mathcal{S}(\mathcal{C},\mathcal{V})$ can be converted into $\mathcal{S}(\Gamma(\mathcal{C}),\Gamma(\mathcal{V}))$ as follows.

Definition 21. Let $\mathcal{S}(\mathcal{C},\mathcal{V})$ be a set of typed terms. Let Γ be a currying such that $\Gamma(\tau_1 \times \cdots \times \tau_p \times \tau_{p+1} \times \cdots \times \tau_m \to \tau_0) = \tau_1 \times \cdots \times \tau_p \to (\tau_{p+1} \times \cdots \times \tau_m \to \tau_0)$. Then a homomorphism from $\mathcal{S}(\mathcal{C},\mathcal{V})$ to $\mathcal{S}(\Gamma(\mathcal{C}),\Gamma(\mathcal{V}))$ induced from Γ, denoted by the same letter Γ, is defined as follows.

$$\Gamma(t) = \begin{cases} (\ (\Gamma(t_0)\ \Gamma(t_1)\ \cdots\ \Gamma(t_p)\ )\ \Gamma(t_{p+1})\ \cdots\ \Gamma(t_m)\ ) \\ \qquad\qquad \text{if } t = ({t_0}^{\tau_1 \times \cdots \times \tau_p \times \tau_{p+1} \times \cdots \times \tau_m \to \tau_0}\ {t_1}^{\tau_1}\ \cdots\ {t_p}^{\tau_p}\ {t_{p+1}}^{\tau_{p+1}}\ \cdots\ {t_m}^{\tau_m}) \\ a^{\Gamma(\tau)} \qquad \text{if } t = a^\tau \in \mathcal{C} \cup \mathcal{V} \\ (\Gamma(t_0)\ \cdots\ \Gamma(t_n)\ ) \qquad \text{otherwise} \quad (t = (t_0\ \cdots\ t_n)) \end{cases}$$

Example 8 (currying). Let $\mathcal{B} = \{N, L\}$ and $\mathcal{C} = \{map^{(N\to N)\times L \to L}, cons^{N \times L \to L}, nil^L\}$. Let $\Gamma((N \to N) \times L \to L) = (N \to N) \to (L \to L)$. Then we have the following curried term.

$$\Gamma((map^{(N\to N)\times L\to L}\ \alpha^{N\to N}\ (cons^{N\times L\to L}\ x^N\ y^L)^L)^L) =$$
$$((map^{(N\to N)\to(L\to L)}\ \alpha^{N\to N})^{L\to L}\ (cons^{N\times L\to L}\ x^N\ y^L)^L)^L \qquad \square$$

Lemma 22. Let t be a term of type σ in $\mathcal{S}(\mathcal{C},\mathcal{V})$. Then $\Gamma(t)$ is a well-typed term of type $\Gamma(\sigma)$ in $\mathcal{S}(\Gamma(\mathcal{C}),\Gamma(\mathcal{V}))$.

Proof. By induction on the structure of t, we can prove the claim. $\square$

For a substitution θ, we define a substitution $\Gamma(\theta)$ by $\Gamma(\theta)(x^{\Gamma(\tau)}) = \Gamma(x^\tau\theta)$. The type τ of hole in a context $C[\,]$ is denoted by $C[\,]_\tau$. Then a context $\Gamma(C)[\,]$ is defined by $\Gamma(C)[\,]_{\Gamma(\tau)} = \Gamma(C[\,]_\tau)$.

The following results are easily proven by induction on structure of a term.

Lemma 23. For a term t and a substitution θ we have $\Gamma(t\theta) = \Gamma(t)\Gamma(\theta)$.

Lemma 24. For a term t^τ and a context $C[\ ]_\tau$ we have $\Gamma(C[t]) = \Gamma(C)[\Gamma(t)]$.

For a typed SRS $\mathcal{R}$ on $\mathcal{S}(\mathcal{C}, \mathcal{V})$ we define a *curried* SRS $\Gamma(\mathcal{R})$ on $\mathcal{S}(\Gamma(\mathcal{C}), \Gamma(\mathcal{V}))$ by $\Gamma(\mathcal{R}) = \{\Gamma(l) \to \Gamma(r) \mid l \to r \in \mathcal{R}\}$. The next result combines termination of $\Gamma(\mathcal{R})$ with termination of $\mathcal{R}$.

Lemma 25. Let Γ be a currying and $\mathcal{R}$ a typed SRS. Then $\Gamma(s) \to_{\Gamma(\mathcal{R})} \Gamma(t)$ for $s \to_{\mathcal{R}} t$.

Proof. Let $s = C[l\theta]$ and $t = C[r\theta]$ for some $l \to r \in \mathcal{R}$, $C[\ ]$, θ. From Lemmas 23 and 24 we have $\Gamma(s) = \Gamma(C[l\theta]) = \Gamma(C)[\Gamma(l)\Gamma(\theta)]$ and $\Gamma(t) = \Gamma(C[r\theta]) = \Gamma(C)[\Gamma(r)\Gamma(\theta)]$. Thus it follows that $\Gamma(s) \to_{\Gamma(\mathcal{R})} \Gamma(t)$. $\square$

Theorem 26. Let Γ be a currying and $\mathcal{R}$ a typed SRS. Then $\mathcal{R}$ is terminating if $\Gamma(\mathcal{R})$ is terminating.

Proof. For a proof by contradiction, suppose an infinite reduction sequence $t_0 \to_{\mathcal{R}} t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} t_3 \to_{\mathcal{R}} \cdots$. Then from Lemma 25 there exists an infinite reduction sequence $\Gamma(t_0) \to_{\Gamma(\mathcal{R})} \Gamma(t_1) \to_{\Gamma(\mathcal{R})} \Gamma(t_2) \to_{\Gamma(\mathcal{R})} \Gamma(t_3) \to_{\Gamma(\mathcal{R})} \cdots$. This is a contradiction to termination of $\Gamma(\mathcal{R})$, and hence the claim holds. $\square$

Note that termination of $\mathcal{R}$ does not guarantee that of $\Gamma(\mathcal{R})$. For example, consider $\mathcal{R} = \{(f\ ((\alpha\ x)\ y)) \to (f\ (a\ b\ c))\}$, where $f^{0\to 0}$, $a^{1\times 1\to 0}$, b^1, c^1 in $\mathcal{C}$ and $\alpha^{1\to 1\to 0}$, x^1, y^1 in $\mathcal{V}$. Take $\Gamma(1 \times 1 \to 0) = 1 \to (1 \to 1)$. We have $\Gamma(\mathcal{R}) = \{(f\ ((\alpha\ x)\ y)) \to (f\ ((a\ b)\ c))\}$. Then $(f\ ((a\ b)\ c)) \to_{\Gamma(\mathcal{R})} (f\ ((a\ b)\ c))$ but not $(f\ (a\ b\ c)) \to_{\mathcal{R}} (f\ (a\ b\ c))$.

In the following examples [1, 6, 10, 11, 12, 14], we regard $\Gamma(\mathcal{R})$ as an untyped SRS and prove termination of it by applying the lexicographic path order $\succ_{lps}$. It should be noted that in our termination method the type information is used only for currying.

Example 9 ($map^{(N\to N)\times L\to L}$). Let $\mathcal{C} = \{map^{(N\to N)\times L\to L}, cons^{N\times L\to L}, nil^L\}$ with $map > cons > nil$ and consider the following typed SRS $\mathcal{R}$. (The type information is omitted in the subsequent examples, as it is easily derived.)

$$\mathcal{R}\begin{cases} (map\ \alpha\ nil) \to nil \\ (map\ \alpha\ (cons\ x\ y)) \to (cons\ (\alpha\ x)\ (map\ \alpha\ y)) \end{cases}$$

Take $\Gamma((N \to N) \times L \to L) = (N \to N) \to (L \to L)$. Then we have the following curried SRS $\Gamma(\mathcal{R})$.

$$\Gamma(\mathcal{R})\begin{cases} ((map\ \alpha)\ nil) \to nil \\ ((map\ \alpha)\ (cons\ x\ y)) \to (cons\ (\alpha\ x)\ ((map\ \alpha)\ y)) \end{cases}$$

From Example 1, $\Gamma(\mathcal{R})$ is terminating. Thus, according to Theorem 26, $\mathcal{R}$ is terminating. $\square$

Example 10 ($fmap^{L\times N\to L}$). Let $\mathcal{C} = \{fmap^{L\times N\to L}, cons^{(N\to N)\times L\to L}, nil^L\}$ with $fmap > cons > nil$ and consider the following typed SRS $\mathcal{R}$.

$$\mathcal{R}\begin{cases}(fmap\ nil\ x) \to nil\\ (fmap\ (cons\ \alpha\ \beta)\ \ x) \to (cons\ (\alpha\ x)\ (fmap\ \beta\ x))\end{cases}$$

Take $\Gamma(L \times N \to L) = L \to (N \to L)$. Then we have the following curried SRS $\Gamma(\mathcal{R})$.

$$\Gamma(\mathcal{R})\begin{cases}((fmap\ nil)\ x) \to nil\\ ((fmap\ (cons\ \alpha\ \beta))\ \ x) \to (cons\ (\alpha\ x)\ ((fmap\ \beta)\ x))\end{cases}$$

From Example 2, $\Gamma(\mathcal{R})$ is terminating. Thus, according to Theorem 26, $\mathcal{R}$ is terminating. □

Example 11 ($fold^{(N\times N\to N)\times N\times L\to N}$). Let $\mathcal{C} = \{fold^{(N\times N\to N)\times N\times L\to N}, cons^{N\times L\to L}, nil^L\}$ with $fold > cons > nil$ and consider the following typed SRS $\mathcal{R}$.

$$\mathcal{R}\begin{cases}(fold\ \alpha\ x\ nil) \to x\\ (fold\ \alpha\ x\ (cons\ y\ z)) \to (\alpha\ y\ (fold\ \alpha\ x\ z))\end{cases}$$

Take $\Gamma((N \times N \to N) \times N \times L \to N) = (N \times N \to N) \times N \to (L \to N)$. Then we have the following curried SRS $\Gamma(\mathcal{R})$.

$$\Gamma(\mathcal{R})\begin{cases}((fold\ \alpha\ x)\ nil) \to x\\ ((fold\ \alpha\ x)\ (cons\ y\ z)) \to (\alpha\ y\ ((fold\ \alpha\ x)\ z))\end{cases}$$

From Example 5, $\Gamma(\mathcal{R})$ is terminating. Thus, according to Theorem 26, $\mathcal{R}$ is terminating. □

Example 12 ($rec^{(N\times A\to A)\times A\times N\to A}$). Let $\mathcal{C} = \{rec^{(N\times A\to A)\times A\times N\to A}), s^{N\to N}, 0^N\}$ with $rec > s > 0$ and consider the following typed SRS $\mathcal{R}$.

$$\mathcal{R}\begin{cases}(rec\ \alpha\ x\ 0) \to x\\ (rec\ \alpha\ x\ (s\ y)) \to (\alpha\ (s\ y)\ (rec\ \alpha\ x\ y))\end{cases}$$

Take $\Gamma((N \times A \to A) \times A \times N \to A) = (N \times A \to A) \times A \to (N \to A)$. Then we have the following curried SRS $\Gamma(\mathcal{R})$.

$$\Gamma(\mathcal{R})\begin{cases}((rec\ \alpha\ x)\ 0) \to x\\ ((rec\ \alpha\ x)\ (s\ y)) \to (\alpha\ (s\ y)\ ((rec\ \alpha\ x)\ y))\end{cases}$$

From Example 6, $\Gamma(\mathcal{R})$ is terminating. Thus, according to Theorem 26, $\mathcal{R}$ is terminating. □

Example 13 ($sort^{(N\times N\to N)\times(N\times N\to N)\times L\to L}$). Let $\mathcal{C} = \{asort^{L\to L}, dsort^{L\to L}$, $sort^{(N\times N\to N)\times(N\times N\to N)\times L\to L}$, $ins^{(N\times N\to N)\times(N\times N\to N)\times L\times N\to L}$, $max^{N\times N\to N}$, $min^{N\times N\to N}$, $s^{N\to N}, 0^N$, $cons^{N\times L\to L}, nil^L\}$ with $asort, dsort > sort > ins > max, min > s > 0 > cons > nil$ and consider the following typed SRS $\mathcal{R}$.

$$\mathcal{R}\begin{cases}(sort\ \alpha\ \beta\ nil) \to nil\\(sort\ \alpha\ \beta\ (cons\ x\ y)) \to (ins\ \alpha\ \beta\ (sort\ \alpha\ \beta\ y)\ x)\\(ins\ \alpha\ \beta\ nil\ y) \to (cons\ y\ nil)\\(ins\ \alpha\ \beta\ (cons\ x\ z)\ y) \to (cons\ (\alpha\ x\ y)\ (ins\ \alpha\ \beta\ z\ (\beta\ x\ y)))\\ \\(max\ 0\ y) \to y\\(max\ x\ 0) \to x\\(max\ (s\ x)\ (s\ y)) \to (s\ (max\ x\ y))\\(min\ 0\ y) \to 0\\(min\ x\ 0) \to 0\\(min\ (s\ x)\ (s\ y)) \to (s\ (min\ x\ y))\\ \\(asort\ z) \to (sort\ min\ max\ z)\\(dsort\ z) \to (sort\ max\ min\ z)\end{cases}$$

Take two curryings $\Gamma_1((N \times N \to N) \times (N \times N \to N) \times L \to L)) = (N \times N \to N) \times (N \times N \to N) \to (L \to L)$ and $\Gamma_2((N \times N \to N) \times (N \times N \to N) \times L \times N \to L)) = (N \times N \to N) \times (N \times N \to N) \to (L \times N \to L)$. Currying $\mathcal{R}$ twice, we have the following curried SRS $\Gamma_2(\Gamma_1(\mathcal{R}))$.

$$\Gamma_2(\Gamma_1(\mathcal{R}))\begin{cases}((sort\ \alpha\ \beta)\ nil) \to nil\\((sort\ \alpha\ \beta)\ (cons\ x\ y)) \to ((ins\ \alpha\ \beta)\ ((sort\ \alpha\ \beta)\ y)\ x)\\((ins\ \alpha\ \beta)\ nil\ y) \to (cons\ y\ nil)\\((ins\ \alpha\ \beta)\ (cons\ x\ z)\ y) \to (cons\ (\alpha\ x\ y)\ ((ins\ \alpha\ \beta)\ z\ (\beta\ x\ y)))\\ \\(max\ 0\ y) \to y\\(max\ x\ 0) \to x\\(max\ (s\ x)\ (s\ y)) \to (s\ (max\ x\ y))\\(min\ 0\ y) \to 0\\(min\ x\ 0) \to 0\\(min\ (s\ x)\ (s\ y)) \to (s\ (min\ x\ y))\\ \\(asort\ z) \to ((sort\ min\ max)\ z)\\(dsort\ z) \to ((sort\ max\ min)\ z)\end{cases}$$

From Example 7, $\Gamma_2(\Gamma_1(\mathcal{R}))$ is terminating. Thus, according to Theorem 26, $\mathcal{R}$ is terminating. □

Acknowledgments. The author would like to thank Keiichiro Kusakari and Takahito Aoto for valuable discussion. Thanks also due to the referees for useful comments and suggestions.

References

1. T. Aoto and T. Yamada, Proving termination of simply typed term rewriting systems automatically, *IPSJ Trans. on Prog. 44, SIG-4* (2003) 67-77, *in Japanese.*
2. F. Baader and T. Nipkow, Term Rewriting and All That, Cambridge University Press (1998).

3. F. Bellegarde and P. Lescanne, Transformation orderings, *Lecture Notes in Comput. Sci. 249* (1987) 69-80.
4. N. Dershowitz, Ordering for term-rewriting systems, *Theoretical Comput. Sci. 17* (1982) 279-301.
5. M. Ferreira, Termination of term rewriting: Well-foundedness, totality and transformation, *PhD thesis*, Dep. of Comput. Sci., Utrecht University (1995).
6. J.-P. Jouannaud and A. Rubio, Higher-Order Recursive Path Orderings, *Procs. 14th IEEE Symp. on Logic in Comput. Sci.* (1999) 402–411.
7. S. Kamin and J.-J. Levy. Two generalizations of the recursive path ordering, *Unpublished Manuscript*, University of Illinois (1980).
8. R. Kennaway, J. W. Klop, M. R. Sleep, F.-J. de Vries, Comparing Curried and Uncurried Rewriting, *J. Symb. Comput. 21(1)* (1996) 15-39.
9. B. Kruskal, Well quasi ordering, the tree theorem and Vazsonyi's conjecture, *Trans. Am. Math. Soc. 95* (1960) 210-225.
10. K. Kusakari, On Proving Termination of Term Rewriting Systems with Higher-Order Variables, *IPSJ Trans. on Prog. 42, SIG-7* (2001) 35-45.
11. M. Linfantsev and L. Bachmair, An LPO-based termination ordering for higher-order terms without λ-abstraction, *Lecture Notes in Comput. Sci. 1479* (1998) 277-293.
12. O. Lysne and J. Piri,. A termination ordering for higher order rewrite systems, *Lecture Notes in Comput. Sci. 914* (1995) 26-24.
13. A. Middeldorp and H. Zantema, Simple termination of rewrite systems, *Theoretical Comput. Sci. 175* (1997) 127-158.
14. F. van Raamsdonk, On Termination of Higher-Order Rewriting, *Lecture Notes in Comput. Sci. 2051* (2001) 261–275.
15. Terese, Term Rewriting Systems, Cambridge University Press (2003).