Tree lifting orderings for termination transformations of term rewriting systems

Takahito Aoto
aoto@jaist.ac.jp

Yoshihito Toyama
toyama@jaist.ac.jp

School of Information Science, JAIST
Asahidai Tatsunokuchi, Ishikawa 923-12, Japan

August 1, 1997

Abstract

A technique to prove the termination of a given term rewriting system (TRS, for short) is presented. We propose tree lifting orderings by which from a given TRS $\mathcal{R}$ candidates for the termination of $\mathcal{R}$ can be obtained—the termination of (at least) one of these candidates guarantees the termination of $\mathcal{R}$. It should be remarked that for a given finite TRS all its candidates can be computed automatically. Our technique is a generalization of the dummy elimination transformation by Ferreira and Zantema (1995) and extends a modularity result on the termination of the direct sum of TRSs by Middeldorp (1989).

1 Introduction

In this paper, a technique to prove the termination of a given term rewriting system (TRS, for short) is presented. We propose *tree lifting orderings* by which from a given TRS $\mathcal{R}$ candidates for the termination of $\mathcal{R}$ can be obtained—the termination of (at least) one of these candidates guarantees the termination of $\mathcal{R}$.

Intended for resolving the termination of a TRS into that of another, some kinds of transformation of TRSs have been proposed [FZ95] [Zan95] [Zan94]. These transformations are especially useful to show the termination of non-simply terminating TRSs, while well-established methods based on precedence orderings, e.g. the recursive path ordering (c.f. [Der87]), are

effective only to show the termination of simply terminating TRSs. Here, a terminating TRS is said to be simply terminating if its termination can be proved by means of a simplification order, otherwise it is non-simply terminating [MZ97].

Among the proposed transformations, the semantic labelling by Zantema [Zan95] is powerful enough to resolve the non-simple termination of a TRS into the simple termination of another TRS [MOZ96]. Transformations by the semantic labelling, however, can not be proceeded automatically (we have to moreover give "semantics"). Our technique is not so powerful as the semantic labelling; instead, for a given finite TRS $\mathcal{R}$ all its candidates whose termination ensures that of $\mathcal{R}$ via our result can be computed automatically.

In the remainder of this section, we briefly describe how the present work was developed from the dummy elimination transformation.

Let a be a fixed function symbol (called a dummy symbol)—we assume it's binary or unary depending on its context. Then each term (tree) is partitioned by the symbol a in an obvious way (see Figure 1), and so by assigning a new constant $\diamondsuit$ at each position of a, one can divide the term (tree) into parts. For example, $f(g(a(f(a(x,y),z),y)),a(g(x),z))$ can be divided into parts $\{f(g(\diamondsuit),\diamondsuit), f(\diamondsuit,z), g(x), x, y, z\}$. Among these parts, the top-part, which results from t by replacing all maximal subterms in t of the form $a(s_1,\ldots,s_n)$ with $\diamondsuit$, is called the cap-part of t (denoted by $\mathrm{cap}(t)$); the rests are dec-parts of t (we denote by $\mathrm{dec}(t)$ the set of dec-parts of t). For example, if $t \equiv f(g(a(f(a(x,y),z),y)),a(g(x),z))$, we have $\mathrm{cap}(t) \equiv f(g(\diamondsuit),\diamondsuit)$ and $\mathrm{dec}(t) = \{f(\diamondsuit,z), g(x), x, y, z\}$.

The dummy elimination transforms a given TRS $\mathcal{R}$ into a TRS $\mathcal{E}(\mathcal{R})$, where

$$\mathcal{E}(\mathcal{R}) = \bigcup_{l\to r\in\mathcal{R}} \{\mathrm{cap}(l) \to u \mid u \in \mathrm{dec}(r) \cup \{\mathrm{cap}(r)\}\}.$$

Then the termination of $\mathcal{R}$ follows from that of $\mathcal{E}(\mathcal{R})$ [FZ95]. Here is an example from [FZ95]. Let

$$\mathcal{R}\left\{\ f(g(x)) \to f(a(g(g(f(x))),g(f(x)))).\right.$$

Then,

$$\mathcal{E}(\mathcal{R})\left\{\begin{array}{l} f(g(x)) \to f(\diamondsuit) \\ f(g(x)) \to g(g(f(x))) \\ f(g(x)) \to g(f(x))). \end{array}\right.$$

It is easy to show the termination of $\mathcal{E}(\mathcal{R})$ using the recursive path ordering, while that of $\mathcal{R}$ can not be shown via simplification orders.

It has to be observed here that the resulting TRS $\mathcal{E}(\mathcal{R})$ does not always satisfy a usual restriction on TRSs, i.e. a variable in the right-hand side of

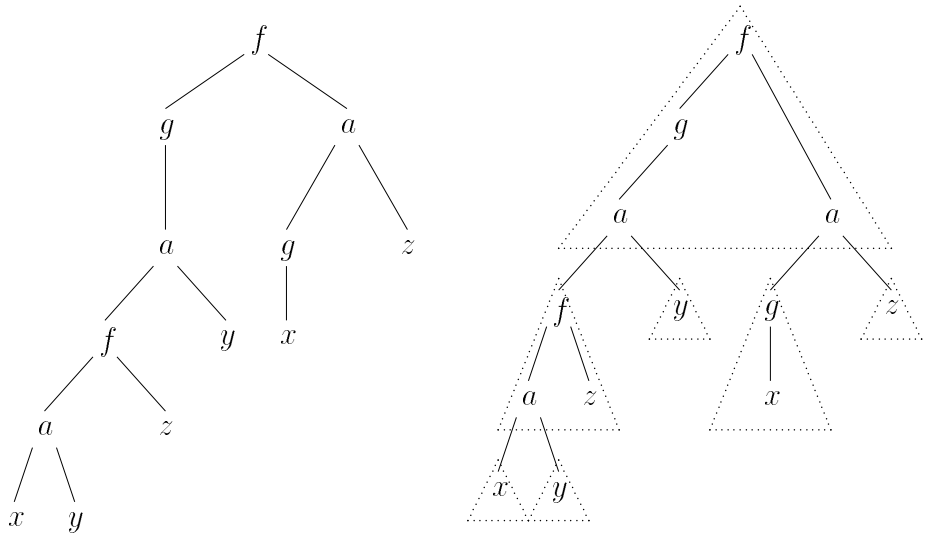

Figure 1: A partition of a term tree

a rule may not appear in the left-hand side of the rule—for example, a rule $f(a(g(x))) \to a(x)$ in $\mathcal{R}$ creates a rule $f(\Diamond) \to x$ in $\mathcal{E}(\mathcal{R})$. In such case, we regard $\mathcal{E}(\mathcal{R})$ non-terminating without loss of generality.

Now, let us assign each term t a tree (called a *meta-tree* of t) consisted of the parts of t with the tree structure inherited from the term tree t. More precisely, the set of nodes of the meta-tree of t is $\mathrm{dec}(t) \cup \{\mathrm{cap}(t)\}$, and if $t \equiv C[a(\vec{s_1}), \ldots, a(\vec{s_n})]$ where C is a context that does not contain a then the top node of the meta-tree of t is $\mathrm{cap}(t)$ and its maximal proper subtrees are meta-trees of $\vec{s_1}, \ldots, \vec{s_n}$. For example, Figure 2 is the meta-tree of $f(g(a(f(a(x,y),z),y)),a(g(x),z))$.

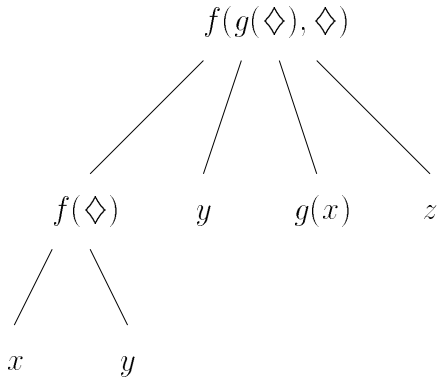

Figure 2: A meta-tree

Suppose $\mathcal{E}(\mathcal{R})$ is terminating. Then $\xrightarrow{+}_{\mathcal{E}(\mathcal{R})}$ is a well-founded order on

nodes of meta-trees. If one looks at a rewriting $s \to_{\mathcal{R}} t$ of $\mathcal{R}$ in terms of meta-trees, it will be observed that a node v of the meta-tree of s changes into nodes $u_1, \ldots, u_k$ of the meta-tree of t and that $u_1, \ldots, u_k$ are $\mathcal{E}(\mathcal{R})$-reducts of v. For example, Figure 3 shows a rewriting of $\mathcal{R}$ given above in which a node $v \equiv f(g(\cdots))$ in the meta-tree of s is changes into nodes $u_1 \equiv f(\diamondsuit)$, $u_2 \equiv g(g(f(\cdots)))$, and $u_3 \equiv g(f(\cdots))$ in the meta-tree of t. Thus, each rewriting of $\mathcal{R}$ replaces a node of the meta-tree with nodes

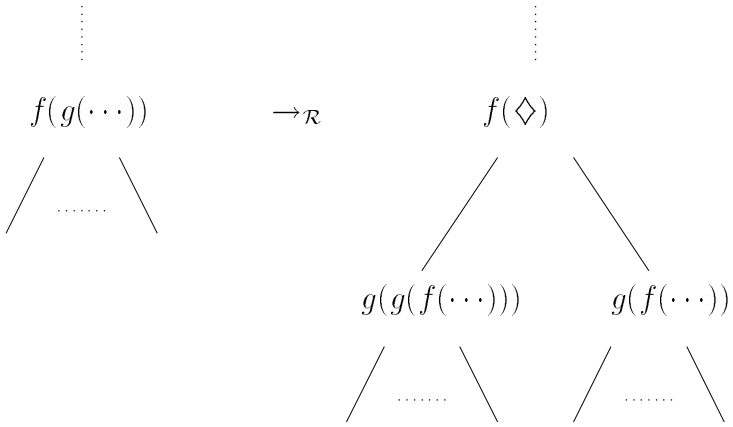

Figure 3: A rewriting in terms of meta-trees

smaller with respect to $\stackrel{+}{\to}_{\mathcal{E}(\mathcal{R})}$. Therefore $\mathcal{R}$ is terminating.

That a meta-tree becomes smaller if its node is replaced by some smaller nodes is a very special case of the recursive path ordering. Can one infer the termination of $\mathcal{R}$ from that of an arbitrary $\mathcal{R}'$, not only from that of $\mathcal{E}(\mathcal{R})$, assuming that for each rule $l \to r \in \mathcal{R}$ the meta-tree of l is larger than that of r by the recursive path ordering on meta-trees induced from $\stackrel{+}{\to}_{\mathcal{R}'}$? The next example answers this question negatively.

Example 1.1. Let

$$\mathcal{R} \left\{ \begin{array}{l} g(c) \to g(f(a(c))) \\ f(a(c)) \to c \end{array} \right. \qquad \mathcal{R}' \left\{ \begin{array}{l} g(c) \to g(f(\diamondsuit)) \\ g(c) \to c. \end{array} \right.$$

Firstly, $\mathcal{R}'$ is terminating (it can be shown easily e.g. by using the recursive path ordering), and so $>= \stackrel{+}{\to}_{\mathcal{R}'}$ is a well-founded order. Since the meta-tree of $g(f(a(c)))$ is $g(f(\diamondsuit))\langle c \rangle$, the meta-tree of $g(c)$ is larger than that of $g(f(a(c)))$ with respect to $>_{rpo}$, for we have $g(c) \to_{\mathcal{R}'} g(f(\diamondsuit))$ and $g(c) \to_{\mathcal{R}'} c$. Also, since the meta-tree $f(a(c))$ is $f(\diamondsuit)\langle c \rangle$, the meta-tree of $f(a(c))$ is larger than that of c with respect to $>_{rpo}$, for c is a proper subtree of $f(\diamondsuit)\langle c \rangle$.

But $\mathcal{R}$ is not terminating as $g(f(a(c))) \to_{\mathcal{R}} g(c) \to_{\mathcal{R}} g(f(a(c))) \to_{\mathcal{R}} \cdots$. Thus, the termination of $\mathcal{R}'$ does not imply that of $\mathcal{R}$, although every rule of $\mathcal{R}$ becomes smaller with respect to $>_{rpo}$.

However, this example only shows that to ensure the termination of $\mathcal{R}$ from that of $\mathcal{R}'$ via the recursive path ordering on meta-trees, we have to restrict a usual sufficient condition of the recursive path ordering that every proper subtree of a tree is smaller than the whole tree. In fact, our result is based on this approach, and it will turns out that this approach is quite successful.

There is also another technical problem to accomplish this approach. The recursive path ordering on meta-trees induced from a reduction order is in general not closed under contexts, i.e. the meta-tree of $C[t]$ needs not be larger than that of $C[s]$ even if the meta-tree of t is larger than that of s. This can be observed in the next example.

EXAMPLE 1.2. Let

$$\mathcal{R} \left\{ \; f(x) \to g(a(c)) \right. \qquad \mathcal{R}' \left\{ \begin{array}{l} f(x) \to g(\Diamond) \\ f(x) \to c. \end{array} \right.$$

The termination of $\mathcal{R}'$ can be shown easily (e.g. by using the recursive path ordering). Let $>$ be a reduction order that includes $\overset{+}{\to}_{\mathcal{R}'}$. Then the meta-tree of $f(x)$ is larger than that of $g(a(c))$ with respect to $>_{rpo}$, but this is not necessarily true for meta-trees of $f(f(x))$ and $f(g(a(c)))$, for we do not have $f(f(x)) > c$ unless $>$ is has the subterm property.

As we have seen, the approach needs to be modified in several points, nevertheless the resulting ordering enriches the application of the dummy elimination transformation technique—our ordering will also be effective against the following examples, while the original dummy elimination is not. The translation of these TRSs will be presented in EXAMPLE 4.7.

EXAMPLE 1.3.

$$\mathcal{R}_1 \left\{ \begin{array}{l} f(f(x)) \to f(a(f(x))) \\ f(a(g(x))) \to a(x) \end{array} \right.$$

$$\mathcal{R}_2 \left\{ \; h(a(g(g(x)))) \to h(a(g(a(g(x))))) \right.$$

$$\mathcal{R}_3 \left\{ \; f(f(a(x))) \to f(a(f(a(x)))) \right.$$

The second direction of our extension is to relax the restriction on the number of dummy symbols. This idea is not new (c.f. [Fer95]). However, by this extension, we can add an arbitrary terminating non-duplicating non-collapsing rules consisted of dummy symbols without loosing correctness of the transformation, because rewriting by these rules does not enlarge the meta-trees. This observation seems to be unnoticed. Thus, our technique also extends a modularity result on the termination of the direct sum of TRSs in [Mid89].

The rest of this paper is organized as follows. In section 2, we review basic notions in term rewriting used in this paper. In section 3, we introduce tree lifting orderings and show that it is a reduction order. In section 4, we show how tree lifting orderings can be used to resolve the termination of a TRS into that of another, and give some examples of transformation.

2 Preliminaries

Let $\mathcal{V}$ be a set of countably infinite variables (denoted by $x, y, z, \ldots$) and $\mathcal{F}$ a set of arity-fixed function symbols (denoted by $f, g, h, \ldots$). Then the set of *terms* (denoted by $s, t, u, \ldots$) built from $\mathcal{F}$ and $\mathcal{V}$ is written as $\mathcal{T}(\mathcal{F}, \mathcal{V})$, or as just $\mathcal{T}$ when $\mathcal{F}$ is obvious from the context. $\mathcal{F}(t)$ and $\mathcal{V}(t)$ are the sets of function symbols and variables that appear in t, respectively. We denote by $\equiv$ the syntactical equality.

Suppose $\Box$ is a new function symbol—we call it a *hole*—of arity 0. The set $\mathcal{C}(\mathcal{F}, \mathcal{V})$ of *contexts* is a set $\mathcal{T}(\mathcal{F} \cup \{\Box\}, \mathcal{V})$. If $\mathcal{F}$ is obvious from a context, we abbreviate $\mathcal{C}(\mathcal{F}, \mathcal{V})$ to $\mathcal{C}$. If a context C has n holes in it and $t_1, \ldots, t_n \in \mathcal{T}$ then $C[t_1, \ldots, t_n]$ is the term obtained from C by replacing holes with $t_1, \ldots, t_n$ from left to right. A term s is a *subterm* of a term t (denoted by $s \trianglelefteq t$) if $t \equiv C[s]$ for some context C with precisely one hole; it is a *proper* subterm (denoted by $s \triangleleft t$) if $s \not\equiv t$.

A *substitution* σ on $\mathcal{T}$ is a mapping from $\mathcal{V}$ to $\mathcal{T}$ with finite $\mathrm{dom}(\sigma) = \{x \in \mathcal{V} \mid \sigma(x) \not\equiv x\}$. A substitution σ is extended to a homomorphism from $\mathcal{T}$ to $\mathcal{T}$. For a substitution σ and a term t, we conventionally write $t\sigma$ instead of $\sigma(t)$.

Given a binary relation R on a set W, the reflexive transitive closure and transitive closure of R are denoted by R^* and R^+, respectively. A binary relation R on $\mathcal{T}$ is said to be *closed under contexts* if sRt implies $C[s]RC[t]$ for any terms s, t and any context C; it is *closed under substitutions* if sRt implies $(s\sigma)R(t\sigma)$ for any terms s, t and any substitution σ. A *rewrite relation* is a binary relation on $\mathcal{T}$ closed under contexts and substitutions. A *strict partial order* $\succ$ on a set W is a transitive irreflexive relation on W; it is *well-founded* if there is no infinite sequence $a_1 \succ a_2 \succ \cdots$ of elements

from W. A *rewrite order* is a rewrite relation that is a strict partial order. A rewrite order is a *reduction order* if it is well-founded.

A pair $\langle l, r\rangle$ of terms is called a *rewrite rule*. We conventionally write $l \to r$ instead of $\langle l, r\rangle$. A rewrite rule $l \to r$ is *collapsing* if $r \in \mathcal{V}$; it is *duplicating* if r contains more occurrences of a variable than l does. A *term rewriting system* is a set of rewrite rules[1]. A TRS is said to be *non-collapsing* (*non-duplicating*) if it contains no collapsing (duplicating, respectively) rules. Given a TRS $\mathcal{R}$, we denote by $\to_{\mathcal{R}}$ the smallest rewrite relation on $\mathcal{T}$ that includes $\mathcal{R}$. A TRS $\mathcal{R}$ is *compatible* with a rewrite order $\succ$ if $l \succ r$ for any $l \to r \in \mathcal{R}$. A TRS $\mathcal{R}$ is said to be *terminating* if it is compatible with a reduction order.

We denote by $\mathrm{Mult}(W)$ the set of finite multisets of elements taken from a set W. The *multiset sum* and *multiset minus* are denoted by $+$ and $\setminus$, respectively; e.g. $[a, b, b, c] + [b, c, c, d] = [a, b, b, b, c, c, c, d]$, and $[a, b, b, c] \setminus [b, c, c, d] = [a, b]$.

Given a strict partial order $>$ on a set W, the *multiset ordering* $\gg$ *on* $\mathrm{Mult}(W)$ *induced from* $>$ is defined as follows:

$$[a_1, \ldots, a_n] \gg [b_1, \ldots, b_m]$$

if and only if

1. $[a_1, \ldots, a_n] \setminus [b_1, \ldots, b_m]$ is non-empty; and
2. for any $b \in [b_1, \ldots, b_m] \setminus [a_1, \ldots, a_n]$, there exists $a \in [a_1, \ldots, a_n] \setminus [b_1, \ldots, b_m]$ with $a > b$.

Clearly, $\gg$ is a strict partial order. The next proposition, which will be often used later, follows immediately from the definition.

PROPOSITION 2.1. Suppose $[a_1, \ldots, a_n] \gg [b_1, \ldots, b_m]$. Then, for any b_i there exists a_j such that $a_j \geq b_i$. Here, $a \geq b$ if and only if $a = b$ or $a > b$.

The following was proved in [DM79].

PROPOSITION 2.2. An order $>$ is well-founded on W if and only if the multiset ordering $\gg$ induced from $>$ is well-founded on $\mathrm{Mult}(W)$.

We denote by $\mathrm{Tr}(W)$ the set of finite trees whose nodes are labeled with elements taken from a set W. Formally, $a\langle\Gamma\rangle \in \mathrm{Tr}(W)$ if $a \in W$ and $\Gamma \in \mathrm{Mult}(\mathrm{Tr}(W))$. We abbreviate $a\langle[T_1, \ldots, T_n]\rangle$ to $a\langle T_1, \ldots, T_n\rangle$.

[1]We omit the usual restriction on TRSs. For, they are implied by the termination of TRSs.

A *subtrees* of a tree S is defined like this: (1) S is a subtree of S; (2) if $S = a\langle T_1, \ldots, T_n\rangle$ then subtrees of $T_1, \ldots, T_n$ are subtrees of S. A subtree of S different from S is a *proper subtree* of S.

Given a strict partial order $>$ on a set W, the *recursive path ordering* $>_{rpo}$ *on* $\mathrm{Tr}(W)$ *induced from* $>$ is defined as follows:

$$T = a\langle T_1, \ldots, T_n\rangle >_{rpo} b\langle S_1, \ldots, S_m\rangle = S$$

if and only if either

1. $T_i \geq_{rpo} S$ for some i;
2. $a > b$ and $T >_{rpo} S_i$ for all $i = 1, \ldots, m$; or
3. $a = b$ and $[T_1, \ldots, T_n] \gg_{rpo} [S_1, \ldots, S_m]$.

Here, $\gg_{rpo}$ is the multiset ordering induced from $>_{rpo}$. Clearly, $>_{rpo}$ is a strict partial order. The following is due to [Der82].

PROPOSITION 2.3. An order $>$ is well-founded on W if and only if the recursive path ordering $>_{rpo}$ induced from $>$ is well-founded on $\mathrm{Tr}(W)$.

3 Tree lifting orderings

In this section, let us fix sets $\mathcal{G}, \mathcal{F}$ of function symbols such that $\mathcal{G} \subseteq \mathcal{F}$, and a constant symbol $\Diamond$ such that $\Diamond \notin \mathcal{F}$.

A sequence of terms $s_1, \ldots, s_m$ is abbreviated to $\vec{s}$. If $\vec{s}$ is $s_1, \ldots, s_m$ and $\vec{t}$ is $t_1, \ldots, t_n$ then $\vec{s}, \vec{t}$ is a sequence $s_1, \ldots, s_m, t_1, \ldots, t_n$.

DEFINITION 3.1.

1. The *root* of a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ is defined by

$$\mathrm{root}(t) = \begin{cases} t & \text{if } t \in \mathcal{V}, \\ f & \text{if } t \equiv f(t_1, \ldots, t_n). \end{cases}$$

2. Let $t \equiv C[t_1, \ldots, t_n]$ $(n \geq 0)$ be a term in $\mathcal{T}(\mathcal{F}, \mathcal{V})$ with $\mathrm{root}(t) \in \mathcal{G}$. We write $t \equiv C[\![t_1, \ldots, t_n]\!]$ if $C \in \mathcal{C}(\mathcal{G}, \emptyset)$ and $\mathrm{root}(t_1), \ldots, \mathrm{root}(t_n) \notin \mathcal{G}$.

3. Let $t \equiv C[t_1, \ldots, t_n]$ $(n \geq 0)$ be a term in $\mathcal{T}(\mathcal{F}, \mathcal{V})$ such that $C \in \mathcal{C}(\mathcal{F}\backslash\mathcal{G}, \mathcal{V})$ and $\mathrm{root}(t_1), \ldots, \mathrm{root}(t_n) \in \mathcal{G}$. Suppose that $t_i \equiv C_i[\![\vec{s_i}]\!]$ for $i = 1, \ldots, n$. We denote by $\mathrm{suc}(t)$ a multiset $[\vec{s_1}, \ldots, \vec{s_n}]$, and by $\mathrm{cap}(t)$ a term $C[\Diamond, \ldots, \Diamond]$.

4. The *rank* of a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ is defined by

$$\mathrm{rank}(t) = \begin{cases} 1 & \text{if } \mathrm{suc}(t) = \emptyset, \\ 1 + \max\{\mathrm{rank}(t_i) \mid t_i \in \mathrm{suc}(t)\} & \text{otherwise.} \end{cases}$$

EXAMPLE 3.2. Let $\mathcal{G} = \{a, b, c\}$. (1) Let $t \equiv f(a(x), g(b(h(c))))$. Then $\mathrm{cap}(t) \equiv f(\Diamond, g(\Diamond))$ and $\mathrm{suc}(t) = [x, h(c)]$. The rank of t is 2. (2) Let $t \equiv f(a(x), f(a(c), z))$. Then $\mathrm{cap}(t) \equiv f(\Diamond, f(\Diamond, z))$ and $\mathrm{suc}(t) = [x]$. The rank of t is 2. (3) Let $t \equiv b(f(x, a(y)))$. Then $\mathrm{cap}(t) \equiv \Diamond$ and $\mathrm{suc}(t) = [f(x, a(y))]$. The rank of t is 3.

Given a reduction order $\succ$ on $\mathcal{T}(\mathcal{F} \backslash \mathcal{G} \cup \{\Diamond\}, \mathcal{V})$, the *tree lifting ordering* $\succ_{tlo(\mathcal{G})}$ *on* $\mathcal{T}(\mathcal{F}, \mathcal{V})$ *induced from* $\succ$ *and* $\mathcal{G}$ is defined as follows:

$$s \succ_{tlo(\mathcal{G})} t$$

if and only if either

TLO:1 $\mathrm{cap}(s) \succ \mathrm{cap}(t)$, and $s \succ_{tlo'(\mathcal{G})} v$ for any $v \in \mathrm{suc}(t)$, or

TLO:2 $\mathrm{cap}(s) \equiv \mathrm{cap}(t)$, and $\mathrm{suc}(s) \succ\!\succ_{tlo'(\mathcal{G})} \mathrm{suc}(t)$;

where

$$s \succ_{tlo'(\mathcal{G})} t$$

if and only if either

TLO':1 $\mathrm{cap}(s)(\succ \cup \rhd)^+ \mathrm{cap}(t)$, and $s \succ_{tlo'(\mathcal{G})} v$ for any $v \in \mathrm{suc}(t)$,

TLO':2 $u \succeq_{tlo'(\mathcal{G})} t$ for some $u \in \mathrm{suc}(s)$, or

TLO':3 $\mathrm{cap}(s) \equiv \mathrm{cap}(t)$, and $\mathrm{suc}(s) \succ\!\succ_{tlo'(\mathcal{G})} \mathrm{suc}(t)$.

Here, $\succeq_{tlo'(\mathcal{G})} = (\succ_{tlo'(\mathcal{G})} \cup \equiv)$ and $\succ\!\succ_{tlo'(\mathcal{G})}$ is the multiset ordering induced from $\succ_{tlo'(\mathcal{G})}$.

The following proposition easily follows from the definition.

PROPOSITION 3.3. Let $\succ$ be a reduction order on $\mathcal{T}(\mathcal{F} \backslash \mathcal{G} \cup \{\Diamond\}, \mathcal{V})$, and $\succ_{tlo(\mathcal{G})}$ and $\succ_{tlo'(\mathcal{G})}$ be as above. Then,

1. $\succ_{tlo(\mathcal{G})} \subseteq \succ_{tlo'(\mathcal{G})}$, and
2. $s \succ_{tlo'(\mathcal{G})} u$ for any $u \in \mathrm{suc}(s)$.

Our goal in this section is to show that $\succ_{tlo(\mathcal{G})}$ is a reduction order. For this purpose, we furthermore fix a reduction order $\succ$ on $\mathcal{T}(\mathcal{F}\backslash\mathcal{G} \cup \{\Diamond\}, \mathcal{V})$, hence $\succ_{tlo(\mathcal{G})}$ and $\succ_{tlo'(\mathcal{G})}$. We first aim to show that $\succ_{tlo(\mathcal{G})}$ is a well-founded strict partial order (LEMMA 3.7).

DEFINITION 3.4. For any term $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$, put

$$tree(t) = \begin{cases} t\langle\rangle & \text{if } t \in \mathcal{T}(\mathcal{F}\backslash\mathcal{G}, \mathcal{V}), \\ \mathrm{cap}(t)\langle tree(t_1), \ldots, tree(t_n)\rangle & \\ & \text{if } \mathrm{suc}(t) = [t_1, \ldots, t_n] \text{ with } n \geq 1. \end{cases}$$

LEMMA 3.5. The relation $\succ_{tlo'(\mathcal{G})}$ is a well-founded strict partial order.

Proof. Let $> = (\succ \cup \rhd)^+$. Then it is readily checked that $>$ is a well-founded strict partial order and that $s \succ_{tlo'(\mathcal{G})} t$ if and only if $tree(s) >_{rpo} tree(t)$. Thus, $\succ_{tlo'(\mathcal{G})}$ is a well-founded strict partial order. ■

By PROPOSITION 2.2, $\succ\succ_{tlo'(\mathcal{G})}$ is also a well-founded strict partial order. In the sequel, we will use this fact and LEMMA 3.5 without explicit mentioning.

LEMMA 3.6. The relation $\succ_{tlo(\mathcal{G})}$ is transitive.

Proof. Suppose $s \succ_{tlo(\mathcal{G})} t \succ_{tlo(\mathcal{G})} u$. We show that $s \succ_{tlo(\mathcal{G})} u$. We distinguish four cases:

1. $s \succ_{tlo(\mathcal{G})} t$ by **TLO:1** and $t \succ_{tlo(\mathcal{G})} u$ by **TLO:1**. Then $\mathrm{cap}(s) \succ \mathrm{cap}(t) \succ \mathrm{cap}(u)$, so $\mathrm{cap}(s) \succ \mathrm{cap}(u)$ by the transitivity of $\succ$. By PROPOSITION 3.3 we have $s \succ_{tlo'(\mathcal{G})} t$, and by assumption $t \succ_{tlo'(\mathcal{G})} u'$ for any $u' \in \mathrm{suc}(u)$. Therefore, we have $s \succ_{tlo'(\mathcal{G})} t \succ_{tlo'(\mathcal{G})} u'$ for any $u' \in \mathrm{suc}(u)$. Thus, $s \succ_{tlo(\mathcal{G})} u$ by **TLO:1**.

2. $s \succ_{tlo(\mathcal{G})} t$ by **TLO:1** and $t \succ_{tlo(\mathcal{G})} u$ by **TLO:2**. Then $\mathrm{cap}(s) \succ \mathrm{cap}(t) \equiv \mathrm{cap}(u)$, so $\mathrm{cap}(s) \succ \mathrm{cap}(u)$. Since $\mathrm{suc}(t) \succ\succ_{tlo'(\mathcal{G})} \mathrm{suc}(u)$, it holds that for any $u' \in \mathrm{suc}(u)$ there exists $t' \in \mathrm{suc}(t)$ such that $t' \succeq_{tlo'(\mathcal{G})} u'$ by PROPOSITION 2.1. Since $s \succ_{tlo'(\mathcal{G})} t'$ for any $t' \in \mathrm{suc}(t)$ by assumption, we have $s \succ_{tlo'(\mathcal{G})} t' \succeq_{tlo'(\mathcal{G})} u'$ for any $u' \in \mathrm{suc}(u)$. Thus, $s \succ_{tlo(\mathcal{G})} u$ by **TLO:1**.

3. $s \succ_{tlo(\mathcal{G})} t$ by **TLO:2** and $t \succ_{tlo(\mathcal{G})} u$ by **TLO:1**. Then $\mathrm{cap}(s) \equiv \mathrm{cap}(t) \succ \mathrm{cap}(u)$, so $\mathrm{cap}(s) \succ \mathrm{cap}(u)$. By PROPOSITION 3.3 we have $s \succ_{tlo'(\mathcal{G})} t$, and by assumption $t \succ_{tlo'(\mathcal{G})} u'$ for any $u' \in \mathrm{suc}(u)$. Therefore, we have $s \succ_{tlo'(\mathcal{G})} t \succ_{tlo'(\mathcal{G})} u'$ for any $u' \in \mathrm{suc}(u)$. Thus, $s \succ_{tlo(\mathcal{G})} u$ by **TLO:1**.

4. $s \succ_{tlo(\mathcal{G})} t$ by **TLO:2** and $t \succ_{tlo(\mathcal{G})} u$ by **TLO:2**. Then $\mathrm{cap}(s) \equiv \mathrm{cap}(t) \equiv \mathrm{cap}(u)$ and $\mathrm{suc}(s) \succ\!\succ_{tlo'(\mathcal{G})} \mathrm{suc}(t) \succ\!\succ_{tlo'(\mathcal{G})} \mathrm{suc}(u)$. Therefore, $s \succ_{tlo(\mathcal{G})} u$ by **TLO:2**.

■

LEMMA 3.7. The relation $\succ_{tlo(\mathcal{G})}$ is a well-founded strict partial order.

Proof. By LEMMA 3.6, it suffices to show $\succ_{tlo(\mathcal{G})}$ is well-founded (because irreflexivity follows from the well-foundedness). Since $\succ_{tlo(\mathcal{G})} \subseteq \succ_{tlo'(\mathcal{G})}$ (PROPOSITION 3.3), $\succ_{tlo(\mathcal{G})}$ is well-founded by LEMMA 3.5. ■

Our next aim is to show that $\succ_{tlo(\mathcal{G})}$ is closed under contexts (LEMMA 3.9).

LEMMA 3.8. For any $C \in \mathcal{C}(\mathcal{F}\backslash\mathcal{G}, \mathcal{V})$, if $s \succ_{tlo'(\mathcal{G})} t$ then $C[s] \succ_{tlo'(\mathcal{G})} t$.

Proof. Suppose that $s \succ_{tlo'(\mathcal{G})} t$. First, note that $\mathrm{cap}(C[u]) = C[\mathrm{cap}(u)]$ and $\mathrm{suc}(C[u]) = \mathrm{suc}(u)$ for any $u \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ and $C \in \mathcal{C}(\mathcal{F}\backslash\mathcal{G}, \mathcal{V})$. Our proof proceeds by induction on $\mathrm{rank}(s) + \mathrm{rank}(t)$. Base step is trivial. For the induction step, we distinguish three cases:

1. $s \succ_{tlo'(\mathcal{G})} t$ by **TLO':1**. Then $\mathrm{cap}(s)(\succ \cup \rhd)^+ \mathrm{cap}(t)$ and $s \succ_{tlo'(\mathcal{G})} u$ for any $u \in \mathrm{suc}(t)$. Since $\mathrm{cap}(C[s]) \equiv C[\mathrm{cap}(s)] \trianglerighteq \mathrm{cap}(s)(\succ \cup \rhd)^+ \mathrm{cap}(t)$, we have $\mathrm{cap}(C[s])(\succ \cup \rhd)^+ \mathrm{cap}(t)$. Also, by the induction hypothesis, $C[s] \succ_{tlo'(\mathcal{G})} u$ for any $u \in \mathrm{suc}(t)$. Therefore, $C[s] \succ_{tlo'(\mathcal{G})} t$ by **TLO':1**.

2. $s \succ_{tlo'(\mathcal{G})} t$ by **TLO':2**. Then $u \succeq_{tlo'(\mathcal{G})} t$ for some $u \in \mathrm{suc}(s)$. Since $u \in \mathrm{suc}(s) = \mathrm{suc}(C[s])$, it holds that $C[s] \succ_{tlo'(\mathcal{G})} t$ by **TLO':2**.

3. $s \succ_{tlo'(\mathcal{G})} t$ by **TLO':3**. Then $\mathrm{cap}(s) \equiv \mathrm{cap}(t)$ and $\mathrm{suc}(s) \succ\!\succ_{tlo'(\mathcal{G})} \mathrm{suc}(t)$. If $C \equiv \Box$ then $C[s] \succ_{tlo'(\mathcal{G})} t$ trivially follows, so we suppose $C \not\equiv \Box$. Then, $\mathrm{cap}(C[s]) \equiv C[\mathrm{cap}(s)] \rhd \mathrm{cap}(s) \equiv \mathrm{cap}(t)$, so $\mathrm{cap}(C[s])(\succ \cup \rhd)^+ \mathrm{cap}(t)$. Since $\mathrm{suc}(s) \succ\!\succ_{tlo'(\mathcal{G})} \mathrm{suc}(t)$ and $\mathrm{suc}(s) = \mathrm{suc}(C[s])$, it holds that for any $t' \in \mathrm{suc}(t)$ there exists $s' \in \mathrm{suc}(C[s])$ such that $s' \succeq_{tlo'(\mathcal{G})} t'$ by PROPOSITION 2.1; hence $C[s] \succ_{tlo'(\mathcal{G})} s' \succeq_{tlo'(\mathcal{G})} t'$ by PROPOSITION 3.3. Therefore, $C[s] \succ_{tlo'(\mathcal{G})} t$ by **TLO':1**.

■

LEMMA 3.9. For any $C \in \mathcal{C}(\mathcal{F}, \mathcal{V})$, if $s \succ_{tlo(\mathcal{G})} t$ then $C[s] \succ_{tlo(\mathcal{G})} C[t]$.

Proof. First, we prove the following claim: for any $C \in \mathcal{C}(\mathcal{F}\backslash\mathcal{G}, \mathcal{V})$, if $s \succ_{tlo(\mathcal{G})} t$ then $C[s] \succ_{tlo(\mathcal{G})} C[t]$. Suppose that $s \succ_{tlo(\mathcal{G})} t$. We distinguish two cases:

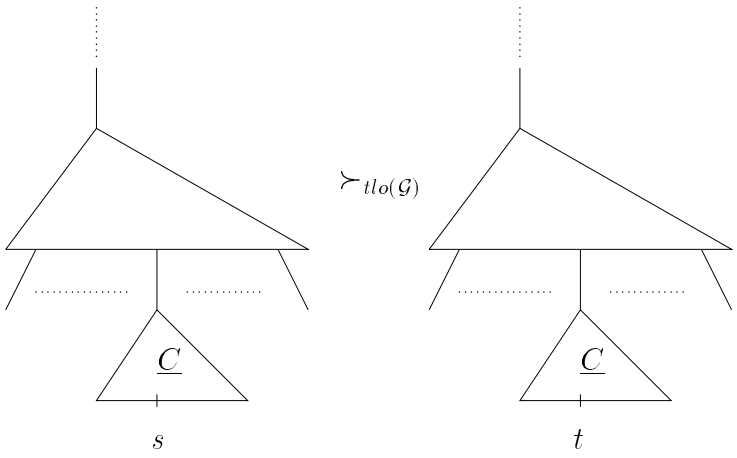

Figure 4: A meta-tree and a context

1. $s \succ_{tlo(\mathcal{G})} t$ by **TLO:1**. Then $\mathrm{cap}(s) \succ \mathrm{cap}(t)$ and $s \succ_{tlo'(\mathcal{G})} v$ for any $v \in \mathrm{suc}(t)$. Since $\succ$ is a reduction order, we have $\mathrm{cap}(C[s]) \equiv C[\mathrm{cap}(s)] \succ C[\mathrm{cap}(t)] \equiv \mathrm{cap}(C[t])$. Also, since $\mathrm{suc}(t) = \mathrm{suc}(C[t])$, for any $v \in \mathrm{suc}(C[t])$, we have $s \succ_{tlo'(\mathcal{G})} v$, hence $C[s] \succ_{tlo'(\mathcal{G})} v$ by Lemma 3.8. Thus, $C[s] \succ_{tlo(\mathcal{G})} C[t]$ by **TLO:1**.

2. $s \succ_{tlo(\mathcal{G})} t$ by **TLO:2**. Then $\mathrm{cap}(C[s]) \equiv C[\mathrm{cap}(s)] \equiv C[\mathrm{cap}(t)] \equiv \mathrm{cap}(C[t])$ and $\mathrm{suc}(C[s]) = \mathrm{suc}(s) \succ\succ_{tlo'(\mathcal{G})} \mathrm{suc}(t) = \mathrm{suc}(C[t])$. Therefore, $C[s] \succ_{tlo(\mathcal{G})} C[t]$ by **TLO:2**. (The end of the proof of the claim.)

Now, let $C \notin \mathcal{C}(\mathcal{F}\backslash\mathcal{G}, \mathcal{V})$. Clearly, we have

$$C \equiv C'[a(\cdots \underline{C} \cdots)]$$

for some $\underline{C} \in \mathcal{C}(\mathcal{F}\backslash\mathcal{G}, \mathcal{V})$ and $C' \in \mathcal{C}(\mathcal{F}, \mathcal{V})$ (see Figure 4). We have $\underline{C}[s] \succ_{tlo(\mathcal{G})} \underline{C}[t]$ by the claim above. Then, using **TLO:2** and **TLO':3**, it is easy to obtain $C[s] \succ_{tlo(\mathcal{G})} C[t]$. ■

We are now going to show that $\succ_{tlo(\mathcal{G})}$ is closed under substitutions (Lemma 3.11). We require the following notations. Let σ be a substitution on $\mathcal{T}(\mathcal{F}, \mathcal{V})$ with $\mathrm{dom}(\sigma) = \{x_1, \ldots, x_n\}$. Suppose that for each $i = 1, \ldots, n$,

$$\sigma(x_i) \equiv C_{x_i}[C^1_{x_i}[\![\overrightarrow{t^{x_i}_1}]\!], \ldots, C^{k_i}_{x_i}[\![\overrightarrow{t^{x_i}_{k_i}}]\!]]$$

where $k_i \geq 0$, $C_{x_i} \in \mathcal{C}(\mathcal{F}\backslash\mathcal{G}, \mathcal{V})$ and $C^1_{x_i}, \ldots, C^{k_i}_{x_i} \in \mathcal{C}(\mathcal{G}, \emptyset)$. Let $\overline{\sigma} : \mathcal{V} \to \mathcal{T}(\mathcal{F}\backslash\mathcal{G} \cup \{\Diamond\}, \mathcal{V})$ and $\underline{\sigma} : \{x_1, \ldots, x_n\} \to \mathrm{Mult}(\mathcal{T}(\mathcal{F}, \mathcal{V}))$ be two mappings defined by

$$\overline{\sigma}(x) = \begin{cases} C_x[\Diamond, \ldots, \Diamond] & \text{if } x \in \{x_1, \ldots, x_n\}, \\ x & \text{otherwise,} \end{cases}$$

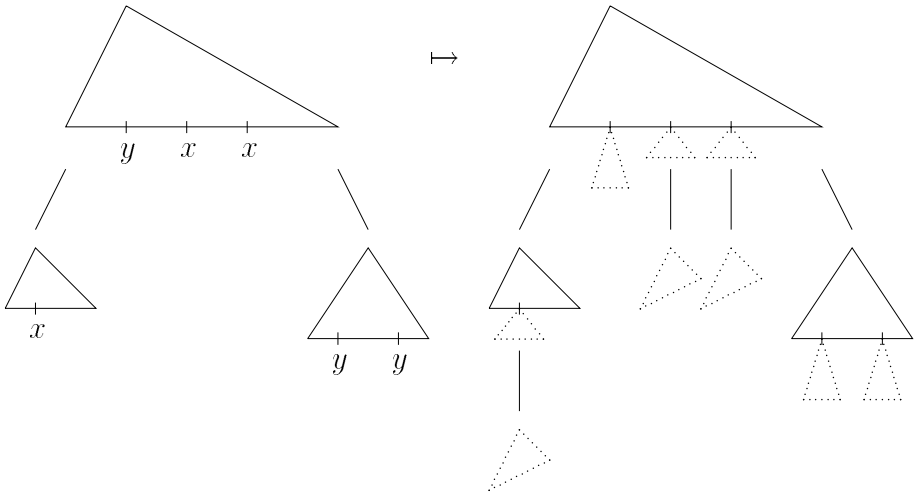

Figure 5: A meta-tree and a substitution

and

$$\underline{\sigma}(x_i) = [\overrightarrow{t_1^{x_i}}, \ldots, \overrightarrow{t_{k_i}^{x_i}}].$$

By the definition, one immediately gets that

$$\mathrm{cap}(s\sigma) \equiv \mathrm{cap}(s)\overline{\sigma} \tag{1}$$

$$\mathrm{suc}(s\sigma) = [u\sigma \mid u \in \mathrm{suc}(s)] + \Sigma_{x \in \mathrm{Var}(\mathrm{cap}(s))} x\underline{\sigma} \tag{2}$$

(see Figure 5). Here, $\mathrm{Var}(t)$ is the multiset of variables occurring in t.

LEMMA 3.10. For any substitution σ, if $s \succ_{tlo'(\mathcal{G})} t$ then $s\sigma \succ_{tlo'(\mathcal{G})} t\sigma$.

Proof. Firstly, suppose θ is a substitution such that $\theta(x) \in \mathcal{T}(\mathcal{F}\backslash\mathcal{G}, \mathcal{V})$ for any $x \in \mathrm{dom}(\theta)$. Since $\succ$ is a reduction order, $u \succ v$ and $x \in \mathcal{V}(v)$ imply $x \in \mathcal{V}(u)$. Therefore, one easily shows $u(\succ \cup \rhd)^+ v$ implies $u\theta(\succ \cup \rhd)^+ v\theta$ for any $u, v \in \mathcal{T}(\mathcal{F}\backslash\mathcal{G} \cup \{\Diamond\}, \mathcal{V})$; hence, by induction on $\mathrm{rank}(s') + \mathrm{rank}(t')$, $s' \succ_{tlo'(\mathcal{G})} t'$ implies $s'\theta \succ_{tlo'(\mathcal{G})} t'\theta$ for any $s', t' \in \mathcal{T}(\mathcal{F}, \mathcal{V})$.

Now, suppose that $s \succ_{tlo'(\mathcal{G})} t$ and let σ be an arbitrary substitution. Our proof proceeds by induction on $\mathrm{rank}(s) + \mathrm{rank}(t)$.

For the base step, suppose that $\mathrm{rank}(s) = \mathrm{rank}(t) = 1$. Then $s \succ_{tlo'(\mathcal{G})} t$ is due to **TLO':1**, so we have $\mathrm{cap}(s\sigma) \equiv \mathrm{cap}(s)\overline{\sigma}(\succ \cup \rhd)^+ \mathrm{cap}(s)\overline{\sigma} \equiv \mathrm{cap}(s\sigma)$. Furthermore, from this follows $\mathrm{suc}(s\sigma) = \Sigma_{x \in \mathrm{Var}(\mathrm{cap}(s\sigma))} x\underline{\sigma} = \Sigma_{x \in \mathrm{Var}(\mathrm{cap}(t\sigma))} x\underline{\sigma} = \mathrm{suc}(t\sigma)$, because $\mathrm{suc}(s) = \mathrm{suc}(t) = \emptyset$. Hence, $s\sigma \succ_{tlo'(\mathcal{G})} v$ for any $v \in \mathrm{suc}(t\sigma)$ by PROPOSITION 3.3. Thus, $s\sigma \succ_{tlo'(\mathcal{G})} t\sigma$ by **TLO':1**.

For the induction step, we distinguish three cases:

1. $s \succ_{tlo'(\mathcal{G})} t$ by **TLO':1**. Then $\mathrm{cap}(s)(\succ \cup \rhd)^+ \mathrm{cap}(t)$ and $s \succ_{tlo'(\mathcal{G})} v$ for any $v \in \mathrm{suc}(t)$. Hence, we have $\mathrm{cap}(s\sigma) \equiv \mathrm{cap}(s)\overline{\sigma}(\succ \cup \rhd)^+ \mathrm{cap}(t)\overline{\sigma} \equiv$

$\mathrm{cap}(t\sigma)$. Let $v \in \mathrm{suc}(t\sigma)$. To show $s\sigma \succ_{tlo'(\mathcal{G})} v$, we distinguish two cases:

(a) $v \equiv v'\sigma$ for some $v' \in \mathrm{suc}(t)$. Then, since $s \succ_{tlo'(\mathcal{G})} v'$, we have $s\sigma \succ_{tlo'(\mathcal{G})} v'\sigma \equiv v$ by the induction hypothesis.

(b) $v \in \underline{\sigma}(x)$ for some $x \in \mathcal{V}(\mathrm{cap}(t))$. Then, since $\mathrm{cap}(s)(\succ \cup \triangleright)^+ \mathrm{cap}(t)$, we have $x \in \mathcal{V}(\mathrm{cap}(s))$; hence $v \in \underline{\sigma}(x) \subseteq \mathrm{suc}(s\sigma)$. Therefore, $s\sigma \succ_{tlo'(\mathcal{G})} v$ by Proposition 3.3.

Thus, $s\sigma \succ_{tlo'(\mathcal{G})} t\sigma$ by **TLO':1**.

2. $s \succ_{tlo'(\mathcal{G})} t$ by **TLO':2**. Then $u \succeq_{tlo'(\mathcal{G})} t$ for some $u \in \mathrm{suc}(s)$. Then, by induction hypothesis, $u\sigma \succeq_{tlo'(\mathcal{G})} t\sigma$. Since $u\sigma \in \mathrm{suc}(s\sigma)$, we have $s\sigma \succ_{tlo'(\mathcal{G})} t\sigma$ by **TLO':2**.

3. $s \succ_{tlo'(\mathcal{G})} t$ by **TLO':3**. Then $\mathrm{cap}(s) \equiv \mathrm{cap}(t)$ and $\mathrm{suc}(s) \succ\succ_{tlo'(\mathcal{G})} \mathrm{suc}(t)$. Hence, we have $\mathrm{cap}(s\sigma) \equiv \mathrm{cap}(s)\overline{\sigma} \equiv \mathrm{cap}(t)\overline{\sigma} \equiv \mathrm{cap}(t\sigma)$. Also, by induction hypothesis, $[u\sigma \mid u \in \mathrm{suc}(s)] \succ\succ_{tlo'(\mathcal{G})} [v\sigma \mid v \in \mathrm{suc}(t)]$. Therefore, $\mathrm{suc}(s\sigma) = [u\sigma \mid u \in \mathrm{suc}(s)] + \Sigma_{x \in \mathrm{Var}(\mathrm{cap}(s))} x\underline{\sigma} \succ\succ_{tlo'(\mathcal{G})} [v\sigma \mid v \in \mathrm{suc}(t)] + \Sigma_{x \in \mathrm{Var}(\mathrm{cap}(t))} x\underline{\sigma} = \mathrm{suc}(t\sigma)$. Thus, $s\sigma \succ_{tlo'(\mathcal{G})} t\sigma$ by **TLO':3**.

■

Lemma 3.11. For any substitution σ, if $s \succ_{tlo(\mathcal{G})} t$ then $s\sigma \succ_{tlo(\mathcal{G})} t\sigma$.

Proof. We distinguish two cases:

1. $s \succ_{tlo(\mathcal{G})} t$ by **TLO:1**. Then $\mathrm{cap}(s) \succ \mathrm{cap}(t)$ and $s \succ_{tlo'(\mathcal{G})} v$ for any $v \in \mathrm{suc}(t)$. Hence, $\mathrm{cap}(s\sigma) \equiv \mathrm{cap}(s)\overline{\sigma} \succ \mathrm{cap}(t)\overline{\sigma} \equiv \mathrm{cap}(t\sigma)$. Let $v \in \mathrm{suc}(t\sigma)$. To show $s\sigma \succ_{tlo'(\mathcal{G})} v$, we distinguish two cases:

   (a) $v \equiv v'\sigma$ for some $v' \in \mathrm{suc}(t)$. Since $s \succ_{tlo'(\mathcal{G})} v'$, we have $s\sigma \succ_{tlo'(\mathcal{G})} v'\sigma \equiv v$ by Lemma 3.10.

   (b) $v \in \underline{\sigma}(x)$ for some $x \in \mathcal{V}(\mathrm{cap}(t))$. Since $\mathrm{cap}(s) \succ \mathrm{cap}(t)$, we have $x \in \mathcal{V}(\mathrm{cap}(s))$; hence $v \in \underline{\sigma}(x) \subseteq \mathrm{suc}(s\sigma)$. Thus, $s\sigma \succ_{tlo'(\mathcal{G})} v$ by Proposition 3.3.

   Therefore, $s\sigma \succ_{tlo(\mathcal{G})} t\sigma$ by **TLO:1**.

2. $s \succ_{tlo(\mathcal{G})} t$ by **TLO:2**. Then $\mathrm{cap}(s) \equiv \mathrm{cap}(t)$ and $\mathrm{suc}(s) \succ\succ_{tlo'(\mathcal{G})} \mathrm{suc}(t)$. Therefore, $\mathrm{cap}(s\sigma) \equiv \mathrm{cap}(s)\overline{\sigma} \equiv \mathrm{cap}(t)\overline{\sigma} \equiv \mathrm{cap}(t\sigma)$. Also, it holds $\mathrm{suc}(s\sigma) \succ\succ_{tlo'(\mathcal{G})} \mathrm{suc}(t\sigma)$ by Lemma 3.10. Thus, $s\sigma \succ_{tlo(\mathcal{G})} t\sigma$ by **TLO:2**.

■

Combining LEMMA 3.7, LEMMA 3.9 and LEMMA 3.11, we arrive at

THEOREM 3.12. Given a reduction order $\succ$ on $\mathcal{T}(\mathcal{F}\backslash\mathcal{G} \cup \{\Diamond\}, \mathcal{V})$, the tree lifting ordering $\succ_{tlo(\mathcal{G})}$ induced from $\succ$ and $\mathcal{G}$ is a reduction order on $\mathcal{T}(\mathcal{F}, \mathcal{V})$.

4 Termination transformations

Tree lifting orderings can be used to show the termination of TRSs. We are now going to show how we can use tree lifting orderings to resolve the termination of a TRS to the well-foundedness of a rewrite order (or, to the termination of another TRS).

To this end, we define an equivalence relation $\simeq_{tlo(\mathcal{G})}$ on $\mathcal{T}(\mathcal{F}, \mathcal{V})$ like this:

$$s \simeq_{tlo(\mathcal{G})} t \text{ if and only if } tree(s) = tree(t).$$

Then, it is readily checked that (1) $\succ_{tlo(\mathcal{G})} \cap \simeq_{tlo(\mathcal{G})} = \emptyset$, and (2) $s \simeq_{tlo(\mathcal{G})} s' \succ_{tlo(\mathcal{G})} t' \simeq_{tlo(\mathcal{G})} t$ implies $s \succ_{tlo(\mathcal{G})} t$. Henceforth, $\simeq_{tlo(\mathcal{G})} \cup \succ_{tlo(\mathcal{G})}$ is denoted by $\succeq_{tlo(\mathcal{G})}$. Clearly, $\succeq_{tlo(\mathcal{G})}$ is a partial order (i.e. reflexive transitive antisymmetric relation) with $\simeq_{tlo(\mathcal{G})}$ as its equivalence relation and $\succ_{tlo(\mathcal{G})}$ as its strict part (i.e. $s \succeq_{tlo(\mathcal{G})} t$ and $t \succeq_{tlo(\mathcal{G})} s$ imply $s \simeq_{tlo(\mathcal{G})} t$, and $s \succeq_{tlo(\mathcal{G})} t$ and $t \not\preceq_{tlo(\mathcal{G})} s$ imply $s \succ_{tlo(\mathcal{G})} t$).

THEOREM 4.1. Let $\mathcal{R}$ be a TRS and $\mathcal{G}$ a set of function symbols. If there exists a reduction order $\succ$ on $\mathcal{T}(\mathcal{F}\backslash\mathcal{G} \cup \{\Diamond\}, \mathcal{V})$ such that $l \succ_{tlo(\mathcal{G})} r$ for any $l \to r \in \mathcal{R}$, then $\mathcal{S} \cup \mathcal{R}$ is terminating for any non-collapsing non-duplicating terminating TRS $\mathcal{S}$ whose function symbols are contained in $\mathcal{G}$.

Proof. Note that $t \to_{\mathcal{S}} s$ implies $t \succeq_{tlo(\mathcal{G})} s$, since $\mathcal{S}$ is non-collapsing and non-duplicating.

Suppose there exists an infinite reduction sequence of $\mathcal{S} \cup \mathcal{R}$. Since $\mathcal{S}$ is terminating, the sequence has the form $t_0 \xrightarrow{*}_{\mathcal{S}} s_0 \to_{\mathcal{R}} t_1 \xrightarrow{*}_{\mathcal{S}} s_1 \to_{\mathcal{R}} t_2 \xrightarrow{*}_{\mathcal{S}} s_2 \to_{\mathcal{R}} \cdots$. Then we have $t_0 \succeq_{tlo(\mathcal{G})} s_0 \succ_{tlo(\mathcal{G})} t_1 \succeq_{tlo(\mathcal{G})} s_1 \succ_{tlo(\mathcal{G})} t_2 \succeq_{tlo(\mathcal{G})} s_2 \succ_{tlo(\mathcal{G})} \cdots$, and so $t_0 \succ_{tlo(\mathcal{G})} t_1 \succ_{tlo(\mathcal{G})} t_2 \succ_{tlo(\mathcal{G})} \cdots$. This contradicts THEOREM 3.12. ■

Now, we claim that the correctness of the dummy elimination transformation [FZ95] also follows from our result. For the definition of $\mathcal{E}(\mathcal{R})$, we refer the reader to the section 1.

COROLLARY 4.2. For a TRS $\mathcal{R}$, if $\mathcal{E}(\mathcal{R})$ is terminating TRS then $\mathcal{R}$ is terminating.

Proof. Let $\succ = \xrightarrow{+}_{\mathcal{E}(\mathcal{R})}$. It is easy to see $l \succ_{tlo(\mathcal{G})} r$ for all $l \to r \in \mathcal{R}$, by applying **TLO:1** and **TLO':1**. ■

REMARK 4.3. It should be remarked here that a much more simple proof for the correctness of the dummy elimination transformation appeared in [MOZ96].

It is clear that if no symbols from $\mathcal{G}$ appears in $\mathcal{R}$, for any reduction relation $\succ$ of $\mathcal{R}$ it holds that $l \succ_{tlo(\mathcal{G})} r$ for all $l \to r \in \mathcal{R}$. Hence, by THEOREM 4.1, we have

COROLLARY 4.4. Let $\mathcal{R}$ and $\mathcal{S}$ be terminating TRSs such that the sets of function symbols that appear in $\mathcal{R}$ and $\mathcal{S}$ are disjoint. Then, if $\mathcal{S}$ is non-collapsing non-duplicating, then $\mathcal{R} \cup \mathcal{S}$ is also terminating.

Thus a modularity result on the termination of the direct sum of TRSs [Mid89] also follows from our result.

Neither results in [Mid89] nor [FZ95] can be applied to the following example.

EXAMPLE 4.5. Let

$$\mathcal{R} \left\{ \begin{array}{ll} f(f(x)) \to f(a(b(f(x)))) & (r1) \\ f(a(g(x))) \to b(x) & (r2) \\ b(x) \to a(x) & (r3). \end{array} \right.$$

Take $\mathcal{G} = \{a, b\}$. Let

$$\hat{\mathcal{R}} \left\{ \begin{array}{l} f(f(x)) \to f(\Diamond) \\ f(\Diamond) \to \Diamond. \end{array} \right.$$

It is easy to see that $\hat{\mathcal{R}}$ is terminating (e.g. using the recursive path ordering). Thus $\xrightarrow{+}_{\hat{\mathcal{R}}}$ is a reduction order. Let $\succ = \xrightarrow{+}_{\hat{\mathcal{R}}}$. Then, one can readily check that $l \succ_{tlo(\mathcal{G})} r$ for any $l \to r \in \{(r1), (r2)\}$. Since $\{(r3)\}$ is a non-duplicating non-collapsing terminating TRS whose function symbols are contained in $\mathcal{G}$, we conclude that $\mathcal{R}$ is terminating by THEOREM 4.1.

REMARK 4.6. The termination of $\mathcal{R}$ in EXAMPLE 4.5 can be shown also by the semantic labelling [Zan95]. However, unlike the semantic labelling, it is decidable whether for a given finite TRS $\mathcal{R}$ our method can be used; that is, one can effectively get all possible candidates whose termination ensures that of $\mathcal{R}$ via our result, for the number of choices of $\mathcal{G}$ is finite.

EXAMPLE 4.7. Like EXAMPLE 4.5, the termination of each TRS $\mathcal{R}_i$ in EXAMPLE 1.3 can be shown using the termination of the following TRS $\mathcal{R}'_i$ ($i \in \{1,2,3\}$).

$$\mathcal{R}'_1 \left\{ \begin{array}{l} f(f(x)) \to f(\diamondsuit) \\ f(\diamondsuit) \to \diamondsuit \end{array} \right.$$

$$\mathcal{R}'_2 \left\{ \begin{array}{l} g(g(x)) \to g(\diamondsuit) \end{array} \right.$$

$$\mathcal{R}'_3 \left\{ \begin{array}{l} f(f(\diamondsuit)) \to f(\diamondsuit) \end{array} \right.$$

One can introduce k-ary $\diamondsuit$ instead of unary $\diamondsuit$ so that some arguments of dummy symbols remain undetached: For example, let $k = 1$ and define $\mathrm{cap}(C[a(s_1, s'_1), \ldots, a(s_k, s'_k)]) \equiv C[\diamondsuit(\mathrm{cap}(s_1)), \ldots, \diamondsuit(\mathrm{cap}(s_k))]$ for C that contains no a (see Figure 6). Now, it is straightforward to generalize

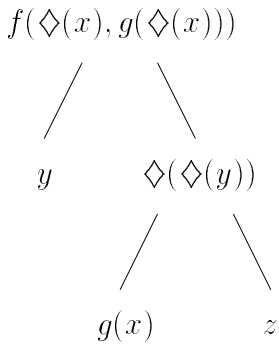

Figure 6: The meta-tree of $f(a(x,y), g(a(x, a(a(y, g(x)), z))))$

THEOREM 3.12 and THEOREM 4.1 for this definition with $\mathcal{S} = \emptyset$.

EXAMPLE 4.8. Let

$$\mathcal{R} \left\{ \begin{array}{l} f(g(x)) \to f(a(x, g(x))) \\ a(x,y) \to x \end{array} \right. \qquad \hat{\mathcal{R}} \left\{ \begin{array}{l} f(g(x)) \to f(\diamondsuit(x)) \\ \diamondsuit(x) \to x \end{array} \right.$$

It is easy to see that $\hat{\mathcal{R}}$ is terminating (e.g. using the recursive path ordering). Thus $\xrightarrow{+}_{\hat{\mathcal{R}}}$ is a reduction order. Let $\succ = \xrightarrow{+}_{\hat{\mathcal{R}}}$. One can readily check that $l \succ_{tlo(\mathcal{G})} r$ for any $l \to r \in \mathcal{R}$. Hence, $\mathcal{R}$ is terminating.

5 Concluding remarks

We have proposed the tree lifting ordering $\succ_{tlo(\mathcal{G})}$ induced from $\succ$ and $\mathcal{G}$, for a set $\mathcal{G} \subseteq \mathcal{F}$ of function symbols and a reduction order $\succ$ on $\mathcal{T}(\mathcal{F} \backslash \mathcal{G} \cup$

$\{\Diamond\}, \mathcal{V})$, and showed that $\succ_{tlo(\mathcal{G})}$ is a reduction order. It was proved that for a TRS $\mathcal{R}$ and a set $\mathcal{G}$ of function symbols, if there exists a reduction order $\succ$ on $\mathcal{T}(\mathcal{F}\backslash\mathcal{G} \cup \{\Diamond\}, \mathcal{V})$ such that $l \succ_{tlo(\mathcal{G})} r$ for any $l \to r \in \mathcal{R}$, then $\mathcal{S} \cup \mathcal{R}$ is terminating for any non-collapsing non-duplicating terminating TRS $\mathcal{S}$ whose function symbols are contained in $\mathcal{G}$. We have shown how this result can be used to infer the termination of a TRS from that of another. Also, it was shown that our technique covers the dummy elimination transformation [FZ95] and a modularity result on the termination of the direct sum of TRSs [Mid89].

References

[Der82] N. Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17:279–301, 1982.

[Der87] N. Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, 3:69–116, 1987.

[DM79] N. Dershowitz and Z. Manna. Proving termination with multiset orderings. *Communications of the ACM*, 22(8):465–476, 1979.

[Fer95] M. C. F. Ferreira. *Termination of term rewriting.* PhD thesis, Utrecht University, 1995.

[FZ95] M. C. F. Ferreira and H. Zantema. Dummy elimination: making termination easier. In *Proc. of the 10th International Conference on Foundation of Computation Theory, Lecture Notes in Computer Science 252*, pages 252–243, 1995.

[Mid89] A. Middeldorp. A sufficient condition for the termination of the direct sum of term rewriting systems. In *Proc. of the 4th annual IEEE Symposium on Login in Computer Science*, pages 396–401, 1989.

[MOZ96] A. Middeldorp, H. Ohsaki, and H. Zantema. Transforming termination by self-labelling. In *Proc. of the 13th International Conference on Automated Deduction, Lecture Notes in Artificial Intelligence 1104*, pages 373–387, 1996.

[MZ97] A. Middeldorp and H. Zantema. Simple termination of rewrite systems. *Theoretical Computer Science*, 175(1):127–158, 1997.

[Zan94] H. Zantema. Termination of term rewriting: interpretation and type elimination. *Journal of Symbolic Computation*, 17:23–50, 1994.

[Zan95] H. Zantema. Termination of term rewriting by semantic labelling. *Fundamenta Informaticae*, 24:89–105, 1995.