NAIST-IS-MT0051093

Master's Thesis

Layered Transducing Term Rewriting Systems and Its Recognizability Preserving Property

Yohei Fujinaka

February 8, 2002

Department of Information Processing
Graduate School of Information Science
Nara Institute of Science and Technology

Master's Thesis
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
MASTER of ENGINEERING

Yohei Fujinaka

Thesis Committee: Hiroyuki Seki, Professor
Minoru Ito, Professor
Yuichi Kaji, Associate Professor

Layered Transducing Term Rewriting Systems and Its Recognizability Preserving Property*

Yohei Fujinaka

Abstract

In this thesis, a new subclass of term rewriting systems (TRSs), layered transducing TRSs (LT-TRSs) is defined and its recognizability preserving property is discussed. A set of ground terms (tree language) T is recognizable if there exists a tree automaton which accepts T. A TRS $\mathcal{R}$ effectively preserves recognizability (EPR) if for any recognizable tree language $\mathcal{L}$, the set of terms obtained by rewriting terms in $\mathcal{L}$ by $\mathcal{R}$ is also recognizable and a tree automaton which accepts it can be effectively constructed. EPR-TRS has good mathematical properties, e.g., reachability, joinability and local confluence are decidable for EPR-TRSs. Unfortunately it is undecidable whether a given TRS belongs to EPR-TRS or not. Therefore decidable subclasses of EPR-TRS have been proposed. These subclasses put a rather strong constraint on the syntax of the right-hand side of a rewrite rule. The class of LT-TRSs contains some EPR-TRSs, e.g., $\{f(x) \to f(g(x))\}$ which do not belong to any of the known decidable subclasses of EPR-TRSs. Bottom-up linear tree transducer, which is a well-known computation model in the tree language theory, is a special case of LT-TRS. In this thesis, two sufficient conditions for an LT-TRS to be an EPR-TRS are presented. Also some properties of LT-TRSs including reachability are shown to be decidable.

Keywords:

Term Rewriting System(TRS), Tree Automaton, Recognizability, Recognizability Preserving Property, Layered Transducing TRS

* Master's Thesis, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-MT0051093, February 8, 2002.

多層変換項書換え系とその正則保存性について*

藤中 洋平

内容梗概

本論文では, 項書換え系の部分クラスとして, 多層変換項書換え系を導入する. 項の集合 T が正則であるとは, T を受理する木オートマトンが存在することである. 項の集合を木言語ともいう. 項書換え系 $\mathcal{R}$ が構成的正則保存であるとは, 任意の正則木言語 $\mathcal{L}$ に対して, $\mathcal{L}$ から $\mathcal{R}$ によって書換えて得られる項全体からなる木言語が正則であり, それを受理する木オートマトンを構成できることをいう. 構成的正則保存項書換え系においては, 到達可能性や会同性, 局所合流性などの重要な性質が決定可能になることが知られている. しかしながら, 与えられた項書換え系が構成的正則保存であるかどうかは決定不能である. これまでにいくつかの決定可能な部分クラスが提案されてきたが, それらは書換え規則の右辺に強い制約を課している. 本論文で提案する多層変換項書換え系のクラスは $\{f(x) \to f(g(x))\}$ のような, 既知の部分クラスには属さない正則保存項書換え系を含んでいる. また, 木言語理論における代表的な計算モデルである線形ボトムアップ型木変換器は多層変換項書換え系の特別な場合である. 本論文では, 多層変換項書換え系が正則保存になるための 2 つの十分条件を与える. さらに, 多層変換項書換え系においては到達可能性などのいくつかの性質が決定可能であることを示す.

キーワード

項書換え系, 木オートマトン, 正則性, 正則保存性, 多層変換項書換え系

* 奈良先端科学技術大学院大学 情報科学研究科 情報処理学専攻 修士論文, NAIST-IS-MT0051093, 2002 年 2 月 8 日.

Acknowledgements

I would especially like to thank Professor Hiroyuki Seki for his invaluable guidance and suggestions to this work. I am also extremely thankful to Professor Minoru Ito and Associate Professor Yuichi Kaji. I express my thanks to Assistant Professor Yoshiaki Takata. Specially, I would like to present great thanks to Mr. Toshinori Takai at National Institute of Advanced Industrial Science and Technology. He suggested an idea, and helped me throughout the research. I also grateful for support of the member of Seki Laboratory.

Contents

1. Introduction

Term Rewriting system (*TRS*) is a well-known computational model which operates on terms (or trees). The following is an example of a TRS:

$$\mathcal{R}_0 = \left\{ \begin{array}{l} add(x, 0) \to x \\ add(x, s(y)) \to s(add(x, y)) \end{array} \right\}$$

where add, s and 0 are function symbols and x and y are variables. In general, a TRS is an arbitrary finite relation on first-order terms and defines a *rewrite relation* on terms. The rewrite relation of a TRS $\mathcal{R}$ is an infinite relation and written as $\to_{\mathcal{R}}^*$. For example, the term $add(s(0), s(0))$ can be 'rewritten' to a term by using the TRS $\mathcal{R}_0$, i.e. $add(s(0), s(0)) \to_{\mathcal{R}}^* s(s(0))$. Intuitively saying, for TRS $\mathcal{R}$, the rewrite relation defined by $\mathcal{R}$ is the minimal relation which contains $\mathcal{R}$ and is closed under contexts and substitutions. An element in a TRS is called a rewrite rule and for a rewrite rule $l \to r$, l is called the left-hand side and r is called a right-hand side. The TRS $\mathcal{R}_0$ above defines the addition (add) of natural numbers in terms of zero (0) and the successor function (s).

A TRS is often used as a formal model of an actual problem in computer science. For example, TRS is applied in the field of automated theorem proving, formal language theory and functional programming languages.

We can easily see that any type 0 grammar in Chomsky's hierarchy can be simulated by a TRS according to the definition that the left-hand and right-hand sides of a rewrite rule can be an arbitrary term. Especially, it is known that any Turing machine can be simulated by TRS which consists of one (left-linear) rewrite rule. Due to its Turing-complete computational power, many important properties such as reachability, confluence and strongly normalizing property are undecidable in general. Consequently, finding an appropriate subclass of TRSs which has sufficient computational power and decidable properties on important problems has been paid attention to for a long time.

Tree automaton is a natural extension of finite-state automaton on strings. A set of ground terms (tree language) T is *recognizable* if there exists a tree automaton which accepts T. Tree automaton inherits good mathematical properties from finite-state automaton. For example, the class of recognizable sets is closed under boolean operations (union, intersection and complementation), and deci-

sion problems such as emptiness and membership are decidable for a recognizable set. Let $\mathcal{L}(\mathcal{A})$ denote the language accepted by a tree automaton $\mathcal{A}$. For a TRS $\mathcal{R}$ and a tree language T, define $(\rightarrow_{\mathcal{R}}^*)(T) = \{t \mid \exists s \in T \text{ s.t. } s \rightarrow_{\mathcal{R}}^* t\}$. A TRS $\mathcal{R}$ *effectively preserves recognizability* (abbreviated as EPR) if for any tree automaton $\mathcal{A}$, $(\rightarrow_{\mathcal{R}}^*)(\mathcal{L}(\mathcal{A}))$ is also recognizable and a tree automaton $\mathcal{A}_*$ such that $\mathcal{L}(\mathcal{A}_*) = (\rightarrow_{\mathcal{R}}^*)(\mathcal{L}(\mathcal{A}))$ can be effectively constructed. Due to the above mentioned properties of recognizable sets, some important problems, e.g., reachability, joinability and local confluence are decidable for EPR-TRSs [8, 9]. Furthermore, with additional conditions, strong normalization property, neededness and unifiability become decidable for EPR-TRSs [4, 11, 14].

The problem to decide whether a given TRS is EPR is undecidable [7], and decidable subclasses of EPR-TRSs have been proposed in a series of works [13, 3, 10, 9, 11, 14]. These subclasses put a rather strong constraint on the syntax of the right-hand side of a rewrite rule. For example, the right-hand side of a rewrite rule in a linear semi-monadic TRS (L-SM-TRS) [3] is either a variable or $f(t_1, t_2, \ldots, t_n)$ where each t_i $(1 \leq i \leq n)$ is either a variable or a ground term. Linear generalized semi-monadic TRS (L-GSM-TRS) [9] and right-linear finite path-overlapping TRS (RL-FPO-TRS) [14] weaken this constraint, but some simple EPR-TRSs such as $\{f(x) \rightarrow f(g(x))\}$ still do not belong to any of the known decidable subclasses of EPR-TRSs. To show that a given TRS $\mathcal{R}$ is EPR, for a given tree automaton $\mathcal{A}$, a tree automaton $\mathcal{A}_*$ such that $\mathcal{L}(\mathcal{A}_*) = (\rightarrow_{\mathcal{R}}^*)(\mathcal{L}(\mathcal{A}))$ should be constructed. The above mentioned restrictions on the right-hand side of a rewrite rule are sufficient conditions for a procedure of automata construction to halt.

In this paper, a new subclass of TRSs, *layered transducing TRSs* (LT-TRSs) is defined and its recognizability preserving property is discussed. Intuitively, an LT-TRS is a TRS such that certain unary function symbols are specified as *markers* and a marker moves from leaf to root in each rewrite step. Bottom-up linear tree transducer [6], which is a well-known computation model in the tree language theory, can be considered as a special case of LT-TRS. We propose a procedure which, for a given tree automaton $\mathcal{A}$ and an LT-TRS $\mathcal{R}$, constructs a tree automaton $\mathcal{A}_*$ such that $\mathcal{L}(\mathcal{A}_*) = (\rightarrow_{\mathcal{R}}^*)(\mathcal{L}(\mathcal{A}))$. The procedure introduces a state $[z, q]$ which is the product of a state z already belonging to $\mathcal{A}_*$ and a marker

q and constructs a transition rule which is the product of a transition rule already in $\mathcal{A}_*$ and a rewrite rule in $\mathcal{R}$.

However, an LT-TRS is not always EPR and the above procedure does not always halt. We present a sufficient condition for the procedure to halt. The subclass of LT-TRSs which satisfy the sufficient condition is still incomparable with any of the known decidable subclasses of EPR-TRSs. Especially, the class contains some EPR-TRSs, such as $\{f(x) \to f(g(x))\}$ mentioned above. Finally, some properties including reachability of LT-TRSs are shown to be decidable.

The rest of the paper is organized as follows. After providing preliminary definitions in section 2, LT-TRS is defined in section 3. A procedure for automata construction is presented and the partial correctness of the procedure is proved in section 4. Sufficient conditions for the construction procedure to halt are presented in section 5. Also some properties including reachability are shown to be decidable for LT-TRS in section 5.

2. Preliminaries

2.1 Term Rewriting Systems

Let Σ be a *signature* and $\mathcal{V}$ be an enumerable set of *variables*. An element in Σ is called a *function symbol* and the *arity* of $f \in \Sigma$ is denoted by $a(f)$. A function symbol c with $a(c) = 0$ is called a *constant*. The set of *terms* is denoted by $\mathcal{T}(\Sigma, \mathcal{V})$ and recursively defined as follows:

1. If $x \in \mathcal{V}$, then $x \in \mathcal{T}(\Sigma, \mathcal{V})$.
2. If $f \in \Sigma$ $(a(f) = n)$ and $t_1, \cdots, t_n \in \mathcal{T}(\Sigma, \mathcal{V})$, then $f(t_1, \cdots, t_n) \in \mathcal{T}(\Sigma, \mathcal{V})$.

The set of variables occurring in t is denoted by $\mathcal{V}ar(t)$. A term t is *ground* if $\mathcal{V}ar(t) = \emptyset$. The set of ground terms is denoted by $\mathcal{T}(\Sigma)$. A ground term in $\mathcal{T}(\Sigma)$ is also called a *Σ-term*. A term is *linear* if no variable occurs more than once in the term. A *substitution* σ is a mapping from $\mathcal{V}$ to $\mathcal{T}(\Sigma, \mathcal{V})$ satisfying that $\{x \mid \sigma(x) \neq x\}$ is finite, and written as $\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ where t_i with $1 \leq i \leq n$ is a term which substitutes for the variable x_i. A substitution σ can be extended to a mapping $\sigma' : \mathcal{T}(\Sigma, \mathcal{V}) \to \mathcal{T}(\Sigma, \mathcal{V})$ in the unique way as follows:

1. If a term t is a variable, then $\sigma'(t) = t$.
2. If a term t is of the form $t = f(t_1, \cdots, t_n)$ where $f \in \Sigma$ and $t_1, \cdots, t_n$ are terms, then $\sigma'(t) = f(\sigma'(t_1), \cdots, \sigma'(t_n))$.

The term obtained by applying a substitution σ to a term t is written as $t\sigma$. $t\sigma$ is called an *instance* of t. A *position* in a term t is defined as a sequence of positive integers which indicates a certain subtree of t if we regard t as a tree. All positions of a term t is denoted by $\mathcal{P}os(t)$ and recursively defined as follows:

1. If t is a variable, then $\mathcal{P}os(t) = \{\lambda\}$.
2. It t is of the form $t = f(t_1, \cdots, t_n)$ where f is a function symbol and $t_1, \cdots, t_n$ are terms, then $\mathcal{P}os(t) = \{\lambda\} \cup \bigcup_{1 \leq i \leq n} \{i \cdot o \mid o \in \mathcal{P}os(t_i)\}$.

A *subterm* of t at a position $o \in \mathcal{P}os(t)$ is denoted by t/o and defined as follows:

1. $t/\lambda = t$.

2. If $o = i \cdot o'$ and $t = f(t_1, \cdots, t_n)$ with $1 \leq i \leq n$, then $t/o = t_i/o'$.

If a term t is obtained from a term t' by replacing the subterms of t' at positions $o_1, \ldots, o_m$ ($o_i \in \mathcal{P}os(t'), o_i$ and o_j are disjoint if $i \neq j$) with terms t_1, ..., t_m, respectively, then we write $t = t'[o_i \leftarrow t_i \mid 1 \leq i \leq m]$.

A *rewrite rule* over a signature Σ is an ordered pair of terms in $\mathcal{T}(\Sigma, \mathcal{V})$, written as $l \rightarrow r$. A *term rewriting system* (*TRS*) over Σ is a finite set of rewrite rules over Σ. For terms t, t' and a TRS $\mathcal{R}$, we write $t \rightarrow_{\mathcal{R}} t'$ if there exists a position $o \in \mathcal{P}os(t)$, a substitution σ and a rewrite rule $l \rightarrow r \in \mathcal{R}$ such that $t/o = l\sigma$ and $t' = t[o \leftarrow r\sigma]$. Define $\rightarrow^*_{\mathcal{R}}$ to be the reflexive and transitive closure of $\rightarrow_{\mathcal{R}}$. Also the transitive closure of $\rightarrow_{\mathcal{R}}$ is denoted by $\rightarrow^+_{\mathcal{R}}$. The subscript $\mathcal{R}$ of $\rightarrow_{\mathcal{R}}$ is omitted if $\mathcal{R}$ is clear from the context. A *redex* (*in* $\mathcal{R}$) is an instance of l for some $l \rightarrow r \in \mathcal{R}$. A *normal form* (in $\mathcal{R}$) is a term which has no redex as its subterm. Let $\mathrm{NF}_{\mathcal{R}}$ denote the set of all ground normal forms in $\mathcal{R}$. A rewrite rule $l \rightarrow r$ is *left-linear* (resp. *right-linear*) if l is linear (resp. r is linear). A rewrite rule is *linear* if it is left-linear and right-linear. A TRS $\mathcal{R}$ is *left-linear* (resp. *right-linear, linear*) if every rule in $\mathcal{R}$ is left-linear (resp. right-linear, linear).

Example 1 Let $\Sigma = \{add, s, 0\}$ where $a(add) = 2$, $a(s) = 1$, and 0 is a constant. Also let $\mathcal{V} = \{x, y\}$. Let us consider the following TRS $\mathcal{R}_0$ which appeared in section 1.

$$\mathcal{R}_0 = \left\{ \begin{array}{l} add(x, 0) \rightarrow x \\ add(x, s(y)) \rightarrow s(add(x, y)) \end{array} \right\}$$

For a ground term $add(s(0), s(0))$, we obtain the following sequence:

$$\begin{array}{rcl} add(s(0), s(0)) & \rightarrow_{\mathcal{R}} & s(add(s(0), 0)) \\ & \rightarrow_{\mathcal{R}} & s(s(0)). \end{array}$$

∎

Definition 1 For a TRS $\mathcal{R}$ and two terms s and t:

1. s and t are *reachable* in $\mathcal{R}$ if $s \rightarrow^*_{\mathcal{R}} t$ or $t \rightarrow^*_{\mathcal{R}} s$.

2. s and t are *joinable* in $\mathcal{R}$ if there is a term u such that $s \rightarrow^*_{\mathcal{R}} u$ and $t \rightarrow^*_{\mathcal{R}} u$.

∎

Theorem 1 *For a given TRS $\mathcal{R}$ and two term s and t, the following problems are undecidable:*

1. *Are s and t reachable in $\mathcal{R}$?*
2. *Are s and t joinable in $\mathcal{R}$?* ∎

Definition 2 For a TRS $\mathcal{R}$:

1. $\mathcal{R}$ is *confluent* if, for terms s, t and t', $s \to_{\mathcal{R}}^* t$ and $s \to_{\mathcal{R}}^* t'$ then t and t' are joinable.
2. $\mathcal{R}$ is *locally confluent* if, for terms s, t and t', $s \to_{\mathcal{R}} t$ and $s \to_{\mathcal{R}} t'$ then t and t' are joinable. ∎

Theorem 2 *The following problems are undecidable:*

1. *Is a given TRS $\mathcal{R}$ confluent?*
2. *Is a given TRS $\mathcal{R}$ locally confluent?* ∎

2.2 Tree Automata

A *tree automaton(TA)* [6] is defined by a 4-tuple $\mathcal{A} = (\Sigma, \mathcal{P}, \Delta, \mathcal{P}_{final})$ where Σ is a signature, $\mathcal{P}$ is a finite set of states, $\mathcal{P}_{final} \subseteq \mathcal{P}$ is a set of final states, and Δ is a finite set of transition rules of the form $f(p_1, \ldots, p_n) \to p$ where $f \in \Sigma$, $a(f) = n$, and $p_1, \ldots, p_n, p \in \mathcal{P}$ or of the form $p' \to p$ where $p, p' \in \mathcal{P}$. A rule with the former form is called a *non-ε-rule* and a rule with the latter form is called an *ε-rule*. A TA $\mathcal{A}$ is *deterministic* if $\mathcal{A}$ has no ε-rule and the left-hand sides of any two transition rules are distinct. Consider the set of ground terms $\mathcal{T}(\Sigma \cup \mathcal{P})$ where we define $a(p) = 0$ for $p \in \mathcal{P}$. A *transition* of a TA can be regarded as a rewrite relation on $\mathcal{T}(\Sigma \cup \mathcal{P})$ by regarding transition rules in Δ as rewrite rules on $\mathcal{T}(\Sigma \cup \mathcal{P})$. For terms t and t' in $\mathcal{T}(\Sigma \cup \mathcal{P})$, we write $t \vdash_{\mathcal{A}} t'$ if and only if $t \to_{\Delta} t'$. The reflexive and transitive closure and the transitive closure of $\vdash_{\mathcal{A}}$ is denoted by $\vdash_{\mathcal{A}}^*$ and $\vdash_{\mathcal{A}}^+$ respectively. If $t \vdash_{\mathcal{A}} t_1 \vdash_{\mathcal{A}} t_2 \vdash_{\mathcal{A}} \cdots \vdash_{\mathcal{A}} t_k = t'$, we write $t \vdash_{\mathcal{A}}^k t'$ and k is called the length of the transition sequence. For a TA $\mathcal{A}$ and $t \in \mathcal{T}(\Sigma)$, if $t \vdash_{\mathcal{A}}^* p_f$ for a final state $p_f \in \mathcal{P}_{final}$, then we say t is *accepted*

by $\mathcal{A}$. The set of ground terms accepted by $\mathcal{A}$ is denoted by $\mathcal{L}(\mathcal{A})$. A set T of ground terms is *recognizable* if there is a TA $\mathcal{A}$ such that $T = \mathcal{L}(\mathcal{A})$. A state p is *reachable* if there exists a Σ-term t such that $t \vdash^*_{\mathcal{A}} p$. It is well-known that for any TA $\mathcal{A}$ we can construct a deterministic TA $\mathcal{A}'$ such that $\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A})$ and $\mathcal{A}'$ contains only reachable states [6]. Recognizable sets inherit some useful properties of regular (string) languages.

Example 2 Let $\mathcal{A}_0 = (\Sigma, \mathcal{P}, \Delta, \mathcal{P}_{final})$ be a TA where $\Sigma = \{f, g, h, a, b\}$, $\mathcal{P} = \{p_1, p_2, p_f\}$, $\mathcal{P}_{final} = \{p_f\}$ and Δ consists of the following transition rules:

$$\begin{array}{llllll} a & \to & p_1, & b & \to & p_2, \\ g(p_1) & \to & p_1, & h(p_2) & \to & p_2, \\ f(p_1, p_2) & \to & p_f. & & & \end{array}$$

For example, for a ground term $f(g(a), h(b))$, we obtain the following sequence:

$$\begin{array}{lll} f(g(a), h(b)) & \vdash^*_{\mathcal{A}_0} & f(g(p_1), h(p_2)) \\ & \vdash^*_{\mathcal{A}_0} & f(p_1, p_2) \\ & \vdash_{\mathcal{A}_0} & p_f. \end{array}$$

Hence, $f(g(a), h(b))$ is accepted by $\mathcal{A}_0$. It can be easily verified that $\mathcal{L}(\mathcal{A}_0) = \{f(g^m(a), h^n(b)) \mid m, n \geq 0\}$. ∎

Lemma 1 *[6] The class of recognizable sets is effectively closed under union, intersection and complementation. For a recognizable set T, the following problems are decidable. (1) Does a given ground term t belong to T? (2) Is T empty?* ∎

2.3 TRS which Preserves Recognizability

For a TRS $\mathcal{R}$ and a set T of ground terms, define $(\to^*_{\mathcal{R}})(T) = \{t \mid \exists s \in T \text{ s.t. } s \to^*_{\mathcal{R}} t\}$. A TRS $\mathcal{R}$ is said to *effectively preserve recognizability* if, for any tree automaton $\mathcal{A}$, the set $(\to^*_{\mathcal{R}})(\mathcal{L}(\mathcal{A}))$ is also recognizable and we can effectively construct a tree automaton which accepts $(\to^*_{\mathcal{R}})(\mathcal{L}(\mathcal{A}))$. In this paper, the class of TRSs which effectively preserve recognizability is written as EPR-TRS.

Theorem 3 *If a TRS $\mathcal{R}$ belongs to EPR-TRS, then the reachability relation and the joinability relation for $\mathcal{R}$ are decidable [8]. It is also decidable whether $\mathcal{R}$ is locally confluent or not [9].* ∎

Unfortunately it is undecidable whether a given TRS belongs to EPR-TRS or not [7]. Therefore decidable subclasses of EPR-TRS have been proposed, for example, ground TRS by Brained [2], right-linear monadic TRS (RL-M-TRS) by Salomaa [13], linear semi-monadic TRS (L-SM-TRS) by Coquidé et al. [3], linear generalized semi-monadic TRS (L-GSM-TRS) by Gyenizse and Vágvölgyi [9], and right-linear finite path overlapping TRS (RL-FPO-TRS) by Takai et al. [14].

Theorem 4 *RL-M-TRS* $\subset$ *RL-FPO-TRS* $\subset$ *EPR-TRS and ground TRS* $\subset$ *L-SM-TRS* $\subset$ *L-GSM-TRS* $\subset$ *RL-FPO-TRS* $\subset$ *EPR-TRS. All inclusions are proper.*

∎

Remark that Gyenizse and Vágvölgyi [9] introduced the notion of Σ-recognizability preserving property. A TRS $\mathcal{R}$ *effectively preserves* Σ*-recognizability* (abbreviated as Σ-EPR) if for any TA $\mathcal{A}$ of which accepting language is over Σ, we can effectively constructs a TA $\mathcal{A}_*$ such that $\mathcal{L}(\mathcal{A}_*) = (\rightarrow^*_{\mathcal{R}})(\mathcal{A})$. They show that there is a difference between Σ-EPR and EPR. Also Réty[12] defined a subclass of TRSs and showed that the class effectively preserves recognizability for the subclass $\mathcal{C}$ of tree languages each of which is a set $\{\ t\sigma \mid t$ is a linear term and σ is a substitution such that $\sigma(x)$ is a constructor term for each $x \in \mathcal{V}ar(t)\ \}$(abbreviated as $\mathcal{C}$-EPR). $\mathcal{R}_3$ of Example 6 in section 4 is not an EPR-TRS but it is $\mathcal{C}$-EPR.

3. Layered Transducing TRS

A new class of TRS named *layered transducing TRS* (*LT-TRS*) is proposed in this section.

Definition 3 Let $\Sigma = \mathcal{F} \cup \mathcal{Q}$ be a signature where $\mathcal{F} \cap \mathcal{Q} = \emptyset$. Suppose that for each $q \in \mathcal{Q}$, $a(q) = 1$. A function symbol in $\mathcal{Q}$ is called a *marker*. A *layered transducing TRS* (*LT-TRS*) is a TRS over Σ in which each rewrite rule has one of the following forms:

(i) $f(q_1(x_1), \cdots, q_n(x_n)) \to q(t)$, or

(ii) $q_1(x_1) \to q(t)$

where $f \in \mathcal{F}$, $q, q_i (1 \leq i \leq n) \in \mathcal{Q}$, $t \in \mathcal{T}(\mathcal{F}, \{x_1, \cdots, x_n\})$ and t is a linear term. ∎

Example 3 Let $g \in \mathcal{F}$, $q \in \mathcal{Q}$, $a(g) = 1$. $\mathcal{R}_1 = \{q(x) \to q(g(x))\}$ is an LT-TRS. Note that this TRS is an EPR-TRS but is not an FPO-TRS [14]. ∎

Example 4 Let $f, g, h \in \mathcal{F}$, $q_1, q_2, q \in \mathcal{Q}$.
$\mathcal{R}_2 = \{f(q_1(x_1), q_2(x_2)) \to q(g(h(x_2), x_1)),\ q_1(x_1) \to q(h(x_1))\}$ is an LT-TRS. ∎

In this paper, we use a, b, c to denote a constant, f, g, h to denote a non-marker symbol, $q, q', q_1, q_2, \ldots$ to denote a marker, $p, p', p_1, p_2, \ldots$ to denote a state and $s, t, t_1, t_2, \ldots$ to denote a term.

4. Construction of Tree Automata

In this section, we will present a procedure which takes an LT-TRS $\mathcal{R}$ and a tree automaton (TA) $\mathcal{A}$ as an input and constructs a TA $\mathcal{A}_*$ such that $\mathcal{L}(\mathcal{A}_*) = (\to_{\mathcal{R}}^*)(\mathcal{L}(\mathcal{A}))$ if the procedure halts. The idea behind the procedure is as follows. Let $\mathcal{A} = (\Sigma, \mathcal{P}, \Delta, \mathcal{P}_{final})$ be a TA. By the definition of $(\to_{\mathcal{R}}^*)(\cdot)$,

if $t \vdash_{\mathcal{A}}^* p$ and $t \to_{\mathcal{R}}^* s$ then $s \vdash_{\mathcal{A}_*}^* p$ also holds.

To satisfy this property, the proposed procedure starts with $\mathcal{A}_0 = \mathcal{A}$ and constructs a series of TAs $\mathcal{A}_0, \mathcal{A}_1, \ldots$. We define $\mathcal{A}_*$ as the limit of this chain of TAs. For example, let $f(p_1, p_2) \to p \in \Delta$ and $f(q_1(x_1), q_2(x_2)) \to q(g(h(x_2), x_1)) \in \mathcal{R}$ and assume that

$$t = f(q_1(t_1), q_2(t_2)) \vdash_{\mathcal{A}}^* f(q_1(p_1'), q_2(p_2')) \vdash_{\mathcal{A}}^* f(p_1, p_2) \vdash_{\mathcal{A}} p. \tag{4.1}$$

Note that $f(q_1(t_1), q_2(t_2)) \to_{\mathcal{R}} q(g(h(t_2), t_1))(= t')$ and hence $\mathcal{A}_*$ is required to satisfy $q(g(h(t_2), t_1)) \vdash_{\mathcal{A}_*}^* p$. The procedure constructs a 'product' rule of $f(p_1, p_2) \to p$ and $f(q_1(x_1), q_2(x_2)) \to q(g(h(x_2), x_1))$ and some auxiliary rules so that $\mathcal{A}_*$ can simulate the transition sequence (4.1) when $\mathcal{A}_*$ reads $q(g(h(t_2), t_1))$. More precisely, new states $[p, q]$ and $\langle h(p_2') \rangle$ are introduced and rules

$$g(\langle h(p_2') \rangle, p_1') \to [p, q],\ h(p_2') \to \langle h(p_2') \rangle \tag{4.2}$$

are constructed. The following transition rule is also added so that $s \vdash_{\mathcal{A}_*}^* [p, q]$ if and only if $q(s) \vdash_{\mathcal{A}_*}^* p$.

$$q([p, q]) \to p. \tag{4.3}$$

When $\mathcal{A}_*$ reads $q(g(h(t_2), t_1))$, we can see by (4.1) that

$$t' = q(g(h(t_2), t_1)) \vdash_{\mathcal{A}}^* q(g(h(p_2'), p_1')). \tag{4.4}$$

$\mathcal{A}_*$ guesses that in a term t such that $t \to_{\mathcal{R}} t'$, the markers q_1 and q_2 were placed above the subterms t_1 and t_2, respectively, and $\mathcal{A}_*$ behaves as if it reads q_1 and q_2 at p_1' and p_2'. That is, $\mathcal{A}_*$ simulates the transition $f(p_1, p_2) \vdash_{\mathcal{A}} p$ by rules (4.2). Also see Fig. 1.

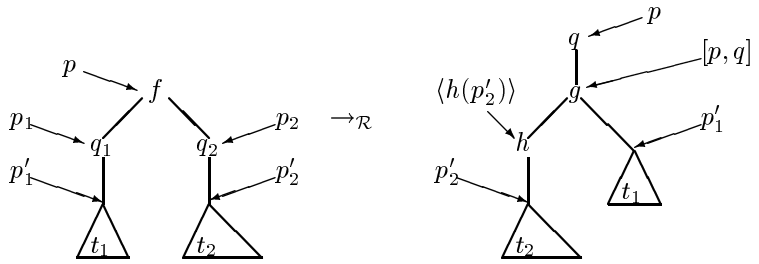

Figure 1. Illustration of automata construction.

$$t' \vdash^*_{\mathcal{A}} q(g(h(p'_2), p'_1)) \vdash_{\mathcal{A}} q(g(\langle h(p'_2)\rangle, p'_1)) \vdash_{\mathcal{A}} q([p,q]) \vdash_{\mathcal{A}} p$$

The last transition is by (4.3); $\mathcal{A}_*$ encounters the marker q at the state $[p,q]$, which means that the guess was correct, and $\mathcal{A}_*$ changes its state to p by forgetting the guess q. The construction of new rules and states is repeated until $\mathcal{A}_i$ saturates. Hence, states with more than one nesting such as $[[[p,q_1],q_2],q_3]$ may be defined in general.

Procedure 1 Suppose $\Sigma = \mathcal{F} \cup \mathcal{Q}$ $(\Sigma \cap \mathcal{Q} = \emptyset)$ and $\forall q \in \mathcal{Q} : a(q) = 1$.

Input : a deterministic tree automaton $\mathcal{A} = (\Sigma, \mathcal{P}, \Delta, \mathcal{P}_{final})$ such that
every state in $\mathcal{P}$ is reachable,
an LT-TRS $\mathcal{R}$ over Σ.

Output : a tree automaton $\mathcal{A}_*$ s.t. $\mathcal{L}(\mathcal{A}_*) = (\rightarrow^*_{\mathcal{R}})(\mathcal{L}(\mathcal{A}))$.

Step1. Let $i := 0$, $\mathcal{A}_0 = (\Sigma, \mathcal{Z}_0, \Delta_0, \mathcal{P}_{final}) := \mathcal{A}$.

In **Step2**–**Step4**, this procedure constructs $\mathcal{A}_1, \mathcal{A}_2, \cdots$ by adding new states and transition rules to $\mathcal{A}_0$.

Step2. Let $i := i+1$, $\mathcal{A}_i = (\Sigma, \mathcal{Z}_i, \Delta_i, \mathcal{P}_{final}) := \mathcal{A}_{i-1}$.

Step3. Do the following (**S**) and (**T**) until $\mathcal{A}_i$ does not change.

(**S**) `if`

(**S1**) $f(z_1, \cdots, z_n) \rightarrow z \in \Delta_{i-1}(f \in \mathcal{F})$, $f(q_1(x_1), \cdots, q_n(x_n)) \rightarrow q(t) \in \mathcal{R}$, $z'_j \in \mathcal{Z}_{i-1}$ s.t. $q_j(z'_j) \vdash^*_{\mathcal{A}_{i-1}} z_j (1 \leq j \leq n)$

`or`

(**S2**) $q_1(x_1) \rightarrow q(t) \in \mathcal{R}$, $z'_1 \in \mathcal{Z}_{i-1}$ s.t. $q_1(z'_1) \vdash^*_{\mathcal{A}_{i-1}} z$

`then`

`if` $t = x_l$ and $[z,q] \neq z'_l$ `then`

add to Δ_i $z'_l \to [z,q]$;
add to $\mathcal{Z}_i$ $[z,q]$;
else let $t = g(t_1, \cdots, t_m)$
add to Δ_i $g(\langle t_1\rho\rangle, \cdots, \langle t_m\rho\rangle) \to [z,q]$;
add to $\mathcal{Z}_i$ $\langle t_1\rho\rangle, \cdots, \langle t_m\rho\rangle, [z,q]$
where $\rho = \{x_j \mapsto z'_j \mid 1 \le j \le m\}$[1] ;
ADDREC$(t_j, i, \rho)(1 \le j \le m)$.

(T) $\forall [z,q] \in \mathcal{Z}_i, \forall q \in \mathcal{Q}$ add to Δ_i $q([z,q]) \to z$.

Step4. If $\mathcal{A}_{i-1} = \mathcal{A}_i$, then let $\mathcal{A}_* := \mathcal{A}_i$ and output $\mathcal{A}_*$, else go to **Step2**. ∎

Procedure 2 (**ADDREC**) This procedure takes a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$, an integer $i \ge 1$ and a substitution $\rho : \mathcal{V} \to \mathcal{Z}_{i-1}$ as an input, and adds new states and transition rules to $\mathcal{A}_i$ so that $t\sigma \vdash^*_{\mathcal{A}_i} \langle t\rho\rangle$ holds for every substitution σ such that $\sigma = \{x_j \mapsto s_j \in \mathcal{T}(\Sigma) \mid s_j \vdash^*_{\mathcal{A}_i} \rho(x_j),\ 1 \le j \le m\}$.

ADDREC(t, i, ρ)=
if $t = x$ then return;
else let $t = h(t_1, \cdots, t_n)$
add to Δ_i $h(\langle t_1\rho\rangle, \cdots, \langle t_n\rho\rangle) \to \langle t\rangle$;
add to $\mathcal{Z}_i$ $\langle t_1\rho\rangle, \cdots, \langle t_n\rho\rangle$;
ADDREC$(t_j, i, \rho)(1 \le j \le n)$.

∎

Example 5 Let $\mathcal{A} = (\Sigma, \mathcal{P}, \Delta, \mathcal{P}_{final})$ be a TA where $\Sigma = \mathcal{F} \cup \mathcal{Q}$, $\mathcal{F} = \{f, g, h, c\}$, $\mathcal{Q} = \{q_1, q_2, q\}$, $\mathcal{P} = \{p_1, p'_1, p_2, p_c, p_f\}$, $\mathcal{P}_{final} = \{p_f\}$ and $\Delta = \{c \to p_c,\ q_1(p_c) \to p'_1,\ q_1(p'_1) \to p_1,\ q_2(p_c) \to p_2,\ f(p_1, p_2) \to p_f\}$. It can be easily verified that $\mathcal{L}(\mathcal{A}) = \{f(q_1(q_1(c)), q_2(c))\}$. We apply Procedure 1 to $\mathcal{A}$ and LT-TRS $\mathcal{R}_2$ of Example 4. For $i = 1$, $f(p_1, p_2) \to p_f \in \Delta_0 = \Delta$ and $f(q_1(x_1), q_2(x_2)) \to q(g(h(x_2), x_1)) \in \mathcal{R}_2$ are considered. Since $q_1(p'_1) \vdash_{\mathcal{A}_0} p_1$ and $q_2(p_c) \vdash_{\mathcal{A}_0} p_2$, condition (**S1**) is satisfied and a rule $g(\langle h(x_2)\rho\rangle, \langle \rho(x_1)\rangle) = g(\langle h(p_c)\rangle, p'_1) \to [p_f, q]$ is added to Δ_1 where $\rho = \{x_1 \mapsto p'_1, x_2 \mapsto p_c\}$. Also a rule $h(p_c) \to \langle h(p_c)\rangle$ is constructed by **ADDREC**$(h(x_2), 1, \rho)$. Consider $q_1(x_1) \to q(h(x_1)) \in \mathcal{R}_2$. Since

[1] For simplicity, for a state $z'_j \in \mathcal{Z}_i$, we identify z'_j with $\langle z'_j\rangle$.

Table 1. The transition rules constructed by Procedure 1.

	(**S**)	(**T**)
$\mathcal{A}_1$	$g(\langle h(p_c)\rangle, p_1') \to [p_f, q]$	$q([p_f, q]) \to p_f$
	$h(p_c) \to \langle h(p_c)\rangle$	
	$h(p_1') \to [p_1, q]$	$q([p_1, q]) \to p_1$
	$h(p_c) \to [p_1', q]$	$q([p_1', q]) \to p_1'$

$q_1(p_1') \vdash_{\mathcal{A}_0} p_1$, condition (**S2**) is satisfied and a rule $h(p_1') \to [p_1, q]$ is constructed. The transition rules constructed in Procedure 1 are listed in Table 1. Since no rule is added in $\mathcal{A}_2$, the procedure halts and we obtain $\mathcal{A}_* = \mathcal{A}_2$ as the output. We can verify that $\mathcal{L}(\mathcal{A}_*) = (\to^*_{\mathcal{R}_2})(\mathcal{L}(\mathcal{A}))$. ∎

Example 6 Let $\mathcal{A} = (\Sigma, \mathcal{P}, \Delta, \mathcal{P}_{final})$, $\Sigma = \mathcal{F} \cup \mathcal{Q}$, $\mathcal{F} = \{c, f\}$, $\mathcal{Q} = \{q\}$, $\mathcal{P} = \mathcal{P}_{final} = \{p\}$, $\Delta = \{c \to p, f(p) \to p, q(p) \to p\}$ and $\mathcal{R}_3 = \{f(q(x)) \to q(f(x))\}$. $\mathcal{R}_3$ is an LT-TRS. Assume that Procedure 1 is executed for $\mathcal{A}$ and $\mathcal{R}_3$. Since $f(p) \to p \in \Delta_0$, $f(q(x)) \to q(f(x)) \in \mathcal{R}_3$ and $q(p) \vdash_{\mathcal{A}_0} p$, $f(p) \to [p, q]$ and $q([p, q]) \to p$ are added to Δ_1. When $i = 2$, the procedure considers $f(p) \to [p, q]$ and $f(q(x)) \to q(f(x))$ and it constructs $f(p) \to [[p, q], q]$ and $q([[p, q], q]) \to [p, q]$ to Δ_2. The procedure repeats a similar construction and does not halt. Note that $\mathcal{R}_3$ is not an EPR-TRS since for a recognizable set $T_1 = \{(fq)^n(c) \mid n \geq 0\}$, $(\to^*_{\mathcal{R}_3})(T_1) \cap \mathrm{NF}_{\mathcal{R}_3} = \{q^n(f^n(c)) \mid n \geq 0\}$ is not recognizable. ∎

In the following, we will prove the soundness ($t \vdash^*_{\mathcal{A}_i} p \in \mathcal{P} \Rightarrow \exists s : s \vdash^*_{\mathcal{A}} p$ and $s \to^*_{\mathcal{R}} t$) and completeness ($s \vdash^*_{\mathcal{A}} p$ and $s \to^*_{\mathcal{R}} t \Rightarrow \exists i \geq 0 : t \vdash^*_{\mathcal{A}_i} p$) of Procedure 1. To show the soundness, we first prove a few technical lemmas. By the definition of Procedure 1, it is not difficult to prove the following three lemmas. Note that by Procedure 1, if $z \in \mathcal{Z}_i$ then either $z \in \mathcal{P}$, z is of the form $[z', q]$ for some $z' \in \mathcal{Z}_{i-1}$ and $q \in \mathcal{Q}$ or z is of the form $\langle t\rho\rangle$ for some term t and substitution $\rho : \mathcal{V} \to \mathcal{Z}_{i-1}$.

Lemma 2 *(i) If $z_1 \vdash^+_{\mathcal{A}_i} z_2$, then z_2 is of the form $[z_2', q_2]$ and z_1 is either in $\mathcal{P}$ or of the form $[z_1', q_1]$.*

(ii) If $f(z_1, \cdots, z_n) \to z \in \Delta_i (i \geq 1, f \in \mathcal{F})$, then z is of the form $[z', q]$ and for each $j (1 \leq j \leq n)$ either z_j (where z_j is the state substituted for each variable) is in $\mathcal{P}$ or of the form $[z'_j, q_j]$. ∎

Lemma 3 *Let $\mathcal{A} = (\Sigma, \mathcal{P}, \Delta, \mathcal{P}_{final})$ be a TA. Assume that every state $p \in \mathcal{P}$ is reachable. If Procedure 1 is executed for $\mathcal{A}$ and an LT-TRS $\mathcal{R}$, then every state $z \in \mathcal{Z}_i$ constructed during the execution of Procedure 1 is reachable.* ∎

Lemma 4 *For a transition rule $g(\langle t_1\rho\rangle, \cdots, \langle t_m\rho\rangle) \to [z, q]$ constructed in (**S1**) of Procedure 1 and a Σ-term s, $s \vdash^*_{\mathcal{A}_i} g(\langle t_1\rho\rangle, \cdots, \langle t_m\rho\rangle) \vdash_{\mathcal{A}_i} [z, q]$ if and only if there exists a substitution $\sigma : \mathcal{V} \to \mathcal{T}(\Sigma)$ such that $s = g(t_1, \cdots, t_m)\sigma$ and for each $x \in \mathcal{V}ar(g(t_1, \cdots, t_m))$, $\sigma(x) \vdash^*_{\mathcal{A}_i} \rho(x)$.* ∎

For example, consider a rule $g(\langle(h(p_c))\rangle, p'_1) \to [p_f, q]$ constructed by Procedure 1 for $\mathcal{A}$ and $\mathcal{R}_2$ of Example 5. For a Σ-term s, $s \vdash^*_{\mathcal{A}_1} g(\langle h(p_c)\rangle, p'_1) \vdash_{\mathcal{A}_1} [p_f, q]$ if and only if s can be written as $g(h(s_2), s_1)$ where s_1 and s_2 are Σ-terms such that $s_1 \vdash^*_{\mathcal{A}_1} p'_1$ and $s_2 \vdash^*_{\mathcal{A}_1} p_c$.

The next lemma states an important property of an ε-transition.

Lemma 5 *If either (i) $[z', q'] \vdash^*_{\mathcal{A}_i} [z, q]$, $s' \vdash^*_{\mathcal{A}_{i-1}} z'$ and $s' \to^*_{\mathcal{R}} q'(t)$ or (ii) $p \vdash^*_{\mathcal{A}_i} [z, q] (p \in \mathcal{P})$, $s' \vdash^*_{\mathcal{A}_{i-1}} p$ and $s' \to^*_{\mathcal{R}} t$ holds, then there exists a Σ-term s such that $s \vdash^*_{\mathcal{A}_{i-1}} z$ and $s \to^*_{\mathcal{R}} q(t)$.*

Proof. We will prove the following slightly more general property than (i) by double induction on i and the number of rewrite steps in $[z', q'] \vdash^*_{\mathcal{A}_i} [z, q]$. (A proof for (ii) is similar.)

> *If $[z', q'] \vdash^*_{\mathcal{A}_i} [z, q]$, $s' \vdash_{\mathcal{A}_{i'}} z$ and $s' \to^*_{\mathcal{R}} q'(t)$ with $i' \geq i - 1$ then there exists a Σ-term s such that $s \vdash^*_{\mathcal{A}_{i'}} z$ and $s \to^*_{\mathcal{R}} q(t)$.*

If $i = 0$ or $[z', q'] = [z, q]$ then the lemma holds clearly. If the number of rewrite steps is not less than one and $i \geq 1$, then we can assume that $[z', q'] \vdash^*_{\mathcal{A}_i} [z'', q''] \vdash_{\mathcal{A}_i} [z, q]$ by Lemma 2. By the inductive hypothesis on the number of rewrite steps, there exists a Σ-term s'' such that

$$s'' \vdash^*_{\mathcal{A}_{i'}} z''(i' \geq i - 1) \text{ and } s'' \to^*_{\mathcal{R}} q''(t). \tag{4.5}$$

Assume that $[z'', q''] \to [z, q] \in \Delta_i$ is constructed in (**S1**). (The proof for the case (**S2**) is similar.) There exist

$$f(z_1, \cdots, z_m) \ \to \ z \in \Delta_{i-1}, \tag{4.6}$$

$$f(q_1(x_1), \cdots, q_n(x_n)) \ \to \ q(x_k) \in \mathcal{R}, \tag{4.7}$$

$$q_j(z'_j) \vdash^*_{\mathcal{A}_{i-1}} z_j (1 \leq j \leq n), \tag{4.8}$$

and $z'_k = [z'', q'']$. The condition (4.8) when $j = k$ can be written as:

$$q_k(z'_k) = q_k([z'', q'']) \vdash^*_{\mathcal{A}_{i-1}} q_k([z^{(3)}, q_k]) \vdash_{\mathcal{A}_{i-1}} z^{(3)} \vdash^*_{\mathcal{A}_{i-1}} z_k. \tag{4.9}$$

By applying the inductive hypothesis on i to $[z'', q''] \vdash^*_{\mathcal{A}_{i-1}} [z^{(3)}, q_k]$, we can see by (4.5) there exists a Σ-term $s^{(3)}$ such that

$$s^{(3)} \vdash^*_{\mathcal{A}_{i'}} z^{(3)} \text{ and } s^{(3)} \to^*_{\mathcal{R}} q_k(t). \tag{4.10}$$

By Lemma 3, there exist Σ-term $\tau_j (1 \leq j \leq n)$ such that $\tau_j \vdash^*_{\mathcal{A}_{i-1}} z'_j$. Let $s = f(q_1(\tau_1), \cdots, s^{(3)}, \cdots, q_n(\tau_n))$, then

$$s \vdash^*_{\mathcal{A}_{i'}} f(z_1, \cdots, z_n) \vdash_{\mathcal{A}_{i'}} z \qquad \text{by (4.6),(4.8),(4.9) and (4.10), and}$$

$$s \to^*_{\mathcal{R}} f(q_1(\tau_1), \cdots, q_k(t), \cdots, q_n(\tau_n)) \to_{\mathcal{R}} q(t) \qquad \text{by (4.7) and (4.10).}$$

Hence, the lemma holds. ∎

Lemma 6 *(Soundness) Let $t \in \mathcal{T}(\Sigma)$, $p \in \mathcal{P}$, $i \geq 1$ and $[z, q] \in \mathcal{Z}_i$.*

(**A**) $t \vdash^*_{\mathcal{A}_i} p \ \Rightarrow$ *there exists a Σ-term s such that $s \vdash^*_{\mathcal{A}_{i-1}} p$ and $s \to^*_{\mathcal{R}} t$.*

(**B**) $t \vdash^*_{\mathcal{A}_i} [z, q] \ \Rightarrow$ *there exists a Σ-term s such that $s \vdash^*_{\mathcal{A}_{i-1}} z$ and $s \to^*_{\mathcal{R}} q(t)$.*

Proof. We will prove (**A**) and (**B**) simultaneously by induction on the length of the transition sequences $t \vdash^*_{A_i} p$ and $t \vdash^*_{A_i} [z, q]$.

(**A**) Assume $t \vdash^*_{\mathcal{A}_i} p$. The following three cases (i)–(iii) should be considered according to the rule applied in the last transition in $t \vdash^*_{\mathcal{A}_i} p$.

(i) The case that $t \vdash^*_{\mathcal{A}_i} z \vdash_{\mathcal{A}_i} p (z \in \mathcal{Z}_i)$ is impossible since $\mathcal{A}$ contains no ε-rule and the right-hand side of an ε-rule constructed in (**S**) dose not belong to $\mathcal{P}$.

(ii) If $t = q(t') \vdash^*_{\mathcal{A}_i} q(z) \vdash_{\mathcal{A}_i} p(q \in \mathcal{Q}, z \in \mathcal{Z}_i)$, then $z \in \mathcal{P}$ or $z = [p, q]$ by Procedure 1.

(ii-a) Assume $t = q(t') \vdash^*_{\mathcal{A}_i} q(p') \vdash_{\mathcal{A}_i} p(p' \in \mathcal{P})$. By applying the inductive hypothesis (**A**) to $t' \vdash^*_{\mathcal{A}_i} p'$, there exists a Σ-term s' such that $s' \vdash^*_{\mathcal{A}_{i-1}} p'$ and $s' \rightarrow^*_{\mathcal{R}} t'$. Clearly, $q(p') \rightarrow p \in \Delta_0$. By letting $s = q(s')$, we can see that $q(s') \vdash^*_{\mathcal{A}_{i-1}} q(p') \vdash_{\mathcal{A}_0} p$, $q(s') \rightarrow^*_{\mathcal{R}} q(t') = t$ and the lemma holds.

(ii-b) Assume $t = q(t') \vdash^*_{\mathcal{A}_i} q([p, q]) \vdash_{\mathcal{A}_i} p$. By applying the inductive hypothesis (**B**) to $t' \vdash^*_{\mathcal{A}_i} [p, q]$, there exists a Σ-term s' such that $s' \vdash^*_{\mathcal{A}_{i-1}} p$ and $s' \rightarrow^*_{\mathcal{R}} q(t') = t$. This implies the lemma.

(iii) Assume $t = f(t_1, \cdots, t_n) \vdash^*_{\mathcal{A}_i} f(z_1, \cdots, z_n) \vdash_{\mathcal{A}_i} p(f \in \mathcal{F},\ p \in \mathcal{P})$. By Lemma 2 (ii), $f(z_1, \cdots, z_n) \rightarrow p \in \Delta_0$ and hence $z_j \in \mathcal{P} (1 \leq j \leq n)$. By the inductive hypothesis (**A**), for each l $(1 \leq l \leq n)$, there exists a Σ-term s_l such that $s_l \vdash^*_{\mathcal{A}_{i-1}} z_l$ and $s_l \rightarrow^*_{\mathcal{R}} t_l$. If we let $s = f(s_1, \cdots, s_n)$, then $f(s_1, \cdots, s_n) \vdash^*_{\mathcal{A}_{i-1}} f(z_1, \cdots, z_n) \vdash_{\mathcal{A}_0} p$ and $f(s_1, \cdots, s_n) \rightarrow^*_{\mathcal{R}} f(t_1, \cdots, t_n) = t$.

(**B**) Assume $t \vdash^*_{\mathcal{A}_i} [z, q]$ and we perform the following case analysis (i)–(iii) by considering the rule applied in the last step of the transition sequence $t \vdash^*_{\mathcal{A}_i} [z, q]$.

(i) If $t \vdash^*_{\mathcal{A}_i} z' \vdash_{\mathcal{A}_i} [z, q] (z' \in \mathcal{Z}_i)$, then by Lemma 2 we further consider the two subcases, namely, (i-a) $z' \in \mathcal{P}$, and (i-b) $z' = [z'', q']$ $(z'' \in \mathcal{Z}_i, q' \in \mathcal{Q})$.

(i-a) By the inductive hypothesis (**A**), there exists a Σ-term s' such that $s' \vdash^*_{\mathcal{A}_{i-1}} z'$ and $s' \rightarrow^*_{\mathcal{R}} t$. (i-b) In a similar way to (i-a), there exists a Σ-term s' such that $s' \vdash^*_{\mathcal{A}_{i-1}} z''$ and $s' \rightarrow^*_{\mathcal{R}} q'(t)$ by the inductive hypothesis (**B**). In either case (i-a) or (i-b), there exists a Σ-term s such that $s \vdash^*_{\mathcal{A}_{i-1}} z$ and $s \rightarrow^*_{\mathcal{R}} q(t)$ by Lemma 5.

(ii) Assume $t = q'(t') \vdash^*_{\mathcal{A}_i} q'(z') \vdash_{\mathcal{A}_i} [z, q] (q' \in \mathcal{Q}, z' \in \mathcal{Z}_i)$. Since the rule $q'(z') \rightarrow [z, q]$ applied in the last step of the transition sequence is constructed in (**T**), $z' = [[z, q], q'] \in \mathcal{Z}_i$. This implies $[z, q] \in \mathcal{Z}_{i-1}$ and $q([z, q]) \rightarrow z \in \Delta_{i-1}$. By the inductive hypothesis (**B**) applied to $t' \vdash^*_{\mathcal{A}_i} [[z, q], q']$, there exists a Σ-term s' such that $s' \vdash^*_{\mathcal{A}_{i-1}} [z, q]$ and $s' \rightarrow^*_{\mathcal{R}} q'(t') (= t)$. Let $s = q(s')$, then $s \vdash^*_{\mathcal{A}_{i-1}} q([z, q]) \vdash_{\mathcal{A}_{i-1}} z$ and $s \rightarrow^*_{\mathcal{R}} q(q'(t')) = q(t)$. Thus, the lemma holds.

(iii) Assume $t = g(t_1, \cdots, t_m) \vdash^k_{\mathcal{A}_i} g(z'_1, \cdots, z'_m) \vdash_{\mathcal{A}_i} [z, q] (g \in \mathcal{F},\ z'_1, \cdots, z'_m \in \mathcal{Z}_i)$. This is the most difficult case. The rule $g(z'_1, \cdots, z'_m) \rightarrow [z, q] \in \Delta_i$ applied in the last step is constructed in (**S**). Assume that the rule is constructed in (**S1**). (The lemma can be shown in a similar way when the rule is constructed in (**S2**).)

By (**S1**), there exist

$$\begin{aligned} f(z_1,\cdots,z_n) &\rightarrow z \in \Delta_{i-1}, && (4.11)\\ f(q_1(x_1),\cdots,q_n(x_n)) &\rightarrow q(g(\xi_1,\cdots,\xi_m)) \in \mathcal{R} \\ &\quad (g \in \mathcal{F},\ \xi_j \in \mathcal{T}(\mathcal{F},\{x_1,\cdots,x_n\})), && (4.12) \end{aligned}$$

$$q_j(z_j'') \vdash^*_{\mathcal{A}_{i-1}} z_j\ (1 \le j \le n). \tag{4.13}$$

By Lemma 4, the assumed transition sequence can be written as:

$$\begin{aligned} t = g(t_1,\cdots,t_m) &= g(\xi_1,\cdots,\xi_m)\sigma \\ &\vdash^*_{\mathcal{A}_i} g(\xi_1,\cdots,\xi_m)\rho \\ &\vdash^*_{\mathcal{A}_i} g(\langle\xi_1\rho\rangle,\cdots,\langle\xi_m\rho\rangle) \\ &= g(z_1',\cdots,z_m') \\ &\vdash_{\mathcal{A}_i} [z,q] \end{aligned} \tag{4.14}$$

where

$$\begin{aligned} \sigma &= \{x_j \mapsto w_j \mid 1 \le j \le n\}, \\ \rho &= \{x_j \mapsto z_j'' \mid 1 \le j \le n\}, \end{aligned} \tag{4.15}$$

$$w_j \vdash^{k_j}_{\mathcal{A}_i} z_j'' \text{ with } k_j \le k (1 \le j \le n). \tag{4.16}$$

Note that for $j(1 \le j \le n)$ such that $x_j \notin \mathcal{V}ar(g(\xi_1,\cdots,\xi_m))$, we use Lemma 3 to guarantee the existence of term w_j.

By Lemma 2, there are two cases to consider. (iii-a) If $z_j'' \in \mathcal{P}$ then by (4.16) and the inductive hypothesis (**A**), there exists a Σ-term u_j such that $u_j \vdash^*_{\mathcal{A}_{i-1}} z_j''$ and $u_j \rightarrow^*_{\mathcal{R}} w_j$. Let $s_j = q_j(u_j)$, then $s_j \vdash^*_{\mathcal{A}_{i-1}} q_j(z_j'') \vdash^*_{\mathcal{A}_{i-1}} z_j$ by (4.13) and $s_j \rightarrow^*_{\mathcal{R}} q_j(w_j)$. (iii-b) If $z_j'' = [z_j^{(3)}, q_j^{(3)}]$ for some $z_j^{(3)}$ and $q_j^{(3)}$ then by the inductive hypothesis (**B**), there exists a Σ-term u_j such that

$$u_j \vdash^*_{\mathcal{A}_{i-1}} z_j^{(3)} \text{ and } u_j \rightarrow^*_{\mathcal{R}} q_j^{(3)}(w_j). \tag{4.17}$$

By Lemma 2, the condition (4.13) can be written as:

$$q_j(z_j'') = q_j([z_j^{(3)}, q_j^{(3)}]) \vdash^*_{\mathcal{A}_{i-1}} q_j([z_j^{(4)}, q_j]) \vdash_{\mathcal{A}_{i-1}} z_j^{(4)} \vdash^*_{\mathcal{A}_{i-1}} z_j. \tag{4.18}$$

By Lemma 5 and (4.17) there exists a Σ-term s_j such that $s_j \vdash^*_{\mathcal{A}_{i-1}} z_j^{(4)} \vdash^*_{\mathcal{A}_{i-1}} z_j$ and $s_j \rightarrow^*_{\mathcal{R}} q_j(w_j)$. In either case (iii-a) or (iii-b), $s_j \vdash^*_{\mathcal{A}_{i-1}} z_j$ and $s_j \rightarrow^*_{\mathcal{R}} q_j(w_j)$.

Let $s = f(s_1, \cdots, s_n)$. Then,

$$s \vdash^*_{\mathcal{A}_{i-1}} f(z_1, \cdots, z_n) \vdash_{\mathcal{A}_i} z \quad \text{by (4.11)},$$

$$\begin{aligned} s &\to^*_{\mathcal{R}} f(q_1(w_1), \cdots, q_n(w_n)) \\ &\to_{\mathcal{R}} q(g(\xi_1, \cdots, \xi_m))\sigma = t \quad \text{by (4.12),(4.14) and (4.15).} \end{aligned}$$

Therefore, the lemma holds. ∎

Lemma 7 *(Completeness)*

$$s \to^*_{\mathcal{R}} t \text{ and } s \vdash^*_{\mathcal{A}_0} p \in \mathcal{P} \quad \Rightarrow \quad \text{there exists an integer } i \geq 0 \text{ such that } t \vdash^*_{\mathcal{A}_i} p \ .$$

Proof. Assume that $s \to^*_{\mathcal{R}} t$ and $s \vdash^*_{\mathcal{A}_0} p$. The lemma is shown by induction on the number of rewrite steps in $s \to^*_{\mathcal{R}} t$. If $s = t$ then the lemma holds clearly. Assume $s \to^*_{\mathcal{R}} t' \to_{\mathcal{R}} t$. By the inductive hypothesis, there exists $i' \geq 0$ such that $t' \vdash^*_{\mathcal{A}_{i'}} p$. For a rewrite step $t' \to_{\mathcal{R}} t$, the applied rewrite rule has the form of either (i) $f(q_1(x_1), \cdots, q_n(x_n)) \to q(u)$, or (ii) $q_1(x_1) \to q(u)$. In either case (i) or (ii), t can be written as $t = t'[o \leftarrow q(u\sigma)]$ where o is the rewrite position and σ is a substitution. Assume $u = g(u_1, \cdots, u_m)$ and $u\sigma = g(t_1, \cdots, t_m)$. (The case when u is a variable is easier and omitted.) Consider the case (i). Since $t' \vdash^*_{\mathcal{A}_{i'}} p$ and t'/o is an instance of $f(q_1(x_1), \cdots, q_n(x_n))$, there exists a transition rule $f(z_1, \cdots, z_n) \to z_0 \in \Delta_{i'}$ and $t' \vdash^*_{\mathcal{A}_{i'}} p$ can be written as $t' = t'[o \leftarrow f(q_1(\sigma(x_1)), \cdots, q_n(\sigma(x_n)))] \vdash^*_{\mathcal{A}_{i'}} t'[o \leftarrow f(q_1(z''_1), \cdots, q_n(z''_n))] \vdash^*_{\mathcal{A}_{i'}} t'[o \leftarrow f(z_1, \cdots, z_n)] \vdash_{\mathcal{A}_{i'}} t'[o \leftarrow z_0] \vdash^*_{\mathcal{A}_{i'}} p$. Since $q_j(z''_j) \vdash^*_{\mathcal{A}_{i'}} z_j$ $(1 \leq j \leq n)$, (**S1**) constructs a transition rule $g(\langle u_1\rho\rangle, \cdots, \langle u_m\rho\rangle) \to [z_0, q]$ where $\rho = \{x_j \mapsto z''_j \mid 1 \leq j \leq n\}$. Since $\sigma(x_j) \vdash^*_{\mathcal{A}_{i'}} z''_j$, by Lemma 4, $g(t_1, \cdots, t_n) \vdash^*_{\mathcal{A}_{i'+1}} [z_0, q]$. The case (ii) can be treated in a similar way to the case (i) and we can show that (**S2**) constructs transition rules which enable $g(t_1, \cdots, t_n) \vdash^*_{\mathcal{A}_{i'+1}} [z_0, q]$. In either case, $q([z_0, q]) \to z_0$ is also added in (**T**) and we can see that $t = t'[o \leftarrow q(g(t_1, \cdots, t_n))] \vdash^*_{\mathcal{A}_{i'+1}} t'[o \leftarrow q([z_0, q])] \vdash_{\mathcal{A}_{i'+1}} t'[o \leftarrow z_0] \vdash^*_{\mathcal{A}_{i'}} p$ and the lemma holds. ∎

Lemma 8 *(Partial Correctness) Let $\mathcal{A} = (\Sigma, \mathcal{P}, \Delta, \mathcal{P}_{final})$ be a deterministic TA, $\mathcal{R}$ be an LT-TRS. Assume that for input $\mathcal{A}$ and $\mathcal{R}$, Procedure 1 constructs a series of tree automata $\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \cdots$. For any term $t \in \mathcal{T}(\Sigma)$ and state $p \in \mathcal{P}$,*

> *there exists a term $s \in \mathcal{T}(\Sigma)$ such that $s \vdash^*_{\mathcal{A}} p$ and $s \to^*_{\mathcal{R}} t$*
> *if and only if there exists $i \geq 0$ such that $t \vdash^*_{\mathcal{A}_i} p$.*

Proof. ($\Rightarrow$) By Lemma 7. ($\Leftarrow$) By induction on i and Lemma 6(**A**). ∎

Lemma 9 *If Procedure 1 halts for a deterministic TA $\mathcal{A}$ and an LT-TRS $\mathcal{R}$, then $\mathcal{L}(\mathcal{A}_*) = (\to^*_{\mathcal{R}})(\mathcal{L}(\mathcal{A}))$ holds for the output $\mathcal{A}_*$ of the procedure.*

Proof. ($\subseteq$) Assume $t \in \mathcal{L}(\mathcal{A}_*)$. Since $\mathcal{A}_* = \mathcal{A}_i$ for some $i \geq 0$, there exists a final state p_f such that $t \vdash^*_{\mathcal{A}_i} p_f$. By Lemma 8, there exists a Σ-term s such that $s \vdash^*_{\mathcal{A}} p_f$ and $s \to^*_{\mathcal{R}} t$. Therefore, $t \in (\to^*_{\mathcal{R}})(\mathcal{L}(\mathcal{A}))$. $\mathcal{L}(\mathcal{A}_*) \supseteq (\to^*_{\mathcal{R}})(\mathcal{L}(\mathcal{A}))$ can be shown in a similar way. ∎

5. Recognizability Preserving Property

In this section, two sufficient conditions for Procedure 1 to halt are proposed. One condition is that the sets of non-marker function symbols occurring in the left-hand sides and the right-hand sides of rewrite rules are disjoint. The other condition is the one which in effect restricts the class of recognizable sets. An LT-TRS $\mathcal{R}$ which satisfies the former condition effectively preserves recognizability. Although the latter condition does not directly give a subclass of LT-TRSs which are EPR, we can show that some properties of LT-TRSs are decidable by using results which are derived concerning the latter condition. For simplicity, if we write a signature Σ as $\Sigma = \mathcal{F} \cap \mathcal{Q}$, then we implicitly assume that $\mathcal{F} \cap \mathcal{Q} = \emptyset$ and $\mathcal{Q}$ is a set of markers.

5.1 I/O-Separated LT-TRS

An LT-TRS $\mathcal{R}$ is *I/O-separated* if $\mathcal{R}$ satisfies the following condition.

Condition 1 For a signature $\Sigma = \mathcal{F} \cup \mathcal{Q}$, $\mathcal{F}$ is further divided as $\mathcal{F} = \mathcal{F}_I \cup \mathcal{F}_O$, $\mathcal{F}_I \cap \mathcal{F}_O = \emptyset$. A function symbol in $\mathcal{F}_I$ (respectively, $\mathcal{F}_O$) is called an *input symbol* (respectively, *output symbol*). For each rewrite rule

$$\begin{cases} f(q_1(x_1), \cdots, q_n(x_n)) \to q(t), \text{ or} \\ q_1(x_1) \to q(t) \end{cases}$$

in $\mathcal{R}$, $f \in \mathcal{F}_I$ and $t \in \mathcal{T}(\mathcal{F}_O, \{x_1, \cdots, x_n\})$. ∎

Lemma 10 *Let $\mathcal{R}$ be an I/O-separated LT-TRS over $\Sigma = \mathcal{F}_I \cup \mathcal{F}_O \cup \mathcal{Q}$. If Procedure 1 is executed for a TA $\mathcal{A}$ and $\mathcal{R}$, then it always halts.*

Proof. Consider a TA $\mathcal{A}_1 = (\Sigma, \mathcal{Z}_1, \Delta_1, \mathcal{P}_{final})$ constructed in Procedure 1 for a TA $\mathcal{A}$ and an I/O separated LT-TRS. Since $\mathcal{R}$ is I/O separated, every rule $g(z_1, \cdots, z_n) \to z$ which is constructed in (**S**) and added to Δ_1 satisfies $g \in \mathcal{F}_O$. Hence, when $i = 2$ no rule is constructed in (**S**). Since (**S**) adds no rules and no states, (**T**) constructs no rules. Therefore, $\mathcal{A}_2 = \mathcal{A}_1$ and Procedure 1 halts. ∎

$\mathcal{R}_1$ in Example 3 and $\mathcal{R}_2$ in Example 4 are both I/O-separated LT-TRSs.

Theorem 5 *Every I/O-separated LT-TRS effectively preserves recognizability.* ∎

A bottom-up tree transducer [6] is a well-known computation model in the theory of tree languages. For a linear bottom-up tree transducer $\mathcal{M}$, if we consider the set of states of $\mathcal{M}$ as a set of markers, $\mathcal{M}$ is a special case of I/O-separated LT-TRS. Hence, the following known property of tree transducer is obtained as a corollary.

Corollary 1 *[6] Every linear bottom-up tree transducer effectively preserves recognizability.* ∎

5.2 Marker-Bounded Sets

For a function symbol f and a ground term t, let $|t|_f$ denote the number of occurrences of f in t:

$$|g(t_1, \cdots, t_n)|_f = \begin{cases} \sum_{i=1}^{n} |t_i|_f + 1 & f = g, \\ \sum_{i=1}^{n} |t_i|_f & f \neq g. \end{cases}$$

For a subset $\Sigma' \subseteq \Sigma$ of function symbols, let $|t|_{\Sigma'} = \sum_{f \in \Sigma'} |t|_f$. For a signature $\Sigma = \mathcal{F} \cup \mathcal{Q}$, a set $T \subseteq \mathcal{T}(\Sigma)$ is *marker-bounded* if the following condition holds:

Condition 2 There exists $k \geq 0$ such that $|t|_{\mathcal{Q}} \leq k$ for each $t \in T$. ∎

Let $\Sigma' \subseteq \Sigma$ be a subset of function symbols. Consider a tree representation of a term t. Let $depth_{\Sigma'}(t)$ denote the maximum number of occurrences of function symbols in Σ' which occur in a single path from the root to a leaf in t. That is, $depth_{\Sigma'}(t)$ is defined as:

$$depth_{\Sigma'}(g(t_1, \cdots, t_n)) = \begin{cases} \max\{depth_{\Sigma'}(t_i) \mid 1 \leq i \leq n\} + 1 & g \in \Sigma', \\ \max\{depth_{\Sigma'}(t_i) \mid 1 \leq i \leq n\} & g \notin \Sigma'. \end{cases}$$

For example, for $\Sigma' = \{f, g\}, \Sigma = \{f, g, h, c\}$, $depth_{\Sigma'}(f(h(g(c)), g(g(c)))) = 3$.

Lemma 11 *Let $\mathcal{R}$ be an LT-TRS over $\Sigma = \mathcal{F} \cup \mathcal{Q}$. If $t \to_{\mathcal{R}}^* t'$ then $depth_{\mathcal{Q}}(t) \leq depth_{\mathcal{Q}}(t')$. If $\mathcal{R}$ contains no rewrite rule of which left-hand side is a constant, then $depth_{\mathcal{Q}}(t) = depth_{\mathcal{Q}}(t')$.*

Proof. The lemma can be easily proved by the form of a rewrite rule of an LT-TRS. ∎

Definition 4 (deg) For each state $z \in \mathcal{Z}_i$, let $deg(z)$ denote the number of nestings in z, which is defined as follows:

$$\begin{cases} deg(p) & = & 0\ (p \in \mathcal{P}), \\ deg([z,q]) & = & deg(z)+1\ (z \in \mathcal{Z}_i,\ q \in \mathcal{Q}), \\ deg(\langle f(t_1,\cdots,t_n)\rangle) & = & \max\{deg(\langle t_j\rangle) \mid 1 \le j \le n\} \\ & = & \max\{deg([z,q]) \mid [z,q] \text{ occurs in } f(t_1,\cdots,t_n)\} \\ & & \text{where } \max\emptyset = 0 \text{ for the empty set } \emptyset\ . \end{cases}$$

∎

Let $\mathcal{A} = (\Sigma, \mathcal{P}, \Delta, \mathcal{P}_{final})$ be a TA. A state $p \in \mathcal{P}$ is *useful* if there exists a Σ-term t, a position $o \in \mathcal{P}os(t)$ and a final state $p_f \in \mathcal{P}_{final}$ such that $t \vdash^*_{\mathcal{A}} t[o \leftarrow p] \vdash^*_{\mathcal{A}} p_f$. It is not difficult to show that for a given TA $\mathcal{A}$, we can construct a deterministic TA $\mathcal{A}'$ which satisfies $\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A})$ and has only useful states.

Lemma 12 *Procedure 1 always halts for a deterministic TA $\mathcal{A}$ and an LT-TRS $\mathcal{R}$ which satisfy the following conditions.*

(1) $\mathcal{L}(\mathcal{A})$ is marker-bounded.

(2) $\mathcal{R}$ contains no rewrite rule of which left-hand side is a constant.

Proof. Let $\mathcal{A}$ and $\mathcal{R}$ be a TA and an LT-TRS which satisfy the conditions of the lemma. Without loss of generality, assume that $\mathcal{A} = (\Sigma, \mathcal{P}, \Delta, \mathcal{P}_{final})$ is deterministic and has only useful states. The proof is by contradiction. Assume that Procedure 1 does not halt for $\mathcal{A}$ and $\mathcal{R}$. For an arbitrary constant l, the number of states z with $deg(z) < l$ and the number of rules which contain only such states are both finite. Since Procedure 1 does not halt, there exists an integer $i \ge 0$ and a state $z \in \mathcal{Z}_i$ with $deg(z) \ge l$. In particular, let k be a constant in Condition 2 for $\mathcal{L}(\mathcal{A})$, then there exists an integer $i \ge 0$ and a state $z \in \mathcal{Z}_i$ with $deg(z) = k' \ge k+1$ and z can be written as:

$$z = [\cdots[[p,q_1],q_2],\cdots,q_{k'}]\ (p \in \mathcal{P}, q_j \in \mathcal{Q}(1 \le j \le k')).$$

By (**T**) in Procedure 1, $q_1(q_2(\cdots(q_{k'}(z))\cdots)) \vdash^*_{\mathcal{A}_i} p$. By Lemma 3, there exists a Σ-term t_1 such that $t_1 \vdash^*_{\mathcal{A}_i} z$. Hence, for $t_2 = q_1(q_2(\cdots(q_{k'}(t_1))\cdots))$, we have

$$t_2 \vdash^*_{\mathcal{A}_i} p. \tag{5.1}$$

Since p is useful, there exists a Σ-term t_3, a position $o \in \mathcal{P}os(t_3)$ and a final state $p_f \in \mathcal{P}_{final}$ such that

$$t_3 \vdash^*_{\mathcal{A}} t_3[o \leftarrow p] \vdash^*_{\mathcal{A}} p_f. \tag{5.2}$$

It follows from (5.1) and the soundness of Procedure 1 that there exists a Σ-term s' such that

$$s' \vdash^*_{\mathcal{A}} p \text{ and } s' \rightarrow^*_{\mathcal{R}} t_2. \tag{5.3}$$

Let $t = t_3[o \leftarrow s']$, then $t \vdash^*_{\mathcal{A}} t_3[o \leftarrow p] \vdash^*_{\mathcal{A}} p_f \in \mathcal{P}_{final}$ by (5.2) and (5.3). Thus, $t \in \mathcal{L}(\mathcal{A})$ holds. Furthermore, since $t \rightarrow^*_{\mathcal{R}} t_3[o \leftarrow t_2]$ by (5.3) and $depth_{\mathcal{Q}}(t_3[o \leftarrow t_2]) \geq k'$, Lemma 11 implies $depth_{\mathcal{Q}}(t) \geq k' \geq k+1$. This conflicts with Condition 2 since $|t|_{\mathcal{Q}} \geq depth_{\mathcal{Q}}(t)$. Therefore, Procedure 1 halts. ∎

Theorem 6 *For any TA $\mathcal{A}$ and an LT-TRS $\mathcal{R}$ which satisfy condition (1) and (2) of Lemma 12, $(\rightarrow^*_{\mathcal{R}})(\mathcal{L}(\mathcal{A}))$ is recognizable.*

Proof. By Lemmas 9 and 12. ∎

Note that $\mathcal{R}_1$, $\mathcal{R}_2$, $\mathcal{R}_3$ in the examples satisfy condition (2) of Lemma 12. As mentioned in section 2.3, a TRS in Réty's subclass [12] is $\mathcal{C}$-EPR. The subclass of TRSs defined in [12] and the subclass of LT-TRSs satisfying condition (1) of Lemma 12 are incomparable. In fact, any non-left-linear TRS in the former class is not an LT-TRS and $\{f(q(x)) \rightarrow q(f(f(x)))\}$ belongs to the latter class but does not belong to the former class. Also the class of marker-bounded sets and $\mathcal{C}$ are incomparable since $\{q((fg)^n)(c) \mid n \geq 0\}$ is marker-bounded but not in $\mathcal{C}$ and also it is easy to see that there is a set $\mathcal{C}$ which is not marker-bounded when we identify a marker with a constructor.

Corollary 2 *For a finite set T of ground terms T and an LT-TRS $\mathcal{R}$ which contains no rewrite rule of which left-hand side is a constant, $(\rightarrow^*_{\mathcal{R}})(T)$ is recognizable.* ∎

Corollary 3 *For an LT-TRS $\mathcal{R}$ which contains no rewrite rule of which left-hand side is a constant, reachability and joinability are decidable.*

Proof. The reachability problem is to decide whether for a given TRS $\mathcal{R}$ and Σ-terms s and t, $s \to_{\mathcal{R}}^* t$ holds or not. It is obvious that $s \to_{\mathcal{R}}^* t$ if and only if $t \in (\to_{\mathcal{R}}^*)(\{s\})$. The latter condition is decidable by Lemma 1 and Corollary 2.

Decidability of joinability can easily be verified by noting that $\exists w \ : \ s \to_{\mathcal{R}}^* w$ and $t \to_{\mathcal{R}}^* w$ if and only if $(\to_{\mathcal{R}}^*)(\{s\}) \cap (\to_{\mathcal{R}}^*)(\{t\}) \neq \emptyset$. ∎

6. Conclusion

In this paper, a new subclass of TRSs called LT TRSs is defined and a sufficient conditions for an LT-TRS to effectively preserve recognizability are provided. The subclass of LT-TRSs satisfying the condition contains simple EPR-TRSs which does not belong to any of the known decidable subclasses of EPR-TRSs.

Extending the proposed class is a future study. It is not difficult to extend Procedure 1 so that a ground term can occur at an argument position of the left-hand side of a rewrite rule. For example, assume that $f(z_1, z_2) \to z \in \Delta_i$ and $f(q_1(x_1), d(c)) \to q(g(x_1))$ $(c, d, f, g \in \mathcal{F},\ q_1, q \in \mathcal{Q})$ is a rewrite rule. If $q_1(z_1') \vdash^*_{\mathcal{A}_{i-1}} z_1$ and $d(c) \vdash^*_{\mathcal{A}_{i-1}} z_2$ then construct a transition rule $g(z_1') \to [z, q]$. We conjecture that even if a marker does not occur at the right-hand side and/or at an argument position of the left-hand side of a rewrite rule (e.g. $f(x, q(y)) \to g(x, y),\ f, g \in \mathcal{F},\ q \in \mathcal{Q}$), we can modify Procedure 1 so that the procedure is still partially correct.

References

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

[2] Brained, W.S.: "Tree generating regular systems," *Inform. and control*, **14**, pp.217–231, 1969.

[3] Coquidé, J.L., Dauchet, M., Gilleron, R. and Vágvölgyi, S.: "Bottom-up tree pushdown automata: classification and connection with rewrite systems," *Theoretical Computer Science*, **127**, pp.69–98, 1994.

[4] Durand, I. and Middeldorp, A.: "Decidable call by need computations in term rewriting (extended abstract)," Proc. of CADE-14, North Queensland, Australia, LNAI **1249**, pp.4–18, 1997.

[5] Fujinaka, Y., Takai, T., Kaji, Y and Seki, H.: "Layered Transducing Term Rewriting System and Its Recognizability Preserving Property," LA Symposium, Kyoto, 2002.

[6] Gécseq, F. and Steinby, M.: *Tree Automata*, Académiai Kiadó, 1984.

[7] Gilleron, R.: "Decision problems for term rewriting systems and recognizable tree languages," Proc. of STACS'91, Hamburg, Germany, LNCS **480**, pp.148–159, 1991.

[8] Gilleron, R. and Tison, S.: "Regular tree languages and rewrite systems," *Fundamenta Informaticae*, **24**, pp.157–175, 1995.

[9] Gyenizse, P. and Vágvölgyi, S.: "Linear generalized semi-monadic rewrite systems effectively preserve recognizability," *Theoretical Computer Science*, **194**, pp.87–122, 1998.

[10] Jacquemard, F.:"Decidable approximations of term rewriting systems," Proc. of RTA96, New Brunswick, NJ, LNCS **1103**, pp.362–376, 1996.

[11] Nagaya, T. and Toyama, Y.: "Decidability for left-linear growing term rewriting systems," Proc. of RTA99, Trento, Italy, LNCS **1631**, pp.256–270, 1999.

[12] Réty, P.: "Regular sets of descendants for constructor-based rewrite systems," Proc. of LPAR'99, Tbilisi, Georgia, LNCS **1705**, pp.148–160, 1999.

[13] Salomaa, K.: "Deterministic tree pushdown automata and monadic tree rewriting systems," *J. Comput. system Sci.*, **37**, pp.367–394, 1988.

[14] Takai, T., Kaji, Y. and Seki, H.: "Right-linear finite path overlapping term rewriting systems effectively preserve recognizability," Proc. of RTA2000, Norwich, U.K., LNCS **1833**, pp.246–260, 2000.