Available online at www.sciencedirect.com
ScienceDirect
Theoretical Computer Science 379 (2007) 298–304
Theoretical Computer Science
www.elsevier.com/locate/tcs

Losing recognizability☆

Sándor Vágvölgyi*

Department of Foundations of Computer Science, University of Szeged, Szeged, Árpád tér 2, H-6720, Hungary

Received 15 May 2006; received in revised form 30 March 2007; accepted 30 March 2007

Communicated by D. Sannella

Abstract

We consider the ranked alphabet Σ consisting of a binary symbol. We give a rewrite system R over Σ such that R effectively preserves recognizability on any ranked alphabet obtained by adding finitely many nullary symbols to Σ. However, R does not preserve recognizability on the ranked alphabet obtained by adding one unary and one nullary symbol to Σ.

Keywords: Rewrite systems; Sets of descendants; Tree automata; Recognizability preserving

1. Introduction

Let Σ be a ranked alphabet, R be a rewrite system over Σ, and L be a tree language over Σ. Then $R^*_\Sigma(L) = \{p \mid q \to^*_R p \text{ for some } q \in L\}$ is the set of descendants of trees in L. A rewrite system R over Σ preserves Σ-recognizability, if for each recognizable tree language L over Σ, $R^*_\Sigma(L)$ is recognizable.

The signature $sign(R)$ of a rewrite system R is the ranked alphabet consisting of all symbols appearing in the rules of R. A rewrite system R over $sign(R)$ preserves recognizability, if for each ranked alphabet Σ with $sign(R) \subseteq \Sigma$, R preserves Σ-recognizability. Gyenizse and Vágvölgyi [6] showed that there is a linear rewrite system R over $sign(R)$ such that R preserves $sign(R)$-recognizability but does not preserve recognizability.

Let R be a rewrite system over a ranked alphabet Σ. We say that R effectively preserves Σ-recognizability if for a given tree automaton $\mathcal{B}$ over Σ, we can effectively construct a tree automaton $\mathcal{C}$ over Σ such that $L(\mathcal{C}) = R^*_\Sigma(L(\mathcal{B}))$. Let R be a rewrite system over the ranked alphabet $sign(R)$. We say that R effectively preserves recognizability if for a given ranked alphabet Σ with $sign(R) \subseteq \Sigma$ and a given tree automaton $\mathcal{B}$ over Σ, we can effectively construct a tree automaton $\mathcal{C}$ over Σ such that $L(\mathcal{C}) = R^*_\Sigma(L(\mathcal{B}))$.

In [5] Gilleron showed that it is undecidable for an arbitrary rewrite system R whether R preserves $sign(R)$-recognizability. Otto [8] showed that it is undecidable for an arbitrary rewrite system R whether or not R preserves recognizability. In spite of Gilleron's and Otto's undecidability results, we know several rewrite systems which

☆ This research was supported by the grants Széchenyi István Scholarship of the Hungarian Ministry of Education and OTKA T 046686 of the Hungarian Research Foundation.

* Tel.: +36 62 546 192; fax: +36 62 546 397.
E-mail address: vagvolgy@inf.u-szeged.hu.

0304-3975/$ - see front matter
doi:10.1016/j.tcs.2007.03.050

effectively preserve recognizability; see [1,2,6,9]. Recently, Seki, Takai, Fujinaka, Kaji [10,11] introduced layered transducing rewrite systems and showed that each I/O-separated layered transducing rewrite system effectively preserves recognizability.

Let R be a rewrite system over $sign(R)$, and let $\Sigma = \{f, \sharp\} \cup sign(R)$, where $f \in \Sigma_2 - sign(R)$ and $\sharp \in \Sigma_0 - sign(R)$. Gyenizse and Vágvölgyi [6] showed that R effectively preserves Σ-recognizability if and only if R effectively preserves recognizability. Gyenizse and Vágvölgyi [7] improved this result for left-linear rewrite systems. They showed the following. Let R be a left-linear rewrite system over $sign(R)$, and let $\Sigma = \{g, \sharp\} \cup sign(R)$, where $g \in \Sigma_1 - sign(R)$ and $\sharp \in \Sigma_0 - sign(R)$. Then R effectively preserves Σ-recognizability if and only if R effectively preserves recognizability. Furthermore, R preserves Σ-recognizability if and only if R preserves recognizability.

In this paper we give a rewrite system R over the ranked alphabet $sign(R) = \{f\}$, where f is a binary function symbol. For each $n \geq 0$, we define the ranked alphabet $\Delta^{(n)}$ by adding n constants $\sharp 1, \ldots, \sharp n$ to $sign(R)$. We show that R effectively preserves $\Delta^{(n)}$-recognizability for $n \geq 0$. On the other hand, we define the ranked alphabet Ω by adding a unary symbol g and a constant $\sharp 1$ to $sign(R)$, and show that R does not preserve Ω-recognizability. Hence R does not preserve recognizability.

This paper is divided into three sections. In Section 2, we recall the necessary notions and notation. In Section 3, we present our results.

2. Preliminaries

We recall and invent some notation, basic definitions and terminology which will be used in the rest of the paper. Nevertheless the reader is assumed to be familiar with the basic concepts of rewrite systems and of tree language theory; see [3,4,7].

Let N be the set of all nonnegative integers. N^* stands for the free monoid generated by N with empty word λ as identity element. Consider the words $\alpha, \beta, \gamma \in N^*$ such that $\alpha = \beta\gamma$. Then we say that β is a prefix of α, and write $\beta \preceq \alpha$. Furthermore, if $\alpha \neq \beta$, then β is a proper prefix of α, and we write $\beta \prec \alpha$. We say that the words $\alpha, \beta \in N^*$ are incomparable if neither $\alpha \preceq \beta$ nor $\beta \preceq \alpha$. For a word $\alpha \in N^*$, $length(\alpha)$ stands for the length of α.

A ranked alphabet is a finite set Σ in which every symbol has a unique rank. For $m \geq 0$, Σ_m denotes the set of all elements of Σ which have rank m. The elements of Σ_0 are called constants. We assume that all ranked alphabets Σ and Δ that we consider have the following property. If $f \in \Sigma_i$, and $f \in \Delta_j$, then $i = j$. In other words, f has the same rank in Σ as in Δ.

Let Y be a set of variables. The set of terms over Σ with variables in Y is denoted by $T_\Sigma(Y)$. If $Y = \emptyset$, then $T_\Sigma(Y)$ is written as T_Σ. A term $t \in T_\Sigma$ is a ground term. For any $t \in T_\Sigma$, $cons(t)$ is the set of all constant symbols that appear in t. A tree language L over Σ is any subset of T_Σ. We specify a countable set $X = \{x_1, x_2, \ldots\}$ of variables which will be kept fixed in this paper. Moreover, we put $X_m = \{x_1, \ldots, x_m\}$, for $m \geq 0$. Hence $X_0 = \emptyset$. We distinguish a subset $\overline{T}_\Sigma(X_n)$ of $T_\Sigma(X_n)$, $n \geq 0$, as follows: a tree $t \in T_\Sigma(X_n)$ is in $\overline{T}_\Sigma(X_n)$ if and only if each variable symbol of X_n appears exactly once in t. For any term $t \in T_\Sigma(X)$, $var(t)$ is the set of all variables occurring in t. Let $Y \subseteq X$. Then $AP_\Sigma(Y) = \{t \in T_\Sigma(Y) \mid Y = var(t)\}$. In other words, $AP_\Sigma(Y)$ consists of all trees $t \in T_\Sigma(Y)$ such that all elements of Y appear in t.

For example, consider the ranked alphabet $\Sigma = \{\$, f\}$, where $\$$ is a constant and f is a binary function symbol. Then $f(f(f(x_1, x_1), x_2), \$) \notin \overline{T}_\Sigma(X_2)$ because variable x_1 appears twice. Furthermore, $f(f(f(x_1, x_3), x_2), \$) \notin \overline{T}_\Sigma(X_4)$ because variable x_4 does not appear in $f(f(f(x_1, x_3), x_2), \$)$. On the other hand, $f(f(f(x_1, x_3), x_2), \$) \in \overline{T}_\Sigma(X_3)$. Furthermore, $f(f(f(x_1, x_2), x_2), \$) \notin AP_\Sigma(X_3)$ because variable x_3 does not appear in $f(f(f(x_1, x_2), x_2), \$)$. On the other hand, $f(f(f(x_1, x_1), x_2), \$) \in AP_\Sigma(X_2)$ and $f(f(f(x_1, x_3), x_2), \$) \in AP_\Sigma(X_3)$.

For any $f \in \Sigma$ and $t \in T_\Sigma(X)$, $size_f(t)$ is the number of occurrences of symbol f in tree t. That is, for any $g \in \Sigma_0$, $size_f(g) = 1$ if $f = g$ and $size_f(g) = 0$ otherwise. For any $m \geq 1$, $h \in \Sigma_m$ and $t_1, \ldots, t_m \in T_\Sigma(Y)$, $size_f(h(t_1, \ldots, t_m)) = 1 + size_f(t_1) + \cdots + size_f(t_m)$ if $h = f$, and $size_f(h(t_1, \ldots, t_m)) = size_f(t_1) + \cdots + size_f(t_m)$ otherwise.

We shall need a few functions on terms. For a term $t \in T_\Sigma(X)$, the height $height(t) \in N$ and the set of positions $POS(t) \subseteq N^*$ are defined by recursion:

(a) if $t \in \Sigma_0 \cup X$, then

$height(t) = 0$,
$POS(t) = \{\lambda\}$;

(b) if $t = f(t_1, \ldots, t_m)$ with $m \geq 1$ and $f \in \Sigma_m$, then
$height(t) = 1 + max\{height(t_i) \mid 1 \leq i \leq m\}$,
$POS(t) = \{\lambda\} \cup \{i\alpha \mid 1 \leq i \leq m \text{ and } \alpha \in POS(t_i)\}$.

We note that $height(t) = max\{length(\alpha) \mid \alpha \in POS(t)\}$.

For each $t \in T_\Sigma(X)$ and $\alpha \in POS(t)$, we introduce the subterm $t/\alpha \in T_\Sigma(X)$ of t at α and the label $lab(t, \alpha) \in \Sigma \cup X$ in t at α as follows:

(a) for $t \in \Sigma_0 \cup X$, $t/\lambda = t$ and $lab(t, \lambda) = t$;

(b) for $t = f(t_1, \ldots, t_m)$ with $m \geq 1$ and $f \in \Sigma_m$, if $\alpha = \lambda$ then $t/\alpha = t$ and $lab(t, \alpha) = f$; otherwise, if $\alpha = i\beta$ with $1 \leq i \leq m$, then $t/\alpha = t_i/\beta$ and $lab(t, \alpha) = lab(t_i, \beta)$.

For $t \in T_\Sigma(X)$, $\alpha \in POS(t)$, and $r \in T_\Sigma(X)$, we define $t[\alpha \leftarrow r] \in T_\Sigma(X)$ as follows.

(i) If $\alpha = \lambda$, then $t[\alpha \leftarrow r] = r$.

(ii) If $\alpha = i\beta$, for some $i \in N$ and $\beta \in N^*$, then $t = f(t_1, \ldots, t_m)$ with $f \in \Sigma_m$ and $t_i \in T_\Sigma(X)$, $1 \leq i \leq m$. Then $t[\alpha \leftarrow r] = f(t_1, \ldots, t_{i-1}, t_i[\beta \leftarrow r], t_{i+1}, \ldots, t_m)$.

Let $t \in T_\Sigma(X)$, $m \geq 1$, and let $\alpha_1, \ldots, \alpha_{m+1} \in POS(t)$ and $r_1, \ldots, r_{m+1} \in T_\Sigma(X)$. Then

$$t[\alpha_1 \leftarrow r_1, \ldots, \alpha_{m+1} \leftarrow r_{m+1}] = u[\alpha_{m+1} \leftarrow r_{m+1}],$$

where $u = t[\alpha_1 \leftarrow r_1, \ldots, \alpha_m \leftarrow r_m]$.

For any trees $t \in T_\Sigma(X_k)$, $k \geq 0$, $t_1, \ldots, t_k \in T_\Sigma(X)$, the term $t[t_1, \ldots, t_k]$ is produced from t by replacing each occurrence of x_i with t_i for $1 \leq i \leq k$. Let $t \in T_\Sigma(X_k)$, $k \geq 0$, $t_1, \ldots, t_k \in T_\Sigma(X)$ be arbitrary. Then we say that $t[t_1, \ldots, t_k]$ is an instance of t.

Consider the ranked alphabet $\Sigma = \{f\}$, where f is a binary function symbol. For each $n \geq 0$, the binary balanced tree of height n is denoted by bbt_n, and is defined as follows. $bbt_0 = x_1$. For each $n \geq 1$,

$$bbt_n = f(bbt_{n-1}, bbt_{n-1}[x_{2^{n-1}+1}, \ldots, x_{2^n}]).$$

For example, $bbt_1 = f(x_1, x_2)$, $bbt_2 = f(f(x_1, x_2), f(x_3, x_4))$.

Let Σ be a ranked alphabet. Then a rewrite system R over Σ is a finite subset of $T_\Sigma(X) \times T_\Sigma(X)$ such that for each $(l, r) \in R$, each variable of r also occurs in l. Elements (l, r) of R are called rules and are denoted by $l \to r$. We define the relations $\to_R$ and $\to_R^*$ in the usual way. Finally, $sign(R) \subseteq \Sigma$ is the ranked alphabet consisting of all symbols appearing in the rules of R.

Let Σ be a ranked alphabet, let R be a rewrite system over Σ, and let $L \subseteq T_\Sigma(X)$. Then $R^*_\Sigma(L) = \{p \in T_\Sigma(X) \mid q \to_R^* p \text{ for some } q \in L\}$ is the set of descendants of trees in L. Obviously, if $L \subseteq T_\Sigma$, then $R^*_\Sigma(L) \subseteq T_\Sigma$. A rewrite system R over Σ preserves Σ-recognizability, if for each recognizable tree language L over Σ, $R^*_\Sigma(L)$ is recognizable. A rewrite system R over $sign(R)$ preserves recognizability, if for each ranked alphabet Σ with $sign(R) \subseteq \Sigma$, R preserves Σ-recognizability.

3. The results

Consider the ranked alphabet $\Sigma = \{f\}$, where f is a binary function symbol. Each tree $t \in T_\Sigma(X)$ with $size_f(t) \geq 2$, is an instance of $f(f(x_1, x_2), x_3)$ or $f(x_1, f(x_2, x_3))$. In the light of this observation, we define the rewrite system S over Σ. Intuitively, the left-hand side of each rule in S is $f(f(x_1, x_2), x_3)$ or $f(x_1, f(x_2, x_3))$. Hence S rewrites those trees $t \in T_\Sigma(X)$ such that $size_f(t) \geq 2$. Applying one of the first twelve rules of S, we can move any subtree one or two levels upwards. Iterating these rewrite steps, we can shift any subtree any number of levels upwards. Applying the thirteenth and fourteenth rules finitely many times, we can build $bbt_m[t, \ldots, t]$, $m \geq 0$, the binary balanced tree of height m where t is substituted for the variables. Thus for any $Y \subseteq X$, S rewrites any tree $t \in AP_\Sigma(Y)$ with $size_f(t) \geq 2$ into any tree $z \in T_\Sigma(Y)$ in the following way. S rewrites t into $bbt_m[t, \ldots, t]$, where $m = height(z)$. Let $w \in \overline{T}_\Sigma(X)$ be such that z can be obtained from w by renaming the variables. Observe that $bbt_m[t, \ldots, t]$ is an instance of w. Hence S moves the suitable variables of t up to the frontier of w which is equal to the frontier of z.

Rewrite system S over Σ consists of the following fourteen rules:

(1) $f(f(x_1, x_2), x_3) \to f(x_1, x_2)$,
(2) $f(f(x_1, x_2), x_3) \to x_3$,
(3) $f(f(x_1, x_2), x_3) \to f(x_1, x_3)$,
(4) $f(f(x_1, x_2), x_3) \to f(x_2, x_3)$,
(5) $f(f(x_1, x_2), x_3) \to x_1$,
(6) $f(f(x_1, x_2), x_3) \to x_2$,
(7) $f(x_1, f(x_2, x_3)) \to x_1$,
(8) $f(x_1, f(x_2, x_3)) \to f(x_2, x_3)$,
(9) $f(x_1, f(x_2, x_3)) \to f(x_1, x_2)$,
(10) $f(x_1, f(x_2, x_3)) \to f(x_1, x_3)$,
(11) $f(x_1, f(x_2, x_3)) \to x_2$,
(12) $f(x_1, f(x_2, x_3)) \to x_3$,
(13) $f(f(x_1, x_2), x_3) \to f(f(f(x_1, x_2), x_3), f(f(x_1, x_2), x_3))$,
(14) $f(x_1, f(x_2, x_3)) \to f(f(x_1, f(x_2, x_3)), f(x_1, f(x_2, x_3)))$.

Lemma 3.1. *Let $t \in T_\Sigma(X)$ be such that $size_f(t) \geq 2$ and let $i \in \{1, 2\}$, $\alpha, \alpha i \in POS(t)$. Then $t \to_S t[\alpha \leftarrow t/\alpha i]$.*

Proof. We observe that $lab(t, \alpha) = f$. We distinguish twelve cases.
Case 1: $\alpha = \lambda$, $i = 1$, and $lab(t, \alpha i) = f$. Then we apply the first rule of S at position α.
Case 2: $\alpha = \lambda$, $i = 1$, and $lab(t, \alpha i) \in X$. Then we apply the seventh rule of S at position α.
Case 3: $\alpha = \lambda$, $i = 2$, and $lab(t, \alpha i) = f$. Then we apply the eighth rule of S at position α.
Case 4: $\alpha = \lambda$, $i = 2$, and $lab(t, \alpha i) \in X$. Then we apply the second rule of S at position α.
Case 5: $\alpha = \beta 1$, $i = 1$, and $lab(t, \alpha i) = f$. Then we apply the first rule of S at position α.
Case 6: $\alpha = \beta 1$, $i = 1$, and $lab(t, \alpha i) \in X$. Then we apply the third rule of S at position β.
Case 7: $\alpha = \beta 1$, $i = 2$, and $lab(t, \alpha i) = f$. Then we apply the eighth rule of S at position α.
Case 8: $\alpha = \beta 1$, $i = 2$, and $lab(t, \alpha i) \in X$. Then we apply the fourth rule of S at position β.
Case 9: $\alpha = \beta 2$, $i = 1$, and $lab(t, \alpha i) = f$. Then we apply the first rule of S at position α.
Case 10: $\alpha = \beta 2$, $i = 1$, and $lab(t, \alpha i) \in X$. Then we apply the ninth rule of S at position β.
Case 11: $\alpha = \beta 2$, $i = 2$, and $lab(t, \alpha i) = f$. Then we apply the eighth rule of S at position α.
Case 12: $\alpha = \beta 2$, $i = 2$, and $lab(t, \alpha i) \in X$. Then we apply the tenth rule of S at position β.
In all twelve cases we get $t \to_S t[\alpha \leftarrow t/\alpha i]$. □

Lemma 3.2. *Let $t \in T_\Sigma(X)$ be such that $size_f(t) \geq 2$ and let $i, j \in \{1, 2\}$, $\alpha, \alpha i, \alpha ij \in POS(t)$. Then $t \to_S t[\alpha \leftarrow t/\alpha ij]$.*

Proof. We observe that $lab(t, \alpha) = f$ and $lab(t, \alpha i) = f$. We distinguish four cases.
Case 1: $ij = 11$. Then we apply the fifth rule of S at position α.
Case 2: $ij = 12$. Then we apply the sixth rule of S at position α.
Case 3: $ij = 21$. Then we apply the eleventh rule of S at position α.
Case 4: $ij = 22$. Then we apply the twelfth rule of S at position α.
In all four cases we get $t \to_S t[\alpha \leftarrow t/\alpha ij]$. □

Lemma 3.3. *Let $t \in T_\Sigma(X)$ be such that $size_f(t) \geq 2$ and let $\alpha, \beta \in POS(t)$ be such that $\alpha \prec \beta$. Then $t \to_S^* t[\alpha \leftarrow t/\beta]$.*

Proof. Let $\gamma \in N^*$ such that $\alpha\gamma = \beta$. We proceed by induction on $length(\gamma)$.
Base Case: $length(\gamma) = 1$. By Lemma 3.1, $t \to_S t[\alpha \leftarrow t/\beta]$.
Induction Step: Assume that the lemma is true for $length(\gamma) = k$, where $k \geq 1$. We now show that the lemma is true for $length(\gamma) = k + 1$. Since $k \geq 1$, $length(\gamma) = k + 1 \geq 2$. We distinguish two cases.
Case 1: $length(\gamma) = 2$. Then by Lemma 3.2 $t \to_S t[\alpha \leftarrow t/\beta]$.
Case 2: $length(\gamma) > 2$. Then $\gamma = i\delta$ for some $i \in N$ and $\delta \in N^*$. Then $size_f(t/\alpha i) \geq 2$. By Lemma 3.1, $t \to_S t[\alpha \leftarrow t/\alpha i]$. By the induction hypothesis, $t[\alpha \leftarrow t/\alpha i] \to_S^* t[\alpha \leftarrow t/\beta]$. Hence $t \to_S^* t[\alpha \leftarrow t/\beta]$. □

Lemma 3.4. *Let $t \in T_\Sigma(X)$ be such that $size_f(t) \geq 2$. Let $m \geq 1$, and let $\alpha_1, \beta_1, \ldots, \alpha_m, \beta_m \in POS(t)$ be such that*

- *for any* $1 \leq i < j \leq m$, α_i *and* α_j *are incomparable, and*
- $\alpha_i \prec \beta_i$ *for* $1 \leq i \leq m$.

Then $t \to_S^* t[\alpha_1 \leftarrow t/\beta_1, \ldots, \alpha_m \leftarrow t/\beta_m]$.

Proof. We proceed by induction on m.

Base Case: $m = 1$. Then we are done by Lemma 3.3.

Induction Step: Assume that the lemma is true for some m. We now show that the lemma is true for $m+1$. By the induction hypothesis,

$$t \xrightarrow[S]{*} t[\alpha_1 \leftarrow t/\beta_1, \ldots, \alpha_m \leftarrow t/\beta_m].$$

Since α_m and α_{m+1} are incomparable, $length(\alpha_{m+1}) \geq 1$. Observe that $\alpha_{m+1} \in POS(t[\alpha_1 \leftarrow t/\beta_1, \ldots, \alpha_m \leftarrow t/\beta_m])$. Thus we have

$$size_f(t[\alpha_1 \leftarrow t/\beta_1, \ldots, \alpha_m \leftarrow t/\beta_m]) \geq 2.$$

Hence by Lemma 3.3,

$$[\alpha_1 \leftarrow t/\beta_1, \ldots, \alpha_m \leftarrow t/\beta_m] \xrightarrow[S]{*} t[\alpha_1 \leftarrow t/\beta_1, \ldots, \alpha_{m+1} \leftarrow t/\beta_{m+1}].$$

Thus $t \to_S^* t[\alpha_1 \leftarrow t/\beta_1, \ldots, \alpha_{m+1} \leftarrow t/\beta_{m+1}]$. □

Lemma 3.5. *Let* $t \in T_\Sigma(X)$ *such that* $size_f(t) \geq 2$. *Then* $t \to_S^* bbt_m[t, \ldots, t]$ *for* $m \geq 0$.

Proof (*Sketch*). By induction on m. In the induction step we apply the thirteenth and fourteenth rules of S. □

Lemma 3.6. *Let* $Y \subseteq X$, $t \in AP_\Sigma(Y)$, *and* $z \in T_\Sigma(Y)$. *If* $size_f(t) \geq 2$, *then* $t \to_S^* z$.

Proof. Let $m = height(z)$. By Lemma 3.5, $t \to_S^* bbt_m[t, \ldots, t]$. Let $w \in \overline{T}_\Sigma(X_k)$, $k \geq 0$, be such that $z = w[x_{i_1}, \ldots, x_{i_k}]$. Then $bbt_m = w[s_1, \ldots s_k]$ for some $s_1, \ldots, s_k \in T_\Sigma(X)$. Hence $bbt_m[t, \ldots, t] = w[v_1, \ldots v_k]$, where for each $1 \leq i \leq k$, $v_i \in T_\Sigma(Y)$ is an instance of s_i and $v_i/\alpha_i = t$ for some position $\alpha_i \in POS(v_i)$. By Lemma 3.4, $bbt_m[t, \ldots, t] \to_S^* z$. □

We now define the rewrite system R over Σ. Let $R = S \cup \{f(x_1, x_2) \to f(x_1, x_1)\}$. Observe that $sign(R) = \{f\} = \Sigma$.

Intuitively, for any $Y \subseteq X$, S rewrites in finitely many steps any tree $t \in AP_\Sigma(Y)$ with $size_f(t) \geq 2$ into any tree $z \in T_\Sigma(Y)$. Hence we get that R also rewrites in finitely many steps any tree $t \in AP_\Sigma(Y)$ with $size_f(t) \geq 2$ into any tree $z \in T_\Sigma(Y)$.

Lemma 3.7. *Let* $Y \subseteq X$ *and* $t \in AP_\Sigma(Y)$. *If* $size_f(t) \geq 2$, *then* $R_\Sigma^*(\{t\}) = T_\Sigma(Y)$.

Proof. By Lemma 3.6, for any $Y \subseteq X$ and $t \in AP_\Sigma(Y)$, if $size_f(t) \geq 2$, then $S_\Sigma^*(\{t\}) = T_\Sigma(Y)$. As $S \subseteq R$, we are done. □

For each $n \geq 0$, we define the ranked alphabet $\Delta^{(n)}$ by adding n constants, $\sharp 1, \ldots, \sharp n$ to Σ. Put it in another way, $\Delta^{(n)} = \Delta_0^{(n)} \cup \Delta_2^{(n)}$, $\Delta_0^{(n)} = \{\sharp 1, \ldots, \sharp n\}$, and $\Delta_2^{(n)} = \{f\}$. Let $\Gamma \subseteq \{\sharp 1, \ldots, \sharp n\}$. Then let $OC_{\Delta^{(n)}}(\Gamma) = \{t \in T_{\Delta^{(n)}} \mid cons(t) \cap \{\sharp 1, \ldots, \sharp n\} = \Gamma\}$. In other words, $OC_{\Delta^{(n)}}(\Gamma)$ consists of all trees $t \in T_{\Delta^{(n)}}$ such that all elements of Γ appear in t and no element of $\{\sharp 1, \ldots, \sharp n\} - \Gamma$ appears in t. From now on we consider R as a rewrite system over $\Delta^{(n)}$. Lemma 3.7 implies the following result.

Lemma 3.8. *Let* $n \geq 0$, $\Gamma \subseteq \{\sharp 1, \ldots, \sharp n\}$, *and let* $t \in OC_{\Delta^{(n)}}(\Gamma)$. *If* $size_f(t) \geq 2$, *then* $R_{\Delta^{(n)}}^*(\{t\}) = T_{\Sigma \cup \Gamma}$.

By direct inspection of R we get the following observations.

Observation 3.9. *Let* $t = f(t_1, t_2)$ *for some* $t_1, t_2 \in \Delta_0^{(n)}$. *Then* $R_{\Delta^{(n)}}^*(\{t\}) = \{f(t_1, t_2), f(t_1, t_1)\}$.

Observation 3.10. *Let* $t \in \Delta_0^{(n)}$. *Then* $R_{\Delta^{(n)}}^*(\{t\}) = \{t\}$.

Theorem 3.11. *Let $n \geq 0$, $\Delta^{(n)} = \Delta_0^{(n)} \cup \Delta_2^{(n)}$, $\Delta_0^{(n)} = \{\sharp 1, \ldots, \sharp n\}$, and $\Delta_2^{(n)} = \{f\}$. Then rewrite system R effectively preserves $\Delta^{(n)}$-recognizability.*

Proof. Let $\mathcal{A}$ be a tree automaton over $\Delta^{(n)}$. We construct a tree automaton $\mathcal{C}$ recognizing the tree language $R^*_{\Delta^{(n)}}(L(\mathcal{A}))$ as follows. We construct a set W of tree automata over $\Delta^{(n)}$. For each subset $\Gamma \subseteq \{\sharp 1, \ldots, \sharp n\}$, $OC_{\Delta^{(n)}}(\Gamma)$ is a recognizable tree language and we construct a tree automaton over $\Delta^{(n)}$ recognizing the tree language $OC_{\Delta^{(n)}}(\Gamma)$. Thus for each subset $\Gamma \subseteq \{\sharp 1, \ldots, \sharp n\}$, we decide whether $L(\mathcal{A})$ contains a tree t in $t \in OC_{\Delta^{(n)}}(\Gamma)$ such that $size_f(t) \geq 2$. If the answer is yes, then we construct a tree automaton $\mathcal{B}$ recognizing the tree language $T_{\Sigma \cup \Gamma}$, and we put $\mathcal{B}$ in the set W. For any $t_1, t_2 \in \{\sharp 1, \ldots, \sharp n\}$, we decide whether $f(t_1, t_2)$ is an element of $L(\mathcal{A})$. If the answer is yes, then we construct a tree automaton $\mathcal{B}$ recognizing the tree language $\{f(t_1, t_2), f(t_1, t_1)\}$, and we put $\mathcal{B}$ in the set W. For each $t \in \{\sharp 1, \ldots, \sharp n\}$, we decide whether t is an element of $L(\mathcal{A})$. If the answer is yes, then we construct a tree automaton $\mathcal{B}$ recognizing the tree language $\{t\}$. Then we put $\mathcal{B}$ in the set W. We construct a tree automaton $\mathcal{C}$ such that $L(\mathcal{C}) = \bigcup\{L(\mathcal{B}) \mid \mathcal{B} \in W\}$. By Lemma 3.8 and Observations 3.9, 3.10, and the construction of $\mathcal{C}$, $L(\mathcal{C}) = R^*_{\Delta^{(n)}}(L(\mathcal{A}))$. □

Theorem 3.12. *Let $\Omega = \Omega_0 \cup \Omega_1 \cup \Omega_2$, $\Omega_0 = \{\sharp 1\}$, $\Omega_1 = \{g\}$, $\Omega_2 = \{f\}$. Then rewrite system R does not preserve Ω-recognizability.*

Proof. Observe that we defined the ranked alphabet Ω by adding the nullary symbol $\sharp 1$ and the unary symbol g to $sign(R)$, that is, $\Omega = sign(R) \cup \{\sharp 1, g\}$. Now we consider R as a rewrite system over Ω. Let $g^0(\sharp 1) = \sharp 1$, and for each $n \geq 1$, let $g^n(\sharp 1) = g(g^{n-1}(\sharp 1))$. Consider the recognizable tree language $L = \{f(g^n(\sharp 1), \sharp 1) \mid n \geq 0\}$ over Ω. Let $t \in L$ be arbitrary. Then $t = f(g^n(\sharp 1), \sharp 1)$ for some $n \geq 0$. Observe that $size_f(t) = 1$. Hence no rule of S is applicable to t. On the other hand, we can apply the rule $f(x_1, x_2) \to f(x_1, x_1)$ to t: $t \to_R f(g^n(\sharp 1), g^n(\sharp 1))$. Observe that no rule of S is applicable to $f(g^n(\sharp 1), g^n(\sharp 1))$. We can apply the rule $f(x_1, x_2) \to f(x_1, x_1)$ to $f(g^n(\sharp 1), g^n(\sharp 1))$: $f(g^n(\sharp 1), g^n(\sharp 1)) \to_R f(g^n(\sharp 1), g^n(\sharp 1))$. Since we get $f(g^n(\sharp 1), g^n(\sharp 1))$ again, $R^*_\Omega(\{t\}) = \{t, f(g^n(\sharp 1), g^n(\sharp 1))\}$. In this way, $R^*_\Omega(L) = L \cup \{f(g^n(\sharp 1), g^n(\sharp 1)) \mid n \geq 0\}$. It should be clear that $R^*_\Omega(L)$ is not a recognizable tree language over Ω. Hence R does not preserve Ω-recognizability. □

By Theorem 3.12, we get the following result.

Consequence 3.13. *Rewrite system R does not preserve recognizability.*

Let R be a rewrite system over the ranked alphabet $sign(R)$, where $sign(R)$ contains at least one constant. For each $n \geq 1$, we define the ranked alphabet $\Delta^{(n)}$ by adding n constants, $\sharp 1, \ldots, \sharp n$ to $sign(R)$, where $\sharp i \notin sign(R)$ for $1 \leq i \leq n$. That is, for each $n \geq 1$, $\Delta^{(n)} = sign(R) \cup \{\sharp 1, \ldots, \sharp n\}$. One might believe the following. If the trs R effectively preserves $sign(R)$-recognizability then for any $n \geq 1$, R preserves $\Delta^{(n)}$-recognizability. We now show that this belief is not justified. Let $sign(R)$ consist of the binary symbol f and constant \$, and let the rewrite system R consist of the following three rules:

$f(\$, \$) \to \$$,
$\$ \to f(\$, \$)$,
$f(x_1, x_2) \to f(x_1, x_1)$.

By induction on $height(t)$, we can show that for each $t \in T_{sign(R)}$, $t \to^*_R \$$ and that for each $t \in T_{sign(R)}$, $\$ \to^*_R t$. Hence for all $t, s \in T_{sign(R)}$, $t \to^*_R s$. Thus for each $t \in T_{sign(R)}$, $R^*_{sign(R)}(\{t\}) = T_{sign(R)}$. Consequently for each nonempty tree language $L \subseteq T_{sign(R)}$, $R^*_{sign(R)}(L) = T_{sign(R)}$. This implies that trs R effectively preserves $sign(R)$-recognizability.

We now show that the trs R does not preserve $\Delta^{(2)}$-recognizability. To this end, for each $n \geq 0$, we define the nth comb $comb_n \in T_{\Delta^{(2)}}$ in the following way. $comb_0 = \sharp 1$. For each $n \geq 1$, $comb_n = f(comb_{n-1}, \sharp 2)$. For example, $comb_3 = f(f(f(\sharp 1, \sharp 2), \sharp 2), \sharp 2)$. Let $CO = \{comb_n \mid n \geq 0\}$. We now show that $R^*_{\Delta^{(2)}}(CO)$ is not a recognizable tree language over $\Delta^{(2)}$. To this end, consider the tree language $M = R^*_{\Delta^{(2)}}(CO) \cap T_{\Delta^{(1)}}$. By the definition of R, CO, and M, we get that $M = \{bbt_n[\sharp 1, \ldots, \sharp 1] \mid n \geq 0\}$. It is well-known that $\{bbt_n[\sharp 1, \ldots, \sharp 1] \mid n \geq 0\}$ is not a recognizable tree language over $\Delta^{(2)}$. The intersection of any two recognizable tree languages is also recognizable. If $R^*_{\Delta^{(2)}}(CO)$ were a recognizable tree language, then M would also be recognizable. Hence $R^*_{\Delta^{(2)}}(CO)$ is not a recognizable tree language over $\Delta^{(2)}$. Thus the trs R does not preserve $\Delta^{(2)}$-recognizability.

Modifying the above arguments we can show that the trs R does not preserve $\Delta^{(1)}$-recognizability either. In the definition of $comb_n$ we replace $\sharp 2$ with $\sharp 1$. The tree language K consists of all trees $s \in T_{\Delta^{(1)}}$ such that for each $\alpha \in POS(s)$, if $lab(s, \alpha) = f$ and $lab(s, \alpha 2) = \sharp 1$, then $lab(s, \alpha 1) = \sharp 1$. Then K is a recognizable tree language over $\Delta^{(1)}$. In the definition of M we replace $\Delta^{(2)}$ by $\Delta^{(1)}$ and we replace $T_{\Delta^{(1)}}$ by K. That is, $M = R^*_{\Delta^{(1)}}(CO) \cap K$. Similarly to the above proof, we can show that $M = \{bbt_n[\sharp 1, \ldots, \sharp 1] \mid n \geq 0\}$. Hence $R^*_{\Delta^{(1)}}(CO)$ is not a recognizable tree language over $\Delta^{(1)}$.

References

[1] W.S. Brainerd, Tree generating regular systems, Inf. Control 14 (1969) 217–231.

[2] J.L. Coquidé, M. Dauchet, R. Gilleron, S. Vágvölgyi, Bottom-up tree pushdown automata: classification and connection with rewrite systems, Theoret. Comput. Sci. 127 (1994) 69–98.

[3] F. Gécseg, M. Steinby, Tree Automata, Akadémiai Kiadó, Budapest, 1984.

[4] F. Gécseg, M. Steinby, Tree Languages, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, vol. 3, Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, 1996, pp. 1–69.

[5] R. Gilleron, Decision problems for term rewriting systems and recognizable tree languages, in: Proc. STACS'91, in: Lecture Notes in Computer Science, vol. 480, 1991, pp. 148–159.

[6] P. Gyenizse, S. Vágvölgyi, Linear generalized semi-monadic rewrite systems effectively preserve recognizability, Theoret. Comput. Sci. 194 (1998) 87–122.

[7] P. Gyenizse, S. Vágvölgyi, A property of left-linear rewrite systems preserving recognizability, Theoret. Comput. Sci. 242 (2000) 474–498.

[8] F. Otto, Some undecidability results concerning the property of preserving regularity, Theoret. Comput. Sci. 207 (1998) 43–72.

[9] T. Takai, Y. Kaji, H. Seki, Right-linear finite path overlapping term rewriting systems effectively preserve recognizability, in: Rewriting Techniques and Applications, in: Lecture Notes in Computer Science, vol. 1833, 2000, pp. 246–260.

[10] T. Takai, H. Seki, Y. Fujinaka, Y Kaji, Layered transducing term rewriting system and its recognizability preserving property, IEICE Trans. Inform. Syst. E86-D (2) (2003) 285–295.

[11] H. Seki, T. Takai, Y. Fujinaka, Y. Kaji, Layered transducing term rewriting system and its recognizability preserving property, in: Sophie Tison (Ed.), Rewriting Techniques and Applications, RTA 2002, Proceedings, in: Lecture Notes in Computer Science, vol. 2378, Springer, 2002, pp. 98–113.