Theoretical Computer Science 262 (2001) 229–240

Theoretical Computer Science

www.elsevier.com/locate/tcs

The component hierarchy of chain-free cooperating distributed regular tree grammars ☆

G. Dányi[a], Z. Fülöp[b,*]

[a] *Deptartment of Foundations of Computer Science, University of Szeged, H-6701 Szeged, P.O. Box 652, Hungary*

[b] *Department of Computer Science, University of Szeged, H-6701 Szeged, P.O. Box 652, Hungary*

Received 15 December 1998; revised 24 February 2000; accepted 27 April 2000
Communicated by A. Salomaa

Abstract

In this paper we show that the component hierarchy of chain-free distributed regular tree grammars cooperating with terminal strategy is infinite with respect to tree language generating capacity. More exactly, we prove that $cf\text{-}CD\text{-}RTG_{\dashv}(n) \subset cf\text{-}CD\text{-}RTG_{\dashv}(2(n-1)^2+3)$, where $n \geqslant 1$ and $cf\text{-}CD\text{-}RTG_{\dashv}(n)$ denotes the class of tree languages generated by chain-free distributed regular tree grammars of at most n components cooperating with terminal strategy. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Cooperating grammar systems; Regular tree grammars; Tree languages

1. Introduction

Cooperating distributed grammar systems were introduced in [2] as a grammatical counterpart of the blackboard model of problem solving [9]. Broadly speaking, here the agents (called also components) are Chomsky grammars and the problem to be solved is to generate a terminal string from a common initial nonterminal. Hence the current state of the solution is a sentential form, which is written on the blackboard. The agent grammars can contribute to obtaining the solution by applying some derivation steps to the sentential form. The cooperation strategy prescribes which agent becomes active, then when it becomes inactive, and the number of the derivation steps an active agent

☆ This research was partially supported by the Hungarian Scientific Research Found, Grant T030084 and by the Foundation for the Hungarian Higher Education and Research, Scholarship Magyary Zoltán.

* Corresponding author.

E-mail addresses: danyi@inf.u-szeged.hu (G. Dányi), fulop@inf.u-szeged.hu (Z. Fülöp).

can perform. The research of the topic became popular in a short time and yielded a lot of results of which a good summary and literature can be found in [3, 10].

Recently in [6], it was realized that the principle of distribution and cooperation can also be applied to tree grammars, tree automata, and tree transducers. (For notions concerning tree automata the reader is referred to [1, 4, 13], information on tree transducers can be found in [5, 11–13]) . Most recently, cooperating distributed hyperedge replacement systems were considered in [8].

In this paper we consider cooperating distributed regular tree grammars (or cd-rtg's). A cd-rtg differs from a cooperating distributed (string) grammar in that it generates trees over a ranked alphabet (instead of strings). As far as the cooperating strategy of the components is concerned, here we consider only the so-called terminal cooperating strategy (called t-mode in the literature of cooperating distributed grammar systems). This means that an agent must be active as far as it can make a derivation step on the sentential form.

Now we turn to discussing the results of this paper. We start with showing the concept of the cooperating distributed regular tree grammar.

A cd-rtg is a system $G=(A,\Sigma,\mathscr{P},S)$, where A is a finite set, called the set of nonterminal symbols, Σ is a ranked alphabet, called the terminal ranked alphabet, $\mathscr{P}=\{P_1,\ldots,P_n\}$ is a finite set of components, and $S\subseteq A$ is the set of start symbols. A component P_i consists of a finite number of rules having the form $a\rightarrow t$, where $a\in A$ and $t\in T_\Sigma(A)$ with $T_\Sigma(A)$ being the set of trees over Σ indexed by A. A sentential form is also an element of $T_\Sigma(A)$ and a terminal sentential form is a tree in T_Σ, which is the set of trees over Σ. We describe the cooperating strategy of components in terms of relations over sentential forms. For an i, where $1\leqslant i\leqslant n$, let us define the relation $\Rightarrow_i$ over sentential forms such that for every $u,v\in T_\Sigma(A)$, $u\Rightarrow_i v$ if v can be obtained from u by applying a derivation step by a rule in P_i to u, and let $\Rightarrow_i^*$ the reflexive, transitive closure of $\Rightarrow_i$. Moreover, define $u\Rightarrow_i^{\dashv} v$ if $u\Rightarrow_i^* v$ and there is no $v'\in T_\Sigma(A)$ such that $v\Rightarrow_i v'$. The terminal cooperating strategy can be described by the relations $\Rightarrow_i^{\dashv}$ with $1\leqslant i\leqslant n$, because the expression $u\Rightarrow_i^{\dashv} v$ can be interpreted as follows. The actual state of the solution (sentential form) was u, then the component P_i became active, it worked on the solution (by applying its rules to it) as far as it could and the state of the solution became v. Thus, the tree language generated by G with terminal cooperating strategy, denoted by $L_{\dashv}(G)$, consists of all trees $v\in T_\Sigma$ for which there is a derivation

$$s=u_0\Rightarrow_{G,i_1}^{\dashv} u_1\Rightarrow_{G,i_2}^{\dashv} u_2\ldots u_{l-1}\Rightarrow_{G,i_l}^{\dashv} u_l=v,$$

for some $s\in S$, $l\geqslant 1$, $1\leqslant i_1,\ldots,i_l\leqslant n$, and $u_0,\ldots,u_l\in T_{\Sigma(A)}$.

Next, we discuss the role of the so-called chain rules appearing in the components. A rule is called a chain rule if it has the form $a\rightarrow b$, where $a,b\in A$. There may be two kinds of chain rules in a component P_i, viz. inner chain rules and terminal chain rules. We call a chain rule $a\rightarrow b$ inner if there is a rule $b\rightarrow t$ in P_i. Since there is a standard construction to eliminate inner chain rules of components (in fact, it is adopted from Chomsky grammars), we can assume, without loss of generality, that there are no such chain rules in the components. A chain rule $a \to b \in P_i$ is a terminal chain rule if there is no rule of the form $b \to t$ in P_i. The application of a terminal chain rule $a \to b \in P_i$ to an occurrence of a in a sentential form $u \in T_\Sigma(A)$ can be interpreted as follows. The component P_i does not contribute to the sentential form u by extending that occurrence of a in it, rather it activates a component P_j to do so in which there is a rule of the form $b \to t$.

Nevertheless, terminal chain rules play an important role in the discussion of cooperating distributed systems. For instance, the proof of the following fundamental result is based on terminal chain rules. Let us denote by $\mathit{CD\text{-}RTG}_{\dashv}(n)$ the class of tree languages generated by cd-rtg's of at most n components, where $n \geqslant 1$, and by $\mathit{CD\text{-}RTG}_{\dashv}$ the class of tree languages generated by all cd-rtg's, in both cases with terminal cooperation strategy. Obviously, $\mathit{CD\text{-}RTG}_{\dashv}(n) \subseteq \mathit{CD\text{-}RTG}_{\dashv}(n+1)$ and $\mathit{CD\text{-}RTG}_{\dashv} = \bigcup_{n=1}^{\infty} \mathit{CD\text{-}RTG}_{\dashv}(n)$. It was shown that $\mathit{CD\text{-}RTG}_{\dashv} = \mathit{CD\text{-}RTG}_{\dashv}(3)$, which says that three components cooperating with terminal strategy are enough to generate all tree languages in $\mathit{CD\text{-}RTG}_{\dashv}$. (The proof of the corresponding celebrated result for cooperating distributed grammar systems, see e.g. [10] for this proof, was adopted to trees in [6].) We can see that terminal chain rules are used intensively in the proof, in fact, two of the three components consists of merely terminal chain rules. Hence, the natural question arises whether the result $\mathit{CD\text{-}RTG}_{\dashv} = \mathit{CD\text{-}RTG}_{\dashv}(3)$ remains valid for cd-rtg's in which there are no terminal chain rules.

In this paper, we prove that the same statement does not hold if we do not allow the presence of terminal chain rules in the components, i.e., if every rule contributes to the solution by at least one terminal symbol. We call such cd-rtg's chain-free cd-rtg's and prove that, for every $n \geqslant 1$, $\mathit{cf\text{-}CD\text{-}RTG}_{\dashv}(n) \subset \mathit{cf\text{-}CD\text{-}RTG}_{\dashv}(2(n-1)^2+3)$, where the prefix *cf* means chain-free. Hence, the component hierarchy of chain-free cd-rtg's does not collapse, i.e., there is no finite number of chain-free components which would be sufficient to generate all tree languages in $\mathit{cf\text{-}CD\text{-}RTG}_{\dashv}$.

2. Definitions

If $\to$ is a binary relation over a set H, then we write $a \to b$ for $(a,b) \in \to$. We denote by $\to^+$ and by $\to^*$ the transitive and the reflexive, transitive closure of $\to$, respectively. Moreover, for two elements $a, b \in H$, we define $a \to^{\dashv} b$ if and only if $a \to^* b$ and there is no $c \in H$ such that $b \to c$.

By a *ranked alphabet* we mean a finite, nonempty set Σ in which every symbol σ has a unique *rank* in the set of nonnegative integers. A symbol of rank m is called an m-ary symbol. For each $m \geqslant 0$, the set of m-ary symbols in Σ is denoted by Σ_m.

Let Σ be a ranked alphabet and A be an arbitrary set disjoint with Σ. The *set of terms* (or rather *trees*) *over Σ indexed by A* is denoted by $T_\Sigma(A)$ and defined to be the smallest set U satisfying the following two conditions:
(i) $A \cup \Sigma_0 \subseteq U$.
(ii) If $n \geqslant 1$, $\sigma \in \Sigma_n$ and $t_1, \ldots, t_n \in U$, then the term $\sigma(t_1, \ldots, t_n)$ is also in U.

The set $T_\Sigma(\emptyset)$ is written as T_Σ and is called the *set of trees over* Σ. A *tree language* L *over* Σ is a subset of T_Σ.

The *height* of a tree $t \in T_\Sigma$ is denoted by height(t) and is defined as follows. If $t \in A \cup \Sigma_0$, then height$(t) = 0$, otherwise, if $t = \sigma(t_1, \ldots, t_n)$, then height$(t) = 1 + \max\{\text{height}(t_i) \mid 1 \leqslant i \leqslant n\}$.

Let A be a finite set and let us fix an order $a_1, \ldots, a_m$ of the elements of A. Moreover, let $t \in T_\Sigma(A)$ and $\boldsymbol{L} = (L_1, \ldots, L_m)$ be a tuple of subsets of $T_\Sigma(A)$. We define the tree language $t \cdot \boldsymbol{L}$ as follows:

(i) For every $\sigma \in \Sigma_0$, $\sigma \cdot \boldsymbol{L} = \{\sigma\}$.

(ii) For every $1 \leqslant i \leqslant m$, $a_i \cdot \boldsymbol{L} = L_i$.

(iii) If $t = \sigma(t_1, \ldots, t_n)$ for some $n \geqslant 1, \sigma \in \Sigma_n$ and $t_1, \ldots, t_n \in T_\Sigma(A)$, then $t \cdot \boldsymbol{L} = \{\sigma(u_1, \ldots, u_n) \mid u_i \in t_i \cdot \boldsymbol{L}$, for every $1 \leqslant i \leqslant n\}$.

Informally, $t \cdot \boldsymbol{L}$ consists of all trees that are obtained by substituting, for every $1 \leqslant i \leqslant m$, elements of L_i for a_i in t such that different occurrences of a_i can be substituted by different elements of L_i.

For a tree language $K \subseteq T_\Sigma(A)$, we define $K \cdot \boldsymbol{L} = \bigcup_{t \in K} t \cdot \boldsymbol{L}$. Moreover, for a tuple $\boldsymbol{K} = (K_1, \ldots, K_m)$, we put $\boldsymbol{K} \cdot \boldsymbol{L} = (K_1 \boldsymbol{L}, \ldots, K_m \boldsymbol{L})$. It is easy to check that $\cdot$ is associative, i.e., $(\boldsymbol{K} \cdot \boldsymbol{L}) \cdot \boldsymbol{M} = \boldsymbol{K} \cdot (\boldsymbol{L} \cdot \boldsymbol{M})$, for every $\boldsymbol{K}, \boldsymbol{L}$, and $\boldsymbol{M}$.

A *cd-rtg*, cf. [6], is a tuple $G = (A, \Sigma, \mathscr{P}, S)$, where

- A is a finite set, called *the set of nonterminal symbols*,
- Σ is a ranked alphabet with $A \cap \Sigma = \emptyset$,
- $\mathscr{P}$ is a finite set of *rule sets* $P_1, \ldots, P_n$, where each rule set $P_i \in \mathscr{P}$ is a finite set of *rules* of the form $a \to r$ with $a \in A$ and $r \in T_\Sigma(A)$, and
- $S \subseteq A$ is the set of *start symbols*.

The set P_i is called the ith *component of G*. We put dom$(P_i) = \{a \in A \mid a \to r \in P_i$ for some $r \in T_\Sigma(A)\}$. Moreover, a rule of the form $a \to b$, where $a, b \in A$, is called a *chain rule*. A cd-rtg with no chain rules is called *chain-free* and the expression chain-free cd-rtg is abbreviated to cf-cd-rtg.

We define *the tree language generated by G with terminal cooperating strategy* (cf. t-mode in [2, 3, 10].

First, for every i with $1 \leqslant i \leqslant n$, the relation $\Rightarrow_i \subseteq T_\Sigma(A) \times T_\Sigma(A)$ is defined in the following way. For any two trees $t, u \in T_\Sigma(A)$, we have $t \Rightarrow_i u$, if and only if there is a rule $a \to r \in P_i$ and u is obtained from t by substituting r for an occurrence of a in t.

The *tree language generated by G with terminal cooperating strategy* (or $\dashv$-strategy) is denoted by $L_\dashv(G)$ and consists of all trees $t \in T_\Sigma$ for which there is a derivation

$$s = u_0 \Rightarrow_{i_1}^{\dashv} u_1 \Rightarrow_{i_2}^{\dashv} u_2 \ldots u_{l-1} \Rightarrow_{i_l}^{\dashv} u_l = t,$$

for some $s \in S$, $l \geqslant 1, 1 \leqslant i_1, \ldots, i_l \leqslant n$, and $u_0, \ldots, u_l \in T_{\Sigma(A)}$.

The class of all tree languages generated by cd-rtg's of at most n components with $\dashv$-cooperation strategy will be denoted by $CD\text{-}RTG_\dashv(n)$, where $n \geqslant 1$. Moreover, $CD\text{-}RTG_\dashv = \bigcup_{n=1}^{\infty} CD\text{-}RTG_\dashv(n)$. The corresponding classes generated by chain-free cd-rtg's are denoted by $cf\text{-}CD\text{-}RTG_\dashv(n)$ and $cf\text{-}CD\text{-}RTG_\dashv$.

A cd-rtg with one component is called a *regular tree grammar* and is written as $G=(A,\Sigma,P,S)$ where $P=P_1$. Moreover, we write $\Rightarrow$ for $\Rightarrow_1$ and $L(G)$ for $L_{\dashv}(G)$. Then it is easy to see that $L(G)=\{t\in T_\Sigma \mid s\Rightarrow^* t$ for some $s\in S\}$. A tree language generated by a regular tree grammar is called a *regular tree language*.

There is a very close connection between regular tree languages and sets of derivation trees of context-free grammars. Hence there is a counterpart of Bar–Hillel's pumping lemma for regular tree languages.

Lemma 2.1 (cf. Lemma II. 10.1 of Gecseg, Steinby [7]). *Let $L\subseteq T_\Sigma$ be a regular tree language, then there exists an integer $N>0$, depending on L, which satisfies the following conditions. For every $t\in L$, if $height(t)>N$, then there are trees $t_1,t_2\in T_\Sigma(\{*\})$ and $t_3\in T_\Sigma$, where $*\notin\Sigma$, $t_2\neq *$, and $*$ occurs exactly once both in t_1 and t_2, such that $t=t_1\overset{*}{\leftarrow}t_2\overset{*}{\leftarrow}t_3$ and, for every $n\geqslant 0$, the tree $t_1(\overset{*}{\leftarrow}t_2)^n\overset{*}{\leftarrow}t_3$ is also in L.*

(Here $t_1\overset{*}{\leftarrow}t_2$ is the tree obtained by substituting t_2 for the only occurrence of $*$ in t_1. Moreover, we define $t_1(\overset{*}{\leftarrow}t_2)^0=t_1$ and $t_1(\overset{*}{\leftarrow}t_2)^i=t_1\overset{*}{\leftarrow}t_2(\overset{*}{\leftarrow}t_2)^{i-1}$, for every $i\geqslant 1$. Notice that $t_1(\overset{*}{\leftarrow}t_2)^1=t_1\overset{*}{\leftarrow}t_2$.)

3. The hierarchy theorem

In this section we show that the set $\{cf\text{-}CD\text{-}RTG_{\dashv}(n) \mid n\geqslant 1\}$ contains infinite proper hierarchy with respect to the inclusion. This means that, broadly speaking, the more components a cf-cd-rtg has, the higher its generating power is.

We prove our result in the following way. First, for each $k\geqslant 1$, we define a tree language $L^{(k)}$ and we show that it can be generated by a cf-cd-rtg of $2k+1$ components. Next, we prove that, for arbitrarily given $k,n\geqslant 1$, if $L^{(k)}$ can be generated by a cf-cd-rtg of n components, then $(n-1)^2\geqslant k$ must hold. (Actually, this is the key lemma to the proof of the infiniteness of the hierarchy.)

The hierarchy theorem comes from these results immediately as follows. On the one hand, for every $k,n\geqslant 1$, if $k>(n-1)^2$, then $L^{(k)}$ cannot be generated by a cf-cd-rtg of n components, i.e., $L^{((n-1)^2+1)}\notin cf\text{-}CD\text{-}RTG_{\vdash}(n)$. On the other hand, $L^{((n-1)^2+1)}\in cf\text{-}CD\text{-}RTG_{\vdash}(2((n-1)^2+1)+1)$. Hence we obtain $cf\text{-}CD\text{-}RTG_{\vdash}(n)\subset cf\text{-}CD\text{-}RTG_{\vdash}((2(n-1)^2+3))$.

Before the formal proof, we need some preparations. Let $G=(A,\Sigma,\mathscr{P},S)$ be a cd-rtg with $\mathscr{P}=\{P_1,\dots,P_n\}$ in the sequel.

A tree $u\in T_\Sigma(A)$ is a *sentential form* of G, if there are $s\in S$, $l\geqslant 1$, $1\leqslant i_1,\dots,i_l\leqslant n$, and $u_0,\dots,u_l\in T_{\Sigma(A)}$, such that

$$s=u_0\Rightarrow_{i_1}^{\dashv}u_1\Rightarrow_{i_2}^{\dashv}u_2\dots u_{l-1}\Rightarrow_{i_l}^{*}u_l=u.$$

The sentential form u is called *component closing*, if, in addition, $u_{l-1}\Rightarrow_{i_l}^{\dashv}u_l=u$ holds. In what follows, we abbreviate the expression component closing just to *closing*. We call *u first* closing sentential form, if it is a closing sentential form with $l=1$. Finally,

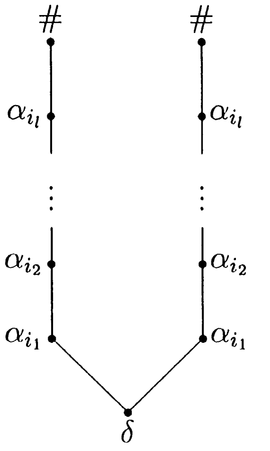

Fig. 1. The tree $\delta(\alpha_{i_1}(\ldots\alpha_{i_l}(\#)\ldots),\alpha_{i_1}(\ldots\alpha_{i_l}(\#)\ldots))$ in $L^{(k)}$.

a closing sentential form u is said to be *useful*, if there are $k \geqslant 0$, $1 \leqslant j_1,\ldots,j_k \leqslant n$, $v_0,\ldots,v_k \in T_{\Sigma(A)}$, and $t \in T_\Sigma$, such that

$$u = v_0 \Rightarrow^{\dashv}_{j_1} v_1 \Rightarrow^{\dashv}_{j_2} v_2 \ldots v_{k-1} \Rightarrow^{\dashv}_{j_k} v_k = t,$$

i.e., a terminal sentential form can be derived from u.

Let $a_1,\ldots,a_m$ be a fixed order of elements of A, i.e., the set of nonterminal symbols of G. For every i with $1 \leqslant i \leqslant n$, we define the tuple $\boldsymbol{K}^{(i)} = (K_1^{(i)},\ldots,K_m^{(i)})$, such that, for every j with $1 \leqslant j \leqslant m$, $K_j^{(i)} = \{t \in T_\Sigma(A) \mid a_j \Rightarrow^{\dashv}_i t\}$. We call $\boldsymbol{K}^{(i)}$ the $\dashv$-tuple generated by P_i. Note that, if $a_j \notin \mathrm{dom}(P_i)$, then $K_j^{(i)} = \{a_j\}$, otherwise, $K_j^{(i)} \subseteq T_\Sigma(A - \mathrm{dom}(P_i))$ implying $a_j \notin K_j^{(i)}$. Moreover, if $a_j \in \mathrm{dom}(P_i)$, then $K_j^{(i)}$ is a regular tree language generated by the regular tree grammar $G_j^{(i)} = (\mathrm{dom}(P_i), \Sigma \cup (A - \mathrm{dom}(P_i)), P_i, \{a_j\})$, i.e., $K_j^{(i)} = L(G_j^{(i)})$. Then $L_\dashv(G)$ can also be expressed as

$$L_\dashv(G) = \cup(a \cdot \boldsymbol{K}^{(i_1)} \cdot \cdots \cdot \boldsymbol{K}^{(i_l)} \cap T_\Sigma \mid a \in S, l \geqslant 1,\ 1 \leqslant i_1,\ldots,i_l \leqslant n).$$

Now we are ready to prove the result of this section. First, for each $k \geqslant 1$, we define the language $L^{(k)}$.

Definition 3.1. Let $k \geqslant 1$. We put $\Sigma^{(k)} = \{\alpha_1,\ldots,\alpha_k,\#\}$, where the rank of $\alpha_1,\ldots,\alpha_k$ is 1 and the rank of # is 0. Moreover, let $\delta \notin \Sigma^{(k)}$ with rank 2. The tree language $L^{(k)}$ is defined as

$$L^{(k)} = \{\delta(t,t) \mid t \in T_{\Sigma^{(k)}}\},$$

i.e., $L^{(k)}$ is the set of trees of the form $\delta(\alpha_{i_1}(\ldots\alpha_{i_l}(\#)\ldots),\alpha_{i_1}(\ldots\alpha_{i_l}(\#)\ldots))$, where $l \geqslant 0$ and $1 \leqslant i_1,\ldots,i_l \leqslant k$ (see Fig. 1).

Now we make two simple, but important observations. The first one gives a lower bound of the number of trees in $L^{(k)}$, which are not higher than a given $h \geqslant 1$.

Observation 3.2. *Let $h \geqslant 1$, then the number of the trees of height h in $L^{(k)}$ is k^{h-1}. Hence, k^{h-1} is also a lower bound of the number of the trees in $L^{(k)}$ of height at most h.*

The second observation shows that $L^{(k)}$ can be generated by a cf-cd-rtg of $2k+1$ components. (We note we guess that $L^{(k)}$ cannot be generated by a cf-cd-rtg of less than $2k+1$ components, however, we cannot prove this yet.)

Observation 3.3. *For every $k \geqslant 1$, $L^{(k)} \in$ cf-CD-RTG$_{\dashv}(2k+1)$ holds.*

Proof. It is easy to check that the following cf-cd-rtg of $2k+1$ components generates $L^{(k)}$:

$$\begin{aligned}
R_1 &: a_0 \to \delta(a_1, a_1),\ \ a_2 \to \alpha_1(a_1)\\
R_2 &: a_1 \to \alpha_1(a_2)\\
&\ \ \vdots\\
R_{k+1} &: a_1 \to \alpha_k(a_2)\\
R_{k+2} &: a_2 \to \alpha_2(a_1)\\
&\ \ \vdots\\
R_{2k} &: a_2 \to \alpha_k(a_1)\\
R_{2k+1} &: a_1 \to \#,\ \ a_2 \to \# \qquad \square
\end{aligned}$$

Now, we are ready to prove the key statement of this section. The following lemma gives a lower bound of the number of components of such cf-cd-rtg's, which generate $L^{(k)}$.

Lemma 3.4. *Let $n, k \geqslant 1$ be arbitrarily fixed integers. If a cf-cd-rtg of n components generates $L^{(k)}$, then $(n-1)^2 \geqslant k$ holds.*

Proof. Suppose that a cf-cd-rtg $G = (A, \Sigma, \mathscr{P}, S)$ generates $L^{(k)}$, i.e., $L_{\vdash}(G) = L^{(k)}$. Clearly, we can assume that $\Sigma = \Sigma^{(k)} \cup \{\delta\}$.

Let $a_1, \ldots, a_m$ and $P_1, \ldots, P_n$ be arbitrarily fixed orders of the elements of A and $\mathscr{P}$, respectively. Denote by $\boldsymbol{K}^{(1)}, \ldots, \boldsymbol{K}^{(n)}$ the $\vdash$-tuples generated by the components of G, respectively, to the fixed order.

Let us investigate, how G derives a tree in $L^{(k)}$. By definition, a derivation begins with a start symbol $a \in S$. Then a component P_i of G is chosen with $a \in \mathrm{dom}(P_i)$ and this will be the first active component. By applying rules of P_i under $\vdash$-strategy, a first closing sentential form $\delta(t_1, t_2) \in T_\Sigma(A)$ is derived from a, such that $a \Rightarrow_i^{\vdash} \delta(t_1, t_2)$. In fact, $\delta(t_1, t_2) \in K_j^{(i)}$, where j is defined by $a = a_j$.

Hence, each tree $t \in L_{\vdash}(G)$ can be associated with an expression of the form

$$\{\delta(t_1, t_2)\} \cdot \boldsymbol{K}^{(i_1)} \cdot \cdots \cdot \boldsymbol{K}^{(i_l)},$$

where $l \geqslant 0$ and $\delta(t_1,t_2) \in T_\Sigma(A)$ is a useful first closing sentential form and $t \in \{\delta(t_1,t_2)\} \cdot \boldsymbol{K}^{(i_1)} \cdot \dots \cdot \boldsymbol{K}^{(i_l)}$. If this inclusion holds, then we also say that the expression $\{\delta(t_1,t_2)\} \cdot \boldsymbol{K}^{(i_1)} \cdot \dots \cdot \boldsymbol{K}^{(i_l)}$ generates t.

Observation 3.5. *There is only a finite number of useful first closing sentential forms.*

Proof. We prove by contradiction. Let us assume that the opposite of the statement holds. This, by the above reasoning, means that there are P_i and a_j, such that $a_j \in S$ and $K_j^{(i)}$ contains an infinite number of useful first closing sentential forms. (Note that all elements of $K_j^{(i)}$ are first closing sentential forms, but there can also be not useful ones in it.)

Since $K_j^{(i)}$ is a regular tree language, there is an integer N satisfying the conditions described in Lemma 2.1. Moreover, since there are infinite number of useful first closing sentential forms in $K_j^{(i)}$, there must be a useful first closing sentential form $\delta(t_1,t_2) \in K_j^{(i)}$ with $t_1,t_2 \in T_{\Sigma^{(k)}}(A)$, such that $\mathrm{height}(\delta(t_1,t_2)) > N$. Since $\delta(t_1,t_2)$ is useful, there is a sequence $\boldsymbol{K}^{(i_1)},\dots,\boldsymbol{K}^{(i_l)}$, such that

$$\{\delta(t_1,t_2)\} \cdot \boldsymbol{K}^{(i_1)} \cdots \boldsymbol{K}^{(i_l)} \cap T_\Sigma \neq \emptyset.$$

Moreover, at least one of t_1 and t_2 can be pumped (cf. Lemma 2.1), such that the resulted trees are all useful first closing sentential forms.

Assume that t_1 can be pumped in the sequel. (The proof for t_2 can be performed in the same way.) This means that there are trees $u_1,u_2 \in T_{\Sigma^{(k)}}(\{*\})$ with $u_2 \neq *$ and $u_3 \in T_{\Sigma^{(k)}}(A)$, such that $t_1 = u_1 \overset{*}{\leftarrow} u_2 \overset{*}{\leftarrow} u_3$ and, for every $j \geqslant 0$, the tree $\delta(u_1(\overset{*}{\leftarrow} u_2)^j \overset{*}{\leftarrow} u_3, t_2)$ is a useful first closing sentential form. This implies that, for any $j > 1$, there are asymmetric trees in $\{\delta(u_1(\overset{*}{\leftarrow} u_2)^j \overset{*}{\leftarrow} u_3, t_2)\} \cdot \boldsymbol{K}^{(i_1)} \cdots \boldsymbol{K}^{(i_l)} \cap T_\Sigma$ and hence in $L_\vdash(G)$, see Fig. 2. On the other hand, we assumed $L_\vdash(G) = L^{(k)}$. This is a contradiction because $L^{(k)}$ consists of only symmetric trees. □

Let us denote the number of useful first closing sentential forms by p. Notice that p is determined by G.

Observation 3.6. *For every useful first closing sentential form $\delta(t_1,t_2)$ and expression of the form $\{\delta(t_1,t_2)\} \cdot \boldsymbol{K}^{(i_1)} \cdot \dots \cdot \boldsymbol{K}^{(i_l)}$, there is at most one $t \in L_\vdash(G)$, such that $t \in \{\delta(t_1,t_2)\} \cdot \boldsymbol{K}^{(i_1)} \cdot \dots \cdot \boldsymbol{K}^{(i_l)}$, i.e., every such expression generates at most one element of $L^{(k)}$.*

Proof. This can also be proved by contradiction as follows. Let us assume that there are $t,t' \in L_\vdash(G)$, such that $t \neq t'$ and $t,t' \in \{\delta(t_1,t_2)\} \cdot \boldsymbol{K}^{(i_1)} \cdot \dots \cdot \boldsymbol{K}^{(i_l)}$. Then there are nonterminals $b_1,b_2 \in A$ and trees $u_1,u_2,u_1',u_2' \in T_\Sigma$, such that t and t' can be written in the form $t = \delta(t_1 \overset{b_1}{\leftarrow} u_1, t_2 \overset{b_2}{\leftarrow} u_2)$ and $t' = \delta(t_1 \overset{b_1}{\leftarrow} u_1', t_2 \overset{b_2}{\leftarrow} u_2')$, respectively. (Note that the trees u_1 and u_1' are in the component of $\boldsymbol{K}^{(i_1)} \cdot \dots \cdot \boldsymbol{K}^{(i_l)}$ belonging to b_1, and similarly, u_2 and u_2' are in the component of $\boldsymbol{K}^{(i_1)} \cdot \dots \cdot \boldsymbol{K}^{(i_l)}$ belonging to b_2.) Since $t \neq t'$, at least one of $u_1 \neq u_1'$ and $u_2 \neq u_2'$ holds. Suppose that $u_1 \neq u_1'$. (The proof for $u_2 \neq u_2'$

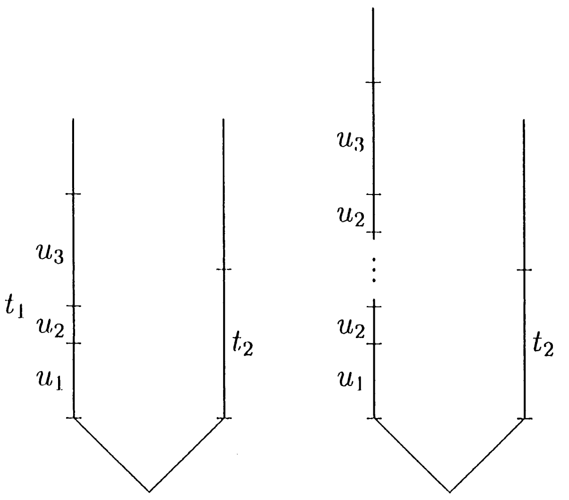

Fig. 2. Pumping u_2 in $t_1 = u_1 \overset{*}{\leftarrow} u_2 \overset{*}{\leftarrow} u_3$.

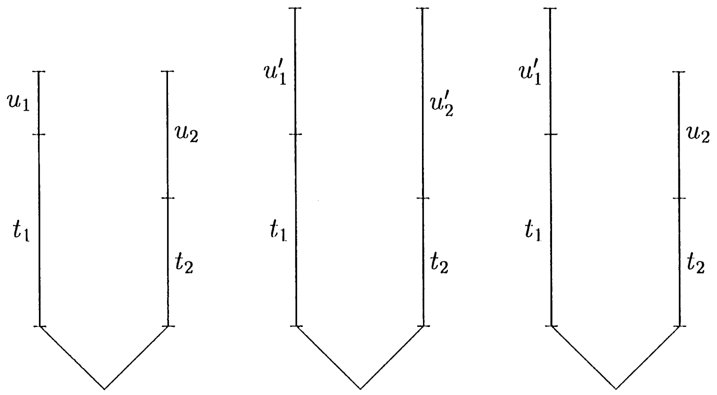

Fig. 3. Replacing u_1 by u'_1.

is similar.) Then the asymmetric tree $\delta(t_1 \overset{b_1}{\leftarrow} u'_1, t_2 \overset{b_2}{\leftarrow} u_2)$ is also in $L_\vdash(G)$ (see Fig. 3), which again contradicts $L_\vdash(G) = L^{(k)}$. $\square$

By Observation 3.6, we obtain that different trees in $L_\vdash(G)$ are generated by different expressions.

Now let $t \in L_\vdash(G)$ and let $\{\delta(t_1,t_2)\} \cdot \boldsymbol{K}^{(i_1)} \cdot \dots \cdot \boldsymbol{K}^{(i_l)}$ be an expression, which generates t. We say that $\{\delta(t_1,t_2)\} \cdot \boldsymbol{K}^{(i_1)} \cdot \dots \cdot \boldsymbol{K}^{(i_l)}$ is *optimal* for t, if there is no $1 \leqslant j \leqslant l$, such that $\{\delta(t_1,t_2)\} \cdot \boldsymbol{K}^{(i_1)} \cdot \dots \cdot \boldsymbol{K}^{(i_{j-1})} \cdot \boldsymbol{K}^{(i_{j+1})} \cdot \dots \cdot \boldsymbol{K}^{(i_l)}$ also generates t. Moreover, an expression is optimal, if it is optimal for some $t \in L_\vdash(G)$. (Notice that, by Observation 3.6, an optimal expression generates exactly one tree in $L_\vdash(G)$. However, a tree $t \in L_\vdash(G)$ can be generated by more optimal expressions.)

Observation 3.7. *If* $\{\delta(t_1,t_2)\}\cdot \boldsymbol{K}^{(i_1)}\cdot \cdots \cdot \boldsymbol{K}^{(i_l)}$ *is an optimal expressions, then there is no* $1 \leqslant j < l$ *such that* $i_j = i_{j+1}$.

Proof. Let $\{\delta(t_1,t_2)\}\cdot \boldsymbol{K}^{(i_1)}\cdot \cdots \cdot \boldsymbol{K}^{(i_l)}$ be an optimal expression for a tree $t \in L_{\vdash}(G)$. If, for some $1 \leqslant j < l$, the equality $i_j = i_{j+1}$ holds, then, since we work with $\vdash$-strategy, $\boldsymbol{K}^{(i_{j+1})}(=\boldsymbol{K}^{(i_j)})$ cannot contribute to t. Hence it can be omitted and thus the expression is not optimal. □

Observation 3.8. *Let* $h \geqslant 1$. *Then* $p(n-1)^{2h}$ *is an upper bound for the number of trees in* $L_{\vdash}(G)$ *of height at most* h.

Proof. Since every such tree is generated by an optimal expression, moreover, an expression generates at most one tree, it is sufficient to give an upper bound for the number of optimal expressions generating trees of height at most h.

Therefore, let $t \in L_{\vdash}(G)$, such that height$(t) \leqslant h$, and let $\{\delta(t_1,t_2)\}\cdot \boldsymbol{K}^{(i_1)}\cdot \cdots \cdot \boldsymbol{K}^{(i_l)}$ be an optimal expression with $t \in \{\delta(t_1,t_2)\}\cdot \boldsymbol{K}^{(i_1)}\cdot \cdots \cdot \boldsymbol{K}^{(i_l)}$. Since the expression is optimal, moreover, G is chain-free, each $\boldsymbol{K}^{(i_j)}$ in that expression must add at least one symbol to the generated tree. Moreover, since a tree of height h consists of $2h+1$ symbols and δ appears already in the first closing sentential form, we obtain that $l \leqslant 2h$ must hold.

Next, we observe that every optimal expression of the above form, generating a tree $t \in L_{\vdash}(G)$ with height$(t) \leqslant h$, can be extended to an (not necessarily optimal) expression of the form $\{\delta(t_1,t_2)\}\cdot \boldsymbol{K}^{(i_1)}\cdot \cdots \cdot \boldsymbol{K}^{(i_{2h})}$, such that

(i) the extended expression has the property described in Observation 3.7 (i.e. there are no identical $\boldsymbol{K}^{(i_j)}$'s after each other in it) and

(ii) it generates t.

(Notice that the components $\boldsymbol{K}^{(i_{l+1})},\ldots,\boldsymbol{K}^{(i_{2h})}$ do not contribute to t.)

Next we determine the number of the extended expressions generating trees of height at most h. Since there are no identical neighbours in the subexpression $\boldsymbol{K}^{(i_1)}\cdot \cdots \cdot \boldsymbol{K}^{(i_{2h})}$, every element $\boldsymbol{K}^{(i_j)}$ can be chosen in $n-1$ different ways and thus the number of subexpressions $\boldsymbol{K}^{(i_1)}\cdot \cdots \cdot \boldsymbol{K}^{(i_{2h})}$ obtained by extension is $(n-1)^{2h}$. Hence the number of extended expressions generating trees of height at most h is $p(n-1)^{2h}$.

This is surely an upper bound of the number of trees generated by G of height at most h, too. □

Now we can finish the proof of the lemma. Recall that $L(G)_{\vdash} = L^{(k)}$. Then, for every $h \geqslant 1$, G also generates all trees in $L^{(k)}$ of height at most h. Comparing the upper bound of the number of such trees generated by G with the lower bound of the number of the ones in $L^{(k)}$, we obtain that, for every $h \geqslant 1$,

$$p(n-1)^{2h} \geqslant k^{h-1}$$

must hold (cf. Observations 3.8 and 3.2).

Since k, $n-1$, and p are positive integers, we can equivalently present the above inequality as

$$\ln p + h \ln(n-1)^2 \geqslant h \ln k - \ln k.$$

Rearranging the components, we get

$$\ln p + \ln k \geqslant h(\ln k - \ln (n-1)^2).$$

Observe that the left-hand side is a positive constant value for the given G and $L^{(k)}$, meanwhile h can be an arbitrarily big positive integer. Hence, the inequality holds for each $h \geqslant 1$, if and only if $\ln k - \ln(n-1)^2 \leqslant 0$, that is,

$$(n-1)^2 \geqslant k.$$

With this we complete the proof of the lemma. □

The above lemma implies that, for instance, $L^{(5)}$ cannot be generated by a cf-cd-rtg of three components, $L^{(10)}$ cannot be generated by a cf-cd-rtg of four components, etc. On the other hand, by Observation 3.3, $L^{(5)}$ and $L^{(10)}$ can be generated by cf-cd-rtg's of 11 and 21 components, respectively. Thus $L^{(11)} - L^{(3)} \neq \emptyset$ and $L^{(21)} - L^{(4)} \neq \emptyset$ hold.

The main result of this paper is an immediate consequence of Lemma 3.4.

Theorem 3.9. *The component hierarchy of cf-cd-rtg's with* $\vdash$*-strategy is infinite. In fact*, $cf\text{-}CD\text{-}RTG_{\vdash}(n) \subset cf\text{-}CD\text{-}RTG_{\vdash}(2(n-1)^2+3)$, *where* $n \geqslant 1$.

Proof. It is enough to show that

$$cf - CD - RTG_{\vdash}(2(n-1)^2+3) - cf - CD - RTG_{\vdash}(n) \neq \emptyset$$

holds for every $n \geqslant 1$.

Let $n \geqslant 1$ be arbitrarily fixed and let $k = (n-1)^2 + 1$. By Lemma 3.4, $L^{(k)} \notin cf\text{-}CD - RTG_{\vdash}(n)$, because $k > (n-1)^2$. Moreover, by Observation 3.3, $L^{(k)} \in cf\text{-}CD\text{-}RTG_{\vdash}(2k+1) = cf\text{-}CD\text{-}RTG_{\vdash}(2(n-1)^2+3)$. □

4. Conclusion

In this paper we showed that the component hierarchy of chain-free distributed regular tree grammars cooperating with terminal strategy is infinite with respect to tree language generating capacity. More exactly, we proved that, for every $n \geqslant 1$, $cf\text{-}CD\text{-}RTG_{\vdash}(n) \subset cf\text{-}CD\text{-}RTG_{\vdash}(2(n-1)^2+3)$. This is despite the fact that the same hierarchy for unrestricted distributed regular tree grammars collapses at $n = 3$.

However, we could not prove that

(i) whether the inclusion $CD\text{-}RTG_{\vdash}(n) \subseteq CD\text{-}RTG_{\vdash}(n+1)$ is strict for every $n \geqslant 1$, and

(ii) whether $CD\text{-}RTG_{\vdash} = cf\text{-}CD\text{-}RTG_{\vdash}$, i.e., whether chain-free cd-rtg's are as power ful as (unrestricted) cd-rtg's with respect to tree language generating capacity.

These questions remain open.

Moreover, it is still also open, if the component hierarchy of chain free cooperating distributed (string) grammar systems is also infinite or not. The investigation of this problem seems to require a technique, which is different from the one applied in this paper. The main reason of this is that a context-free string grammar can write terminals in any part of a sentential form, while a regular tree grammar can extend the sentential form only at its leaves.

References

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

[2] E. Csuhaj-Varjú, J. Dassow, On cooperating distributed grammar systems, J. Inform. Process. Cybernet, EIK 26 (1990) 49–63.

[3] E. Csuhaj-Varjú, J. Dassow, J. Kelemen, G. Paun, Grammar Systems: A Grammatical Approach to Distribution and Cooperation, Gordon and Breach, London, 1994.

[4] P.J. Downey, Formal languages and recursion schemes, Ph.D. Thesis, Harvard University, 1973.

[5] J. Engelfriet, Bottom-up and top-down tree transformations – a comparison, Math. Systems Theory 9 (1975) 198–231.

[6] Z. Fülöp, Cooperating tree processing devices, Comput. Artificial Intelligence 18 (1999) 37–71.

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

[8] S. Maneth, Cooperating distributed hyperedge replacement grammars, Grammars 1 (1999) 193–208.

[9] P.H. Nii, Blackboard systems, in: A. Barr, P.R. Cohen, E.A. Feigenbaum (Eds.), The Handbook of Artificial Intelligence, Addison-Wesley, Reading, MA, 1989.

[10] G. Paun, Grammar Systems: A Grammatical Approach to Distribution and Computation, Lecture Notes in Computer Science, vol. 944, Proc. ICALP 95, Springer, Berlin, 1995, pp. 429–443.

[11] J.W. Thatcher, Generelized^2 sequential machines, J. Comput. System Sci. 4 (1970) 339–367.

[12] J.W. Thatcher, Tree automata: an informal survey, in: A.V. Aho (Ed.), Currents in the Theory of Computing, Prentice-Hall, Englewood Cliffs, NJ, 1973, pp. 143–172.

[13] J.W. Thatcher, J.B. Wright, Generalized finite automata theory with an application to a decision problem of second order logic, Math. Systems Theory 2 (1968) 57–81.