ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume **12**, Number 2, 2009, 175–189

Cooperating Distributed Grammar Systems with Permitting Grammars as Components

Erzsébet CSUHAJ-VARJÚ[1,2], Tomáš MASOPUST[3], György VASZIL[1]

[1] Computer and Automation Research Institute, Hungarian Academy of Sciences
Kende u. 13-17, 1111 Budapest, Hungary
E-mail: {csuhaj,vaszil}@sztaki.hu

[2] Department of Algorithms and Their Applications, Faculty of Informatics
Eötvös Loránd University
Pázmány Péter sétány 1/c, 1117 Budapest, Hungary

[3] Faculty of Information Technology, Brno University of Technology
Božetěchova 2, Brno 61266, Czech Republic
E-mail: masopust@fit.vutbr.cz

Abstract. This paper studies cooperating distributed grammar systems working in the terminal derivation mode where the components are variants of permitting grammars. It proves that although the family of permitting languages is strictly included in the family of random context languages, the families of random context languages and languages generated by permitting cooperating distributed grammar systems in the above mentioned derivation mode coincide. Moreover, if the components are so-called left-permitting grammars, then cooperating distributed grammar systems in the terminal mode characterize the class of context-sensitive languages, or if erasing rules are allowed, the class of recursively enumerable languages. Descriptional complexity results are also presented. It is shown that the number of permitting components can be bounded, in the case of left-permitting components with erasing rules even together with the number of nonterminals.

1. Introduction

The tools of formal language theory are often used to describe phenomena occurring in natural languages or in living developmental systems. To study these topics, it

is sometimes desirable to construct generative mechanisms which are easy to handle but still have the required descriptive power. Context-free grammars have several properties which make them convenient and useful as a basis when more powerful generative mechanisms are created.

A way of increasing the power of context-free grammars is studied by the theory of regulated rewriting which, given a context-free grammar, adds some type of control mechanism for the restriction of the use of the rules in such a way that some of the usual context-free derivations are eliminated. Thus, the obtained words form a subset of the original context-free language generated by the grammar without the controlling mechanism. Since these subsets can also be non-context-free languages, these mechanisms are more powerful than the context-free grammars. (See [3] and [5] for more details.)

Random context grammars, a class of regulated rewriting devices motivated by these ideas, operate by controlling the use of the rules based on the presence or absence of certain symbols in the sentential form [13]. Each rule of a random context grammar has an associated permitting and forbidding set of nonterminals, and roughly speaking, the rule can only be applied to sentential forms which contain all permitting symbols, but do not contain any of the forbidding symbols. Random context grammars characterize the class of recursively enumerable languages, but if erasing rules are not allowed, they are able to generate only a subclass of the class of context-sensitive languages, see [3].

Another possibility of describing non-context-free languages with rewriting mechanisms using context-free rules is provided by the theory of grammar systems, see [2, 4] for more information. Cooperating distributed grammar systems (CD grammar systems in short) were introduced in [1] to model the so-called blackboard type of problem solving architectures. A CD grammar system consists of several component grammars, these are the problem solving agents, which generate a common sentential form by taking turns in the rewriting process. The sentential form represents the blackboard, the current state of the problem, which the agents might modify according to a certain protocol until the solution appears, that is, until a terminal string is generated. In this paper we consider the cooperation protocol called terminal mode (or t-mode) of derivation which is known to increase the generative power of the components: CD grammar systems with context-free components working in the t-mode of derivation are more powerful than context-free grammars, they characterize the class of ET0L languages, the languages generated by so-called extended tabled interactionless Lindenmayer systems.

In the following we study the generative mechanisms obtained by using restricted variants of random context grammars as components of a cooperating distributed grammar system, and investigate how much of the power that was lost by the different type of restrictions can be recovered by the use of several grammars and the t-mode of cooperating derivation.

For CD grammar systems with forbidding grammars as components (these are random context grammars with no permitting symbols) the problem was investigated in [9, 10], it was shown that forbidding CD grammar systems have the same power as random context grammars (with or without erasing rules), while so-called left-forbidding

grammar CD grammar systems characterize the class of recursively enumerable or context sensitive languages depending on whether erasing rules are allowed or not. (In the case of left-forbidding grammars, the non-appearance of the forbidding symbols is checked only on the left side of the rewritten nonterminal symbol.) These latter results are especially interesting since left-forbidding random context grammars without the cooperating distributed framework characterize only the class of context-free languages.

Here we are interested in permitting and in left-permitting grammars, variants of random context grammars with no forbidding symbols. If erasing rules are not allowed, permitting random context grammars are strictly weaker than random context grammars, see [6], but their power is regained using the cooperating distributed framework and the terminal mode of derivation. We will show that the power of CD grammar systems with permitting grammars equals the power of general random context grammars in both cases, with or without erasing rules. Moreover, in the case of left-permitting components, the use of CD grammar systems not only compensates for the possibility of checking the non-appearance of forbidding symbols, but even more, these systems characterize the class of context sensitive or recursively enumerable languages, depending on whether erasing rules are allowed or not, thus, in the non-erasing case, they are more powerful than random context grammars.

Finally, we also present some descriptional complexity results. We show that the number of components can be bounded, and moreover, if left-permitting components with erasing rules are used, this can be done together with bounding the number of nonterminals at the same time.

2. Preliminaries

This paper assumes that the reader is familiar with formal language theory, more information can be found in [3, 11, 12]. An alphabet V is a finite set of symbols, V^* represents the free monoid generated by V where the unit of V^* is denoted by λ, and $V^+ = V^* - \{\lambda\}$. The cardinality of a finite set A is denoted by $|A|$, $|w|$ denotes the length of the word $w \in V^*$, and $alph(w) \subseteq V$ is the set of symbols occurring in w. Let $\mathscr{L}(CF)$, $\mathscr{L}(CS)$, and $\mathscr{L}(RE)$ denote the families of context-free, context-sensitive, and recursively enumerable languages, respectively.

A *random context grammar* is a quadruple $G = (N, T, P, S)$, where N is the alphabet of nonterminals, T is the alphabet of terminals such that $N \cap T = \emptyset$, $S \in N$ is the start symbol, and P is a finite set of productions of the form $(A \to x, Per, For)$, where $A \to x$ is a context-free production, i.e., $A \in N$, $x \in V^*$ with V denoting $N \cup T$, and $Per, For \subseteq N$. For two words $u, v \in V^*$ and the production $(A \to x, Per, For) \in P$, the relation $uAv \Rightarrow uxv$ holds, provided that

- $Per \subseteq alph(uv)$, and
- $alph(uv) \cap For = \emptyset$.

The transitive closure, and the reflexive and transitive closure of $\Rightarrow$ is denoted by $\Rightarrow^+$ and $\Rightarrow^*$, respectively. The language generated by G is defined as $L(G) = \{w \in T^* : S \Rightarrow^* w\}$.

A *permitting grammar* is a random context grammar $G = (N, T, P, S)$, where for each production $(A \to x, Per, For) \in P$ it holds that $For = \emptyset$.

A *left-permitting grammar* is a quadruple $G = (N, T, P, S)$, where N, T, P, and S are the same as in the case of a permitting grammar. For $u, v \in V^*$ and $(A \to x, Per, \emptyset) \in P$, the relation $uAv \Rightarrow uxv$ holds, provided that

- $Per \subseteq alph(u)$,

that is, the permitting symbols must appear on the left side of the rewritten nonterminal. The language generated by G is defined as $L(G) = \{w \in T^* : S \Rightarrow^* w\}$.

In case of (left-) permitting grammars, the forbidding sets are omitted; i.e., we write $(A \to x, Per)$ instead of $(A \to x, Per, \emptyset)$, and such a production is called a (left-) permitting production.

The families of languages generated by random context grammars, permitting grammars, and left-permitting grammars are denoted by $\mathscr{L}(RC)$, $\mathscr{L}(P)$, and $\mathscr{L}(\ell P)$, respectively, and by $\mathscr{L}(RC - \lambda)$, $\mathscr{L}(P - \lambda)$, and $\mathscr{L}(\ell P - \lambda)$ if they are generated by grammars without erasing rules.

As we have already mentioned, $\mathscr{L}(P-\lambda) \subset \mathscr{L}(RC-\lambda)$, permitting random context grammars are strictly weaker then random context grammars (if erasing rules are not allowed). A similar relation is not known for left-permitting grammars, but clearly, $\mathscr{L}(CF) \subseteq \mathscr{L}(\ell P)$, any context-free grammar is also a left-permitting grammar. The following example also demonstrates that $\mathscr{L}(CF) \subset \mathscr{L}(\ell P)$, left-permitting grammars are more powerful that context-free grammars. The precise relation between the families $\mathscr{L}(P)$ and $\mathscr{L}(\ell P)$ is also an open question.

Example 1. Let $G = (\{S, A, C, A', C'\}, \{a, b, c\}, P, S)$ be a left-permitting grammar, and let P contain the following productions:

$$\begin{aligned} P \;=\; \{&(S \to AC, \emptyset), (A \to aA'b, \emptyset), (A \to ab, \emptyset), (A' \to A, \emptyset), \\ &(C \to cC', \{A'\}), (C \to c, \emptyset), (C' \to C, \{A\})\}. \end{aligned}$$

It is not difficult to see that $L(G) = \{a^n b^n c^m : n \geq m \geq 1\}$ which is a non-context-free language.

Now we present the notion of a cooperating distributed grammar system, see [1, 2, 4] for more details.

A *cooperating distributed grammar system* (a CD grammar system in short) is a construct $\Gamma = (N, T, P_1, P_2, \ldots, P_n, S)$, where N is a set of nonterminals, T is a set of terminals, $N \cap T = \emptyset$, $S \in N$ is the start symbol, and for $1 \leq i \leq n$, each component P_i is a set of context-free productions. For $u, v \in V^*$ and $1 \leq i \leq n$, let $u \Rightarrow_{P_i} v$ denote a rewriting step performed by the application of a production from P_i. We say that u derives v by productions from P_i in the terminal mode, written as $u \Rightarrow^t_{P_i} v$, if $u = u_0 \Rightarrow_{P_i} u_1 \Rightarrow_{P_i} \ldots \Rightarrow_{P_i} u_k = v$ for some $k \geq 1$, and there is no $w \in V^*$ such that $v \Rightarrow_{P_i} w$. The language generated by Γ in the terminal derivation mode (or t-mode) is defined as $L(\Gamma) = \{w \in T^* :$ there is an $\ell \geq 1$ such that $S \Rightarrow^t_{P_{i_1}} w_1 \Rightarrow^t_{P_{i_2}} w_2 \Rightarrow^t_{P_{i_3}} \ldots \Rightarrow^t_{P_{i_\ell}} w_\ell = w,\ 1 \leq i_j \leq n,\ 1 \leq j \leq \ell\}$.

The family of languages generated in the terminal mode by CD grammar systems with n components where $n \geq 1$, is denoted by $\mathscr{L}(CD, CF, n)$, or $\mathscr{L}(CD, CF-\lambda, n)$ if the components contain only non-erasing productions.

It is known that $\mathscr{L}(CD, CF, 2) = \mathscr{L}(CD, CF-\lambda, 2) = \mathscr{L}(CF)$, but

$$\mathscr{L}(CD, CF, n) = \mathscr{L}(CD, CF-\lambda, n) = \mathscr{L}(ET0L), \text{ for } n \geq 3,$$

where $\mathscr{L}(ET0L)$ denotes the class of languages generated by extended tabled interactionless Lindenmayer systems (see for example [8]).

A *permitting (or left-permitting) cooperating distributed grammar system* is a construct $\Gamma = (N, T, P_1, P_2, \ldots, P_n, S)$, where the components P_i, $1 \leq i \leq n$, are sets of permitting productions. The language generated by Γ in the t-mode is defined in the same way as for CD grammar systems, the productions are used as in permitting (left-permitting) grammars.

The families of languages generated by permitting (or left-permitting) CD grammar systems with n components where $n \geq 1$, working in the t-mode are denoted by $\mathscr{L}(CD,P,n)$ (or $\mathscr{L}(CD,\ell P,n)$), and $\mathscr{L}(CD,P-\lambda,n)$ (or $\mathscr{L}(CD,\ell P-\lambda,n)$) if the components are non-erasing. If the number of components is not considered, n is omitted.

3. The power of permitting CD grammar systems

Now we show that the class of languages generated by permitting CD grammar systems in the t-mode coincides with the class of random-context languages. First we need a lemma.

Lemma 1. *For each random context grammar G, there is a random context grammar G' with $L(G) = L(G')$ and productions of the form $(A \to x, Per, For)$ satisfying the property that $A \notin For$.*

Proof. Let $G = (N, T, P, S)$ be a random context grammar. Let us construct the random context grammar $G' = (N \cup N', T, P', S)$ as follows. Let $N' = \{A' : A \in N\}$, $N \cap N' = \emptyset$, and $P' = \{(A \to A', \emptyset, N'), (A' \to x, Per, For) : (A \to x, Per, For) \in P\}$. Then, G and G' generate the same language and G' satisfies the required property.□

Now we demonstrate that every (non-erasing) random context grammar can be simulated by a (non-erasing) permitting CD grammar system working in the t-mode. As a result, cooperating permitting components are able to compensate the absence of forbidding sets, which is surprising because considering the cooperating distributed framework only with context-free components, the generated language family (the family of ET0L languages) is properly included in the family of forbidding languages (see [3, 5]).

Theorem 2. *$\mathscr{L}(RC) \subseteq \mathscr{L}(CD,P)$ and $\mathscr{L}(RC-\lambda) \subseteq \mathscr{L}(CD,P-\lambda)$.*

Proof. Let $G = (N, T, P, S)$ be a random context grammar satisfying the property from Lemma 1, and let the productions of P be labeled by numbers from 1 to $n = |P|$.

Let

$$\Gamma = (N \cup N' \cup N_{[]}, T, P_0, P_1, \ldots, P_n, S)$$

be a permitting CD grammar system where $N' = \{A' : A \in N\}$ and $N_{[]} = \{[x] : (A \to x, Per, For) \in P, A \in N\}$, with the set of productions

$$P_0 = \{(A' \to A, \emptyset), ([x] \to x, \emptyset) : A' \in N', [x] \in N_{[]}\},$$

and for each rule $i : (A \to x, Per, For) \in P$, $1 \leq i \leq n$,

$P_i = \{(A \to [x], Per), ([x] \to [x], \{[x]\})\ \} \cup \{(X \to X, \emptyset), (Y \to Y', \emptyset) : X \in For, Y \in N - For\}$.

We prove that $L(\Gamma) = L(G)$.

To prove that $L(G) \subseteq L(\Gamma)$, consider $uAv \Rightarrow uxv$, a derivation step of a derivation in G according to a production $i : (A \to x, Per, For) \in P$. We show that $uAv \Rightarrow^t_{P_i} h(u)[x]h(v) \Rightarrow^t_{P_0} uxv$, where h is a homomorphism from $(T \cup N)$ to $(T \cup N')$ defined by $h(X) = X'$, $X \in N$, and $h(a) = a$, $a \in T$. Clearly, by productions from P_i, the required occurrence of A may be replaced with $[x]$ and all other nonterminal symbols can be primed because there are no symbols from *For* in uv. After these replacements, as there is only one symbol $[x]$ in $h(u)[x]h(v)$, component P_i is blocked; i.e., we have $uAv \Rightarrow^t_{P_i} h(u)[x]h(v)$. Now, by productions from P_0, we obtain $h(u)[x]h(v) \Rightarrow^t_{P_0} uxv$.

To prove the inclusion $L(\Gamma) \subseteq L(G)$, consider a t-mode derivation step of Γ, $\alpha_1 \Rightarrow^t_{P_i} \alpha_2$, for some $\alpha_1 \in (N \cup T)^*$, and P_i, $1 \leq i \leq n$, with the rule $i : (A \to x, Per, For) \in P$. Assume that $\alpha_1 = u_0 A u_1 A u_2 \ldots A u_r$, for some $r \geq 0$ ($r = 0$ implies that there is no A in α_1), where $u_0 u_1 \ldots u_r \in (V - (For \cup \{A\}))^*$. This follows from the fact that the derivation is terminating: If there appeared a symbol $X \in For$ in the sentential form, the derivation would keep replacing X with X for ever. Thus, $\alpha_2 = h(u_0)\delta_1 h(u_1)\delta_2 h(u_2) \ldots \delta_r h(u_r)$, where $\delta_j \in \{[x], A'\}$, $1 \leq j \leq r$. Notice that there is no applicable production in P_i if and only if there is no more than one occurrence of $[x]$ in α_2; otherwise, $[x]$ is replaced with $[x]$ for ever.

Now, the only component able to rewrite α_2 is P_0, thus, in another t-mode derivation step we get $\alpha_2 \Rightarrow^t_{P_0} \alpha_3 = uxv$, where for u, v, the property that $uAv = \alpha_1$ holds. This means that either $\alpha_1 = \alpha_3$, or the results of the rewriting process using the two consecutive t-mode derivation steps of P_i and P_0 of Γ can also be achieved by the rule $i : (A \to x, Per, For)$ of G. □

On the other hand, the following theorem proves that cooperating distributed systems of permitting components in the t-mode of derivation are at most as powerful as random context grammars.

Theorem 3. *$\mathscr{L}(CD,P) \subseteq \mathscr{L}(RC)$ and $\mathscr{L}(CD,P-\lambda) \subseteq \mathscr{L}(RC-\lambda)$.*

Proof. Let $\Gamma = (N, T, P_1, P_2, \ldots, P_n, S)$ be a permitting CD grammar system with n components, $n \geq 1$. Construct the random context grammar $G = (N', T \cup \{c\}, P', S')$, where c is a new symbol, $c \notin T$. The set of nonterminals is $N' = N \cup \{X' : X \in N\} \cup N''$ where

$$\begin{aligned} N'' \quad = \quad & \{\ [Q_i], \langle p, Q_i\rangle, [i]\ :\ Q_i \subseteq \{r, r'\ :\ r \in P_i\},\ p \in Q_i,\ \text{where } |Q_i| = |P_i| \\ & \text{and}\ \ p_j, p'_k \in Q_i \text{ implies } p_j \neq p_k, 1 \leq j, k \leq |P_i|,\ 1 \leq i \leq n\}. \end{aligned}$$

P' is constructed as follows.

1. For each $(A \to x, Per) \in P_i$, add

   (a) $(A \to x, Per \cup \{[i]\}, \emptyset)$

   to P'.

2. For $1 \leq i, \ell \leq n$, add

   (a) $(S' \to S[i], \emptyset, \emptyset)$,

   (b) $([i] \to [\{p_1, p_2, \ldots, p_{|P_i|}\}], \emptyset, \emptyset)$ where $P_i = \{p_1, p_2, \ldots, p_{|P_i|}\}$,

   (c) $([\{p'_1, p'_2, \ldots, p'_{|P_i|}\}] \to [\ell], \emptyset, \emptyset)$ where $P_i = \{p_1, p_2, \ldots, p_{|P_i|}\}$, and

   (d) $([i] \to c, \emptyset, \emptyset)$,

   to P'.

3. For all Q_i defined as above and for all $p_j = (A_j \to x_j, Per_j) \in (Q_i \cap P_i)$, $1 \leq i \leq n$, $1 \leq j \leq |P_i|$, add also to P' the rules

   (a) $([Q_i] \to [(Q_i - \{p_j\}) \cup \{p'_j\}], \emptyset, \{A_j\})$,

   (b) $([Q_i] \to \langle p_j, Q_i \rangle, \{A_j\}, \emptyset)$,

   (c) $(A_j \to A'_j, \{\langle p_j, Q_i \rangle\}, \{A'_j\})$,

   (d) $(\langle p_j, Q_i \rangle \to [p_j, Q_i], \{A'_j\}, \{X\})$, where $X \in Per_j$,

   (e) $(A'_j \to A_j, \emptyset, \emptyset)$,

   (f) $([p_j, Q_i] \to [(Q_i - \{p_j\}) \cup \{p'_j\}], \emptyset, \{A'_j\})$.

Informally, productions constructed in (1a) simulate the applications of the corresponding productions of the ith component of Γ. By (2b) and (2d), the grammar stops the simulation of Γ's productions and either it starts to verify that there is no applicable production of the ith component of Γ (see productions constructed in (3a) to (3f)), or it finishes the derivation by replacing $[i]$ with c. In (3a) to (3f), for each production $p_j : (A_j \to x_j, Per_j)$ of the component P_i, the grammar verifies that p_j is not applicable as follows.

(3a): If there is no A_j in the sentential form, then p_j is not applicable.

(3b)–(3f): If there is A_j, i.e., the sentential form is uA_jv, for some u, v, but at least one of the permitting symbols is not occurring in uv, i.e., there is no X for some $X \in Per_j$ in uv, then p is not applicable. Finally,

(2c): if there is no applicable production, the random context grammar can change the simulated component.

Formally, we shall prove that $L(\Gamma) \cdot c = L(G)$. To prove that $L(\Gamma) \cdot c \subseteq L(G)$, consider a successful derivation of Γ. Such a derivation is of the form $S \Rightarrow^t \alpha_1 \Rightarrow^t \alpha_2 \Rightarrow^t \ldots \Rightarrow^t \alpha_k$ where $\alpha_k \in T^*$ for some $k \geq 1$. Consider a sub-derivation of the form $\alpha_m \Rightarrow^t_{P_i} \alpha_{m+1}$, for some $1 \leq i \leq n$, $1 \leq m < k$, according to a sequence s of productions from P_i. Consider a sentential form $\alpha_m[i]$. Then, by the sequence of the corresponding productions to the productions of s (see (1a) of the construction), by a production constructed in (2b), by productions constructed in (3a) to (3f), and by (2c), we have

$$\alpha_m[i] \Rightarrow^* \alpha_{m+1}[i] \Rightarrow \alpha_{m+1}[P_i] \Rightarrow^{|P_i|} \alpha_{m+1}[P_i'] \Rightarrow \alpha_{m+1}[\ell]$$

in G, where $P_i' = \{p_1', p_2', \ldots, p_{|P_i|}'\}$ for $P_i = \{p_1, p_2, \ldots, p_{|P_i|}\}$, and where P_ℓ is a component which continues the derivation of Γ from α_{m+1} to α_{m+2}. The proof now proceeds by induction.

If $m+1 = k$, then instead of (2b), a production constructed under (2d) is applied; i.e., we have

$$\alpha_m[i] \Rightarrow^* \alpha_{m+1}[i] \Rightarrow \alpha_{m+1}c\,.$$

As $\alpha_{m+1} \in T^*$, we have $\alpha_{m+1}c \in L(G)$.

To prove that $L(G) \subseteq L(\Gamma) \cdot c$, consider a successful derivation of G. Such a derivation is of the form $S' \Rightarrow S[i] \Rightarrow^* w[j] \Rightarrow wc$ for some $1 \leq i, j \leq n$, $w \in T^*$. Consider one derivation step of this derivation, say $\alpha \Rightarrow \beta$. Then, there are the following possibilities:

1. The derivation step is according to a production $(A \to x, W \cup \{[i]\})$, i.e., $\alpha = uAv[i]$ for some u, v, and $W \subseteq \mathit{alph}(uv)$. Clearly, $(A \to x, W) \in P_i$ is applicable to uAv in Γ; i.e., $uAv \Rightarrow uxv$ in Γ.

2. If $\alpha = u[i]$ for some u, i, and $([i] \to c, \emptyset, \emptyset)$ is applied, then no other production is applicable, i.e., $\beta = uc \in T^* \cdot c$.

3. If $\alpha = u[i]$ for some u, i, and $([i] \to [P_i], \emptyset, \emptyset)$ is applied, then by the construction of G, if a terminal word can be derived from $u[P_i]$, then no production of P_i is applicable to u. First, only productions constructed in (3a), (3b), and (2c) are applicable. If a production constructed in (3a) is applied, replacing a nonterminal with p by a nonterminal with p', $p : (A \to x, W) \in P_i$, then there is no A in u; i.e., p is not applicable in Γ. If a sequence of productions constructed in (3b)–(3f) is applied, replacing a nonterminal with p by a nonterminal with p', then $A \in \mathit{alph}(u)$, i.e., $u = u'Av'$, for some u', v', but there is $X \in W$ such that $X \notin \mathit{alph}(u'v')$; i.e., $W \not\subseteq \mathit{alph}(u'v')$. Again, p is not applicable in Γ. Finally, a production of the form $[\{p_1', p_2', \ldots, p_{k_i}'\}] \to [\ell], \emptyset, \emptyset)$ is applied; i.e., all symbols of P_i are primed, which means that there is no applicable production in P_i.

   Thus, the derivation of G is of the form $u[i] \Rightarrow u[P_i] \Rightarrow^* u[\ell]$ and there is no rule of P_i applicable to u.

The proof now proceeds by induction.

The above considerations show that $L(G) = L(\Gamma)\cdot c$. This is sufficient to prove our statement since (non-erasing) random context grammars are closed under restricted homomorphisms (see [3, Lemma 1.3.3]), that is, there is a random context grammar H such that $L(\Gamma) = L(H)$. □

The following result is an immediate consequence of the previous two theorems.

Corollary 4. *$\mathscr{L}(CD,P) = \mathscr{L}(RC)$ and $\mathscr{L}(CD,P-\lambda) = \mathscr{L}(RC-\lambda)$.*

4. The power of left-permitting CD grammar systems

Now we show that if we have left-permitting components, we can characterize the class of recursively enumerable languages (or $\mathscr{L}(CS)$ if erasing rules are not allowed).

Theorem 5. $\mathcal{L}(CD, \ell P - \lambda) = \mathcal{L}(CS)$ and $\mathcal{L}(CD, \ell P) = \mathcal{L}(RE)$.

Proof. It is obvious that $\mathcal{L}(CD, \ell P-\lambda) \subseteq \mathcal{L}(CS)$ and $\mathcal{L}(CD, \ell P) \subseteq \mathcal{L}(RE)$ holds, thus we only need to show the opposite inclusions.

We start with $\mathcal{L}(CS) \subseteq \mathcal{L}(CD, \ell P - \lambda)$. Let $L \subseteq T^*$ be a context-sensitive language. Then $L = \bigcup_{a\in T}(L_a \cdot a)$ where $L_a = L/\{a\}$, the right quotient of L with the language $\{a\}$. Since $\mathcal{L}(CS)$ is closed under right quotient, and since $\mathcal{L}(CD, \ell P - \lambda)$ is obviously closed under union, it is sufficient to show that $(L \cdot a) \in \mathcal{L}(CD, \ell P - \lambda)$ for any context-sensitive $L \subseteq T^*$ and $a \in T$.

Let $G = (N, T, P, S)$ be a context-sensitive grammar. In the following we show that $L(G) \cdot a \in \mathcal{L}(CD, \ell P - \lambda)$. Without the loss of generality we may assume that S does not occur on the right side of the productions of P, and that all productions are of one of the following forms: $AB \to CD$, $A \to BC$, $A \to a$, with $A, B, C, D \in N$, $a \in T$. Let $P = P_1 \cup P_2$ where P_1 denotes the set of context-free productions, P_2 denotes the set of non-context-free productions of P.

We construct a left-permitting CD grammar system Γ such that $L(\Gamma) = L(G) \cdot a$ where $a \in T$. For the sake of conciseness, we use the notation $(P_{r,1}, P_{r,2}, P_{r,3})_{r\in P_2}$ to denote the $3 \cdot |P_2|$ components containing a three-tuple of production sets for each rule in P_2.

Let $\Gamma = (N', T, P_{ini}, P_{CF}, P'_{CF}, (P_{r,1}, P_{r,2}, P_{r,3})_{r\in P_2}, P_4, P_{fin}, S')$ with

$$N' = N \cup \bigcup_{r\in P}(N_r \cup N'_r \cup N''_r \cup N'''_r \cup N^{iv}_r \cup N^{v}_r \cup N^{vi}_r) \cup \{\$, \$_r, \$'_r, \$''_r \ : \ r \in P\}$$

where for any $r \in P$, $N_r = \{X_r \ : \ X \in N\}$ and for any $\alpha \in \{', '', ''', ^{iv}, ^{v}, ^{vi}\}$, $N^{\alpha}_r = \{X^{\alpha}_r \ : \ X \in N\}$. Γ consists of the following components.

$$P_{ini} = \{S' \to S\$\}, \quad P_{CF} = \{A \to A', A \to \alpha' \ : \ A \to \alpha \in P\},$$

$$P'_{CF} = \{A' \to A \ : \ A \in N\}$$

where for a string $\alpha \in (N \cup T)^+$, α' denotes $h(\alpha)$ for $h : N \cup T \to N' \cup T$ with $h(X) = X'$, $X \in N$ and $h(x) = x$, $x \in T$.

For each rule $r \in P_2$ of the form $AB \to CD$, Γ also has the components

$$\begin{aligned}
P_{r,1} &= \{B \to B_r, \$ \to \$_r, \$'_r \to \$'_r, \$''_r \to \$''_r\} \cup \{(X'_r \to X''_r, \{B_r\}) \ : \ X \in N\} \cup \\
&\quad \{\$_s \to \$_s, \$'_s \to \$'_s, \$''_s \to \$''_s \ : \ s \neq r\} \cup \{(B_r \to B''_r, \{B_r\})\} \cup \\
&\quad \{X \to X'_r, X'''_r \to X'''_r, X^v_r \to X^v_r \ : \ X \in N\}, \\
P_{r,2} &= \{A'_r \to A_r, \$'_r \to \$'_r, \$''_r \to \$''_r\} \cup \{X''_r \to X'''_r, X'_r \to X^v_r \ : \ X \in N\} \cup \\
&\quad \{(X'''_r \to X^{iv}_r, \{A_r\}), (X^v_r \to X^{vi}_r, \{A_r\})\} \cup \\
&\quad \{\$_s \to \$_s, \$'_s \to \$'_s, \$''_s \to \$''_s \ : \ s \neq r\}, \\
P_{r,3} &= \{X^{iv}_r \to X, X'''_r \to X'''_r, X^v_r \to X, X^{vi}_r \to X^{vi}_r \ : \ X \in N\} \cup \\
&\quad \{(\$_r \to \$'_r, \{A_r\}), (\$'_r \to \$''_r, \{B_r\})\} \cup \\
&\quad \{A_r \to C, B_r \to D, \$_r \to \$_r\},
\end{aligned}$$

which work together with

$$P_4 = \{\$''_r \to \$, \$_r \to \$_r, \$'_r \to \$'_r \ : \ r \in P\},$$

and then finally, one more component

$$P_{fin} \quad = \quad \{\$ \to a\} \cup \{X \to X \ : \ X \in N'\}.$$

The sentential forms w generated by G correspond to $w\$$ generated by Γ, $S\$$ is produced by the component P_{ini} from the start symbol S'. The context-free rules of G, the rules in P_1, are simulated by the components P_{CF} and P'_{CF}.

The context-sensitive rules are simulated by the three components $P_{r,i}$, $1 \leq i \leq 3$, associated to each rule r of the form $AB \to CD$. If we start with a sentential form $w\$$ where rule r is applicable to $w \in (N \cup T)^*$, we can use the component $P_{r,1}$ to produce a string of the form (1) $u'_r B_r v''_r \$_r$ or (2) $u'_r \$_r$ where $u'_r \in (N'_r)^*$, $v''_r \in (N''_r)^*$, and w is the unprimed, unindexed version of $u'_r B_r v''_r$ or u'_r. Because of the rules of the form $\$_r \to \$_r$, the components other than $P_{r,2}$ do not allow finishing the derivation.

If we have $u'_r \$_r$ and $P_{r,2}$ is used, then we get (2.1) $u^v_r A_r v^{vi}_r \$_r$ or (2.2) $u^v_r \$_r$ with $u^v_r \in (N^v_r)^*$, $v^{vi}_r \in (N^{vi}_r)^*$.

In the second case, the derivation either cannot be finished because of the rules of the form $X^v_r \to X^v_r$ and $\$_r \to \$_r$, or the sentential form can be changed to its unprimed form, $w\$_r$, by $P_{r,3}$ and then $P_{r,1}$ becomes applicable again.

In the first case, the symbol $\$_r$ is changed to $\$'_r$ by $P_{r,3}$ (there is no other component which can be used to continue the derivation) and then the derivation cannot be continued because of the presence of the rules $\$'_r \to \$'_r$.

If we have $u'_r B_r v''_r \$_r$ and $P_{r,2}$ is used, then we get strings of the form (1.1) $u^v_r B_r v'''_r \$_r$ or (1.2) $u^v_{r,1} A_r u^{vi}_{r,2} B_r v^{iv}_r \$_r$ with $u^v_r, u^v_{r,1} \in (N^v_r)^*$, $u^{vi}_{r,2} \in (N^{vi}_r)^*$, $v'''_r \in (N'''_r)^*$, $v^{iv}_r \in (N^{iv}_r)^*$.

In the first case, because of the presence of the rules of the form $X'''_r \to X'''_r$ and $\$_r \to \$_r$, the derivation cannot be continued. In the second case, because of the presence of the rules of the form $X^v_r \to X^v_r$ and $\$_r \to \$_r$, the derivation can only continue by the rules of $P_{r,3}$ which only enable a successful derivation if $u^{vi}_{r,2} = \lambda$.

In this case, the primed and indexed nonterminals are changed back to their original form, A_rB_r is rewritten to CD, and $\$_r$ is changed to $\$''_r$.

The symbol $\$''_r$ can be rewritten to \$ by component P_4, and then the simulation of another rule can begin.

If a string $z\$$ with $z \in T^*$ is obtained, then P_{fin} can be used to finish the derivation producing za. From the considerations above we can see that $za \in L(\Gamma)$ if and only if $z \in L(G)$, thus, we have shown that $\mathcal{L}(CS) \subseteq \mathcal{L}(CD, \ell P - \lambda)$.

To show that $\mathcal{L}(RE) \subseteq \mathcal{L}(CD, \ell P)$, we use the same construction but we start with a grammar in which we also allow rules of the form $A \to \lambda$. This way the constructed left-permitting CD grammar system also contains erasing rules, so we obtain that each recursively enumerable language is in $\mathcal{L}(CD, \ell P)$. □

5. On the size of permitting and left permitting CD grammar systems

Now we show that the number of components of permitting and left-permitting CD grammar systems can be reduced to two. This result is especially interesting since CD grammar systems with two context-free components generate only context-free languages, that is, the cooperating distributed framework itself can only increase the power of context-free components if at least three of them are present in the system.

Theorem 6. *$\mathscr{L}(CD, X, n) = \mathscr{L}(CD, X, 2)$ for all $n \geq 3$ and $X \in \{P,\ P - \lambda,\ \ell P,\ \ell P - \lambda\}$.*

Proof. Let $n \geq 3$ and let $\Gamma = (N, T, P_1, P_2, \ldots, P_n, S)$ be a (left-)permitting CD grammar system, and let $V = N \cup T$. Construct the permitting CD grammar system $\Gamma' = (N', T, P'_1, P'_2, S')$, with

$$N' = N \cup \{[z, i], \langle z, i\rangle : z \in V \cup \{\lambda\}, 1 \leq i \leq n\},$$

and P'_1 containing the rules

1. $(\langle z, i\rangle \to [z, i], \emptyset)$, for $z \in V \cup \{\lambda\},\ 1 \leq i \leq n$, and
2. for each $(A \to x, W) \in P_i$, where $x = x_1x_2 \ldots x_k$, $k \geq 1$, $x_j \in V$, $1 \leq j \leq k$, or $x = x_1 = \lambda$
   (a) $([A, i] \to [x_1, i]x_2 \ldots x_k, W)$,
   (b) $(A \to x, W \cup \{[z, i]\})$, for $z \in V \cup \{\lambda\}$,
   (c) $(A \to x, (W - \{Y\}) \cup \{[Y, i]\})$, for $Y \in W$.

The other component is

$$\begin{aligned} P'_2 \quad = \quad & \{(S' \to \langle S, i\rangle, \emptyset), ([z, i] \to \langle z, j\rangle, \emptyset) \ :\ z \in V \cup \{\lambda\},\ 1 \leq i, j \leq n\} \cup \\ & \{([z, i] \to z, \emptyset) \ :\ z \in T \cup \{\lambda\},\ 1 \leq i \leq n\}. \end{aligned}$$

The system works as follows. P_2' chooses a component of Γ to be simulated, say P_i, remembering this in the first nonterminal symbol of the sentential form, say $[z,i]$. Then, P_1' simulates a terminating derivation of P_i and it is not difficult to see that $(A \to x, W) \in P_i$ is applicable in Γ if and only if (at least) one of the following productions is applicable in Γ:

- production $(A \to x, W \cup \{[z,i]\}) \in P_1'$ if $z \neq A$ and $z \notin W$, or
- in case of $z = A$, production $([A,i] \to [x_1,i]x_2 \dots x_k, W) \in P_1'$, or
- in case of $z \in W$, production $(A \to x, (W - \{z\}) \cup \{[z,i]\}) \in P_1'$.

Thus, if $[z,i]$ is present in the sentential form, then P_1' keeps rewriting, simulating the rules of P_i, as long as possible. Then P_2' chooses the new component to be simulated by changing $[z,i]$ to $\langle z,j\rangle$, after which the process is repeated.

Since the special nonterminals $[z,i]$ or $\langle z,i\rangle$ always occupy the leftmost position in the string, the simulation works not only in the case of permitting, but also in the case of left-permitting components. □

Now we show that not only the number of components, but also the number of nonterminals can be bounded, moreover, both of them simultaneously, in the case of left-permitting systems with erasing rules.

The proof of the statement is based on simulation of Geffert normal form grammars for recursively enumerable languages. By [7] it is known that for every recursively enumerable language L over an alphabet T there exist a grammar $G = (N, T, P \cup \{AB \to \lambda, CD \to \lambda\}, S)$ with $L = L(G)$, such that $N = \{S, S', A, B, C, D\}$ and P contains only context-free productions of the form $S \to zSx$ where $z \in \{A,C\}^*$, $x \in T$, $S \to S'$, $S' \to uS'v$ where $u \in \{A,C\}^*$, $v \in \{B,D\}^*$, and $S' \to \lambda$.

Thus, a word $w \in T^*$ belongs to L if and only if at some step of the derivation in G the obtained sentential form is of the form $\alpha\beta w$, where $\alpha \in \{A,C\}^*$, $\beta \in \{B,D\}^*$ and $\alpha = h(\beta^R)$ where β^R denotes the reverse of β, and $h : \{B,D\} \to \{A,C\}$ is a homomorphism with $h(B) = A$, $h(D) = C$.

Theorem 7. *Any language $L \in \mathcal{L}(RE)$ can be generated by a left-permitting CD grammar system with 6 components and 19 nonterminals.*

Proof. Let L be an arbitrary recursively enumerable language and let $G = (N, T, P \cup \{AB \to \lambda, CD \to \lambda\}, S)$ be a grammar in Geffert normal form, above, which generates L. To prove the statement, we construct a CD grammar system Γ with left-permitting grammars as components that generates L in the t-mode derivation. The fact, that any language in $\mathcal{L}(CD, \ell P, 6)$ is recursively enumerable can be obtained by standard simulations.

Let $\Gamma = (N, T, P_1, P_2, \dots, P_6, \bar{S})$, where

$$N = \{S, \bar{S}, S', Z, Z', Z'', Z'''\} \cup \{X, \bar{X}, \tilde{X} \ : \ X \in \{A,B,C,D\}\},$$

and let $P' = \{(X \to \alpha, \emptyset) \ : \ X \to \alpha \in P\}$. Now let

$$P_1 \ = \ P' \cup \{(\bar{S} \to SZ, \emptyset), (Z' \to Z', \emptyset), (Z'' \to Z'', \emptyset), (Z''' \to Z''', \emptyset)\ \},$$

$$
\begin{aligned}
P_2 &= \{(X \to \bar{X}, \{C\}), (X \to \bar{X}, \{A\}) \ : \ X \in \{A, B, C, D\}\} \cup \\
&\quad \{(X \to \tilde{X}, \emptyset) \ : \ X \in \{B, D\} \ \} \cup \\
&\quad \{(X \to X, \{\tilde{B}\}), (X \to X, \{\tilde{D}\}) \ : \ X \in \{\bar{B}, \bar{D}, \tilde{B}, \tilde{D}\} \ \} \cup \\
&\quad \{(Z \to Z', \{A\}), (Z \to Z', \{C\}), (Z' \to Z'', \{\tilde{B}\}), (Z' \to Z'', \{\tilde{D}\}) \ \} \cup \\
&\quad \{(Z \to Z, \emptyset), (Z' \to Z', \emptyset)\}, \\
P_3 &= \{(A \to \lambda, \emptyset), (\tilde{B} \to \lambda, \emptyset), (\tilde{D} \to \tilde{D}, \emptyset), (C \to C, \emptyset), (Z'' \to Z''', \emptyset), \\
&\quad (Z \to Z, \emptyset), (Z' \to Z', \emptyset)\}, \\
P_4 &= \{(C \to \lambda, \emptyset), (\tilde{D} \to \lambda, \emptyset), (\tilde{B} \to \tilde{B}, \emptyset), (A \to A, \emptyset), (Z'' \to Z''', \emptyset), \\
&\quad (Z \to Z, \emptyset), (Z' \to Z', \emptyset) \ \}, \\
P_5 &= \{(\bar{X} \to X, \emptyset) \ : \ X \in \{A, B, C, D\} \cup \{(Z'' \to Z'', \emptyset), (Z''' \to Z, \emptyset) \ \}, \\
P_6 &= \{(Z''' \to \lambda, \emptyset), (Z \to Z, \emptyset), (Z' \to Z', \emptyset), (Z'' \to Z'', \emptyset) \ \} \cup \\
&\quad \{(X \to X, \emptyset) \ : \ X \in \{Y, \bar{Y}, \tilde{Y} \ : \ Y \in \{A, B, C, D\}\} \ \}.
\end{aligned}
$$

In the following, let us refer to Z as the end-marker. The derivation starts with component P_1 that derives a word of the form $\alpha\beta wZ$, where $\alpha \in \{A, C\}^*$, $\beta \in \{B, D\}^*$ and $\alpha = h(\beta)^R$. Then, the only component which can successfully continue the derivation is P_2. (The other components, because of the rules $Z \to Z$, would not be able to leave the sentential form.)

In the case of successful generation, P_2 first checks whether the sentential form $\alpha\beta wZ$ contains at least one letter from the set $\{A, C\}$. If this property holds, then the end-marker Z is changed to Z', in any other case the component is not able to stop with the derivation due to the existence of productions $Z \to Z$. Then P_2 rewrites letters in $\alpha\beta$ as follows: it leaves the first letter unchanged (there is no A or C preceding the first symbol), replaces all letters except the last letter preceding wZ' by its barred version and the symbol preceding wZ' by its version with tilde. Notice that if a letter which is not the symbol preceding wZ' is replaced by its version with tilde, then the component will not be able to leave the sentential form, just as in the case when no letter is replaced by its version with the tilde. Otherwise, Z' is changed to Z'' and the derivation continues either by component P_3 or by P_4. (Notice that P_1, P_5, and P_6 would also be able to continue the derivation, but due to production $Z'' \to Z''$, they would not be able to stop.)

Component P_2 finishes the derivation with a sentential form $Y\bar{\alpha_1}\tilde{\beta_1}\tilde{X}Z''$, where $Y \in \{A, C\}$, $X \in \{B, D\}$, and $Y\alpha_1\beta_1 X = \alpha\beta$. Suppose that $YX = AB$. Then, component P_3 removes letters A and $\tilde{B}$ and rewrites Z'' to Z'''. If $YX \neq AB$, then P_3 will never be able to stop with the derivation. Analogously, P_4 is able to cancel letters C and $\tilde{D}$ from the left-hand side and the right-hand side of $C\alpha_1\beta_1\tilde{D}$, respectively, and to rewrite Z'' to Z'''; in any other case the component enters a never ending derivation.

In the next step component P_5 rewrites the barred letters to their non-barred versions - if they exist - and replaces Z''' with Z. If symbols from $\{A, B, C, D\}$ still exists and Z''' is changed for Z, then the derivation continues with the interplay of components P_2, P_3, P_5 and P_2, P_4, P_5, respectively. If no symbol from $\{A, B, C, D\}$ exists, then component P_6 finishes the derivation. Notice that P_6 can stop with the

derivation only if the sentential form consists of zero or more terminal letters and Z'''.

Due to the construction of the production sets, every word of L and only words of L can be generated by Γ. □

6. Conclusions

In the previous sections, we have investigated the power and descriptional complexity parameters of permitting and left-permitting CD grammar systems. The definition of a derivation step in random context grammars by a rule $r : (A \to \alpha, Per)$ was given in such a way that r is applicable to uAv if $Per \subseteq alph(uv)$, that is, if $A \in Per$, then we have required that A appears in the sentential form in at least two copies. An equivalent definition of the direct derivation step for random context (permitting) grammars is also used in the literature, which considers also the rewritten nonterminal; i.e., $uAv \Rightarrow u\alpha v$ provided that there is a production $r : (A \to \alpha, Per)$, $Per \subseteq alph(uAv)$, that is, if $A \in Per$, then r may be applicable even in the case when $A \notin alph(uv)$. Contrary to "single" random context grammars where these two definitions are equivalent, we do not know whether they are also equivalent in case of CD grammar systems with (permitting) random context grammars as components.

Acknowledgments. T. Masopust was supported by the Czech Ministry of Education under the Research Plan No. MSM 0021630528.

References

[1] CSUHAJ-VARJÚ E., DASSOW J., *On cooperating/distributed grammar systems*, Journal of Information Processing and Cybernetics (EIK), **26**, 1990, pp. 49–63.

[2] CSUHAJ-VARJÚ E., DASSOW J., KELEMEN J., PĂUN G., *Grammar Systems: A Grammatical Approach to Distribution and Cooperation*, Gordon and Breach Science Publishers, Topics in Computer Mathematics 5, Yverdon, 1994.

[3] DASSOW J., PĂUN G., *Regulated Rewriting in Formal Language Theory*, Springer-Verlag, Berlin, 1989.

[4] DASSOW J., PĂUN G., ROZENBERG G., *Grammar systems*, in [11], Vol. **II**, pp. 155–213.

[5] DASSOW J., PĂUN G., SALOMAA A., *Grammars with controlled derivations*, in [11], Vol. **II**, pp. 101–154.

[6] EWERT S., VAN DER WALT A., *A pumping lemma for permitting random context languages*, Theoretical Computer Science, **270**, 2002, pp. 959–967.

[7] GEFFERT V., *Context-free-like forms for phrase structure grammars*, in M. P. Chytil, L. Janiga, V. Koubek (eds.), *Proc. Mathematical Foundations of Computer Science 1988*, 13th Symp. Carlsbad, Czechoslovakia, August 29–September 2, 1988. Volume **324** of Lecture Notes in Computer Science, Springer, Berlin, 1988, pp. 309–317.

[8] KARI L., ROZENBERG G., SALOMAA A., *L systems*, in [11], Vol. **I**, pp. 253–328.

[9] MASOPUST T., *On the terminating derivation mode in cooperating distributed grammar systems with forbidding components*, International Journal of Foundations of Computer Science, **20**, 2009, pp. 331–340.

[10] GOLDEFUS F., MASOPUST T., MEDUNA A., *Left-forbidding cooperating distributed grammar systems*, submitted.

[11] ROZENBERG G., SALOMAA A., eds, *Handbook of Formal Languages*, volumes **I**–**III**, Springer-Verlag, Berlin, 1997.

[12] SALOMAA A., *Formal languages*, Academic Press, New York, 1973.

[13] VAN DER WALT A.P.J., *Random context grammars*, *Proceedings of the Symposium on Formal Languages*, 1970.