Technical Report 2004/3,
Theoretical Computer Science Research Group, MTA SZTAKI, Budapest, 2004.

On A Competence-based Cooperation Strategy in CD Grammar Systems[*]

Erzsébet Csuhaj -Varjú
Computer and Automation Research Institute
Hungarian Academy of Sciences
Kende u. 13-17, H-1111 Budapest, Hungary
E-mail: csuhaj@sztaki.hu

Jürgen Dassow
Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
PSF 4120, D-39016 Magdeburg, Germany
E-mail: dassow@iws.cs-uni.magdeburg.de

Markus Holzer
Institute für Informatik
Technische Universität München
Arcisstrasse 21, D-80290 München, Germany
E-mail: holzer@in.tum.de

Abstract

In this paper context-free cooperating distributed (CD) grammar systems are examined where the cooperation strategy used by the component grammars is based on their competence (capability) in rewriting. The power of a derivation mode is studied where a component is allowed to start the generation only if its competence level on the sentential form is greater than or equal to the competence level of any of the other components and stops with the derivation if it does not have this property anymore. The competence level of a component on a string is the number of different nonterminal occurrences in this word that can be rewritten by its production set. It is shown that these CD grammar systems are more powerful than the Russian parallel grammars and at most as powerful as the programmed grammars with appearance checking. If the finite index restriction is applied, then the language class of these system is exactly the class of programmed languages with finite index. We also demonstrate an example of a non-$ET0L$ language generated by these constructions.

[*]The work of the first two authors was supported by a research cooperation, performed in the frame of the Hungarian-German Intergovernmental S&T Cooperation Programme, supported by the Office of Research and Development Division of the Hungarian Ministry of Education and its foreign contractual partner, BMBF. The work of the third author was supported by project Centre of Excellence in Information Technology, Computer Science and Control, ICA1-CT-2000-70025, HUN-TING project, workpackage 5.

1 Introduction

Cooperating distributed grammar systems (CD grammar systems, for short) are distributed models of language which can be considered as syntactic frameworks for the blackboard architectures known from the theory of cooperative distributed problem solving [2]. A blackboard architecture consists of several autonomous agents which jointly solve a problem in turn, in such way that the agents have access to a global database, called the blackboard, which stores information on the actual state of the problem solution and the problem solving process. The problem is solved by modifying the contents of the blackboard step by step. Furthermore, the blackboard is the only mean of communication among the agents. A cooperating distributed grammar system is a construct, where several grammars jointly generate words of a language, in turn, in such way that any moment of time one grammar is allowed to perform a derivation step on the actual sentential form. This grammar is selected according to the cooperation protocol of the grammars in the system, to the so-called derivation mode or cooperation strategy. The reader can easily observe that the grammars correspond to the agents, the sentential form in generation corresponds to the blackboard, and the generated language represents the set of possible problem solutions. The idea of cooperating grammars dates back to 1978, when Meersman and Rozenberg [8], motivated by the theory of two-level grammars, introduced this term, but it has only been extensively and intensively explored after Csuhaj-Varjú and Dassow [2] introduced the notion in a more general form, namely, as a cooperating/distributed grammar system, and related that to the above concepts of distributed artificial intelligence, to the blackboard architectures. The interested reader can find further information in [3] and [6]. An on-line annotated bibliography on the area can be found at [4], see http://www.sztaki.hu/mms/bib.html.

According to the cooperation protocol in [8], a component grammar of the system is allowed to derive the sentential form if it is able to replace any nonterminal occurrence in this string. This strategy was later called sf-mode of derivation [1]. If this property does not hold anymore, another grammar should continue the generation. In [2] another derivation mode was introduced and examined, called the t-mode of derivation, where a grammar is allowed to start with the generation if it is able to perform at least one derivation step and it must continue the derivation until this property holds. If the grammar is not able to execute any derivation step on the current string anymore and the string is not a terminal word, another grammar must continue the derivation. If such grammar does not exist, the derivation stops without generating a terminal word. It is easy to see that both strategies are based on a property of the grammar that is related to the current sentential form, namely, on its ability of replacing a nonterminal occurrence in that. We also can say that a grammar is *competent* on the sentential form if it is able to rewrite at least one nonterminal occurrence in this string. Thus, according to the derivation mode of Meersman and Rozenberg [8] a grammar is allowed to work on the sentential form if it is completely competent on the string (the agent is able to contribute to the solution of any open subproblem), and the derivation mode of Csuhaj-Varjú and Dassow [2] requires that after starting its work, the grammar continues the derivation until it is not competent anymore, that is, the agent contributes with its full competence to the problem solving process. It has turned out, that the first cooperation strategy, based on the complete competence of the grammars, is more powerful than the other one, these grammar systems with context-free components determine the class of programmed languages with appearance checking, while the other derivation mode, the t-mode of derivation leads to the power of $ET0L$ systems.

In this paper we continue the above line of examinations, namely, we introduce and study context-free CD grammar systems with another cooperation strategy, based on the competence level of the grammars, called the max-mode of derivation. In this case, a grammar is allowed to start the derivation of the sentential form if its competence level on the string is greater than or equal to the competence level of any of the other grammars and it must continue the generation until it does not have this property anymore. The idea behind the concept is to know whether or not it is a reasonable strategy if any moment of time the agent that contributes to the problem solving is one of the most competent ones.

We proved that the language class generated by these constructions (without λ-rules) is included in the class of programmed languages with appearance checking, that is, any language which can be obtained by a context-free CD grammar system with the max-mode of derivation can be obtained with the sf-mode of derivation, that is, cooperation based on maximal competence level is at most as powerful as cooperation based on complete competence. The question whether this relation is proper or not is open. Moreover, we showed that if we restrict the class of languages generated by context-free CD grammar systems in the max-mode of derivation to that of with finite index, then we obtain exactly the class of programmed languages with finite index. This result means that if the number of open subproblems is bounded by a finite number at any stage of the problem solving process based on the max-mode cooperation strategy, then this process can be programmed only by using "if-then" rules.

Furthermore, although we do not know in full details the relation of the class of $ET0L$ languages and the class of languages of context-free CD grammar systems working in the max-mode of derivation, we prove that a known subclass of the $ET0L$ language class, namely, the class of Russian parallel languages is strictly included in the language class of the above CD grammar systems. This result demonstrates that there are cases when the cooperation strategy based on the full competence of the components and the cooperation strategy based on selecting a grammar which is one of the most competent ones among the components are equally powerful. We also demonstrate here an example for a non-$ET0L$ language generated by these CD grammar systems, and thus we prove that these constructions determine languages that cannot be generated under the full competence strategy.

2 Preliminaries

Throughout the paper we assume that the reader is familiar with formal language theory. For further information consult [3, 7, 10].

The set of nonempty words over an alphabet V is denoted by V^+, if the empty string, λ, is included, then we use notation V^*. A set of strings $L \subseteq V^*$ is said to be a language over V.

For a string $w \in V^*$, we denote the length of w by $|w|$, and for a set of symbols $U \subseteq V$ we denote by $|w|_U$ the number of occurrences of letters U in w.

For a finite language L, the number of strings in L is denoted by $card(L)$.

We specify a context-free grammar by $G = (N, T, P, S)$, where N is the set of nonterminals, T is the set of terminals, P is the set of context-free productions and S is the start symbol. We use the notation $dom(P)$ for the set $\{A \in N \mid$ there is a production $A \to \alpha \in P\}$.

Throughout the paper, we shall use the following variants of context-free grammars with regulated rewriting. For details we refer to [5].

By a context-free *programmed grammar with appearance checking* we mean a quadruple $G = (N, T, P, S)$, where N and T are the set of nonterminals and the set of terminals, respectively, as in the case of context-free grammars, S is the start symbol, and P is a finite set of rules of the form $(r : A \to \alpha, \sigma(r), \phi(r))$, where $A \to \alpha$ is a context-free production over $N \cup T$, the core production of the rule labelled by r, and $\sigma(r)$ and $\phi(r)$ are two sets of labels of such core rules.

A direct derivation step $u \Longrightarrow_G v$ in G is as follows: Either the core rule of some rule $(r : A \to \alpha, \sigma(r), \psi(r))$ can be applied to obtain v from u and then we use the next rule from the set of productions labelled with $\sigma(r)$ or $A \to \alpha$ cannot be applied and then we pass on to a rule with a label from $\psi(r)$ in the next step. If $\psi(r)$ is the empty set for any rule in P, then we speak about a context-free *programmed grammar* (without appearance checking).

By a context-free *Russian parallel grammar* we mean a quadruple $G = (N, T, P, S)$ whose components are defined as in a context-free grammar and the set P is divided into two disjoint sets, P_1 and P_2. The direct derivation relation $u \Longrightarrow_G v$ is defined as follows: Either $u = u_1 A_1 u_2$, $v = u_1 \alpha u_2$, where $u_1, u_2 \in (N \cup T)^*$, $A \in N$, and $A \to \alpha \in P_1$, or $u = u_1 A u_2 \ldots u_n A u_{n+1}$, $n \geq 1$, $v = u_1 \alpha u_2 \ldots u_n \alpha u_{n+1}$, where $u_1 u_2 \ldots u_n u_{n+1}$ is in $(N \cup T \setminus \{A\})^*$, $A \in N$, and $A \to \alpha$ is in P_2.

If P_1 is the empty set, then we speak of an (context-free) *Indian parallel grammar.* Obviously, an empty set P_2 defines the customary context-free grammar.

We also will refer to the notion of an $ET0L$ system (an $ET0L$ grammar). An $ET0L$ system is an n+3-tuple $G = (N, T, P_1, \ldots, P_n, S)$, with $n \geq 1$, where N,T, and S are defined as in the case of context-free grammars, that is, the set of nonterminals, terminals, and the start symbol, and P_i, for $1 \leq i \leq n$, is a complete set of context-free productions over $(N \cup T)^*$. This means that for any symbol $X \in (N \cup T)$,the production set P_i has a rule with X being on its left-hand side. The direct derivation in an $ET0L$ system G is defined as follows: For two strings $x = x_1 \ldots x_r$, $y = y_1 \ldots, y_r$, $r \geq 1$, $x_i \in (N \cup T), y_i \in (N \cup T)^*$, $1 \leq i \leq r$, we say that x directly derives y, denoted by $x \Longrightarrow_G y$, if $x_i \to y_i \in P_j$ holds for $1 \leq i \leq r$, for some j, $1 \leq j \leq r$.

If no confusion can arise, we may omit the subscript G from the above notations $\Longrightarrow_G$.

For a grammar G, of the above types, $L(G)$ denotes the language generated by G.

Finally, we define restricted variants of the above language generating mechanisms, called grammars with finite index. For details the reader should consult

[5].

Let G be a grammar of arbitrary type, and let N, T, S be its nonterminal alphabet, terminal alphabet and start symbol, respectively. For a derivation

$$d : S = w_1 \Longrightarrow w_2 \Longrightarrow \ldots \Longrightarrow w_r = w \in T^*$$

according to G, we set

$$Ind(d, G) = max\{|w_i|_N \mid 1 \leq i \leq r\},$$

and, for $w \in T^*$, we define

$$Ind(w, G) = min\{Ind(d, G) \mid d \text{ is a derivation for } w \text{ in } G\}.$$

The index of the grammar G is defined as

$$Ind(G) = sup\{Ind(w, G) \mid w \in L(G)\}.$$

For a language L in the family of languages generated by grammars of some type X, we define

$$Ind_X(L) = inf\{Ind(G) \mid L(G) = L, G \in X\}.$$

When no danger of confusion turns out, then subscript X can be omitted.

For a language family $\mathcal{L}(X)$, we set

$$\mathcal{L}_n(X) = \{L \mid L \in \mathcal{L}(X), Ind_X(L) \leq n\},\ n \geq 1,$$

$$\mathcal{L}_{fin}(X) = \bigcup_{n \geq 1} \mathcal{L}_n(X).$$

Now we introduce some notations.

We denote the class of context-free languages by $\mathcal{L}(CF)$, the class of $ET0L$ languages by $\mathcal{L}(ET0L)$, and the class of recursively enumerable languages by $\mathcal{L}(RE)$.

If no λ-rules are allowed, then we denote the class of languages of Indian parallel, Russian parallel, programmed, programmed with appearance checking grammars by $\mathcal{L}(IP(CF))$, $\mathcal{L}(RP(CF))$, $\mathcal{L}(PR(CF))$, and $\mathcal{L}(PR_{ac}(CF))$, respectively. If λ-rules are allowed in the case of these grammars, we replace (CF) with (CF, λ) for in the notation of the corresponding language class. If we would like to refer to both cases, then, we write $(CF, [\lambda])$ instead of (CF). If the finite index restriction is applied, then we write $\mathcal{L}_{fin}$ instead of $\mathcal{L}$. We use analogously the notations $\mathcal{L}(ET0L)$, $\mathcal{L}(ET0L, \lambda)$, and $\mathcal{L}(ET0L, [\lambda])$. We note that $\mathcal{L}(CF) = \mathcal{L}(CF, \lambda)$ and $\mathcal{L}(ET0L) = \mathcal{L}(ET0L, \lambda)$.

The following relations are well-known among the above classes of languages [5]:

1. $\mathcal{L}(CF) \subset \mathcal{L}(RP(CF)) \subset \mathcal{L}(ET0L) \subset \mathcal{L}(PR_{ac}(CF, [\lambda]))$,

2. $\mathcal{L}(IP(CF)) \subset \mathcal{L}(RP(CF))$,

3. $\mathcal{L}(IP(CF))$ and $\mathcal{L}(CF)$ are incomparable,

4. $\mathcal{L}(PR(CF, [\lambda])) \subset \mathcal{L}(PR_{ac}(CF, [\lambda]))$, and $\mathcal{L}(PR_{ac}(CF, \lambda)) = \mathcal{L}(RE)$,

5. $\mathcal{L}(RP(CF,[\lambda])) \subset \mathcal{L}(PR_{ac}(CF,\lambda))$,
6. $\mathcal{L}(RP(CF))$ and $\mathcal{L}(PR(CF))$ are incomparable.
7. $\mathcal{L}_{fin}(PR(CF)) = \mathcal{L}_{fin}(PR_{ac}(CF))$,
8. $\mathcal{L}_{fin}(PR(CF)) = \mathcal{L}_{fin}(PR(CF,\lambda))$,
9. $\mathcal{L}_{fin}(RP(CF)) \subset \mathcal{L}_{fin}(PR(CF))$.

Languages demonstrating the 6. relation are $L_1 = \{a^n b^n c^n \mid n \geq 1\} \notin \mathcal{L}(RP(CF,[\lambda]))$ and $L_2 = \{a^{2^n} \mid n \geq 1\} \notin \mathcal{L}(PR(CF))$. (See [7] for details.)

3 Definitions

In the following we introduce the notion of a context-free CD grammar system where the components cooperate according to a derivation strategy which is based on the competence level of the component grammars in rewriting, related to the current string in generation.

Definition 3.1 *Let $G = (N,T,P,S)$ be a context-free grammar and let $w \in (N \cup T)^*$. We say that production set P is of competence level k on w, $k \geq 0$, if $|dom(P) \cap alph_N(w)| = k$ holds.*

Throughout, for denoting P with competence level k on w, we use notation $clev(P,w) = k$.

In other words, production set P is of competence level k on w if there are exactly k different nonterminals of G with an occurrence in w such that these symbols can be rewritten by a production in P. If $clev(P,w) \geq 1$, then we say P is *competent* on w. If $k = 0$, then either the production set is not competent on the string having at least one nonterminal occurrence or the string is a terminal word.

Now we recall the notion of a context-free CD grammar system according to [6].

By a *context-free CD grammar system* we mean an $n+3$-tuple $\Gamma = (N,T,P_1,\ldots,P_n,S)$, $n \geq 1$, where N, T, S are the set of nonterminals, the set of terminals and the start symbol, as in the case of context-free grammars, and P_i, with $1 \leq i \leq n$, are finite sets of context-free productions over $(N \cup T)$, called the components of the system. Observe that the quadruple $G_i = (N,T,P_i,S)$, for $1 \leq i \leq n$, with N,T,P_i,S as above, is a grammar, therefore we can also speak about a component grammar or a grammar of a CD grammar system.

For two sentential forms, u and v over $(N \cup T)^*$, we say v is directly derived (derived) from u in Γ by a component P_i, $1 \leq i \leq n$, denoted by $u \Longrightarrow_{P_i} v$ ($u \Longrightarrow^*_{P_i} v$), if v is generated from u by a direct derivation step (by a derivation) using production set P_i.

When jointly generating terminal words from the start symbol, the component grammars of the CD grammar system can follow different strategies for cooperation.

In the following we introduce a cooperation strategy which is based on the competence level of the components with respect to the actual string in generation.

Definition 3.2 *Let $\Gamma = (N, T, P_1, \ldots P_n, S)$, with $n \geq 1$, be a context-free CD grammar system and let $u \in (N \cup T)^*$. We say that component P_i, for $1 \leq i \leq n$, is a most competent one on u among the components of Γ, if $clev(P_i, u) \geq clev(P_j, u)$ holds for each component P_j, where $1 \leq j \leq n$.*

Now we define a derivation mode for CD grammar systems where the grammars work under the following cooperation strategy: at any step of the derivation there is no component having greater competence on the actual sentential form than the active component and the grammar remains active until it does not have this property anymore.

Definition 3.3 *Let $\Gamma = (N, T, P_1, \ldots P_n, S)$, $n \geq 1$, be a context-free CD grammar system and let*

$$d : S = w_0 \Longrightarrow^*_{P_{j_1}} w_1 \Longrightarrow^*_{P_{j_2}} \ldots w_{r-1} \Longrightarrow^*_{P_{j_r}} w_r = w,$$

$r \geq 1$, $w \in T^*$, $w_i \in (N \cup T)^*$, $1 \leq i \leq r+1$, $j_1, \ldots, j_r \in \{1, \ldots, n\}$ be a derivation in Γ.

We say that d is a derivation in the max-mode in Γ or a max-mode derivation in Γ if the following hold:

- *Let $w_i = w_{i,0} \Longrightarrow_{P_{j_i}} w_{i,1} \Longrightarrow_{P_{j_i}} \ldots \Longrightarrow_{P_{j_i}} w_{i,s_i} = w_{i+1}$, $1 \leq i \leq r$, $s_i \geq 1$ be a subderivation of d. Then, $clev(P_{j_i}, w_{i,k}) \geq clev(P_l, w_{i,k})$, $1 \leq l \leq n$, $0 \leq k \leq s_{i-1}$, and*
- *$clev(P_{j_i}, w_{i+1}) < clev(P_{j_{i+1}}, w_{i+1})$, $1 \leq i \leq r-1$.*

That is, when a component P_{j_i}, $1 \leq i \leq r$, starts deriving word w_{i-1}, then its competence level on w_{i-1} must be greater than or equal to the competence level of the other component grammars and P_{j_i} stops with the derivation when it does not have this property anymore.

Definition 3.4 *Let $\Gamma = (N, T, P_1, \ldots P_n, S)$, with $n \geq 1$, be a context-free CD grammar system. The language $L_{max}(\Gamma)$, called the language of max-derivations of Γ, is defined as*

$$L_{max}(\Gamma) = \{w \in T^* \mid d : S \Longrightarrow^* w,\ d\ is\ a\ max-derivation\ in\ \Gamma\}.$$

We demonstrate the notion with an example.

Example 3.1 *Let $\Gamma = (N, T, P_1, P_2, P_3, S)$ be a CD grammar system defined as follows: $N = \{A, B, A', B'\}$, $T = \{a, b, c\}$, and*

$$\begin{aligned} P_1 &= \{A \to A', B \to B', S \to AB\}, \\ P_2 &= \{A' \to aAb, B' \to Bc\}, \\ P_3 &= \{A' \to ab, B' \to c\}. \end{aligned}$$

Then $L_{max}(\Gamma) = \{a^n b^n c^n \mid n \geq 1\}$. This can be seen as follows. The derivation starts with applying production $S \to AB$ of component P_1, since it is the only competent component in the system. Then, P_1 still remains a most competent one, thus, either $AB \Longrightarrow A'B$ or $AB \Longrightarrow AB'$ follows. At this moment, any production set is at the same level of competence on $A'B$ or AB', namely, each component is able to perform only one replacement, thus, the derivation will still be continued by P_1, leading to sentential form $A'B'$ in both cases. Then, either component P_2 or P_3 can continue the derivation. Suppose that the derivation is continued by P_2, the case of P_3 is treated analogously. Now, either A' is replaced by aAb or B' with Bc, both cases result in a situation when each component will be on equal competence level on the string. Then, as in the previous case, P_2 continues the derivation and generates $aAbBc$. Thus, again, component P_1 can be active on the sentential form. Repeating this procedure as many times as it is necessary and using component P_3 in the final phase, we obtain the language above.

The above language is a non-context-free context-sensitive language that cannot even be generated by a Russian parallel grammar [5].

Finally, we introduce some notations.

Notation 1 *The class of languages generated by context-free CD grammar systems without λ-rules in the max-mode of derivation is denoted by $\mathcal{L}(CD_{max}(CF))$. If λ-rules are allowed, then we replace CF by CF, λ in the notation, if we would like to refer to both cases, then we write $CF, [\lambda]$ instead of CF. Furthermore, if we consider the subclass obtained by the finite index restriction, that is, the class of languages of context-free CD grammar systems working in the max-mode of derivation with finite index, then we write $\mathcal{L}_{fin}$ instead of $\mathcal{L}$.*

4 Generative capacity

In this section we deal with the generative power of context-free CD grammar systems working in the *max*-mode of derivation. We show that the class of languages generated by these systems strictly includes the class of Russian parallel languages and we demonstrate how programmed grammars with appearance checking simulate the working of these constructions. Moreover, we prove that if we apply the finite index restriction, then the obtained language class is the class of languages of programmed grammars with finite index. We also demonstrate an example for a non-$ET0L$ language generated by these CD grammar systems.

Theorem 4.1

$$\mathcal{L}(RP(CF, [\lambda])) \subset \mathcal{L}(CD_{max}(CF, \lambda)).$$

Proof. Let $G = (N, T, P, S)$ be a Russian parallel grammar with $P = P_1 \cup P_2$, where P_1 is the set of productions applied in the customary context-free manner and P_2 contains the rules that are applied in the same way as in the case of Indian parallel grammars. (Note that any of P_1 and P_2 can be the empty set.) Suppose that the productions in P are uniquely labelled and let us denote by $Lab(P)$ the set of these labels. We construct a CD grammar system $\Gamma = (N', T, P_1, \ldots, P_l, S)$, with $l \geq 1$, which, in the max-mode of derivation, generates $L(G)$. N' will consist of letters of N and some new symbols as follows. For each production $r : A \rightarrow \alpha$ in P_1, where $A \in N$, $\alpha \in (N \cup T)^*$, r is the label of the production, we introduce new nonterminals, A'_r, A''_r, and for each production $s : B \rightarrow \beta$ in P_2, where $B \in N$, $\beta \in (N \cup T)^*$, s is the label of the production, we introduce a new nonterminal B_s. Furthermore, we add a further new nonterminal, F, called the trap symbol, to the nonterminal set.

Components of Γ are defined in the following way.

For any production $r : A \rightarrow \alpha$ in P_1, where $A \in N$, $\alpha \in (N \cup T)^*$, and r is the label of the production, Γ has two components, namely, $\{A \rightarrow A'_r A''_r\}$ and $\{A'_r \rightarrow \alpha, A''_r \rightarrow \lambda\} \cup \{C_q \rightarrow F | q \in Lab(P), q : C \rightarrow \gamma \in P, C \in N, \gamma \in (N \cup T)^*, q \neq r\}$.

For any production $s : B \rightarrow \beta$ in P_2, where B is a nonterminal, β is a string in $(N \cup T)^*$, and s is the label of the production, Γ has two components $\{B \rightarrow B_s\}$ and $\{B_s \rightarrow \beta\} \cup \{C_q \rightarrow F | q \in Lab(P), q : C \rightarrow \gamma \in P, C \in N, \gamma \in (N \cup T)^*, q \neq s\}$.

We prove that Γ simulates the derivations in G. Suppose that at some stage of the derivation in Γ the actual sentential form is u, where $u \in (N \cup T)^*$. (When starting the derivation with S, this is exactly the case.)

At this moment any component of the form $\{C \rightarrow \gamma\}$ with an occurrence of C in u, where $C \in N$, $\gamma \in (N' - N)^+$, can be active, since these components are the most competent grammars among the grammars of the system, namely, they are of competence level one. Let us choose nondeterministically one of them. Suppose that this component is of the form $\{C \rightarrow C'_r C''_r\}$, that is, it was constructed to a production $r : C \rightarrow \gamma$ in P_1, to be applied in the context-free manner in G. The next component to be applied must be $\{C'_r \rightarrow \gamma, C''_r \rightarrow \lambda\} \cup \{A_q \rightarrow F | q \in Lab(P), q : A \rightarrow \alpha \in P, A \in N, \alpha \in (N \cup T)^*, q \neq r\}$, since its competence is two. Then, this component rewrites C'_r to γ and cancels C''_r, and thus, simulates the execution of the production $C_r \rightarrow \gamma$ in the context-free manner. If the sentential form u intended to be derived further by component $\{C \rightarrow C'_r C''_r\}$ contains an indexed nonterminal, say, A_q, then, the component applied after this component introduces at least one occurrence of the trap symbol F and the derivation will never lead to a terminal word.

Suppose now that to continue the derivation from u, a component of the form $\{C \rightarrow C_s\}$ is selected, that was constructed to production $s : C \rightarrow \gamma$ in P_2, to be applied in G in the Indian parallel manner. Then, this component replaces all occurrences of C in u with C_s, and finishes its activity. To continue the derivation, there are the following possibilities: A component $\{C_s \rightarrow \gamma\} \cup \{A_q \rightarrow F | q \in Lab(P), q : A \rightarrow \alpha \in P, A \in N, \alpha \in (N \cup T)^*, q \neq s\}$ will be active and it replaces all occurrences of C_s in the new string with γ, thus, simulates the application of production $s : C \rightarrow \gamma$ in G in the Indian parallel manner.

Or, a component, $\{A \rightarrow A'_r A''_r\}$, with competence level one, that was constructed for the simulation of the application of a context-free rule in P_1, will be active. If this is the case, then, at the next step component $\{A'_r \rightarrow \alpha, A''_r \rightarrow \lambda\} \cup \{C_q \rightarrow F | q \in Lab(P), q : C \rightarrow \gamma \in P, C \in N, \gamma \in (N \cup T)^*, q \neq r\}$ has to be selected, and then, the trap symbol F will be introduced for replacing at least one occurrence of C_s, and thus , the derivation never will end in a terminal word. Analogously, if after applying component $\{C \to C_s\}$, another component $\{B \to B_r\}$, introduced for simulating a production applied in the Indian parallel manner, is executed, then the derivation in the next step will introduce a trap symbol in the sentential form. In this case either component $\{C_s \to \gamma\} \cup \{A_q \to F | q \in Lab(P), q : A \to \alpha \in P, A \in N, \alpha \in (N \cup T)^*, q \neq s\}$ or component $\{B_r \to \beta\} \cup \{A_q \to F | q \in Lab(P), q : A \to \alpha \in P, A \in N, \alpha \in (N \cup T)^*, q \neq r\}$ has to be applied, since they both are at competence level at least two. But any of these two cases would lead to the introduction of symbol F. By the above explanations we can see that the sequence of the active components in the derivations which lead to a terminal word in Γ follows the sequence of the applied rules in a terminating derivation in G, that is, if a rule p in a successful derivation of G is applied, then in the simulating derivation in Γ the two components constructed for p are and must be applied after each other in the appropriate order. Moreover, the terminating derivations in G and only that are simulated by the terminating derivations in Γ in a correct manner, that is, $L(G) = L(\Gamma)$ holds. Thus, $\mathcal{L}(RP(CF, [\lambda])) \subseteq \mathcal{L}(CD_{max}(CF, \lambda))$. Because of Example 1, the inclusion is proper. Hence the result follows. ■

Corollary 4.1 $\mathcal{L}(IP(CF, [\lambda])) \subset \mathcal{L}(CD_{max}(CF, [\lambda]))$.

It is known that the class of Russian parallel languages is strictly included in the class of $ET0L$ languages [5]. The following statement demonstrates that CD grammar systems working in the *max*-mode of derivation are able to generate languages outside from the $ET0L$ language class. Notice that $\mathcal{L}(ET0L) = \mathcal{L}(ET0L, \lambda)$.

Theorem 4.2

$$\mathcal{L}(CD_{max}(CF)) \setminus \mathcal{L}(ET0L) \neq \emptyset.$$

Proof. To prove the statement, we construct a CD grammar system Γ which, in the *max*-mode of derivation, generates a non-$ET0L$ language. To help the legibility, we list only the components of the system; the nonterminal set and the terminal set of Γ can easily be determined by these productions. In the following, capital letters denote nonterminals and small letters denote terminals. The system starts its work from axiom S.

The productions sets of Γ are defined as follows:

$$\begin{array}{ll}
P_1 = \{S \to ASB, S \to AB\}, & P_2 = \{B \to B'b\}, \\
P_3 = \{B' \to Bb\}, & P_4 = \{A \to A_1A_2\}, \\
P_5 = \{A_1 \to c, A_2 \to c, B \to B'b\}, & P_6 = \{A_1 \to c, A_2 \to c, B' \to Bb\}, \\
P_7 = \{B \to C\}, & P_8 = \{B' \to C\}, \\
P_9 = \{A \to F, C \to a\}. &
\end{array}$$

We show that $L = L(\Gamma) = \{c^{2n}(ab^s)^n \mid s \geq n \geq 1\}$. Then, by applying homomorphism $h : \{a,b,c\} \to \{a,b,c\}$, defined by $h(a) = a$, $h(b) = b$, $h(c) = \lambda$, we obtain $h(L(\Gamma)) = \{(ab^s)^n \mid s \geq n \geq 1\}$, and this language is not an $ET0L$ language [5, 9]. Since $ET0L$ languages are closed under homomorphisms [9], $L = L(\Gamma)$ must not be an $ET0L$ language, thus, the statement holds.

Now we prove that Γ generates in the *max*-mode of derivation L.

The derivation starts with applying production $S \to ASB$ of component P_1 and continues with applying this rule several times and ends with applying production $S \to AB$ of this component. The sentential form we have yield is of the form A^nB^n for some n, $n \geq 1$. Then, components P_2, P_5,P_7, or P_4, P_9 can continue the derivation, since they are of equal competence level on the string and the most competent ones. If P_9 is applied, then trap symbol F will be introduced in the sentential form and we do not obtain a terminal word. If we apply P_7, then all Bs will be changed for C, and the next applicable component will be P_9 and thus the derivation will end with a word contaning the trap symbol, F.

Suppose that the next component we apply is P_2 (or P_5). Then, the component finishes its work with a string of the form $A^n(B'b)^n$. The possible continuation can be done by components P_3 or P_6 or by components P_4 or P_9. As in the previous case, we can exclude the application of P_9. Suppose now that component P_3 will be active, and the active period of components P_2 (or P_5) and P_3 (or P_6) following each other is repeated several times. Since A is present in the sentential form, during this phase, by the former reasoning, we can exclude the application of component P_7 for rewriting Bs. Then, a string of the form $A^n(Bb^r)^n$ is obtained, where $r > 1$. Now the derivation is continued by component P_4 (and not with P_9), and the new string will be of the form $w_1A_1A_2w_2(Bb^r)^n$, where $w_1w_2 = A^{n-1}$. At this point, P_5 must continue the derivation, by rewriting all symbols from $\{A_1, A_2, B\}$. This component will replace A_1 with c, A_2 with c and all occurrences of B with $B'b$. Until the sentential form contains at least one symbol from $\{A_1, A_2, B\}$, this component remains with the highest competence level among the grammars. Then, components P_3, P_6, or components P_4, P_9 can follow. As in the previous cases, we cannot choose P_9. If P_4 is selected, then, again, a letter A is changed for A_1 and A_2, and then component P_6 must be activated. Examining components P_2, P_3, P_4, P_5, and P_6, we can see that their interplay leads to a sentential form of the form $c^{2n}(Bb^s)^n$, where $s \geq n$. The relation $s \geq n$ follows from that the elimination of any letter A from the sentential form induces the introduction of at least n b-s in the sentential form. We should notice that the elimination of A-s from the sentential form can precede or can be combined with the increasing of the letters b in the sentential form by the work of components P_2, P_3, P_5, and P_6, that is, we could choose component P_4 instead of P_2 to be active at the beginning, for example, the result will be the same, the number s being the exponent of b-s in the generated terminal words will be at least as big as the number n of the exponent of the subwords of the form (ab^s).

Now, at some stage, either the B-s or the B'-s in the sentential form are changed for C by the activity of P_7 or P_8, and then, the procedure is finished by component P_9 that rewrites each C onto a. But, P_9 can be active only in the last phase of the derivation when no A is present in the sentential form. Otherwise, if some occurrences of A are found in the string, and the active component is P_9, then trap symbol F is introduced in the sentential form. Thus, we obtain that the terminal words which can be generated by Γ are of the form $c^{2n}(ab^s)^n$, where $s \geq n \geq 1$. Hence, we proved the result. ■

Obviously, any language generated by a CD grammar system working with the *max*-

mode of derivations is a recursively enumerable language, thus, it can be generated by a programmed grammar with appearance checking and with λ-rules. Furthermore, for any context-free CD grammar system without λ-rules and working in the max-mode of derivation a simulating programmed grammar with appearance checking and without λ-rules can be found.

Theorem 4.3

$$\mathcal{L}(CD_{max}(CF,[\lambda])) \subseteq \mathcal{L}(PR_{ac}(CF,[\lambda])).$$

Proof. Let $\Gamma = (N, T, P_1, \ldots, P_m, S)$, $m \geq 1$, be a context-free CD grammar system working in the max-mode of derivations. To prove the statement, we construct a programmed grammar, G, with appearance checking such that $L(G) = L(\Gamma)ab$ holds, where a, b are terminal symbols of G not in $N \cup T$. Since the programmed languages with appearance checking are closed under right derivative (see [5]), this assumption does not mean the loss of the generality. In the construction of the simulating programmed grammar $G = (N', T \cup \{a, b\}, P', S')$, with appearance checking, we use the following ideas. Suppose that $N = \{A_1, \ldots, A_n\}$, $n \geq 1$. At any step of each derivation in the CD grammar system Γ, we can represent the occurrence of the nonterminals of the system in the actual sentential form by an n-dimensional vector $(X_1, \ldots, X_n)$ where $X_i = 1$ if nonterminal A_i is present in the string and $X_i = 0$ if the nonterminal does not occur in it. Suppose now that a derivation step was successfully performed by component P_j, for some j, $1 \leq j \leq m$, in Γ on a sentential form u and it resulted in sentential form v. Then, using the information provided by the above vector and examining the productions of the components, we can decide which component can continue the derivation of v. By definition, this is P_j if P_j is still is of the highest competence level on v among the components and it is another component, if P_j is not among the most competent grammars on v. These two dynamically changing parameters, represented by nonterminals, namely, the nonterminal occurrence vector and the recent component to be activated, form an implicit regulation over the use of the productions in Γ, which can be used in the simulation of the working of Γ by a programmed grammar with appearance checking.

The functioning of the programmed grammar G is as follows: Until the derivation ends by obtaining a terminal word, any sentential form derived by G is of the form $u(X_1, \ldots, X_n)C_j$, where u is a string over $(N \cup T)$ and it is exactly the sentential form of Γ at the corresponding stage of the actual simulated derivation, $(X_1, \ldots, X_n)$ is a nonterminal symbol of G representing the nonterminal occurrence vector of Γ over u, and C_j, $1 \leq j \leq m$, is a nonterminal symbol of G denoting that component P_j is active on u. Then, a production of P_j is tried to be applied to u. If the application is successful, resulting in a string v, then the new nonterminal, corresponding to the nonterminal occurrence vector of v according to Γ is going to be determined (the occurrence of each symbol of N in the string is checked). Then, using the information provided by the nonterminal occurrence vector of v and knowing which was the last active component (letter C_j), the nonterminal (C_k) representing the next component of Γ to be activated in the simulated derivation will be determined. After this, as previously, a production of this component is going to be performed, and the above checking/selecting procedure will be repeated. If the chosen production of the active component grammar fails to be applied, another one is tried to be performed. The derivation successfully ends if a nonterminal representing the occurrence vector with only 0 components is obtained. Then this nonterminal and nonterminal C_k, which corresponds to the recently active component, are replaced by letters a and b, respectively in the sentential form. If no further active component can be found to a nonterminal occurrence vector, then the derivation ends without deriving a terminal word, and this holds for the simulating grammar G as well. By this explanation, we can see that terminal words of G are of the form wab, where the words w are exactly the terminal words of Γ.

Now we construct grammar G. For legibility, we give the productions together with some explanations. The set of nonterminals can easily determined by these rules. To help the reader in following the construction, we use labels which provide information on the role of the productions. By definition, productions are uniquely labelled by different labels; for a set of productions $P \subseteq P'$, we denote by $Lab(P)$ the set of labels of its productions. Moreover, without loss of the generality, we may assume that $A_1 = S$ and there is at least one component of Γ which has a production with S on its left-hand side. Now, production set P' is given as follows. It is the disjoint union of production sets $P_{ini} \cup P_{exe} \cup P_{test} \cup P_{vector-id} \cup P_{act} \cup P_{finish}$, where the different subsets are defined in the following way.

- First, we have production set P_{ini} with productions of the form

  $(ini_i : S' \to S(1,0,\ldots,0)C_i, Lab(P_i), \emptyset)$, for any k, $1 \leq k \leq m$, where P_i has a production of the form $S \to \alpha$, for some $\alpha \in (N \cup T)^*$.

- Then, for any production $p : A \to \alpha$ in P_i, $1 \leq i \leq m$, P' contains a production of the following form, which, together, determine production set P_{exe}. These productions are of the form

  $(p : A \to \alpha, \{test_1\}, Lab(P_i))$, where $test_1$ is the label of a production.

  Production $p : A \to \alpha$ of P_i is tried to be applied (supposing that P_i was the active component), and if the application was successful, then the new sentential form is to be tested according to the occurrence of the nonterminals in N. If the application of p failed, then another production of P_i is tried to be performed.

- For $i = 1, \ldots, n-1$ let

  $(test_i : A_i \to A_i, [A_i = 1], [A_i = 0])$, where

  $([A_i = 1] : (X_1, \ldots, X_i, \ldots X_n) \to (X_1, \ldots, 1, \ldots, X_n), test_{i+1}, \emptyset)$ and

  $([A_i = 0] : (X_1, \ldots, X_i, \ldots X_n) \to (X_1, \ldots, 0, \ldots, X_n), test_{i+1}, \emptyset)$.

  Moreover, let

  $(test_n : A_n \to A_n, [A_n = 1], [A_n = 0])$, where

  $([A_n = 1] : (X_1, \ldots, X_n) \to (X_1, \ldots, 1), Lab(P_{vector-id}), \emptyset)$ and

  $([A_n = 0] : (X_1, \ldots, X_n) \to (X_1, \ldots, 0), Lab(P_{vector-id}), \emptyset)$.

These productions form production set P_{test} and they are for checking whether or not A_i, with $1 \leq i \leq n$, appears in the sentential form, and rewrite the nonterminal representing the current nonterminal occurrence vector, $(X_1, \ldots, X_n)$, by modifying X_i to 1 or 0, according to the result of the check. After testing the appearance of the last nonterminal, A_n, the procedure continues by simulating how the next component grammar in Γ to be activated is determined, that is, which of the most competent grammars will continue the derivation (if such one exists). The simulating derivation in G will continue by a production from a set of rules $P_{vector-id}$.

Rules $\{r_1, \ldots, r_{2^n}\}$ form the production set $P_{vector-id}$ and they are for identifying the nonterminal that was determined above, according to the nonterminal occurrence vector of the new string in Γ. Each rule in $P_{vector-id}$ identifies a vector $v_k = (X_1^{(k)}, \ldots, X_n^{(k)})$, $X_i^{(k)} \in \{1, 0\}$, $1 \leq i \leq n$, $1 \leq k \leq 2^n$. For vectors being different from the zero vector, that is, from the vector having 0 at each position, each rule is of the form

$(r_k : (X_1^{(k)}, \ldots, X_n^{(k)}) \to (X_1^{(k)}, \ldots, X_n^{(k)}), Lab(P_{act_{r_k}}), Lab(P_{vector-id}))$, $2 \leq k \leq 2^n$.

The number of vectors is 2^n, since there are 2^n different combinations of 1 and 0 at n positions. We suppose that the zero vector is labelled by v_1 and the corresponding rule is labelled by r_1.

That is, the production tries to verify the nonterminal representing the nonterminal occurrence vector of the actual string in the simulated derivation. If the check is successful, then the procedure follows with simulating the choice of the next active grammar (the derivation continues at some production of $P_{act_{r_k}}$) , if the check fails, the appearance of another nonterminal is tried to be verified.

For the zero vector, v_1, we have production set P_{finish} with productions

$(r_1 : (0, \ldots, 0) \to (0, \ldots, 0), \{finish_1\}, Lab(P_{vector-id}))$,

where the corresponding rules are

$(finish_1 : (0, \ldots, 0) \to a, Lab(P_{fin}), \emptyset)$ and

P_{fin} consists of the rules

$(fin_j : C_j \to b, \emptyset, Lab(P_{fin}))$, for $1 \leq j \leq m$.

That is, if no nonterminal from N is found in the sentential form, then the nonterminal representing v_1 and the nonterminal C_j, being present in the sentential form and representing the last active component are rewritten onto a and b, respectively. Thus, a terminal word is derived.

Now let us return to the simulation of determining the next active component in the derivation in Γ.

- This is done by productions in $P_{act_{r_k}}$, defined as follows. For each nonterminal representing a nonterminal occurrence vector v_k, $2 \leq k \leq 2^n$, defined above, we define $Act(v_k)$ as the set of components of Γ which are of the highest competence level, among the grammars, on the longest prefixes over $(N \cup T^*)$ of the sentential forms which have nonterminal occurrence vector v_k. Now, for each v_k with $Act(v_k) \neq \emptyset$ we have productions in P_{actr_k} of the form

$(q_{j,k} : C_j \to C_l, Lab(P_k), Lab(P_{actr_k}))$, for $P_j \notin Act(v_k)$, $P_l \in Act(v_k)$, $1 \leq j, l \leq n$, and,

$(q_{j,j} : C_j \to C_j, Lab(P_j), Lab(P_{actr_k}))$, for $P_j \in Act(v_k)$, $1 \leq j \leq n$.

That is, if the simulated component remains of the highest competence level, it will continue the derivation, but if it has lost this property, another grammar will start with the generation.

For $v_k \neq v_1$ with $Act(v_k) = \emptyset$, that is, when the sentential form still contains nonterminals from N but there is no grammar which is able to continue the derivation, we have productions

$(t_j : C_j \to F, \emptyset, \emptyset)$, where F is a new nonterminal, called a trap symbol. In this case the derivation will abnormally terminate, since it cannot be continued and the sentential form is not a terminal word. The set of all productions, which are in some P_{actr_k}, $2 \leq k \leq 2^n$, is denoted by P_{act}.

Now we prove that any terminal word which can be generated by Γ can also be generated by G. Let

$$d : S = w_0 \Longrightarrow^*_{P_{j_1}} w_1 \Longrightarrow^*_{P_{j_2}} \dots \Longrightarrow^*_{P_{j_{r-1}}} w_r \Longrightarrow^*_{P_{j_r}} w_{r+1} = w,$$

$r \geq 0$, $w \in T^*$, $w_i \in (N \cup T)^*$, $0 \leq i \leq r$, and $j_1, \dots, j_r \in \{1, \dots, n\}$ be a *max*-derivation in Γ.

Moreover, let $w_i = w_{i,0} \Longrightarrow w_{i,1} \Longrightarrow \dots w_{i,s_i} = w_{i+1}$, $1 \leq i \leq r$, $s_i \geq 1$.

Then a production of the form $(ini_i : S' \to S(1,0,\dots,0)C_{j_1}, Lab(P_{j_1}), \emptyset)$ must be found in this production set, by the definition of P_{ini}, to indicate that the simulation of derivation d begins.

Then, the derivation in G continues by repeating the next sequence of productions as follows: for $i = 0$ to $i = r$ and for $k = 0$ to $k = s_i$ the following procedure is executed: first, production $p_{i,k}$, which is applied in the derivation step $w_{i,k} \Longrightarrow w_{i,k+1}$ by P_{j_i}, is performed according to P_{exe}, and then productions of P_{test} are applied. This results in checking which nonterminals occur in $w_{i,k+1}$. Then, the procedure continues at $P_{vector-id}$, and results in identifying the corresponding nonterminal occurrence vector. Then, the derivation continues at production set P_{act}. If that production is applied which is constructed according to the nonterminal occurrence vector in $w_{i,k+1}$, then nonterminal C_{j_i} is updated, and either it remains unchanged, or, in the case of w_{i,s_i}, it is changed to $C_{j_{i+1}}$. Then, the derivation continues again at production set P_{exe} with a rule with core production either from P_{j_i} or $P_{j_{i+1}}$, depending on the result of the previous step.

Then the procedure is repeated, as many times as it is necessary. Since the derivation in Γ ends with a terminal word, a nonterminal occurrence $(0,\dots,0)$ in the generated sentential form in G is guaranteed. Thus, the derivation in G also will terminate successfully by applying productions of P_{finish}. By the construction of the production sets of G we can see that any terminal word of Γ can be generated in G as the longest prefix of a terminal word over $(N \cup T)^*$ and not more. Moreover, it also can be seen that if Γ does not contain λ-rules, then G does not contain any λ-rule either. By the property that the programmed languages with appearance checking are closed under right derivative, $L(\Gamma) \in \mathcal{L}(PR_{ac}(CF, [\lambda]))$ holds. If Γ does not contain λ-rules, then $L(\Gamma) \in \mathcal{L}(PR_{ac}(CF))$ holds. Hence the result follows. ■

It is an open problem whether or not the class of languages determined by the context-free CD grammar systems working in the max-mode of derivation is strictly included in the class of programmed grammars with appearance checking. The following statement demonstrates that if we restrict the class of CD grammar systems working in the max-mode of derivation to the subclass of these systems with finite index, then the generated language class is exactly the class of languages of programmed grammars with finite index.

Theorem 4.4

$$\mathcal{L}_{fin}(CD_{max}(CF, [\lambda])) = \mathcal{L}_{fin}(PR(CF)).$$

Proof. Inclusion $\mathcal{L}_{fin}(CD_{max}(CF, [\lambda])) \subseteq \mathcal{L}_{fin}(PR(CF))$ follows from the previous statement and its proof.

To show that the equality holds, now we prove that for any programmed grammar $G = (N, T, P, S)$ with finite index we can construct a context-free CD grammar system $\Gamma = (N', T, P_1, \ldots, P_n, S')$, $n \geq 1$, such that $L_{max}(\Gamma) = L(G)$ holds and Γ is of finite index under the max-mode of derivation. Let $N = \{A_1, \ldots, A_s\}$, $s \geq 1$, and let us assume that $A_1 = S$.

Since G is of finite index, therefore we can list all sets of its nonterminals $C = \{A_{j_1}, \ldots, A_{j_l}\}$, $1 \leq l \leq Ind(G)$ which represent a set of nonterminal occurrences in a sentential form appearing in a derivation in G. By [5], Lemma 3.1.4, pp. 155, we know that for each matrix and thus, for any programmed grammar with finite index an equivalent grammar of the same type can be constructed where all the nonterminal occurrences in any sentential form are pairwise different. Let us denote by $Noc(G)$ the set of these sets of nonterminal occurrences, and let us denote by $C_1 = \{S\}$. Let $Lab(P)$ be the set of labels of productions of G. Let Γ have each nonterminal of G as a nonterminal symbol and some further new nonterminal letters. Now for any nonterminal A_i of G, with $1 \leq i \leq n$, and production of the form $(p : A_i \to \alpha, \sigma(p))$, with $\alpha \in (N \cup T)^*$, and for any $C \in Noc(G)$ with $A_i \in C$ let (A_i, p^C) be a new symbol, a new nonterminal of Γ, not in $(N \cup T)$. These nonterminals will refer to the case when production $(p : A_i \to \alpha, \sigma(p))$ is applied to a sentential form of G with a set of nonterminal occurrences C. Moreover, for any letter $p \in Lab(G)$, and $C \in Noc(G)$ let us introduce new symbols p^C and $(p^C)'$ which are not in $(N \cup T)$ and they are different from the new letters (A_i, p^C), defined above. Furthermore, let p_0, Y, F be further new nonterminals. Symbol p_0 helps in initializing the simulation of the derivation of G in Γ, Y is an auxiliary symbol, and F is a trap symbol playing role in finishing the derivation in Γ.

Now we shall construct the CD grammar system Γ. For legibility, we list only the components of the system.

For any nonterminal A_i, $1 \leq i \leq n$, in G and for any production of the form $(p : A_i \to \alpha, \sigma(p))$ in P, for any $C \in Noc(G)$, where $C = \{A_{j_1}, \ldots, A_{j_C}\}$, $j_C \geq$

1, with $A_i \in C$, and for any $q \in \sigma(p)$, CD grammar system Γ has components $P_{p,C,q,1}$, $P_{p,C,q,2}$, $P_{p,C,q,3}$, and $P_{p,C,q,4}$, defined as follows: $P_{p,C,q,1}$ has the following productions: $p^C \to p^C$, $A_{j_k} \to A_{j_k}$, for $1 \leq i \neq j_k \leq n_c$, and $A_i \to (A_i, p^C)$, $Y \to \lambda$, moreover, we add $X \to F$ for any other nonterminal letters of Γ not in $\{p^C, Y, (A_i, p^C)\} \cup C$.

This component indicates that the application of production p is planned to a sentential form with nonterminal occurrence set C. Moreover, it removes the possible appearance of auxiliary symbol Y from the sentential form and thus finishes the simulation of the production applied before p.

Let $P_{p,C,q,2}$ be defined with the following productions: $p^C \to (q^D)'$, where $D \in Noc(G)$ and D is the set of nonterminals occurring in the sentential form of G obtained from C after applying production p and $q \in Lab(p)$, and production q can be applied for a sentential form with nonterminal occurrence set D. Moreover let $P_{p,C,q,2}$ contain also productions $A_{j_k} \to A_{j_k}$, $1 \leq j_k \neq i \leq n_c$, and $(A_i, p^c) \to (A_i, p^C)$.

This component indicates which production is intended to be applied after p.

Let $P_{p,C,q,3}$ given with the following productions: $(q^D)' \to (q^D)'$, $A_{j_k} \to A_{j_k}$, $1 \leq j_k \neq i \leq n_c$, and $(A_i, p^C) \to \alpha Y$.

This component starts simulating the application of rule p and indicates that we intend to continue the computation by applying production q to a sentential form with nonterminal occurrence set D.

Moreover, let

$P_{p,C,q,4}$ given with the following productions: $(q^D)' \to q^D$, $A_{j_k} \to A_{j_k}$, $1 \leq j_k \neq i \leq n_c$, and $Y \to Y$.

This component makes sure that production q is intended to be applied.

To start with the simulation of the derivations of G by derivations in Γ the following component of Γ is defined.

Let $r_1, \ldots, r_l$ be the set of labels of productions in G that are for rewriting symbol S, that is, let $(r_j : A_1 = S \to \alpha_j, \sigma(r_j))$, $1 \leq j \leq l$. Then we add component P_0 with the following rules: $p_0 \to r_j^{C_1} SY$, for any rule r_j defined above.

We also need components for finishing the derivations.

For each nonterminal p^C with $p \in Lab(G)$ and $C \in Noc(G)$ we define component $P_{p,C}^{fin}$ with the productions

$p^C \to \lambda$, $Y \to \lambda$, $A_i \to F$, $1 \leq i \leq n$.

Furthermore let $S' = p_0$.

Now we prove that any derivation which can be performed in G can be simulated with a derivation in Γ and reversely. To show this, we examine the derivation in Γ.

It is obvious that any derivation in Γ starts with the functioning of component P_0. Then, after applying a rule, the sentential form is of the form $r_j^{C_1} A_1 = r_j^{\{S\}} SY$, that is, it is of the form $p^C u$, where $p \in Lab(P)$, $C \in Noc(G)$, and $u \in (N \cup \{Y\} T)^*$ with at least one occurrence of Y and exactly one occurrence of a nonterminal of G in u. Moreover, $(p : A_i \to \alpha, \sigma(p))$ is a rule in P and $A_i \in C$.

Now, let us suppose that the sentential form in Γ is exactly in the form $p^C u$, like above, and suppose that at this step of the derivation a new component has to start its work. Let us examine the components of Γ. The only components which are able to continue without introducing trap symbol F into the sentential form are that ones which are able to rewrite every nonterminal in the sentential form u and the marker nonterminal p^C, or the corresponding component that belongs to the group of components defined for finishing the derivation, but this case we will discuss later. Let us suppose now that the derivation will be continued. Then a successful continuation without an occurrence of the trap symbol is only possible if C is exactly the set of nonterminals occurring in u. Notice that by the construction of the component triples $P_{p,C,q,i}$, $i = 1, 2, 3$, if the marker nonterminal in the sentential form is p^C, then every nonterminal letter in C appears in the sentential form and only these nonterminals appear in it. Moreover, component $P_{q,D,r,1}$, for some rule r, exists only if the letter rewritten by q is in D. Now, let us suppose that the derivation can be continued and let $P_{p,C,q,1}$ be the production set to be activated. (We guess that the next production to be applied is q, that is $q \in \sigma(p)$.) Now, the component rewrites nonterminal A_i to nonterminal (A_i, p^C) and then it looses property being a most competent one. The next component which is able to continue the derivation is $P_{p,C,q,2}$. This component rewrites symbol p^C to symbol $(q^D)'$, indicating that the next production to be applied is chosen as q and the set of nonterminal occurrences of the sentential form obtained from u by applying p is the set D. After applying the production, component $P_{p,C,q,2}$ is not the most competent component anymore, so a new component must continue the derivation. The only component for this purpose is $P_{p,C,q,3}$, which rewrites (A_i, p^C) to αY. Then, this component looses the property being the most competent grammar and since $P_{p,C,q,4}$ is more competent than this one, it continues the derivation. Thus, we obtained a sentential form of the form $q^D v$, where v has exactly one occurrence of Y, exactly as as $p^C u$. It is also easy to see that the functioning of the components simulated the application of production p to sentential form u in G. If the set of nonterminals in D is not equal to the set of nonterminals of G which appear in v or nonterminal A_j, with $(q : A_j \rightarrow \beta, \sigma(q))$ does not appear in v, then the next component will introduce trap symbol F and the derivation will never terminate with a terminal word. Thus, no derivation in Γ can occur which does not correspond to a correct derivation of G.

Finally, we have to finish the derivation and remove marker nonterminals p^C. This is possible with activating the corresponding member of the group of components, $P^{fin}_{p,C}$. This component removes the marker symbol p^C and symbol Y without introducing the trap symbol, F, if and only if p^C is the only nonterminal in the sentential form.

From the above explanations we can see that the terminating derivations of Γ simulate all terminating derivations of G and only that. Thus, $L_{max}(\Gamma) = L(G)$ holds. Moreover, Γ is with finite index under the max-mode of derivation.

Since we have $\mathcal{L}_{fin}(CD_{max}(CF, [\lambda])) = \mathcal{L}_{fin}(PR(CF)) \subseteq \mathcal{L}_{fin}(CD_{max}(CF, \lambda))$, we proved the equality. ■

We obtain as corollary the following statement.

Corollary 4.2

$$\mathcal{L}_{fin}(CD_{max}(CF, \lambda)) = \mathcal{L}_{fin}(CD_{max}(CF).$$

5 Final remarks

The working of CD grammar systems under the max-mode of derivations very much differs from the functioning of grammars with the customary regulation mechanisms in rewriting and from the behaviour of the different variants of CD grammar systems have been studied so far. While in the latter cases a sequence of productions to be applied under functioning can be prescribed, in the case of this new cooperation strategy these standard regulation techniques do not work. Thus, we think, CD grammar systems with max-mode of derivations can also introduce new aspects in regulated rewriting. Determining the precise relation of their language class to the known classes of languages is a topic of further research.

References

[1] H. Bordihn, E. Csuhaj-Varjú: On competence and completeness in CD grammar systems. Acta Cybernetica 12(4) (1996), 347-361.

[2] E. Csuhaj-Varjú, J. Dassow: On cooperating/distributed grammar systems. Journal of Information Processing and Cybernetics EIK 26(1990), 49-63.

[3] E. Csuhaj-Varjú, J. Dassow, J. Kelemen, Gh. Păun: Grammar Systems - A Grammatical Approach to Distribution and Cooperation. Topics in Computer Mathematics 5. Gordon and Breach Science Publishers, Yverdon, 1994.

[4] E. Csuhaj-Varjú, Gy. Vaszil: An annotated bibliography of grammar systems. See: http://www.sztaki.hu/mms/bib.html

[5] J. Dassow, Gh. Păun: Regulated Rewriting in Formal Language Theory. EATCS Monograph on Theoretical Computer Scince 18, Springer-Verlag, Berlin-Heidelberg-New York, 1989.

[6] J. Dassow, Gh. Păun, G. Rozenberg: Grammar Systems. Handbook of Formal Languages. Vol 2., Chapter 4. (G. Rozenberg, A. Salomaa, eds.), Springer Verlag, Berlin-Heidelberg-New York, 1997, 155-213.

[7] Handbook of Formal Languages, Vol. 1-3., (G. Rozenberg, A. Salomaa, eds.), Springer-Verlag, Berlin-Heidelberg-New York, 1997.

[8] R. Meersman, G. Rozenberg: Cooperating grammar systems. In: Proc. MFCS'78, Lecture Notes in Computer Science, Springer-Verlag, Berlin, 1978, 364-373.

[9] G. Rozenberg, A. Salomaa: The Mathematical Theory of L Systems. Academic Press, New York, 1980.

[10] A. Salomaa: Formal Languages. Academic Press, New York, 1973.