Journal of Universal Computer Science, vol. 6, no. 12 (2000), 1165-1184
submitted: 3/2/00, accepted: 30/8/00, appeared: 28/12/00

Grammar Systems With Negated Conditions in Their Cooperation Protocols

Henning Bordihn
(Fakultät für Informatik, Otto-von-Guericke-Universität,
Postfach 4120, D-39016 Magdeburg, Germany
bordihn@iws.cs.uni-magdeburg.de)

Markus Holzer[1]
(Département d'I.R.O., Université de Montréal, C.P. 6128,
succ. Centre-Ville, Montréal (Québec), H3C 3J7 Canada
holzer@iro.umontreal.ca)

Abstract: The investigation on Boolean operations on the stop conditions of derivation modes for cooperating distributed grammar systems is continued by considering the logical negation of such conditions. The focus is on the negation of the t-mode of derivation, where such non-t-components may stop rewriting only if they still have a production applicable to the current sentential form. In many cases, hybrid cooperating distributed grammar systems with non-t-components turn out to give new characterizations of the class of programmed context-free languages or recurrent programmed context-free languages, where the latter class coincides with the biologically motivated family of languages generated by ET0L systems with random context. Thus, the results presented in this paper can shed new light on some longstanding open problems in the theory of regulated rewriting.

Key Words: Formal languages, grammar systems, programmed grammars, regulated rewriting.

Category: F.4.2, F.4.3

1 Introduction

The theory of cooperating distributed grammar systems—for an overview we refer to [Dassow, Păun, Rozenberg 1997]—has become a vivid field in formal language theory since its origin in [Csuhaj-Varjú, Dassow 1990], with forerunner papers [Meersman, Rozenberg 1978] and [Atanasiu, Mitrana 1989]. Its motivation is twofold: on the one hand, it is stemming from the syntax of programming languages since grammar systems can be seen as generalization of two-level substitution grammars to a multi-level concept. On the other hand, they serve as formal language description of multi-agent systems due to their close relations to models in artificial intelligence for distributed problem solving like blackboard models. Furthermore, in recent papers grammar systems have been considered as sequential counterparts of tabled Lindenmayer systems [Bordihn, Csuhaj-Varjú, Dassow 1999]. For applications of grammar systems in various fields of computer science we refer, e.g., to [Boldt and Jürgensen 2000] and [Freund and Kelemenova 2000]. In particular we want to mention the use of grammar systems in modeling

[1] Part of the work was done while the author was at Wilhelm-Schickard Institut für Informatik, Universität Tübingen, Sand 13, D-72076 Tübingen, Germany.

human-machine interfaces, in particular interfaces between a human and a computer using non-standardized spoken language [Aydin, Jürgensen, and Robbins 2000], in autolexical syntax [Jiménez-López and Martín-Vide 2000], and for the recognition of hand-written characters [Neubauer and Summerer 2000].

A cooperating distributed grammar system (CDGS, for short) consists of a finite set of (context-free) grammars, called components, performing derivation steps on a common sentential form in turns, according to some cooperation protocol. Simple such protocols are the so-called $*$-mode, $\leq k$-mode, $= k$-mode, or $\geq k$-mode, where a component, once started, has to perform an arbitrary number, at most k, exactly k, or at least k derivation steps, respectively. In terms of multi-agent systems, the components correspond to the independent problem solving agents, the sentential form to the current state of the problem solving, and the generated language represents the set of problem solutions. In many papers dealing with CDGS's, competence as an important feature of agents is formalized and used as basis of the cooperation protocol (for a thorough discussion see, e.g., [Bordihn, Csuhaj-Varjú 1996]). Mainly, the so-called t-mode has been considered (cf., e.g., [Csuhaj-Varjú, Dassow 1990] and [Csuhaj-Varjú, Dassow, Kelemen, Păun 1994]): each component which starts to work on the sentential form has to perform as many derivation steps as possible, i.e., it has to continue its work as long as it can contribute to the problem solving by having a production which is applicable to the current sentential form. Also in the first paper [Meersman, Rozenberg 1978] on grammar systems, a competence based derivation mode for the work of the components was formulated: when enabled, a component has to proceed the derivation until and unless it is totally competent, this means it is able to replace any nonterminal symbol occurring in the sentential form.

In this paper, we shall investigate CDGS's working according to another cooperation protocol based on the competence of its components: only competent components are allowed to interrupt their work and to hand over the sentential form, more precisely a component must have at least one production which is still applicable when it stops rewriting. This way it is prevented that one agent dominates the problem solving what is in line with the concept of fairness in grammar systems as considered in [Dassow, Mitrana 1996], where it has turned out that the fairness restriction leads to an increase in the generative power of the grammar systems in many cases. Clearly, the stop condition for this mode is nothing else but the logical negation of the stop condition for the t-mode. From this point of view, we continue the research on Boolean combinations of the "classical" derivation modes, sometimes referred to as *internal hybrid modes*.

In [Fernau, Freund, Holzer 1998] and also in [Fernau, Holzer, Freund 1997] and [Bordihn, Holzer 1999], the Boolean AND combinations are thoroughly investigated, partially coming to interesting characterizations of language families determined by grammars with controlled derivations (regulated rewriting), even involving the finite index restriction, which is of dynamic nature, by purely structural means. Moreover, the authors presented new characterizations of the external hybridization modes of CDGS's introduced by [Mitrana 1993], where each component may work in its own, specific mode. The Boolean OR combination is not worth considering because it trivially leads to external hybrid modes. In particular, a component working in $(f_1 \vee f_2)$-mode corresponds to two components, one in f_1- and the other one in f_2-mode. Therefore, the negation is the only "basic" Boolean function on the predicates formalizing derivation modes which has not yet been investigated. This paper aims to fill this gap. As an aside, we shall see that the negation of the other classical derivation modes, besides the t-mode, is of less interest. Moreover, we also investigate non-t hybrid modes in the sense of [Fernau, Freund, Holzer 1998] and [Fernau, Holzer, Freund 1997].

Finally, let us mention that, by means of the above sketched non-t-mode, we shall present further characterizations of typical language families encountered in the theory of grammars with regulated rewriting such as programmed context-free grammars and sub-families thereof. Especially, hybrid CDGS's having t- and non-t-components characterize the family of recurrent programmed context-free languages, a language class which is of interest because of its relation to the biologically motivated family of ET0L languages with random context [von Solms 1976] and since it forms an intermediate class between the families of context-free random context and programmed context-free languages generated by grammars without appearance checking [Fernau, Wätjen 1998]. The latter classes play a role in the present paper when determining the generative power of hybrid CDGS's with non-t-components in combination with other classical modes. So we hope that the reader can gain a deeper insight into the nature of (recurrent) programmed versus random context grammars without appearance checking such that new light is shed on some longstanding open questions in the field of regulated rewriting.

2 Definitions

We assume the reader to be familiar with the basic notions of formal languages, as contained in [Dassow, Păun 1989]. In general, we have the following conventions: $\subseteq$ denotes inclusion, while $\subset$ denotes strict inclusion. The set of positive integers is denoted by $\mathbb{N}$ and the cardinality of a set M by $|M|$. The empty word is denoted by λ. For $x \in V^*$, where V is some alphabet, and for $W \subseteq V$, let $|x|_W$ denote the number of occurrences of letters from W in x. If W is a singleton set $\{a\}$, we simply write $|x|_a$ instead of $|x|_{\{a\}}$. We consider two languages L_1 and L_2 to be equal if and only if $L_1 \setminus \{\lambda\} = L_2 \setminus \{\lambda\}$, and we simply write $L_1 = L_2$ in this case.

The families of languages generated by regular, context-free, context-sensitive, general type-0 Chomsky grammars, and ET0L systems are denoted by $\mathcal{L}(\mathrm{REG})$, $\mathcal{L}(\mathrm{CF})$, $\mathcal{L}(\mathrm{CS})$, $\mathcal{L}(\mathrm{RE})$, and $\mathcal{L}(\mathrm{ET0L})$, respectively. We attach $-\lambda$ in our notations if erasing rules are not permitted. Details about these families can be found in [Dassow, Păun 1989]. The class of finite languages is denoted by $\mathcal{L}(\mathrm{FIN})$.

A *programmed grammar* (see, for instance, [Rosenkrantz 1969]) is a septuple $G = (N, T, P, S, \Lambda, \sigma, \phi)$, where N, T, and $S \in N$ are the set of nonterminals, the set of terminals, and the start symbol, respectively. In the following we use V_G to denote the set $N \cup T$. P is the finite set of productions $\alpha \to \beta$, and Λ is a finite set of labels (for the productions in P), such that Λ can be also interpreted as a function which outputs a production when being given a label; σ and ϕ are functions from Λ into the set of subsets of Λ. For (x, r_1) and (y, r_2) in $V_G^* \times \Lambda$ and $\Lambda(r_1) = (\alpha \to \beta)$, we write $(x, r_1) \Rightarrow (y, r_2)$ if and only if either $x = x_1 \alpha x_2$, $y = x_1 \beta x_2$ and $r_2 \in \sigma(r_1)$, or $x = y$ and rule $\alpha \to \beta$ is not applicable to x, and $r_2 \in \phi(r_1)$. In the latter case, the derivation step is done in *appearance checking* *mode*. The set $\sigma(r_1)$ is called success field and the set $\phi(r_1)$ failure field of r_1. As usual, the reflexive transitive closure of $\Rightarrow$ is denoted by $\stackrel{*}{\Rightarrow}$. The language generated by G is defined as

$$L(G) = \{\, w \in T^* \mid (S, r_1) \stackrel{*}{\Rightarrow} (w, r_2) \text{ for some } r_1, r_2 \in \Lambda \,\}.$$

The family of languages generated by programmed grammars containing only context-free core rules is denoted by $\mathcal{L}(\mathrm{P}, \mathrm{CF}, \mathrm{ac})$. We replace CF by $\mathrm{CF} - \lambda$ in that notation if erasing rules are forbidden. When no appearance checking features are involved, i.e., $\phi(r) = \emptyset$ for each label $r \in \Lambda$, we are led to the families $\mathcal{L}(\mathrm{P}, \mathrm{CF})$ and $\mathcal{L}(\mathrm{P}, \mathrm{CF} - \lambda)$. A special variant of programmed grammars are recurrent programmed grammars introduced in [von Solms 1976]. A programmed context-free grammar G is a *recurrent programmed context-free grammar* if for every $p \in \Lambda$ of G, if $\phi(p) = \emptyset$, then $p \in \sigma(p)$, and if $\phi(p) \neq \emptyset$, then $p \in \sigma(p) = \phi(p)$. The corresponding language families are denoted by $\mathcal{L}(\mathrm{RP}, \mathrm{CF}, \mathrm{ac})$ and $\mathcal{L}(\mathrm{RP}, \mathrm{CF} - \lambda, \mathrm{ac})$; when no appearance checking features are involved, i.e., $\phi(r) = \emptyset$ for each label $r \in \Lambda$, we omit ac in that notation, again.

We use bracket notations like $\mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda]) \subset \mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda], \mathrm{ac})$ in order to say that the statement holds both in case of forbidding erasing productions and in the case of admitting erasing productions (neglecting the bracket contents).

A *cooperating distributed grammar system (CDGS)* of degree n, with $n \geq 1$, is an $(n+3)$-tuple $G = (N, T, S, P_1, \ldots, P_n)$, where N, T are disjoint alphabets of nonterminals and terminals, respectively, $S \in N$, and $P_1, \ldots, P_n$ are context-free rule sets called components. For $x, y \in (N \cup T)^*$ and $1 \leq i \leq n$, we write $x \Rightarrow_i y$ if and only if $x = x_1 A x_2$, $y = x_1 z x_2$ for some $A \to z \in P_i$. Hence, subscript i refers to the component to be used. By $\Rightarrow_i^{\leq k}$, $\Rightarrow_i^{=k}$, $\Rightarrow_i^{\geq k}$, $\Rightarrow_i^*$ we denote a derivation consisting of at most k steps, exactly k steps, at least k steps, an arbitrary number of steps, respectively, executed by component P_i. Furthermore, we write $x \Rightarrow_i^t y$ if and only if $x \Rightarrow_i^* y$ and there is no z such that $y \Rightarrow_i z$.

Combining the former three modes with the t-mode requirement we obtain the modes $(t \wedge \leq k)$, $(t \wedge = k)$, and $(t \wedge \geq k)$ which are defined as follows (see, e.g., [Fernau, Freund, Holzer 1998] and [Fernau, Holzer, Freund 1997]): there exists a derivation which satisfies both properties in common, e.g., $x \Rightarrow_i^{(t \wedge \leq k)} y$ if and only if there exists an m-step derivation from x to y using P_i such that $m \leq k$ and there is no z such that $y \Rightarrow_i z$.

Applying the negation to the non-combined modes leads us to, e.g., $x \Rightarrow_i^{\bar{t}} y$, which is defined as follows: $x \Rightarrow_i^{\bar{t}} y$ if and only if $x \Rightarrow_i^* y$ and there is a z such that $y \Rightarrow_i z$. In fact this is the most interesting negated derivation mode, because the non-$\leq k$-, and non-$\geq k$-modes are nothing other than synonyms for the $\geq k+1$- and $\leq k-1$-modes, respectively. Moreover, the non-$*$-mode is useless, because nothing can be derived with this mode. Finally, any non-$= k$-component can be simulated by two identical components, one running in $\leq k-1$-mode, the other one running in $\geq k+1$-mode. Hybrid CDGS's, where each component may work in its own specific mode, have been introduced in [Mitrana 1993] and are formally defined as follows. Let

$$\begin{aligned} D = \{*, t\} \cup \{\bar{t}\} \cup \{\leq k, = k, \geq k \mid k \in \mathbb{N}\,\} \\ \cup \{\, (t \wedge \leq k), (t \wedge = k), (t \wedge \geq k) \mid k \in \mathbb{N}\,\} \\ \cup \{\, (\bar{t} \wedge \leq k), (\bar{t} \wedge = k), (\bar{t} \wedge \geq k) \mid k \in \mathbb{N}\,\}. \end{aligned}$$

A *hybrid CDGS* is a construct $G = (N, T, S, (P_1, f_1), \ldots, (P_n, f_n))$, where N, T, S, $P_1, \ldots, P_n$ are as in a CDGS and $f_i \in D$, with $1 \le i \le n$. The language generated by G is

$$L(G) = \{\, w \in T^* \mid S \Rightarrow_{i_1}^{f_{i_1}} w_1 \Rightarrow_{i_2}^{f_{i_2}} \cdots \Rightarrow_{i_{m-1}}^{f_{i_{m-1}}} w_{m-1} \Rightarrow_{i_m}^{f_{i_m}} w_m = w \text{ with } m \ge 1,\ 1 \le i_j \le n, \text{ and } 1 \le j \le m \,\}.$$

If $F \subseteq D$, then the family of languages generated by [λ-free] context-free hybrid CDGS's with degree at most n working with modes from F only, is denoted by $\mathcal{L}(\mathrm{HCD}_n, \mathrm{CF}[-\lambda], F)$. When the number of components is not restricted, we write $\mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], F)$. If F is a singleton $\{f\}$, we simply write $\mathcal{L}(\mathrm{CD}_n, \mathrm{CF}[-\lambda], f)$, where $n \in \mathbb{N} \cup \{\infty\}$.

Since the negation of the t-mode is the only challenging one among the negations of the classical modes, we restrict ourselves to the $\bar{t}$-mode—sometimes we also write non-t-mode instead of $\bar{t}$-mode. Moreover, starting from the non-t-mode, we may build combined modes as it was done with the usual t-mode. In this way we obtain the $(\bar{t} \wedge \le k)$-, $(\bar{t} \wedge = k)$-, and $(\bar{t} \wedge \ge k)$-mode.

On the other hand, the reader may have observed, that for each $n \in \mathbb{N} \cup \{\infty\}$ we have $\mathcal{L}(\mathrm{CD}_n, \mathrm{CF}[-\lambda], \bar{t}) = \{\emptyset\}$, since with the non-$t$-mode it is not possible to terminate a derivation. Therefore, in the remainder of this paper we only deal with hybrid CDGS's with non-t-components.

In order to clarify our definitions, we give a short example:

Example 1. The hybrid CDGS $G = (N, T, S, (P_1, \bar{t}), (P_2, \bar{t}), \ldots, (P_6, \bar{t}), (P_7, *))$ with nonterminals $N = \{S, A, B, A', B', F\}$, terminals $T = \{a, b, c\}$, and the production sets

$$\begin{array}{ll}
P_1 = \{S \to AB, A \to F\}, & P_5 = \{B' \to B, A \to F\}, \\
P_2 = \{A \to aA'b, B \to F\}, & P_6 = \{A \to ab, B \to F\}, \quad \text{and} \\
P_3 = \{B \to B'c, A' \to F\}, & P_7 = \{B \to c\}, \\
P_4 = \{A' \to A, B' \to F\}, &
\end{array}$$

generates the language $L(G) = \{\, a^n b^n c^n \mid n \ge 1 \,\}$. This can be seen as follows.

Obviously, it is possible to generate abc *via* the sentential form AB. Starting from a form $a^k A b^k B c^k$, with $k \ge 0$, the only applicable tables are P_2, P_6, and P_7. Note that the use of P_1, P_3, and P_5 is impossible at this moment because the derivation by these non-t-components would immediately terminate. Analogously, it is easy to see that in the first case, i.e., when applying P_2, the rule $A \to aA'b$ has to be used and not $B \to F$ (note that F serves as failure symbol), and then the system is forced to proceed with P_3, P_4, and P_5 in this order. In this way, we arrive at $a^{k+1} A b^{k+1} B c^{k+1}$. In the other case, i.e., after applying P_6, the system can only terminate using P_7 generating $a^{k+1} b^{k+1} c^{k+1}$. In the final case, i.e., after applying P_7, the generating process stops, because no component of G is applicable anymore. This shows that $L(G) = \{\, a^n b^n c^n \mid n \ge 1 \,\}$.

3 Combining t or $(t \wedge \ge k)$ with $\bar{t}$

In this section we study grammar systems, where the components work, when enabled, in t-mode ($(t \wedge \ge k)$-mode) or in non-t-mode. In the first part of this section we deal with the combination of t and non-t-components, and in the second part, with the $(t \wedge \geq k)$-mode combined with non-t-components. It turns out that these two combinations are closely related, and characterize the class of recurrent programmed context-free languages. Since this class is known to be equal to the ET0L languages with random context, the main result in the first part of this section can be seen as a natural generalization of

$$\mathcal{L}(\mathrm{CD}_\infty, \mathrm{CF}[-\lambda], t) = \mathcal{L}(\mathrm{ET0L}),$$

which was shown in [Csuhaj-Varjú, Dassow 1990]. Hence, adding non-t-components to a CDGS working in t-mode, behaves like adding random context to an ET0L system.

Before we come to the first theorem, we have to introduce the notion of ET0L systems with random context. An ET0L system with random context is a construct $G = (\Sigma, H, \omega, \Delta, oc, noc)$. Here, $(\Sigma, H, \omega, \Delta)$ is an ordinary ET0L system, where Σ is the total alphabet, $\Delta \subseteq \Sigma$ is the terminal alphabet, H is the set of tables (finite substitutions from Σ into Σ^*), and $\omega \in \Sigma^+$ is the axiom. Furthermore, oc and noc are functions from H to the subsets of Σ. For x and y in Σ^*, we write $x \Rightarrow_h y$ for some h in H if and only if all letters in $oc(h)$ occur in x, no letter of $noc(h)$ occurs in x, and $y \in h(x)$. The language generated by G is defined as

$$L(G) = \{\, w \in \Delta^* \mid \omega \Rightarrow_{h_{i_1}} w_1 \Rightarrow_{h_{i_2}} \cdots \Rightarrow_{h_{i_m}} w_m = w \text{ with } \\ m \geq 0, \text{ and } h_{i_j} \in H, \text{ for } 1 \leq j \leq m \,\}.$$

In [von Solms 1976] it was shown that every language generated by a recurrent programmed context-free grammar (with appearance checking) is a language which can be generated by an ET0L system with random context and *vice versa*. Obviously, by the proof given there, the results remain valid if λ-rules are forbidden on both sides. Using this characterization we are ready to prove the following theorem:

Theorem 1. $\mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{t\} \cup \{\bar{t}\}) = \mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac})$.

Proof. First, we show how to simulate a hybrid CDGS

$$G = (N, T, S, (P_1, f_1), (P_2, f_2), \ldots, (P_n, f_n))$$

with $f_i \in \{t, \bar{t}\}$, $P_i = \{p_{i1}, p_{i2}, \ldots, p_{in_i}\}$, and $p_{ij} = (A_{ij} \to w_{ij})$, for $1 \leq i \leq n$ and $1 \leq j \leq n_i$. We construct a recurrent programmed grammar

$$G' = (N \cup \{S', F\}, T, P, S', \Lambda, \sigma, \phi),$$

where S' and F are new symbols and the production set P as well as the label set Λ and the functions σ and ϕ are implicitly defined below.

The initialization is done by the rule $\Lambda(init) = (S' \to S)$ with the success field $\sigma(init) = \{init\} \cup \{\, p_{ij} \mid 1 \leq i \leq n \text{ and } 1 \leq j \leq n_i \,\}$ and failure field $\phi(init) = \emptyset$.

For $1 \leq i \leq n$ and $1 \leq j \leq n_i$, a rule p_{ij} is simulated by the production $\Lambda(p_{ij}) = (A_{ij} \to w_{ij})$. We have to distinguish two cases: in case $f_i = t$ the success and failure field for p_{ij} is defined as $\sigma(p_{ij}) = \{p_{ij}\} \cup \{\, p_{ik} \mid 1 \leq k \leq n_i \,\} \cup \{t_{i1}\}$ and $\phi(p_{ij}) = \emptyset$. After the simulation of the rules it has to be checked whether the t-mode condition is met, i.e., it has to be tested whether the actual sentential form does not contain any left hand-side of the production set. This is done by the rules $\Lambda(t_{ij}) = (A_{ij} \to F)$ with $\sigma(t_{ij}) = \phi(t_{ij}) = \{t_{ij}\} \cup \{t_{ij+1}\}$ if $1 \leq j < n_i$, and $\sigma(t_{in_i}) = \phi(t_{in_i}) = \{t_{in_i}\} \cup \{\, p_{k\ell} \mid 1 \leq k \leq n \text{ and } 1 \leq \ell \leq n_k \,\}$. In the other case, i.e, $f_i = \bar{t}$, we set $\sigma(p_{ij}) = \{p_{ij}\} \cup \{\, p_{ik} \mid 1 \leq k \leq n_i \,\} \cup \{\, t_{ik} \mid 1 \leq k \leq n_i \,\}$ and $\phi(p_{ij}) = \emptyset$. As above, after the simulation, the necessary check for the mode has to be done. For the non-t-mode this means that in the actual sentential form has to appear at least one left hand-side of the production set. This check is done by $\Lambda(t_{ij}) = (A_{ij} \to A_{ij})$ with $\sigma(t_{ij}) = \{t_{ij}\} \cup \{\, p_{k\ell} \mid 1 \leq k \leq n \text{ and } 1 \leq \ell \leq n_k \,\}$ and $\phi(t_{ij}) = \emptyset$.

This completes the description of the equivalent recurrent context-free grammar G' with appearance checking. Note, that the recurrent programmed grammar has λ-productions only if the hybrid CDGS has.

Now we have to show the other inclusion. By a standard argument the involved hybrid CDGS language family is closed under union and embraces the finite languages. Let $L \subseteq T^*$ be in $\mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac})$, then

$$L = \bigcup_{a \in T} (a \cdot \delta_a(L)) \cup (L \cap T) \cup (L \cap \{\lambda\}),$$

where $\delta_a(L) = \{\, w \in T^+ \mid aw \in L \,\}$. Since L is in $\mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac})$, language $\delta_a(L)$ is in $\mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac})$ due to the closure properties of that family under derivatives, which can be proved by standard arguments. Thus, it is sufficient for the proof of the present assertion to show that $\{a\} \cdot \delta_a(L)$ is in $\mathcal{L}(\mathrm{HCD}, \mathrm{CF}[-\lambda], \{t\} \cup \{\bar{t}\})$ provided that $\delta_a(L)$ is a recurrent programmed context-free language.

Here we use now the characterization of recurrent programmed context-free languages in terms of ET0L systems with random context. Let

$$G = (\Sigma, H, \omega, \Delta, oc, noc)$$

be an ET0L system with random context generating $\delta_a(L)$. Without loss of generality assume $H = \{h_1, h_2, \ldots, h_n\}$ and $\omega = S$. Furthermore, let $m = \max\{\, |oc(h)| \mid h \in H \,\}$ and $\Sigma^{\leq m}$ is an abbreviation for $\{\, w \in \Sigma^* \mid |w| \leq m \,\}$.

The constructed hybrid CDGS has an alphabet of nonterminals

$$N = \{S', F\} \cup \{\, p_i, [p_i, \alpha] \mid 0 \leq i \leq n \text{ and } \alpha \in \Sigma^{\leq m} \,\} \\ \cup \{\, [A, i] \mid A \in \Sigma \text{ and } 0 \leq i \leq n \,\},$$

the unions being disjoint, an alphabet of terminals $T = \Delta$, and axiom S'. The production sets and their modes are given below.

The simulation starts with component

$$P_{init} = \{S' \to [p_0, \lambda][S, 0]\},$$

which runs in t-mode. For $1 \leq i \leq n$, define the colouring components $P_{col,i}$ which run in t-mode and select a table $h_i \in H$ as follows:

$$P_{col,i} = \{[p_0, \lambda] \to [p_i, \lambda]\} \cup \{[A, 0] \to [A, i] \mid A \in \Sigma\,\}.$$

Then, for each table h_i in H with $oc(h_i) = \{A_1, A_2, \ldots, A_{m'}\}$ and $m' \leq m$, define the non-t-components $P_{i,j}$, with $1 \leq j \leq m'$, which check the occurrence of A_j in the actual sentential form: Set

$$P_{i,j} = \{\, [p_i, w] \to [p_i, wA_j] \mid w \in \Sigma^{j-1} \,\} \cup \{[A_j, i] \to [A_j, i]\},$$

Due to the construction of the production sets, grammar G is forced to apply $P_{i,1}$, $P_{i,2}$, up to $P_{i,m'}$ in sequence. Before the simulation of h_i can start its work, the non-occurrence of the letters in $noc(h_i)$ has to be tested. This is done with the below given t-component:

$$P_{i,m'+1} = \{[p_i, A_1 \ldots A_{m'}] \to p_i\} \cup \{\, [A, i] \to F \mid A \in noc(h_i) \,\}.$$

Now the simulation of h_i is done by the t-component

$$P_i = \{p_i \to [p_0, \lambda]\} \cup \{\, [A, i] \to g(w) \mid w \in h_i(A) \,\} \cup \{[p_i, v] \to F \mid v \in \Sigma^{\leq m}\},$$

where g is the homomorphism such that $g(A) = [A, 0]$, for $A \in \Sigma$.

Finally, in order to terminate the derivation process one applies the following production set which runs in t-mode:

$$P_{term} = \{[p_0, \lambda] \to a\} \cup \{\, [A, 0] \to A \mid A \in \Delta \,\} \cup \{\, [A, 0] \to F \mid A \in \Sigma \setminus \Delta\}.$$

This completes the description of G.

The reader may easily verify that the constructed hybrid CDGS simulates the original ET0L system with random context and generates $a \cdot \delta_a(L)$. Observe, that the hybrid CDGS has λ-productions only if the programmed grammar has. Thus, the claim follows. □

Now let us turn our attention to the hierarchy induced by the number of components. Here we find the following situation:

Theorem 2. *Let $n \in \mathbb{N} \cup \{\infty\}$, with $n \geq 4$, then we have*

$$\begin{aligned} \mathcal{L}(\mathrm{CF}) = \mathcal{L}(\mathrm{HCD}_1, \mathrm{CF}[-\lambda], \{t\} \cup \{\bar{t}\}) &\subseteq \mathcal{L}(\mathrm{HCD}_2, \mathrm{CF}[-\lambda], \{t\} \cup \{\bar{t}\}) \\ &\subseteq \mathcal{L}(\mathrm{HCD}_3, \mathrm{CF}[-\lambda], \{t\} \cup \{\bar{t}\}) \\ &\subseteq \mathcal{L}(\mathrm{HCD}_n, \mathrm{CF}[-\lambda], \{t\} \cup \{\bar{t}\}) \\ &= \mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac}). \end{aligned}$$

Proof. The first equality and the inclusions are trivial. Finally, four components can do the job of an arbitrary number of components, which can be proved by an adaption of the construction given in [Mitrana 1993]. □

We have not been able to settle the question of which of the above inclusions are strict ones. Clearly, it is known that

$$\mathcal{L}(\mathrm{ET0L}) \subseteq \mathcal{L}(\mathrm{HCD}_3, \mathrm{CF}[-\lambda], \{t\} \cup \{\bar{t}\})$$

since any ET0L system can be simulated by a CDGS with three components working in t-mode of derivation [Csuhaj-Varjú, Dassow 1990], but it is open whether or not this inclusion is getting strict when also non-t-components are allowed. We do not even know whether or not this inclusion still holds if one or two of the components run in non-t-mode instead of t-mode. We are only able to prove the following two lemmas.

Lemma 3. *Any E0L language is generated by a hybrid CDGS with two t-components and one non-t-component.*

Proof. Consider an E0L system $G = (\Sigma, h, \omega, \Delta)$, where Σ is the total alphabet, $\Delta \subseteq \Sigma$ the set of terminals, $\omega \in \Sigma^*$ the axiom, and h is the production set. Let $\Delta' = \{ a' \mid a \in \Delta \}$ and $\Sigma'' = \{ a'' \mid a \in \Sigma \}$ and let g_1 and g_2 be the following morphisms: $g_1(a) = a$ if $a \in \Sigma \setminus \Delta$, $g_1(a) = a'$ if $a \in \Delta$, and $g_2(a) = a''$ for each $a \in \Sigma$.

We construct the hybrid CDGS

$$H = (\Sigma \cup \Delta' \cup \Sigma'' \cup \{S, T, T', T_1, T_2, F\}, \Delta, \{P_1, P_2, P_3\}, S),$$

the unions being disjoint, having the components

$$\begin{aligned} P_1 &= \{T' \to T, T_1 \to F, T_2 \to F\} \cup \{ a \to F \mid a \in \Delta \} \\ &\quad \cup \{ g_1(a) \to g_2(w) \mid a \to w \in h \}, \\ P_2 &= \{S \to T g_1(\omega), T \to T', T \to T_1, T_2 \to \lambda\} \cup \{g_2(a) \to g_1(a) \mid a \in \Sigma\}, \end{aligned}$$

and

$$P_3 = \{T_1 \to T_2, T_2 \to T_2\} \cup \{ g_1(a) \to a \mid a \in \Delta \},$$

where P_1 and P_2 are working in t-mode and P_3 in non-t-mode.

The derivation process has to start by applying P_2 yielding either $T_1 g_1(\omega)$ or $T' g_1(\omega)$. Whenever a sentential form $T_1 g_1(v)$, with $v \in \Sigma^*$, has been derived by P_2, one can successfully continue the derivation only with component P_3, since F is a failure symbol which can never be rewritten. Then one can reach a terminal string if and only if $v \in \Delta^*$ by

$$T_1 g_1(v) \Rightarrow_3^{\bar{t}} T_2 v \Rightarrow_2^{t} v$$

disregarding useless applications of P_2 in intermediate steps.

Starting from a form $T' g_1(v)$, one can simulate one derivation step $v \Rightarrow w$ in G by the use of P_1 and P_2 in this sequence leading to $T g_1(w)$. Any further derivation in H introduces a failure symbol. In conclusion, $L(H) = L(G)$ holds. □

Lemma 4. *There is a hybrid CDGS with one t-component and two non-t components generating a non-context-free language.*

Proof. We show how to generate the language $L = \{ a^n b^n c^m \mid 1 \leq n \leq m\}$. Let G be a hybrid CDGS with the set of nonterminals $N = \{S, F, A, A', B, B'\}$, the set of terminals $T = \{a, b, c\}$, and the production sets

$$\begin{aligned} P_1 &= \{S \to AB, A \to aA'b, B \to B'c, A \to ab, B \to c\}, \\ P_2 &= \{B' \to B, A' \to F\}, \end{aligned}$$

and

$$P_3 = \{A' \to A, B \to F\},$$

where the P_1 runs in t-mode whereas P_2 and P_3 are non-t-components.

Observe, that only the first component is applicable to the axiom S and to any sentential form $a^k A b^k B c^\ell$, with $0 \leq k \leq \ell$. Note that derivations of words in which only one nonterminal occurs cannot be continued to terminating derivations due to the non-t features of P_2 and P_3. Thus, we only need to discuss the cases that P_1 yields $a^{k+1} b^{k+1} c^{\ell+1} \in L$ or $a^{k+1} A' b^{k+1} B' c^{\ell+1}$. In the first case, the derivation is finished, in the second case only P_2 is applicable leading to $a^{k+1}A'b^{k+1}Bc^{\ell+1}$. Now, the derivation can be continued either by P_3 leading to $a^{k+1}Ab^{k+1}Bc^{\ell+1}$ or by P_1. In the latter case, by our remarks given above, only rule $B \to B'c$ can be used, and we have to continue with P_2, again. Now, the reader can readily see that $L(G) = L$. $\square$

Now consider the $(t\wedge\geq k)$-mode of derivation in combination with the non-t-mode. For ordinary CDGS's working in $(t\wedge\geq k)$-mode it was shown in [Fernau, Freund, Holzer 1998] that

$$\mathcal{L}(\mathrm{CD}_\infty, \mathrm{CF}[-\lambda], (t\wedge\geq 1)) = \mathcal{L}(\mathrm{ET0L})$$

and

$$\mathcal{L}(\mathrm{CD}_\infty, \mathrm{CF}[-\lambda], (t\wedge\geq k)) = \mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac}) \quad \text{if } k \geq 2.$$

Here we show that adding non-t-mode components to CDGS's working in $(t\wedge\geq k)$-mode for $k \geq 2$ does not improve the generating power of the underlying systems, whereas $(t\wedge\geq 1)$-components in combination with non-t-components are sufficient to simulate the class of recurrent programmed context-free grammars.

Corollary 5. *For each $k \geq 1$,*

$$\mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t\wedge\geq k)\}) = \mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac}).$$

Proof. Observe, that the $(t\wedge\geq 1)$-mode and the ordinary t-mode coincide. Moreover, by the prolongation technique as used in [Fernau, Freund, Holzer 1998] a component working in $(t\wedge\geq 1)$-mode can be simulated by a component working in $(t\wedge\geq k)$-mode for arbitrary $k \geq 2$. With this, together with Theorem 1, we already have

$$\begin{aligned}\mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac}) &= \mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{t\} \cup \{\bar{t}\}) \\ &= \mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t\wedge\geq 1)\}) \\ &\subseteq \mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t\wedge\geq k)\}).\end{aligned}$$

Finally, combining the simulation of the non-t-components as described in the proof of Theorem 1, and the $(t\wedge\geq k)$-components as described in [Fernau, Freund, Holzer 1998] *via* a recurrent programmed context-free grammar leads to the inclusion

$$\mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t\wedge\geq k)\}) \subseteq \mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac}),$$

which proves our assertion. $\square$

Remark. Further generalizations of Theorem 1 and Corollary 5 concerning the non-t-mode are the following. Observe, that the non-t-mode equals the modes $(\bar{t}\wedge\leq 1)$, $(\bar{t}\wedge= 1)$, and $(\bar{t}\wedge\geq 1)$. Moreover, the $(\bar{t}\wedge\leq 1)$-mode is identical with the $(\bar{t}\wedge\leq k)$-mode for arbitrary $k \geq 1$. Finally, note that by a simple adaption of the $P_{i,j}$-components from Theorem 1 and the fact that $(\bar{t}\wedge\geq k)$-components can be simulated by a recurrent programmed grammar—the simulation of a $(\bar{t}\wedge\geq k)$-component is quite similar to a simulation of a $(t\wedge\geq k)$-component which can be found in [Fernau, Freund, Holzer 1998] and [Fernau, Holzer, Freund 1997]—we obtain that instead of using non-t-components in the result of Theorem 1 and Corollary 5 one can use modes from the set

$$\{\, (\bar{t}\wedge\leq k) \mid k \in \mathbb{N} \,\} \cup \{(\bar{t}\wedge= 1)\} \cup \{\, (\bar{t}\wedge\geq k) \mid k \in \mathbb{N} \,\}.$$

We have to leave open the question whether the non-t-mode can be replaced by the $(\bar{t} \wedge = k)$-mode for $k \geq 2$.

Since the $(t \wedge \geq 1)$-mode and the ordinary t-mode coincide, the following result is obvious by Theorem 2. We state the result without proof.

Theorem 6. *Let $n \in \mathbb{N} \cup \{\infty\}$, with $n \geq 4$, then we have*

$$\begin{aligned}
\mathcal{L}(\mathrm{CF}) &= \mathcal{L}(\mathrm{HCD}_1, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge \geq 1)\}) \\
&\subseteq \mathcal{L}(\mathrm{HCD}_2, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge \geq 1)\}) \\
&\subseteq \mathcal{L}(\mathrm{HCD}_3, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge \geq 1)\}) \\
&\subseteq \mathcal{L}(\mathrm{HCD}_n, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge \geq 1)\}) = \mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac}). \qquad \square
\end{aligned}$$

Remark. As in the case of Theorem 2 we also have not been able to settle the question of which of the inclusions in the above given theorem are strict ones. Obviously, Lemma 3 and 4 remain valid if one uses $(t \wedge \geq 1)$-components instead of t-components.

For general k the situation is only a little bit different.

Theorem 7. *Let $n \in \mathbb{N} \cup \{\infty\}$, with $n \geq 3$. For every $k \in \mathbb{N}$, with $k \geq 2$,*

$$\begin{aligned}
\mathcal{L}(\mathrm{CF}) &= \mathcal{L}(\mathrm{HCD}_1, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge \geq k)\}) \\
&\subset \mathcal{L}(\mathrm{HCD}_2, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge \geq k)\}) \\
&\subseteq \mathcal{L}(\mathrm{HCD}_n, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge \geq k)\}) = \mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac}).
\end{aligned}$$

Proof. The inclusions themselves are trivial and the first equality is obvious. For the strictness of the inclusion

$$\mathcal{L}(\mathrm{CF}) \subseteq \mathcal{L}(\mathrm{HCD}_2, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge \geq k)\})$$

we restrict ourselves to the case $k = 2$. The simulation for arbitrary k is quite similar and left to the reader. Let G be a hybrid CDGS with nonterminals $\{S, S', A, B, A', B', F\}$, terminals $\{a, b, c\}$, axiom S, and the production sets

$$P_1 = \{S \to S', S' \to AB, A' \to aAb, B' \to Bc, A' \to ab, B' \to c\}$$

and

$$P_2 = \{A \to A', B \to B', A' \to F, B' \to F\}.$$

Here the production set P_1 runs in $(t \wedge \geq 2)$-mode and P_2 in non-t-mode. The language generated by G is the non-context-free language $\{\, a^n b^n c^n \mid n \geq 1 \,\}$.

Observe, that F is a failure symbol that one cannot get rid off whenever it has been introduced into the sentential form. Thus, in the following argumentation, we can ignore any derivation leading to a form in which F occurs.

Obviously, it is possible to derive AB *via* P_1, and hence by P_2 and P_1 again the word abc. Starting from a sentential form $a^kAb^kBc^k$, with $k \geq 0$, the only applicable table is P_2, which results in either $a^kAb^kBc^k$, $a^kA'a^kBc^k$, $a^kAb^kB'c^k$, or $a^kA'a^kB'c^k$. The only word for which we can successfully continue the derivation using P_1 is $a^kA'a^kB'c^k$, which results in the following words $a^{k+1}Ab^{k+1}Bc^{k+1}$, $a^{k+1}Ab^{k+1}c^{k+1}$, $a^{k+1}b^{k+1}Bc^{k+1}$, or $a^{k+1}b^{k+1}c^{k+1}$. Since $a^{k+1}Ab^{k+1}c^{k+1}$ and $a^{k+1}b^{k+1}Bc^{k+1}$ only contain one nonterminal, no terminal word can be derived from these words. Thus, the only way to continue the derivation with P_2 is to use $a^{k+1}Ab^{k+1}Bc^{k+1}$. This shows that G generates $\{ a^n b^n c^n \mid n \geq 1 \}$.

Finally, the equality

$$\mathcal{L}(\mathrm{HCD}_n, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge \geq k)\}) = \mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac})$$

for $n \geq 3$ and $k \geq 2$ already follows from [Fernau, Freund, Holzer 1998], where it was shown that every CDGS with an arbitrary number of components working in $(t \wedge \geq k)$-mode can be simulated by such a CDGS with three components. There two components serve as switches between the nonterminal colours and one component does the actual simulation. □

4 Combining $(t \wedge = k)$ or $(t \wedge \leq k)$ with $\bar{t}$

In this section we consider hybrid CDGS's with components working together according the $(t \wedge = k)$ and non-t protocol. Unless stated otherwise, all results given for the $(t \wedge = k)$-mode are also valid for the $(t \wedge \leq k)$-mode. For CDGS's with components only working in the former mode, it was shown in [Fernau, Holzer, Freund 1997] that for each $k \in \mathbb{N}$,

$$\mathcal{L}(\mathrm{CD}_\infty, \mathrm{CF}[-\lambda], (t \wedge = k)) = \mathcal{L}_{fin}(\mathrm{P}, \mathrm{CF}[-\lambda]),$$

where $\mathcal{L}_{fin}(\mathrm{P}, \mathrm{CF}[-\lambda])$ denotes the family of languages of finite index generated by programmed context-free grammars without appearance checking. Loosely speaking, the index of a grammar is the maximal number of nonterminals simultaneously appearing in a sentential form during a terminating derivation, considering the most economical derivation for each string. For a precise definition we refer to, e.g., [Dassow, Păun 1989].

The next theorem shows that adding non-t-components to a CDGS working in $(t \wedge = k)$-mode skips the finite index property of the above mentioned result (and adds the appearance checking feature).

Theorem 8. *For each $k \in \mathbb{N}$, we have*

$$\mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge = k)\}) = \mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda], \mathrm{ac}).$$

Proof. The inclusion from left to right is obvious, because arbitrary Boolean combinations of derivation modes can be simulated by a programmed context-free grammar in a straightforward manner. We briefly describe the other direction.

By a standard argument the involved hybrid CDGS language family is closed under union and embraces the finite languages. Let $L \subseteq T^*$ be a language in $\mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda], \mathrm{ac})$, then

$$L = \bigcup_{a \in T} (a \cdot \delta_a(L) \cap L) \cup (L \cap T) \cup (L \cap \{\lambda\}).$$

Since L is in $\mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda], \mathrm{ac})$, language $\delta_a(L) = \{ w \in T^+ \mid aw \in L \}$ is in $\mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda], \mathrm{ac})$ due to the closure properties of that family under derivatives. Thus, it is sufficient for the proof of the present assertion to show that $\{a\} \cdot \delta_a(L)$ is in $\mathcal{L}(\mathrm{HCD}, \mathrm{CF}[-\lambda], \{t\} \cup \{(t \wedge = k)\})$ provided that $\delta_a(L)$ is a programmed context-free language. In the remainder we restrict ourselves to the case $k = 1$. The result generalizes to arbitrary k, by using the prolongation technique.

Let $G = (N, T, P, S, \Lambda, \sigma, \phi)$ be a programmed grammar generating $\delta_a(L)$. Without loss of generality assume that for all p in Λ, label p does *not* belong to both $\sigma(p)$ and $\phi(p)$. Further assume that $N \cap \Lambda = \emptyset$.

We construct a hybrid CDGS with nonterminals

$$N' = \{S', F\} \cup N \cup \{\, p, p' \mid p \in \Lambda \,\} \cup \{\, [A, p] \mid A \in N \text{ and } p \in \Lambda \,\},$$

terminals T, and axiom S'. To start the derivation, we use S' as axiom and the $(t \wedge = 1)$-component

$$P_{init} = \{S' \to pS \mid p \in \Lambda\,\}.$$

Then for each rule $\Lambda(p) = (A \to w, \sigma, \phi)$ we construct the following production sets. The below given non-t-components $P_{p,1}$ and $P_{p,2}$ are only successfully applicable in the order $P_{p,1}$ followed by $P_{p,2}$ to a sentential form containing label p and at least one nonterminal A, i.e., a form $p\alpha A\beta$. The components are defined as

$$P_{p,1} = \{A \to [A, p], p \to F\} \quad \text{and} \quad P_{p,2} = \{\, p \to p', [A, p] \to F\}$$

Note, that after the application of $P_{p,1}$ and $P_{p,2}$ some, but at least one, of the A's in $p\alpha A\beta$ are replaced by $[A, p]$ symbols.

The simulation of and application of $A \to w$ is done with the help of

$$P_{p,3} = \{[A, p] \to w\} \cup \{\, q \to F \mid q \in \Lambda\,\} \cup \{\, q' \to F \mid q \in \Lambda \setminus \{p\}\,\},$$

which runs in $(t \wedge = 1)$-mode. Since this component runs in $(t \wedge = 1)$-mode the actual sentential form must contain at most one $[A, p]$ nonterminal, i.e., it must look like $p'\alpha[A, p]\beta$. Finally, to get rid off the primed label symbol again a $(t \wedge = 1)$ component is applied:

$$P_{p,4} = \{\, p' \to q \mid q \in \sigma(p)\,\} \cup \{\, [B, q] \to F \mid B \in N \text{ and } q \in \Lambda\,\}.$$

Here the rules $[B, q] \to F$ forces the hybrid CDGS to use $P_{p,3}$ before. Hence, we find

$$(\alpha A\beta, p) \Rightarrow (\alpha w\beta, q) \quad \text{with } q \in \sigma(p) \text{ in the programmed grammar } G$$

if and only if

$$p\alpha A\beta \Rightarrow^{\bar{t}}_{p,1} p\alpha[A, p]\beta \Rightarrow^{\bar{t}}_{p,2} p'\alpha[A, p]\beta \Rightarrow^{(t\wedge=1)}_{p,3} p'\alpha w\beta \Rightarrow^{(t\wedge=1)}_{p,4} q\alpha w\beta.$$

In case the considered sentential form $p\alpha$ does not contain A's and $[A, q]$'s, the component

$$P_{p,5} = \{\, p \to q \mid q \in \phi(p)\,\} \cup \{\, A \to F, [A, q] \to F \mid q \in \Lambda\,\},$$

running in $(t \wedge = 1)$-mode results in $q\alpha$ with $q \in \phi(p)$, i.e.,

$$(\alpha, p) \Rightarrow (\alpha, q) \quad \text{with } q \in \phi(p) \text{ in the programmed grammar } G$$

if and only if $p\alpha \Rightarrow^{(t\wedge=1)}_{p,5} q\alpha$.

To terminate the derivation process the component

$$P_{term} = \{\, p \to a \mid p \in \Lambda \,\} \cup \{\, A \to F, [A,p] \to F \mid A \in N \text{ and } p \in \Lambda \,\},$$

running in $(t \wedge =1)$-mode is used. This completes the description of the hybrid CDGS. Obviously, the constructed grammar system simulates the original programmed context-free grammar correctly. Moreover, note that the CDGS has λ-productions only if the programmed grammar has λ-productions. □

Remark. In what concerns a generalization of the above given theorem to components other than non-t ones, in sense of the remark given after Corollary 5, we find the following situation. Obviously, non-t is identical to $(\bar{t} \wedge \leq k)$ for arbitrary $k \in \mathbb{N}$, $(\bar{t} \wedge =1)$, and $(\bar{t} \wedge \geq 1)$. Moreover, slight modifications on the components $P_{p,1}$ and $P_{p,2}$ in the proof of the above given theorem, shows that also the $(\bar{t} \wedge =k)$- and $(\bar{t} \wedge \geq k)$-modes can be used instead of the non-t-mode to simulate a programmed grammar. Since programmed grammars can simulate arbitrary Boolean combinations of derivations modes we find that the result of Theorem 8 still holds if one uses a mode from the set

$$\{\, (\bar{t} \wedge \leq k), (\bar{t} \wedge =k), (\bar{t} \wedge \geq k) \mid k \in \mathbb{N} \,\}$$

instead of non-t.

Now let us turn our attention to the number of components in a hybrid CDGS with non-t and $(t \wedge =k)$-components. The easiest case is to have one component only. By the definition of the $(t \wedge =k)$-mode, every derivation has length exactly k, so that we only get finite languages.

Lemma 9. *For every* $k \in \mathbb{N}$, $\mathcal{L}(\mathrm{FIN}) = \mathcal{L}(\mathrm{HCD}_1, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge = k)\})$. □

For two components, the situation is a little bit more involved, because we have to distinguish whether the components work in the $(t \wedge =1)$- or in the $(t \wedge =k)$-mode, for $k \geq 2$.

Lemma 10. *For each* $k \in \mathbb{N}$, $\mathcal{L}(\mathrm{CF}) \subseteq \mathcal{L}(\mathrm{HCD}_2, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge = k)\})$. *The inclusion is strict if* $k \geq 2$.

Proof. Let $k = 1$ and $G = (N, T, S, P)$ be a context-free grammar. We construct a hybrid CDGS H with nonterminals $N \cup \{\, A' \mid A \in N \,\} \cup \{F\}$, terminals T, axiom S and the production sets

$$P_1 = \{\, A \to A' \mid A \in N \,\} \cup \{\, A' \to F \mid A \in N \,\}$$

and

$$P_2 = \{\, A' \to w \mid A \to w \text{ is in } P \,\}.$$

Here production set P_1 runs in non-t-mode and P_2 in $(t \wedge =1)$-mode. This completes the description of H. The reader may easily verify that $\alpha A \beta \Rightarrow \alpha w \beta$ using $A \to w$ in P if and only if $\alpha A \beta \Rightarrow_1^{\bar{t}} \alpha A' \beta \Rightarrow_2^{(t \wedge = 1)} \alpha w \beta$. This shows that the hybrid CDGS H is equivalent to the context-free grammar G. Therefore,

$$\mathcal{L}(\mathrm{CF}) \subseteq \mathcal{L}(\mathrm{HCD}_2, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge = 1)\}).$$

For arbitrary k, the inclusions follow form the above given construction and the prolongation technique as used in [Fernau, Freund, Holzer 1998]. For the strictness of these inclusions we use the hybrid CDGS constructed in the proof of Theorem 7, but now the first component has to run in $(t \wedge = 2)$-mode. For arbitrary k one has to adapt the construction given in Theorem 7 accordingly. The details are left to the reader. □

Remark. Obviously, $\mathcal{L}(\mathrm{CF}) \subseteq \mathcal{L}(\mathrm{HCD}_2, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{(t \wedge \leq 1)\})$. We do not know whether the above Theorem holds for the $(t \wedge \leq k)$-mode in general.

5 Combining the remaining classical modes with the $\bar{t}$-mode

As the title of this section indicates, we are going to combine the remaining classical modes $*$, $\leq$, $\geq$, and $=$ with the non-t-mode in a hybrid CDGS.

Before we come to the first theorem of this section we have to introduce the notion of context-free grammar with random context. A *context-free grammar with random context* is a construct $G = (N, T, P, S, oc, noc)$. Here, (N, T, P, S) is an ordinary context-free grammar, where N is the nonterminal alphabet, T is the terminal alphabet, P is a finite set of context-free productions, and $S \in N$ is the axiom. Furthermore, oc and noc are functions from P into the set of subsets of N. For the sake of simplicity and clarity, we shall write productions $p = (A \to w)$ of a random context grammar in the form $(A \to w, oc(p), noc(p))$. For x and y in $(N \cup T)^*$, we write $x \Rightarrow y$ if and only if $x = z_1 A z_2$, $y = z_1 w z_2$, $p = (A \to w, oc(p), noc(p))$ is a rule in P, and all symbols of $oc(p)$ occur in $z_1 z_2$ but no symbol in $noc(p)$ occurs in $z_1 z_2$. The language generated by G is defined as

$$L(G) = \{\, w \in T^* \mid S \overset{*}{\Rightarrow} w \,\},$$

where $\overset{*}{\Rightarrow}$ denotes the reflexive and transitive closure of the yield relation $\Rightarrow$.

Now we are ready to start our investigations with the $*$-mode of derivation. There we find the following situation:

Theorem 11. $\mathcal{L}(\mathrm{CF}) \subset \mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{*\} \cup \{\bar{t}\}) \subseteq \mathcal{L}(\mathrm{RC}, \mathrm{CF}[-\lambda])$.

Proof. The first inclusion is trivial and the strictness can be seen by Example 1. The second inclusion is shown next. Let G be a context-free hybrid CDGS of arbitrary degree n with components working in $*$- and non-t-mode. We define $X_i = \{\, [A, i] \mid A \in N \cup T \,\} \cup \{[L, i]\}$ for $1 \leq i \leq n$ and we construct the context-free random context grammar

$$H = (N \cup \bigcup_{i=1}^{n} X_i \cup \{S'\}, T, P, S'),$$

where the unions are being disjoint, and P is the union of the sets P_{init}, P_1, P_2, and P_{term} of random context productions. With

$$P_{init} = \{\, (S' \to [S, i], \emptyset, \emptyset) \mid 1 \leq i \leq n \,\}.$$

one may choose a component to be applied first. Then

$$\begin{aligned} P_1 = \{\, &(A \to w, \{[B,i]\}, \emptyset) \mid (A \to w) \in P_i \text{ and } [B,i] \in X_i \text{ for } 1 \le i \le n \,\} \\ &\cup \{\, ([A,i] \to [a,i]v, \emptyset, \emptyset) \mid (A \to av) \in P_i, \text{ for } 1 \le i \le n, \\ &\qquad\qquad \text{and } av \in (N \cup T)^+ \,\} \\ &\cup \{\, ([A,i] \to [L,i], \emptyset, \emptyset) \mid (A \to \lambda) \in P_i, \text{ for } 1 \le i \le n \,\}. \end{aligned}$$

simulates the application of productions in the chosen component that is specified by the colour, i.e., either a nonterminal $[A,i]$ or $[L,i]$ is present in the sentential form. To change the component to be simulated rules from P_2 are used:

$$\begin{aligned} P_2 = \{\, &([A,i] \to [A,j], \{B\}, \emptyset) \mid [A,i] \in X_i,\ [A,j] \in X_j,\ 1 \le i,j \le n, \\ &\qquad (B \to w) \in P_i, \text{ and } P_i \text{ is a } \bar{t}\text{-component} \,\} \\ &\cup \{\, ([A,i] \to [A,j], \emptyset, \emptyset) \mid [A,i] \in X_i,\ [A,j] \in X_j,\ 1 \le i,j \le n, \\ &\qquad (A \to w) \in P_i, \text{ and } P_i \text{ is a } \bar{t}\text{-component} \,\} \\ &\cup \{\, ([A,i] \to [A,j], \emptyset, \emptyset) \mid [A,i] \in X_i,\ [A,j] \in X_j,\ 1 \le i,j \le n, \text{ and} \\ &\qquad P_i \text{ is a } *\text{-component} \,\}. \end{aligned}$$

Observe, that the colour that corresponds to a non-t-component can be changed only if there is still a production of that component applicable whereas the simulation of $*$-components can be interrupted at any moment.

To terminate the derivation process use

$$\begin{aligned} P_{term} = \{\, &([a,i] \to a, \emptyset, \emptyset) \mid a \in T,\ 1 \le i \le n, \text{ and } P_i \text{ is a } *\text{-component} \,\} \\ &\cup \{\, ([L,i] \to \lambda, \emptyset, \emptyset) \mid 1 \le i \le n \text{ and } P_i \text{ is a } *\text{-component} \,\}. \end{aligned}$$

Please note that after deleting a symbol $[L,i]$ by some production in the last set, no further derivation step can be done. Thus, such deletion completes a derivation of a terminal string as the final step or it blocks the derivation. Concerning the use of a production of the form $([a,i] \to a, \emptyset, \emptyset)$, we encounter the same situations as when deleting a $[L,i]$. Moreover, whenever H produces a terminal string by a production of the last group, we must have applied a production in the previous derivation step which corresponds to a $*$-component of G, since non-t-components cannot terminate themselves. By our remarks, the reader can readily see that $L(H) = L(G)$ holds.

Now, let G be λ-free. Then we construct H as above omitting all productions with a symbol $[L,i]$ on the left-hand or on the right-hand side. Clearly, such productions are not needed in this case, and we still have $L(H) = L(G)$. □

Since $\mathcal{L}(\mathrm{RC}, \mathrm{CF}[-\lambda]) \subseteq \mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda])$, which was shown in [Fernau, Wätjen 1998], we obtain the following corollary:

Corollary 12. $\mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{*\} \cup \{\bar{t}\}) \subseteq \mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda])$. □

In what concerns the remaining classical modes, the situation is quite similar, but the upper bound we know is a little bit weaker.

Theorem 13. *1.* $\mathcal{L}(\mathrm{CF}) \subset \mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{\le k\} \cup \{\bar{t}\}) \subseteq \mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda])$.

2. $\mathcal{L}(\mathrm{CF}) \subset \mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{=k\} \cup \{\bar{t}\}) \subseteq \mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda])$.
3. $\mathcal{L}(\mathrm{CF}) \subset \mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{\geq k\} \cup \{\bar{t}\}) \subseteq \mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda])$.

Proof. The strict inclusions of $\mathcal{L}(\mathrm{CF})$ in the families generated by the hybrid CDGS's under consideration can be shown very similarly as in Theorem 11. In the following, we only prove $\mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{=k\} \cup \{\bar{t}\}) \subseteq \mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda])$. The other inclusions can be shown with similar arguments. Let

$$G = (N, T, S, (P_1, f_1), \ldots, (P_n, f_n))$$

be a context-free hybrid CDGS of arbitrary degree n with components working in $=k$- and non-t-mode. For $1 \leq i \leq n$, let $P_i = \{p_{i1}, p_{i2}, \ldots, p_{im_i}\}$ with $p_{ij} = (A_{ij} \to w_{ij})$, $1 \leq j \leq m_i$. We set $T' = \{a' \mid a \in T\}$, define the morphism h by $h(a) = a'$ if $a \in T$, and $h(A) = A$ if $A \in N$. Then we construct the programmed context-free grammar

$$H = (N \cup T' \cup \{S', L\}, T, P, \Lambda, \sigma, \phi, S'),$$

the unions being disjoint, and P as well as the label set Λ and the functions σ and ϕ are implicitly defined below.

The initialization is done by the rule $\Lambda(init) = (S' \to S)$ with the success field $\sigma(init) = \{\, [i, j, 0] \mid 1 \leq i \leq n \text{ and } 1 \leq j \leq m_i \,\}$ and $\phi(init) = \emptyset$. With the rule $\Lambda(init)$ going to some label $[i, j, 0]$ one may choose a component and a production in this component to be applied first.

The application of productions in the chosen component is simulated by the below given rules. We have to distinguish two cases: (1) If f_i equals the $=k$-mode, we define for $p_{ij} = (A_{ij} \to w_{ij}) \in P_i$ and $0 \leq \ell \leq k$ the rules $\Lambda([i, j, \ell]) = (A_{ij} \to h(w_{ij}))$ if $w_{ij} \neq \lambda$, and $\Lambda([i, j, \ell]) = (A_{ij} \to L)$ otherwise. The successor fields are $\sigma([i, j, \ell]) = \{[i, j', \ell + 1] \mid 1 \leq j' \leq m_i\}$ if $0 \leq \ell < k - 1$ and $\sigma([i, j, k]) = \{\, [i', j', 0] \mid 0 \leq i' \leq n,\ 1 \leq j' \leq m_{i'} \,\} \cup \{\, term_a \mid a \in T \cup \{L\} \,\}$. Finally set $\phi([i, j, \ell]) = \emptyset$, for any occurring i, j, and ℓ. These productions simulate the application of productions in the chosen component and allow to leave any $=k$-component if it has performed exactly k steps. (2) If f_i equals $\bar{t}$, then define for $p_{ij} = (A_{ij} \to w_{ij}) \in P_i$ the rules $\Lambda([i, j, 0]) = (A_{ij} \to h(w_{ij}))$ if $w_{ij} \neq \lambda$, and $\Lambda([i, j, 0]) = (A_{ij} \to L)$ otherwise, with success field $\sigma([i, j, 0]) = \{[i, j', 0] \mid 1 \leq j' \leq m_i\} \cup \{\, c_{i,B} \mid B \text{ is a left-hand side in } P_i \,\}$ and $\phi([i, j, 0]) = \emptyset$.

Moreover, for $B \in N$ set $\Lambda(c_{i,B}) = (B \to B)$, define the success field to be $\sigma(c_{i,B}) = \{\, [i', j', 0] \mid 0 \leq i' \leq n, 1 \leq j' \leq m_{i'} \,\}$ and $\phi(c_{i,B}) = \emptyset$. The former rules simulate the application of productions of the chosen component, and the latter rules associated with the labels $c_{i,B}$ check whether there is still a production of the non-t-component applicable.

To terminate the derivation process one uses the rules $\Lambda(term_a) = (a' \to a)$ if $a \in T$ and $\Lambda(term_L) = (L \to \lambda)$. The sucessor field $\sigma(term_b)$ equals the set $\{\, term_c \mid c \in T \cup \{L\} \,\}$ and $\phi(term_b) = \emptyset$ for $b \in T \cup \{L\}$.

By our remarks, the reader can readily see that $L(H) = L(G)$ holds. Now, let G be λ-free. Then we construct H as above omitting all productions with L on the left-hand or on the right-hand side. Clearly, such productions are not needed in this case, and we still have $L(H) = L(G)$. □

6 Conclusions

We examined the generative power of hybrid cooperating distributed grammar systems with non-t-components in addition to components working in well-investigated modes. It turned out that adding the non-t feature to CDGS's with one specific mode strictly enlarges the generative capacity in many cases. More precisely, we found the relations summarized in [Tab. 1], where at the intersection of the ith row, marked by a mode f_i, and column j, there appears either a family $\mathcal{L}$ which coincides with the family X at the top of the column having f_i-mode components or $\mathcal{L}_1 \subset \cdot \subseteq \mathcal{L}_2$ meaning that $\mathcal{L}_1 \subset X \subseteq \mathcal{L}_2$ holds.

Derivation mode f	$\mathcal{L}(\mathrm{CD}_\infty, \mathrm{CF}[-\lambda], f)$	$\mathcal{L}(\mathrm{HCD}_\infty, \mathrm{CF}[-\lambda], \{\bar{t}\} \cup \{f\})$
$*$	$\mathcal{L}(\mathrm{CF})$	$\mathcal{L}(\mathrm{CF}) \subset \cdot \subseteq \mathcal{L}(\mathrm{RC}, \mathrm{CF}[-\lambda])$
$\leq k$		$\mathcal{L}(\mathrm{CF}) \subset \cdot \subseteq \mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda])$
$=k;\ \geq k$		
t; $(t \wedge \geq 1)$	$\mathcal{L}(\mathrm{ET0L})$	$\mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac})$
$(t \wedge \leq k)$; $(t \wedge = k)$	$\mathcal{L}_{fin}(\mathrm{P}, \mathrm{CF}[-\lambda])$	$\mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda], \mathrm{ac})$
$(t \wedge \geq k)$ with $k \geq 2$	$\mathcal{L}(\mathrm{RP}, \mathrm{CF}[-\lambda], \mathrm{ac})$	

Table 1: Results for (hybrid) CDGS's summarized.

Interestingly, several new characterizations of the family of recurrent programmed languages are given, generated both in case with and in case without appearance checking. Furthermore, by adding non-t-components to $(t \wedge = k)$- or $(t \wedge \leq k)$-components, a "huge jump" in the generative capacity to the whole family of programmed languages is observed.

Acknowledgments

This work has partially been supported by Deutsche Forschungsgemeinschaft (DFG) grant La 618/3-2, by the National Sciences and Engineering Research Council (NSERC) of Canada grants OGP0089786 and RGPIN 9979-98, and by the *Fonds pour la Formation de Chercheurs et l'Aide à la Recherche* (FCAR) of Québec grants 00ER0642 and 91-ER-0642.

References

[Atanasiu, Mitrana 1989] Atanasiu, A., Mitrana, V.: "The modular grammars"; International Journal of Computer Mathematics, 30 (1989), 101–122.

[Aydin, Jürgensen, and Robbins 2000] Aydin, S., H. Jürgensen, and L. Robbins (2000). Dialogues as co-operating grammars. See Boldt and Jürgensen (2000).

[Boldt and Jürgensen 2000] Boldt, O. and H. Jürgensen (Eds.) (2000). Pre-Proceedings of Descriptional Complexity of Automata, Grammars and Related Structures, Number Report No. 555, London, Ontario, Canada. Department of Computer Science, The University of Western Ontario.

[Bordihn, Csuhaj-Varjú 1996] Bordihn, H., Csuhaj-Varjú, E.: "On competence and completeness in CD grammar systems"; Acta Cybernetica, 12, 4 (1996), 347–360.

[Bordihn, Csuhaj-Varjú, Dassow 1999] Bordihn, H., Csuhaj-Varjú, E., Dassow, J.: "CD grammar systems versus L systems"; In G. Păun and A. Salomaa (Eds.), Grammatical Models of Multi Agent Systems, pp. 18–32. Gordon and Breach (1999).

[Bordihn, Holzer 1999] Bordihn, H., Holzer, M.: "On a hierarchy of languages generated by cooperating distributed grammar systems"; Information Processing Letters, 69, 2 (1999), 59–62.

[Csuhaj-Varjú, Dassow 1990] Csuhaj-Varjú, E., Dassow, J.: "On cooperating/distributed grammar systems"; J. Inf. Process. Cybern. EIK (formerly Elektron. Inf.verarb. Kybern.), 26 1/2 (1990), 49–63.

[Csuhaj-Varjú, Dassow, Kelemen, Păun 1994] Csuhaj-Varjú, E., Dassow, J., Kelemen. J., Păun, Gh.: "Stratified grammar systems"; Computers and Artificial Intelligence, 13, 5 (1994), 409–422.

[Dassow, Mitrana 1996] Dassow, J., Mitrana, V.: "Fairness in grammar systems"; Acta Cybernetica, 12, 4 (1996), 331–345.

[Dassow, Păun 1989] Dassow, J., Păun, Gh.: "Regulated Rewriting in Formal Language Theory"; Volume 18 of EATCS Monographs in Theoretical Computer Science. Springer (1989).

[Dassow, Păun, Rozenberg 1997] Dassow, J., Păun, Gh., Rozenberg, G.: "Grammar systems"; In G. Rozenberg and A. Salomaa (Eds.), Handbook of Formal Languages, Volume 2, pp. 155–213. Springer (1997).

[Fernau, Freund, Holzer 1998] Fernau, H., Freund, R., Holzer, M.: "Hybrid modes in cooperating distributed grammar systems: internal versus external hybridization"; Accepted for publication in Theoretical Computer Science (1998).

[Fernau, Holzer, Freund 1997] Fernau, H., Holzer, M., Freund, R.: "Bounding resources in cooperating distributed grammar systems"; In S. Bozapalidis (Ed.), Proceedings of the 3rd International Conference Developments in Language Theory, Aristotle University of Thessaloniki (1997), 261–272.

[Fernau, Wätjen 1998] Fernau, H., Wätjen, D.: "Remarks on regulated limited ET0L systems and regulated context-free grammars"; Theoretical Computer Science, 194, 1/2 (1998), 35–55.

[Freund and Kelemenova 2000] Freund, R. and A. Kelemenova (Eds.) (2000). Proceedings of the International Workshop Grammar Systems 2000, Opava, Czech Republic. Silesian University.

[Jiménez-López and Martín-Vide 2000] Jiménez-López, M. D. and C. Martín-Vide (2000). Grammar systems and autolexical syntax: Two theories, one single idea. See Freund and Kelemenova (2000), pp. 283–296.

[Meersman, Rozenberg 1978] Meersman, R., Rozenberg, G.: "Cooperating grammar systems"; In Proceedings of Mathematical Foundations of Computer Science, Volume 64 of LNCS, Springer (1978), 364–374.

[Mitrana 1993] Mitrana, V.: "Hybrid cooperating/distributed grammar systems"; Computers and Artificial Intelligence, 12, 1 (1993), 83–88.

[Neubauer and Summerer 2000] Neubauer, M. and M. Summerer (2000). A graph controlled array grammar system with predefined teams for the recognition of hand-written characters. See Freund and Kelemenova (2000), pp. 309–328.

[Rosenkrantz 1969] Rosenkrantz, D. J.: "Programmed grammars and classes of formal languages"; Journal of the Association for Computing Machinery, 16, 1 (1969), 107–131.

[von Solms 1976] von Solms, S. H.: "Some notes on ET0L-languages"; International Journal of Computer Mathematics, 5, A (1976), 285–296.