Cooperating Distributed Grammar Systems: Components with Nonincreasing Competence

Maurice H. ter Beek[1], Erzsébet Csuhaj-Varjú[2,⋆], Markus Holzer[3,⋆⋆], and György Vaszil[4]

[1] Istituto di Scienza e Tecnologie dell'Informazione 'A. Faedo', Consiglio Nazionale delle Ricerche, Via G. Moruzzi 1, 56124 Pisa, Italy
maurice.terbeek@isti.cnr.it
[2] Computer and Automation Research Institute Hungarian Academy of Sciences, Kende utca 13–17, 1111 Budapest, Hungary
csuhaj@sztaki.hu
[3] Institut für Informatik, Universität Giessen Arndtstraße 2, 35392 Giessen, Germany
holzer@informatik.uni-giessen.de
[4] Computer and Automation Research Institute Hungarian Academy of Sciences, Kende utca 13–17, 1111 Budapest, Hungary
vaszil@sztaki.hu

Abstract. We study the generative power of CD grammar systems (CDGSs) that use a cooperation protocol based on the level of competence on a sentential form — and the competence of their components does not increase during rewriting. A component of a CDGS is k-competent on a sentential form if it can rewrite exactly k different nonterminals appearing in that string. A CDGS with components that are nonincreasing in competence works in $=k$-competence mode of derivation if no k-competent component can ever become ℓ-competent, for some $\ell > k$. We introduce a static and a dynamic definition to impose this restriction, and we relate the generative power of such CDGSs working either in a sequential or in a parallel manner and according to the aforementioned cooperation protocol, for various k, with that of context-free forbidding random context grammars, (random context) ET0L systems, and context-free programmed grammars with appearance checking.

1 Introduction

A grammar system is a set of grammars that under a specific cooperation protocol generate one language. The idea to consider — contrary to the classical paradigm "one

⋆ Also affiliated with: Department of Algorithms and Their Applications, Faculty of Informatics, Eötvös Loránd University, Pázmány Péter sétány 1/c, 1117 Budapest, Hungary.

⋆⋆ Part of the work was done while the author was at Institut für Informatik, Technische Universität München, Boltzmannstraße 3, 85748 Garching bei München, Germany.

J. Kelemen and A. Kelemenová (Eds.): Păun Festschrift, LNCS 6610, pp. 70–89, 2011.

grammar generating one language" — a set of cooperating grammars generating one language first appeared in [1]. An intensive exploration of the potential of grammar systems was not undertaken until [2] established a link between cooperating distributed grammar systems (CDGSs) and blackboard systems as known from artificial intelligence. Such a system consists of autonomous agents, a blackboard, and a control mechanism. The latter dictates rules which the agents must respect during their joint effort to solve a problem stated on the blackboard. The only way for the agents to communicate is *via* the blackboard, which represents the current state of problem solving. If the problem solving is successful, the solution appears on the blackboard. CDGSs form a language-theoretic framework for modelling blackboard systems. Agents are represented by grammars, the blackboard is represented by the sentential form, control is regulated by a cooperation protocol of the grammars, and the solution is represented by a terminal word. By now, grammar systems form a well-established and well-recognized area in the theory of formal languages [3,4].

In this paper, we introduce two variants of cooperation protocols for CDGSs, based on the level of competence that a component has on a sentential form during a derivation. We consider cooperation protocols that allow a component to start rewriting when a competence condition is satisfied, and that require it to do so as long as the grammar satisfies this condition. Intuitively, a component is k-competent on a sentential form if it can rewrite exactly k different nonterminals appearing in the sentential form. This particular cooperation protocol is called the $=k$-comp.-mode of derivation. The more different nonterminals of a sentential form a component is able to rewrite, the higher its (level of) competence on that sentential form. Restricting in this way the rewriting of the sentential form to components that have a certain (level of) competence, provides a formal interpretation of the requirement that agents must be competent enough before being able to participate in the problem solving taking place on the blackboard. Competence-based cooperation protocols have already been studied extensively in the literature, see, e.g., [1,2,5,6,7,8]. The variants we introduce here examine the consequences of imposing that no k-competent component can ever become ℓ-competent, for some $\ell > k$, by rewriting the sentential form. We introduce both a static and a dynamic definition of CDGSs whose components' competence is nonincreasing during a derivation, thus providing a formal interpretation of the assumption that the competence of agents does not increase while participating in the problem solving on the blackboard. In the first case we impose restrictions on the productions of the CDGSs, while in the second case we introduce a further restriction on the used competence-based derivation mode.

We show that such CDGSs, working in $=k$-comp.-mode of derivation and context-free rewriting either sequentially or in parallel, are very powerful, by relating them to context-free forbidding random context languages, (random context) ET0L languages [9], and languages generated by context-free programmed grammars with appearance checking. More precisely, we prove that CDGSs with nonincreasing competence in the static or in the dynamic sense working in $=1$-comp.-mode of derivation, rewriting in parallel, characterize the family of languages generated by ET0L systems. We prove that the same holds for CDGSs with nonincreasing competence in the static sense working in $=1$-comp.-mode of derivation, rewriting sequentially, while CDGSs

with nonincreasing competence in the dynamic sense working in $=1$-comp.-mode of derivation are strictly more powerful: their generative power equals that of context-free forbidding random context grammars. We moreover prove that CDGSs with nonincreasing competence in the static or in the dynamic sense working in $=k$-comp.-mode of derivation, with $k \geq 2$, rewriting in parallel, characterize the family of languages generated by context-free recurrent programmed grammars with appearance checking or — equivalently — that of random context ET0L languages. Finally, we show that CDGSs with nonincreasing competence in the static sense working in $=k$-comp.-mode of derivation, with $k \geq 2$, rewriting sequentially, are at least as powerful as random context ET0L languages, but their exact generative power is an open problem, while CDGSs with nonincreasing competence in the dynamic sense working in $=2$-comp.-mode of derivation, rewriting sequentially, can generate all recursively enumerable languages. In Section 6 we will provide a table that summarizes all results.

We thus provide yet another characterization of the family of languages generated by random context ET0L systems, which was shown to be related to CDGSs with nonstandard derivation modes, see, e.g., [6,7,10]. This language family is of particular interest as it coincides with that of context-free recurrent programmed languages and as such forms an intermediate class between the families of context-free random context languages and context-free programmed languages generated by grammars without appearance checking [11]. In fact, since we show that quite simple component grammars suffice to simulate random context ET0L systems, we demonstrate that it is indeed the cooperation protocol that is very powerful. We hope our results can help to gain more insight in a longstanding open problem in the theory of regulated rewriting: are context-free programmed grammars more powerful than context-free recurrent programmed grammars?

2 Preliminaries

We assume familiarity with basic formal language theory (see, e.g., [12,13]). We denote set difference by $\setminus$, set inclusion by $\subseteq$, strict set inclusion by $\subset$, cardinality of a finite set M by $|M|$, and the empty word by λ. We consider two languages L_1 and L_2 equal, and write $L_1 = L_2$, iff $L_1 \setminus \{\lambda\} = L_2 \setminus \{\lambda\}$.

An *ET0L system* is a quadruple $G = (\Sigma, H, \omega, \Delta)$, with alphabet Σ, finite set of complete tables H, axiom $\omega \in \Sigma^+$, and terminal alphabet $\Delta \subseteq \Sigma$. A *complete table* is a finite set of context-free rules, *i.e.*, elements of $\Sigma \times \Sigma^*$, which includes a rule for every $a \in \Sigma$. For $x, y \in \Sigma^*$, we write $x \Rightarrow_h y$ iff $x = a_1 \dots a_n$, $y = z_1 \dots z_n$, and for all $1 \leq i \leq n$, $a_i \to z_i \in h$ for some $h \in H$. The language generated by G is defined as

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

The language family generated by ET0L systems is denoted by $\mathcal{L}(\text{ET0L})$.

A *context-free random context grammar* is a quadruple $G = (N, T, P, S)$, with N and T its sets of nonterminals and terminals, $S \in N$ its start symbol, and P its finite set of context-free random context rules, *i.e.*, triples of the form $(A \to z, Q, R)$, where

$A \to z$ is a context-free production with $A \in N$ and $z \in (N \cup T)^*$, and $Q, R \subseteq N$ are its *permitting* and *forbidding* random context. For $x, y \in (N \cup T)^*$, we write $x \Rightarrow y$ iff $x = x_1 A x_2$, $y = x_1 z x_2$, all symbols of Q appear in $x_1 x_2$, and no symbol of R appears in $x_1 x_2$. If either Q and/or R is empty, then the corresponding context check is omitted. The language generated by G is defined as $L(G) = \{ w \in T^* \mid S \Rightarrow^* w \}$, where $\Rightarrow^*$ is the reflexive transitive closure of $\Rightarrow$. If all permitting random contexts are empty, then G is called a context-free forbidding random context grammar and the corresponding family of languages is denoted by $\mathcal{L}(\mathrm{fRC}, \mathrm{CF})$.

It is known (see, e.g., [12,13]) that $\mathcal{L}(\mathrm{ET0L}) \subset \mathcal{L}(\mathrm{fRC}, \mathrm{CF}) \subset \mathcal{L}(\mathrm{RE})$, where $\mathcal{L}(\mathrm{RE})$ denotes the family of recursively enumerable languages.

A context-free programmed grammar is a 7-tuple $G = (N, T, P, S, \Lambda, \sigma, \phi)$ with finite set of nonterminals N, finite set T of terminals, axiom $S \in N$, finite set P of context-free productions $\alpha \to \beta$, with $\alpha \in N$ and $\beta \in (N \cup T)^*$, and finite set Λ of labels (for the productions in P). Set Λ is a function that given a label outputs a production; σ and ϕ are functions from Λ into 2^{Λ}. For $(x, r_1), (y, r_2) \in (N \cup T)^* \times \Lambda$ and $\Lambda(r_1) = (\alpha \to \beta)$, we write $(x, r_1) \Rightarrow (y, r_2)$ iff either $x = x_1 \alpha x_2$, $y = x_1 \beta x_2$, with $x_1, x_2 \in (N \cup T)^*$, and $r_2 \in \sigma(r_1)$, or $x = y$, rule $\alpha \to \beta$ is not applicable to x, and $r_2 \in \phi(r_1)$. In the latter case, the derivation step is in appearance checking mode. Set $\sigma(r_1)$ is called the success field and set $\phi(r_1)$ the failure field of r_1. The language generated by G is defined as

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

in which $\stackrel{*}{\Rightarrow}$ denotes the reflexive transitive closure of $\Rightarrow$. Grammar G is a context-free recurrent programmed grammar if for every $p \in \Lambda$, $p \in \sigma(p)$, and if $\phi(p) \neq \emptyset$, then $\sigma(p) = \phi(p)$. The family of languages generated by context-free [recurrent] programmed grammars with appearance checking is denoted by $\mathcal{L}([\mathrm{r}]\mathrm{PR}, \mathrm{CF}, \mathrm{ac})$.

The family of languages generated by random context ET0L systems [9] is denoted by $\mathcal{L}(\mathrm{RC}, \mathrm{ET0L})$. It is known (see, e.g., [12,13]) that $\mathcal{L}(\mathrm{ET0L}) \subset \mathcal{L}(\mathrm{RC}, \mathrm{ET0L}) = \mathcal{L}(\mathrm{rPR}, \mathrm{CF}, \mathrm{ac}) \subseteq \mathcal{L}(\mathrm{PR}, \mathrm{CF}, \mathrm{ac}) = \mathcal{L}(\mathrm{RE})$.

3 Competence in CD Grammar Systems

A *cooperating distributed grammar system* (CDGS) of degree $n \geq 1$ is an $(n+3)$-tuple $G = (N, T, \alpha, P_1, \ldots, P_n)$, with disjoint alphabets N of nonterminals and T of terminals, axiom $\alpha \in (N \cup T)^*$, and components $P_1, \ldots, P_n$ that are finite sets of context-free productions, *i.e.*, productions of the form $A \to z$, for $A \in N$ and $z \in (N \cup T)^*$. This definition of CDGSs differs from the usual one by allowing arbitrary words from $(N \cup T)^*$ as axioms.

In this paper, we consider both sequential and parallel context-free rewriting. Let $1 \leq i \leq n$, let $\mathrm{dom}(P_i) = \{ A \in N \mid A \to z \in P_i \}$ be the *domain* of P_i, and let $x, y \in (N \cup T)^*$. Then we define a single *sequential rewriting step* as $x \Rightarrow_i y$ iff $x = x_1 A x_2$ and $y = x_1 z x_2$, for some $A \to z \in P_i$. To facilitate parallel rewriting, we first associate to each component P_i a finite substitution h_i defined as $h_i(A) = \{ z \mid A \to z \in P_i \}$ if $A \in \mathrm{dom}(P_i)$ and $h_i(A) = \{A\}$ if $A \in (N \cup T) \setminus \mathrm{dom}(P_i)$. Then

we define a *parallel rewriting step* as $x \Rightarrow_i y$ iff $y \in h_i(x)$. In both types of rewriting paradigms, subscript i thus refers to the component being used.

Recall from [8] the notion of competence that components of a CDGS have on a sentential form. Component P_i, with $1 \leq i \leq n$, is called k-*competent* on a sentential form x in $(N \cup T)^*$ iff $|\text{alph}_N(x) \cap \text{dom}(P_i)| = k$, where $\text{alph}_N(x) = \{ A \in N \mid x \in (N \cup T)^* A (N \cup T)^* \}$, *i.e.*, it denotes the set of all nonterminals occurring in x. We abbreviate the *(level of) competence* of component P_i on x by $\text{clev}_i(x)$, *i.e.*, $\text{clev}_i(x) = |\text{alph}_N(x) \cap \text{dom}(P_i)|$.

Based on the (level of) competence, we now recall from [6] the $=k$-competence-based cooperation protocol for CDGSs.[1] Let $k \geq 1$. Then

$x \Rightarrow_i^{=k\text{-comp.}} y$ iff there is a derivation

$$x = x_0 \Rightarrow_i x_1 \Rightarrow_i \cdots \Rightarrow_i x_{m-1} \Rightarrow_i x_m = y$$

for some $m \geq 1$ and it satisfies
(1) $\text{clev}_i(x_j) = k$ for $0 \leq j < m$ and $\text{clev}_i(x_m) \neq k$, or
(2) $\text{clev}_i(x_0) = k$, $\text{clev}_i(x_j) \leq k$ for $1 \leq j \leq m$, and $y \in T^*$.

Let $\Rightarrow^{=k\text{-comp.}}$ denote $\Rightarrow_i^{=k\text{-comp.}}$ for some i, with $1 \leq i \leq n$. The reflexive transitive closure of $\Rightarrow^{=k\text{-comp.}}$ is denoted by $\overset{*}{\Rightarrow}^{=k\text{-comp.}}$. The language generated by G in the $=k$-comp.-mode of derivation is

$$L_{=k\text{-comp.}}(G) = \{ w \in T^* \mid \alpha \overset{*}{\Rightarrow}^{=k\text{-comp.}} w \}.$$

The family of languages generated by CDGSs working in the $=k$-comp.-mode of derivation is denoted by $\mathcal{L}(\text{CD}, \text{CF}, =k\text{-comp.})$ for sequential rewriting and by $\mathcal{L}(\text{CD}, \text{parCF}, =k\text{-comp.})$ for parallel rewriting.

Example 1. Let $G = (N, T, \alpha, P_1, \ldots, P_8)$ be a CDGS with nonterminals $N = \{A, A', B, B', C, D\}$, terminals $T = \{a, b, c\}$, axiom AB and components

$P_1 = \{A \to aA'b, B' \to B', C \to C\}$, $P_5 = \{A' \to C, B \to B\}$,
$P_2 = \{A \to A, B \to B'c, C \to C\}$, $P_6 = \{A \to A, A' \to A', B' \to D\}$,
$P_3 = \{A' \to A, B \to B, C \to C\}$, $P_7 = \{B' \to B', C \to \lambda\}$, and
$P_4 = \{A' \to A', B' \to B, C \to C\}$, $P_8 = \{D \to \lambda\}$.

When working in the $=1$-comp.-mode of derivation, G generates the language $L_{=1\text{-comp.}}(G) = \{ a^n b^n c^n \mid n \geq 1 \}$, independently of whether it rewrites in a sequential or parallel manner. This can be seen as follows.

Each of the components P_1, P_3, P_5 and P_6 is 1-competent on the axiom. However, all but P_1 are unable to alter the axiom, and these components thus remain 1-competent forever. In those cases, the derivation enters a loop. Given the axiom, the only two-step derivation that does not loop is $AB \Rightarrow_1^{=1\text{-comp.}} aA'bB \Rightarrow_2^{=1\text{-comp.}} aA'bB'c$.

Now a choice must be made. Either we apply P_5 to derive $aA'bB'c \Rightarrow_5^{=1\text{-comp.}} aCbB'c \Rightarrow_6^{=1\text{-comp.}} aCbDc$, after which the derivation can be finished by

[1] In [6,7], we investigated also the $\leq k$- and $\geq k$-competence-based cooperation protocols.

$aCbDc \Rightarrow_7^{=1\text{-comp.}} abDc \Rightarrow_8^{=1\text{-comp.}} abc$ (or, instead, by applying P_8 before P_7). Otherwise we apply P_3 to derive $aA'bB'c \Rightarrow_3^{=1\text{-comp.}} aAbB'c \Rightarrow_4^{=1\text{-comp.}} aAbBc$, after which this sequence of applications of P_1, P_2, P_3 and P_4 can be repeated $n-1$ times, for some $n \geq 1$, to obtain $a^nAb^nBc^n$, from which we can derive $a^nAb^nBc^n \Rightarrow_1^{=1\text{-comp.}} a^nA'b^nBc^n \Rightarrow_2^{=1\text{-comp.}} a^nA'b^nB'c^n \Rightarrow_5^{=1\text{-comp.}} a^nCb^nB'c^n \Rightarrow_6^{=1\text{-comp.}} a^nCb^nDc^n \Rightarrow_7^{=1\text{-comp.}} a^nb^nDc^n \Rightarrow_8^{=1\text{-comp.}} a^nb^nc^n$ (or, instead, by applying P_8 before P_7). Clearly indeed the language $L_{=1\text{-comp.}}(G) = \{ a^nb^nc^n \mid n \geq 1 \}$ is generated.

We studied the generative power of CDGSs working in the $=k$-comp.-mode of derivation and rewriting sequentially (in [6]) or in parallel (in [7]), and obtained the inclusion chains $\mathcal{L}(\text{ET0L}) \subseteq \mathcal{L}(\text{CD}, \text{parCF}, =1\text{-comp.}) \subseteq \mathcal{L}(\text{CD}, \text{parCF}, f\text{-comp.}) = \mathcal{L}(\text{rPR}, \text{CF}, \text{ac})$ and $\mathcal{L}(\text{ET0L}) \subset \mathcal{L}(\text{fRC}, \text{CF}) \subseteq \mathcal{L}(\text{CD}, \text{CF}, =1\text{-comp.}) \subseteq \mathcal{L}(\text{CD}, \text{CF}, f\text{-comp.}) = \mathcal{L}(\text{RE})$, for $f \in \{ =k \mid k \geq 2 \}$.

4 CD Grammar Systems with Nonincreasing Competence

In CDGSs working in $=k$-comp.-mode of derivation, a component may start rewriting a sentential form provided it is k-competent on the string, and it must rewrite as long as it remains k-competent: only when it is no longer k-competent, another (k-competent) component may start. The moment a component is forced to quit rewriting, it is either j-competent, for some $j < k$, or ℓ-competent, for some $\ell > k$. In this paper, we introduce CDGSs in which the latter case simply cannot occur: no k-competent component can ever become ℓ-competent, for some $\ell > k$, by rewriting the sentential form, *i.e.*, a component's competence is *nonincreasing* during rewriting. We consider two ways of imposing this: a static and a dynamic one.

A CDGS $G = (N, T, \alpha, P_1, \ldots, P_n)$ is with *statically nonincreasing* competence, denoted by sni-CDGS, iff for all $1 \leq i \leq n$ and for all $A \to z$ in P_i,

$$(\text{alph}_N(z) \setminus \{A\}) \cap \text{dom}(P_i) = \emptyset.$$

In an sni-CDGS, no production with left-hand side A in a component P_i, with $1 \leq i \leq n$, is thus allowed to have any nonterminals other than A from $\text{dom}(P_i)$ in its right-hand side. This definition is strict: it does not take into account the nonterminals that are actually present in the sentential form. Assume a CDGS G with a component $P = \{A \to aB, B \to Cb\}$. Clearly G is not an sni-CDGS. However, P is 1-competent on a sentential form AC and cannot increase its competence by rewriting. On the contrary, while P is also 1-competent on a sentential form AAC, in this case applying P would increase its competence. This calls for a dynamic definition to formalize the intuitive notion of forbidding competence-increasing rewriting steps.

A CDGS $G = (N, T, \alpha, P_1, \ldots, P_n)$ works in *dynamically nonincreasing* competence-based fashion iff it works according to the *dynamically nonincreasing* $=k$-competence-based derivation mode, denoted by $=k$-comp.dni, defined as follows. Let $k \geq 1$. Then $x \Rightarrow_i^{=k\text{-comp.dni}} y$ iff there is a derivation $x = x_0 \Rightarrow_i x_1 \Rightarrow_i \cdots \Rightarrow_i x_{m-1} \Rightarrow_i x_m = y$, for some $m \geq 1$, and it satisfies:

(1) $\mathrm{clev}_i(x_j) = k$ for $0 \leq j < m$ and $\mathrm{clev}_i(x_m) < k$, or
(2) $\mathrm{clev}_i(x_0) = k$, $\mathrm{clev}_i(x_j) \leq k$ for $1 \leq j \leq m$, and $y \in T^*$.

As usual, $\Rightarrow^{=k\text{-comp.dni}}$ denotes $\Rightarrow_i^{=k\text{-comp.dni}}$ for some i, with $1 \leq i \leq n$, and the reflexive transitive closure of $\Rightarrow^{=k\text{-comp.dni}}$ is denoted by $\overset{*}{\Rightarrow}^{=k\text{-comp.dni}}$. The language generated by G in the $=k$-comp.dni-mode of derivation is

$$L_{=k\text{-comp.dni}}(G) = \{\, w \in T^* \mid \alpha \overset{*}{\Rightarrow}^{=k\text{-comp.dni}} w \,\}.$$

The family of languages generated by sni-CDGSs working in $=k$-comp.-mode of derivation is denoted by $\mathcal{L}(\text{sni-CD}, \text{CF}, =k\text{-comp.})$ if rewriting sequentially and by $\mathcal{L}(\text{sni-CD}, \text{parCF}, =k\text{-comp.})$ if rewriting in parallel. Likewise, the family of languages generated by CDGSs working in $=k$-comp.dni-mode of derivation is denoted by $\mathcal{L}(\text{CD}, \text{CF}, =k\text{-comp.dni})$ if rewriting sequentially, and by $\mathcal{L}(\text{CD}, \text{parCF}, =k\text{-comp.dni})$ if rewriting in parallel.

It is not difficult to see in Example 1 that G is an sni-CDGS and (thus) working in a dynamically nonincreasing competence-based fashion.

5 The Generative Power of Nonincreasing Competence

CDGSs working in $=1$-comp.-mode and rewriting in parallel, independently of whether they are sni-CDGSs or work in a dynamically nonincreasing competence-based fashion, characterize the class of ET0L languages.

Theorem 1
$\mathcal{L}(\text{sni-CD}, \text{parCF}, =1\text{-comp.}) = \mathcal{L}(\text{CD}, \text{parCF}, =1\text{-comp.dni}) = \mathcal{L}(\text{ET0L})$.

Proof. The statement is proved once we prove the three inclusions $\mathcal{L}(\text{ET0L}) \subseteq \mathcal{L}(\text{sni-CD}, \text{parCF}, =1\text{-comp.}) \subseteq \mathcal{L}(\text{CD}, \text{parCF}, =1\text{-comp.dni}) \subseteq \mathcal{L}(\text{ET0L})$. As the second inclusion is trivial, we prove the first and the last.

[$\mathcal{L}(\text{ET0L}) \subseteq \mathcal{L}(\text{sni-CD}, \text{parCF}, =1\text{-comp.})$] Let $G = (\Sigma, H, \omega, \Delta)$ be an ET0L system with $H = \{h_1, \ldots, h_k\}$. Without loss of generality, we assume ω equals $S \in \Sigma$. We define a CDGS G' with the disjoint union $N' = \{\, X^{(i)} \mid X \in \Sigma,\ 0 \leq i \leq k+2 \,\} \cup \{F\}$ as nonterminals, set of terminals Δ disjoint from N', axiom $S^{(0)}$ and the components defined below.

Simulating the application of one table of H consists of 3 phases: simulating the selected table by encoding the sentential form, applying the table, and decoding the sentential form. The coding is done by the components

$$P_{\text{encode},i,X} = \{X^{(0)} \to X^{(i)}\} \cup \{\, Y^{(\ell)} \to F \mid Y \in \Sigma,\ \ell \in \{1, \ldots, k+2\} \setminus \{i\} \,\},$$

for all $1 \leq i \leq k+2$ and $X \in \Sigma$. An application of $h_i \in H$ is simulated by

$$P_{\text{apply},i,X} = \{\, X^{(i)} \to w^{(k+1)} \mid X \to w \in h_i \,\} \cup \{\, Y^{(\ell)} \to F \mid Y \in \Sigma,\ \ell \in \{0, \ldots, k+2\} \setminus \{i, k+1\} \,\},$$

for all $1 \leq i \leq k$ and $X \in \Sigma$, where for $w = x_1 \cdots x_t$, with $x_j \in \Sigma$ and $1 \leq j \leq t$, $w^{(k+1)} = x_1^{(k+1)} \cdots x_t^{(k+1)}$, with $x_j^{(k+1)} \in N'$. Decoding is simulated by

$$P_{\text{decode},X} = \{X^{(k+1)} \to X^{(0)}\} \cup \{Y^{(\ell)} \to F \mid Y \in \Sigma,\ \ell \in \{1,\ldots,k+2\} \setminus \{k+1\}\},$$

for all $X \in \Sigma$. After encoding the sentential form to code $(k+2)$ with components $P_{\text{encode},k+2,X}$, the derivation can be finished, for all $X \in \Sigma$, by

$$P_{\text{finish},X} = \{\, X^{(k+2)} \to X \mid X \in \Delta \,\} \cup \{\, X^{(k+2)} \to F \mid X \in \Sigma \setminus \Delta \,\} \cup \{\, Y^{(\ell)} \to F \mid Y \in \Sigma,\ \ell \in \{0,\ldots,k+1\} \,\},$$

Now G' can simulate G. Take a sentential form $x = x_1 \cdots x_t$, with $x_j \in \Sigma$ and $1 \le j \le t$. Assume that applying $h_i \in H$ of G leads to $y = y_1 \cdots y_r$, with $y_j \in \Sigma$ and $1 \le j \le r$. Starting from $x^{(0)} = x_1^{(0)} \cdots x_t^{(0)}$, with $x_j^{(0)} \in N'$ and $1 \le j \le t$, G' derives $y^{(0)} = y_1^{(0)} \cdots y_r^{(0)}$, with $y_j^{(0)} \in N'$ and $1 \le j \le r$, as follows.

First components $P_{\text{encode},i,X}$ rewrite each $X^{(0)} \in \text{alph}_{N'}(x^{(0)})$ to $X^{(i)}$. Their 1-competence ensures using the same i for each symbol. Next components $P_{\text{apply},i,X}$ apply a rule of h_i for each symbol, producing $y^{(k+1)} = y_1^{(k+1)} \cdots y_r^{(k+1)}$, with y as above. The components are 1-competent on the sentential form, so each symbol must be rewritten before applying the decoding components. The latter components rewrite each $X^{(k+1)} \in \text{alph}_{N'}(y^{(k+1)})$ to $X^{(0)}$ and another table may be simulated. To obtain a terminal word the components must be applied in this order.

From the description of G' we see that if $x_1^{(0)} \cdots x_t^{(0)}$, with $x_j^{(0)} \in N'$ and $1 \le j \le t$, is a sentential form of G', then so is $x_1 \cdots x_t$, with $x_j \in \Sigma$ and $1 \le j \le t$. Hence, starting from axiom $S^{(0)}$ and considering that using $P_{\text{encode},k+2,X}$ and $P_{\text{finish},X}$, with $X \in \Sigma$, may produce $x_1 \cdots x_t$, with $x_j \in \Sigma$ and $1 \le j \le t$, from the sentential form $x_1^{(0)} \cdots x_t^{(0)}$, with $x_j^{(0)} \in N'$ and $1 \le j \le t$, then we see that G' correctly simulates G.

[$\mathcal{L}(\text{CD}, \text{parCF}, =1\text{-comp.dni}) \subseteq \mathcal{L}(\text{ET0L})$] Let $G = (N, T, \alpha, P_1, \ldots, P_n)$ be a CDGS working in $=1$-comp.dni-mode, rewriting in parallel. To simulate G, we construct an ET0L system $G' = (\Sigma, H, S, T)$, with $\Sigma = M \cup N \cup T \cup \{S, F\}$, with $M = \{\, p_{i,A} \mid A \in \text{dom}(P_i),\ 1 \le i \le n \,\}$ such that S, F and all symbols in M (called labels from now on) are new symbols not appearing in $N \cup T$, and the set H of tables defined below. By definition, all unspecified symbols in a table are rewritten identically.

Assume a derivation in G starts by applying P_i, for some $1 \le i \le n$, to α. As G works in $=1$-comp.dni-mode, only one nonterminal from $\text{dom}(P_i)$ occurs in α. The simulation starts by applying $h_{\text{start}} = \{\, S \to p\alpha \mid p \in M \,\}$. This results in a sentential form $p_{i,A}\alpha$, for some $A \in \text{dom}(P_i)$ and $1 \le i \le n$, in which $p_{i,A}$ is a label from M. Symbol $p_{i,A}$ indicates that P_i (1-competent due to the presence of A) is simulated. As G works in dynamically nonincreasing competence-based fashion, P_i only stops rewriting α if all occurrences of A in α (possibly introduced by successive applications) are replaced. However, P_i may contain more productions with left-hand side A and, as G works in dynamically nonincreasing competence-based fashion, P_i may rewrite occurrences of certain $B \in \text{dom}(P_i)$ introduced meanwhile. Note that two distinct nonterminals from $\text{dom}(P_i)$ may never occur in a sentential form. To simulate the components of G, we construct the tables

$$h_{\text{apply},i,A} = \{\, A \to z \mid A \to z \in P_i \,\} \cup \{\, p_{i,A} \to p_{i,A}, p_{i,A} \to p_{i,B}, \\ B \to F \mid B \in \mathrm{dom}(P_i), B \neq A \,\} \cup \{\, p \to F \mid p \in M \setminus \{p_{i,A}\} \,\},$$

for all $A \in \mathrm{dom}(P_i)$ and $1 \leq i \leq n$. Table $h_{\text{apply},i,A}$, for some $A \in \mathrm{dom}(P_i)$ and $1 \leq i \leq n$, is applied to the sentential form $p_{i,A}\alpha$ until it has rewritten all occurrences of A in α (also those introduced by successive applications). As an F (which can never be rewritten) is introduced in case A is not the only nonterminal from $\mathrm{dom}(P_i)$ appearing in α, in any successful derivation of G' this table is only applied if P_i could be applied to α in a derivation of G. Moreover, the productions replacing all labels $p \neq p_{i,A}$ by F guarantee that this table is only applied if label $p_{i,A}$ is present in the sentential form. Note that this table can rewrite label $p_{i,A}$ by $p_{i,B}$, for some $B \in \mathrm{dom}(P_i)$, to faithfully mimic the fact that G works in dynamically nonincreasing competence-based fashion. It remains to guarantee that this table is indeed applied to sentential form $p_{i,A}\alpha$ until it has rewritten all occurrences of A in α (possibly introduced earlier). Therefore, we construct the tables

$$h_{\text{test},i,A} = \{\, p_{i,A} \to p \mid p_{i,A}, p \in M,\ p \neq p_{i,A} \,\} \cup \{\, B \to F \mid B \in \mathrm{dom}(P_i) \,\},$$

for all $A \in \mathrm{dom}(P_i)$ and $1 \leq i \leq n$. By applying one of these tables, G' can start simulating the application of another component from G (indicated by label p). However, clearly no successful derivation exists in case not all occurrences of A in the sentential form are replaced. The simulation described so far is repeated for all components that are applied in a successful derivation in G. To successfully finish the simulation, we construct table

$$h_{\text{finish}} = \{\, p \to \lambda \mid p \in M \,\} \cup \{\, A \to F \mid A \in \Sigma \setminus (M \cup T) \,\}.$$

Clearly no successful derivation exists if this table is applied before the only nonterminal symbol remaining in the sentential form is a label from M.

The situation is quite different for CDGSs working in $=1\text{-comp.}$-mode and rewriting sequentially: the two definitions of nonincreasing competence lead to different language classes. While sni-CDGSs working in $=1\text{-comp.}$-mode and rewriting sequentially still characterize the class of ET0L languages, CDGSs working in $=1\text{-comp.dni}$-mode and rewriting sequentially characterize the strictly more powerful class of languages generated by context-free forbidding random context grammars. To prove this, we need a normal-form result for context-free forbidding random context grammars [14]; for the reader's convenience we include the proof of the next lemma.

Lemma 1. *Any context-free forbidding random context language L can be generated by a context-free forbidding random context grammar (N, T, P, S) whose productions $(A \to z, \emptyset, R) \in P$ satisfy $z \notin (N \cup T)^* R (N \cup T)^*$,* i.e.*, the forbidding contexts of productions do not appear in their right-hand sides.*

Proof. Let $G = (N, T, P, S)$ be a context-free forbidding random context grammar so that $L(G) = L$. We construct a context-free forbidding random context grammar

$G' = (N' \cup N'', T, P', S)$ in normal form, with $N' = \{ A' \mid A \in N \}$, $N'' = \{ A'' \mid A \in N \}$ (N, N', and N'' are pairwise disjoint), and

$$P' = \{ (A \to A', \emptyset, N''), (A' \to g(z), \emptyset, R \cup N') \mid (A \to z, \emptyset, R) \in P \} \cup \{ (A'' \to A, \emptyset, N') \mid A \in N \},$$

with the homomorphism $g : (N \cup T)^* \to (N'' \cup T)^*$ defined by $g(a) = a''$ if $a \in N$ and $g(a) = a$ if $a \in T$.

Consider a production $p = (A \to z, \emptyset, R)$ and a sentential form with occurrences of A and no nonterminal from R (otherwise no successful derivation exists for this p). The application of p in G is simulated as follows in G'. First an A is primed (the derivation will block if another nonterminal is primed), then A' is replaced by $g(z)$ (blocking the priming of nonterminals). To continue successfully, all (if any) doubly primed nonterminals are unprimed. This results in a sentential form without (doubly) primed nonterminals, and another production from P can be simulated.

We are now ready for the next theorem.

Theorem 2
$\mathcal{L}(\text{ET0L}) = \mathcal{L}(\text{sni-CD}, \text{CF}, =1\text{-comp.}) \subset \mathcal{L}(\text{fRC}, \text{CF}) = \mathcal{L}(\text{CD}, \text{CF}, =1\text{-comp.dni})$.

Proof. Recall that $\mathcal{L}(\text{ET0L}) \subset \mathcal{L}(\text{fRC}, \text{CF})$. We first prove the two inclusions that together prove the first equality in the statement of this theorem, in both cases using the same ideas that we used to prove Theorem 1.

[$\mathcal{L}(\text{ET0L}) \subseteq \mathcal{L}(\text{sni-CD}, \text{CF}, =1\text{-comp.})$] We simulate an ET0L system $G = (\Sigma, H, \omega, \Delta)$ with the CDGS G' that we defined in the proof of Theorem 1 to prove the inclusion $\mathcal{L}(\text{ET0L}) \subseteq \mathcal{L}(\text{sni-CD}, \text{parCF}, =1\text{-comp.})$. This CDGS simulates the application of some table $h_i \in H$ in three phases: simulating h by encoding the sentential form, applying h, and decoding the sentential form. The reader may verify that G' is indeed an sni-CDGS and that the simulation of G still works if G' rewrites sequentially.

[$\mathcal{L}(\text{sni-CD}, \text{CF}, =1\text{-comp.}) \subseteq \mathcal{L}(\text{ET0L})$] Let $G = (N, T, \alpha, P_1, \ldots, P_n)$ be an sni-CDGS working in $=1$-comp.-mode and rewriting sequentially. To simulate G, we construct an ET0L system $G' = (\Sigma, H, S, T)$, where $\Sigma = M \cup N \cup T \cup \{S, F\}$, with $M = \{ p_{i,A} \mid A \in \text{dom}(P_i),\ 1 \le i \le n \}$ and S, F and all symbols in M (called labels from now on) are new symbols not appearing in $N \cup T$, and H contains the tables defined below. As usual, all unspecified symbols in a table are rewritten identically.

Let G start a derivation from α by applying P_i, for some $1 \le i \le n$. Working in $=1$-comp.-mode, α contains only one occurrence of $A \in \text{dom}(P_i)$. The simulation thus starts by applying $h^{\text{start}} = \{ S \to p\alpha \mid p \in M \}$. This results in a sentential form $p_{i,A}\alpha$, for some $A \in \text{dom}(P_i)$ and $1 \le i \le n$, in which $p_{i,A}$ is a label from M indicating the simulation of P_i (1-competent due to the presence of A). As G is an sni-CDGS, component P_i only stops rewriting when all occurrences of A in α (also those introduced by successive applications of P_i along the way) are replaced. Component P_i may however contain more productions with left-hand side A. We thus construct tables

$$h_{i,A}^{\text{apply}} = \{(A \to z) \in P_i\} \cup \{B \to F,\ p \to F \mid B \neq A,\ B \in \text{dom}(P_i),\ p \in M \setminus \{p_{i,A}\}\}$$

for all $A \in \mathrm{dom}(P_i)$ and $1 \leq i \leq n$. The idea is to apply $h^{\text{apply}}_{i,A}$, for some $A \in \mathrm{dom}(P_i)$ and $1 \leq i \leq n$, to the sentential form $p_{i,A}\alpha$ until all occurrences of A in α are rewritten (also those introduced by its successive applications). Symbol F (that cannot be rewritten) is introduced in case A is not the only nonterminal from $\mathrm{dom}(P_i)$ in α, guaranteeing that in a successful derivation of G' this table is only applied if P_i could be applied to α according to G. The productions replacing all labels $p \neq p_{i,A}$ by F guarantee that this table is only applied if label $p_{i,A}$ is present in the sentential form. It remains to guarantee that this table is indeed applied to the sentential form $p_{i,A}\alpha$ until it has rewritten all occurrences of A in α (including those introduced by earlier applications). To this aim, we construct the tables

$$h^{\text{test}}_{i,A} = \{\, p_{i,A} \to p \mid p_{i,A}, p \in M,\ p \neq p_{i,A} \,\} \cup \{\, B \to F \mid B \in \mathrm{dom}(P_i) \,\},$$

for all $A \in \mathrm{dom}(P_i)$ and $1 \leq i \leq n$. By applying one of these tables, G' can start simulating the application of another component from G (indicated by label p). However, clearly no successful derivation exists if not yet all occurrences of A in the sentential form have been replaced. The above simulation is repeated for all components applied in a successful derivation in G. To successfully finish simulating a derivation in G we apply table

$$h^{\text{finish}} = \{\, p \to \lambda \mid p \in M \,\} \cup \{\, A \to F \mid A \in \Sigma \setminus (M \cup T) \,\}.$$

No successful derivation exists if h^{finish} is applied before the only nonterminal symbol remaining is a label from M. This proves the inclusion.

We now prove the second equality in the statement of this theorem.

[$\mathcal{L}(\text{fRC}, \text{CF}) \subseteq \mathcal{L}(\text{CD}, \text{CF}, =1\text{-comp.dni})$] Let $G = (N, T, P, S)$ be a forbidding random context grammar such that every random context rule $(A \to z, \emptyset, R)$ in P satisfies $z \notin (N \cup T)^* R (N \cup T)^*$. Note that $A \in R$ is possible. Without loss of generality, let each random context rule have a unique label p (the set of all labels is Λ). To simulate G we construct G', with disjoint union of sets of nonterminals $N' = N \cup \{F\} \cup \{A' \mid A \in N\} \cup \{A_p \mid A \in N, p \in \Lambda\}$, terminals T, axiom S and the components defined below.

A simulation of the application of a random context rule $(A \to z, \emptyset, R)$ has two phases: Simulating it by marking all nonterminals A in the sentential form and applying it at appropriate places. The marking is as follows:

(1) In case $A \notin R$ we introduce the component $P_{\text{mark},p} = \{A \to A_p\} \cup \{\, B \to F \mid B \in R \,\} \cup \{\, B_q \to F \mid B \in N,\ q \in \Lambda,\ B_q \neq A_p \,\}$.
(2) In case $A \in R$ we introduce the component $P_{\text{mark},p} = \{A \to A',\ A' \to A_p\} \cup \{\, B \to F \mid B \in R \,\} \cup \{\, B_q \to F \mid B \in N,\ q \in \Lambda,\ B_q \neq A_p \,\}$.

Finally, after the marking, the derivation may continue by the component

$$P_{\text{apply},p} = \{A_p \to A, A_p \to z\} \cup \{\, B_q \to F \mid B \in N,\ q \in \Lambda \,\}.$$

This completes the description of the CDGS G'. Next we explain how G' can be used to simulate the forbidding random context grammar G.

Consider applying the random context rule $(A \to z, \emptyset, R)$ labelled p to a sentential form α with at least one occurrence of A and no symbol from R. Recall $z \notin (N \cup T)^* R (N \cup T)^*$. The sentential form α is thus

$$\alpha = \alpha_1 A \alpha_2 A \alpha_3 \dots \alpha_{n-1} A \alpha_n,$$

where $n \geq 2$ and $\alpha_i \in ((N \setminus \{A\}) \cup T)^*$, for $1 \leq i \leq n$. Then we consider two cases, according to whether $A \notin R$ or $A \in R$. In the former case the random context rule can be applied several times to α, obtaining a sentential form

$$\alpha' = \alpha_1 \beta_1 \alpha_2 \beta_2 \alpha_3 \dots \alpha_{n-1} \beta_{n-1} \alpha_n,$$

where β_i, for $1 \leq i \leq n$, is either A or z. These derivations can be mimicked by applying the 1-competent component $P_{\text{mark},p}$, which verifies that no symbol from R is present and replaces every A by A_p, followed by an application of $P_{\text{apply},p}$, which is also 1-competent. During this application, all A_p are replaced by either A or z — note that if all A_p are replaced by A, we get the original sentential form α. This shows G' can generate the sentential form α'. Note how shortcuts are circumvented by rules introducing F. Finally, in the latter case, *i.e.*, if $A \in R$, we observe that whenever random context rule p can be applied to α (assumed to contain no symbol from R) it must satisfy $\alpha = \alpha_1 A \alpha_2$, where $\alpha_i \in ((N \setminus \{A\}) \cup T)^*$, for $1 \leq i \leq 2$, leading to $\alpha' = \alpha_1 z \alpha_2$. This derivation can again be mimicked by $P_{\text{mark},p}$ followed by $P_{\text{apply},p}$, where the dynamically nonincreasing feature comes into play. Observe that if α contains at least two occurrences of A and no symbol from R, then $P_{\text{mark},p}$ is 1-competent at the start. Hence rewriting one A by A' results in an increase in competence. Since this is not allowed if the CDGS G' works in $=1\text{-comp.dni}$-mode, the derivation blocks. Thus, the only way to successfully apply $P_{\text{mark},p}$ on α requires $\alpha = \alpha_1 A \alpha_2$ with $\alpha_i \in ((N \setminus \{A\}) \cup T)^*$, for $1 \leq i \leq 2$. Then it is easy to see that the sentential form $\alpha' = \alpha_1 z \alpha_2$ can be generated by G', proving the inclusion.

$[\mathcal{L}(\text{CD}, \text{CF}, =1\text{-comp.dni}) \subseteq \mathcal{L}(\text{fRC}, \text{CF})]$ Let $G = (N, T, \alpha, P_1, \dots, P_n)$ be a CDGS working in $=1\text{-comp.dni}$-mode and rewriting sequentially. To simulate G we construct a context-free forbidding random context grammar $G' = (N', T, P, S')$ with $N' = N \cup \{S'\} \cup \{ p_i \mid 1 \leq i \leq n \}$ and S' and all p_i, for $1 \leq i \leq n$, symbols not appearing in N.

Assume G starts a derivation by applying P_i, for some $1 \leq i \leq n$, to α. As G works in $=1\text{-comp.dni}$-mode, only one nonterminal from $\text{dom}(P_i)$ occurs in α. The simulation starts by applying a random context rule $(S' \to p_i \alpha, \emptyset, \emptyset)$, for $1 \leq i \leq n$, leading to sentential form $p_i \alpha$ showing P_i is simulated next. As G works in $=1\text{-comp.dni}$-mode, P_i only stops rewriting if all occurrences of some nonterminal A in α are replaced (also those introduced by successive applications). However, as P_i may have several productions with left-hand side A it might also rewrite occurrences of certain $B \in \text{dom}(P_i)$. No two distinct nonterminals from $\text{dom}(P_i)$ may ever occur both in the sentential form. To simulate P_i, for $1 \leq i \leq n$, we construct the random context rules

$$(A \to z, \emptyset, \{ p_j \mid 1 \leq j \leq n,\ i \neq j \} \cup (\text{dom}(P_i) \setminus \{A\})),$$

for every $A \to z \in P_i$. The random context rules can only be applied to a sentential form $p_i \alpha$ and only one nonterminal from $\text{dom}(P_i)$ can occur. Rewriting continues until

all occurrences of nonterminal A from $\mathrm{dom}(P_i)$ in α are rewritten (including those introduced by successive applications). Then another component P_j, for $1 \leq j \leq n$, of G may continue the derivation. We therefore introduce for every i and j with $1 \leq i, j \leq n$ and $i \neq j$, the random context rules $(p_i \to p_j, \emptyset, \mathrm{dom}(P_i))$, ensuring that the label is changed from p_i to p_j iff no nonterminal from $\mathrm{dom}(P_i)$ occurs. Hence, component P_i has become zero-competent on the current sentential form.

The simulation described above is now repeated for all components that are applied by G in a successful derivation. Finally, to successfully finish simulating a derivation by G, we apply the rule $(p_i \to \lambda, \emptyset, N)$, for $1 \leq i \leq n$. It is clear that none of these rules can be applied as long as one nonterminal is present. This proves the inclusion.

We now turn our attention to CDGSs working in $=k$-comp.-mode, for some $k \geq 2$, and rewriting in parallel. Independently of whether the CDGSs are sni-CDGSs or work in a dynamically nonincreasing competence-based fashion, for $k \geq 2$ the family of languages generated by context-free recurrent programmed grammars with appearance checking is characterized.

Theorem 3
$\mathcal{L}(\text{sni-CD}, \text{parCF}, =k\text{-comp.}) = \mathcal{L}(\text{CD}, \text{parCF}, =k\text{-comp.dni}) = \mathcal{L}(\text{rPR}, \text{CF}, \text{ac})$, *for* $k \geq 2$.

Proof. Once we show $\mathcal{L}(\text{rPR}, \text{CF}, \text{ac}) \subseteq \mathcal{L}(\text{sni-CD}, \text{parCF}, =k\text{-comp.}) \subseteq \mathcal{L}(\text{CD}, \text{parCF}, =k\text{-comp.dni}) \subseteq \mathcal{L}(\text{rPR}, \text{CF}, \text{ac})$, for $k \geq 2$, the statement is proved. Since the second inclusion is trivial, we only prove the other two.

[$\mathcal{L}(\text{rPR}, \text{CF}, \text{ac}) \subseteq \mathcal{L}(\text{CD}, \text{parCF}, =k\text{-comp.}, \text{sni})$] Let $k = 2$. Generalizing the proof to the case $k > 2$ is straightforward and left to the reader. Let $G = (N, T, P, S, \Lambda, \sigma, \phi)$ be a recurrent programmed grammar with appearance checking. We assume productions of the form $p : (A \to z, \sigma(p), \phi(p))$, with success field $\sigma(p)$ and failure field $\phi(p)$. As G is a recurrent programmed grammar with appearance checking, each production $p \in \Lambda$ is such that $p \in \sigma(p)$, and either $\phi(p) = \emptyset$ or $\phi(p) = \sigma(p)$. Without loss of generality, assume only p_1 is able to rewrite S. To simulate G, we construct an sni-CDGS G' with disjoint union $N' = N \cup \{ p, p' \mid p \in \Lambda \} \cup \{F, Z\}$ of nonterminals, terminals T disjoint from N', axiom Sp_1Z and the components defined next. For each production $p : (A \to z, \sigma(p), \phi(p))$ such that $q \in \sigma(p)$ with $q \neq p$, we construct the components

$$
\begin{aligned}
P_{p,q,\text{present}} &= \{A \to zq, A \to A, p \to p'\} \cup \{ r \to F \mid r \in \Lambda \setminus \{q, p\} \}, \\
P_{p,q,\text{clean}} &= \{p \to \lambda, q \to q\} \text{ and} \\
P_{p,q,\text{clean}'} &= \{p' \to \lambda, q \to q\},
\end{aligned}
$$

and in case $\phi(p) \neq \emptyset$, with $\phi(p) = \sigma(p)$, the additional components

$$P_{p,q,\text{absent}} = \{A \to F,\ p \to q,\ Z \to Z\}.$$

Note that at any time, two types of markers are present in the sentential form. A general marker Z, sometimes used to guarantee that components are 2-competent, and a specific marker p (or its primed version), which is the label of the context-free production being simulated.

First assume $p : (A \to z, \sigma(p), \phi(p))$ is such that $\phi(p) = \emptyset$. Also assume a sentential form with one or more occurences of A. Then each $P_{p,q,\text{present}}$, with $q \in \sigma(p)$, with $q \neq p$, is 2-competent. Note that the moment $P_{p,q,\text{present}}$ replaces p by its primed version, it is no longer 2-competent. No successful derivation exists if $p \to p'$ is applied before $A \to zq$ introduced q. After one occurrence of A is replaced by zq, then either another occurrence of A is replaced by zq or p is primed. In the latter case, only $P_{p,q,\text{clean}'}$ is 2-competent and its application deletes p'. In the former case, both $P_{p,q,\text{clean}}$ and $P_{q,p,\text{clean}} = \{q \to \lambda, p \to p\}$ are 2-competent. Applying $P_{p,q,\text{clean}}$ leads to deleting p, while $P_{q,p,\text{clean}}$ leads to deleting q. In both cases, we thus obtain a sentential form ready to simulate a new context-free production, labelled with either q or p. Note that productions $r \to F$, for all $r \in \Lambda \setminus \{q, p\}$, in $P_{p,q,\text{present}}$ guarantee that only occurrences of q or p occur in a sentential form before the production q or p, respectively, is simulated.

Note that in G, the application of $A \to z$ can be repeated as long as there are occurrences of A in the sentential form, since by definition $p \in \sigma(p)$. The fact that G' must be an sni-CDGS, however, forces us to require that $q \neq p$ in $P_{p,q,\text{present}}$, which thus indicates that the context-free production q from the success field of p must be applied next. Nevertheless, we have seen that several occurrences of A can be replaced, thus simulating $p \in \sigma(p)$. Hence q indicates that *eventually* this context-free production from the success field of p must be applied, but not necessarily *immediately*.

Now assume $p : (A \to z, \sigma(p), \phi(p))$ is such that $\sigma(p) = \phi(p) \neq \emptyset$, and a sentential form without any A. If it contains label p, then for each $q \in \phi(p)$ we have a 2-competent $P_{p,q,\text{absent}}$. After a $P_{p,q,\text{absent}}$ replaced p by q, it is no longer 2-competent and we get a sentential form ready to simulate the production labelled q, thus simulating the "failed" application of p.

Finally, a derivation can finish only when no more nonterminals other than Z or p, for some $p \in \Lambda$, appear in the sentential form, in which case

$$P_{\text{finish}} = \{\, p \to \lambda \mid p \in \Lambda \,\} \cup \{Z \to \lambda\}$$

is 2-competent and it removes all remaining nonterminals from the sentential form until a terminal word is obtained. Note that an earlier application of component P_{finish} (*i.e.*, when the sentential form still contains nonterminals other than those from $\{\, p \mid p \in \Lambda \,\} \cup \{Z\}$) blocks a successful derivation. This is because such an application would remove either $p \in \Lambda$ or Z, but not both. If $p \in \Lambda$ is removed too early, then the only possible 2-competent component is P_{absent} but in that case it would introduce an F and block a successful derivation. If Z is removed too early, then a successful derivation is blocked due to the fact that P_{finish} can no longer become 2-competent and the sentential form thus always contains some $p \in \Lambda$.

We described the sni-CDGS G'. The reader can verify that if productions are applied in a different order, no successful derivation exists. Working in $=2$-comp.-mode and rewriting in parallel, G' simulates the recurrent programmed grammar G with appearance checking and generates $L(G)$.

[$\mathcal{L}(\text{CD}, \text{parCF}, =k\text{-comp.dni}) \subseteq \mathcal{L}(\text{rPR}, \text{CF}, \text{ac})$] We consider a CDGS $G = (N, T, \alpha, P_1, \ldots, P_n)$ working in $=k$-comp.dni-mode and rewriting in parallel. To simulate G, we construct the recurrent programmed grammar $G' = (N', T, P, S, \Lambda, \sigma, \phi)$

with appearance checking. In G', the nonterminals $N' = N \cup \{ A' \mid A \in N \} \cup \{S, F\}$ are such that S and F are new symbols not appearing in $N \cup T$, while P, Λ, σ and ϕ are as defined below.

For all P_i, with $1 \leq i \leq n$, and all $X \subseteq \mathrm{dom}(P_i)$, with $|X| = k$, we construct the set of productions $P_{i,X}^{\text{present}} \cup P_{i,X}^{\text{absent}} \cup P_{i,X}^{\text{apply}} \cup P_{i,X}^{\text{test}} \cup P_{i,X}^{\text{decode}} \cup P_{i,X}^{\text{test}'}$ as below. For all $Y \subseteq \mathrm{dom}(P_i)$, with $|Y| < k$, we construct as follows the set of productions $P_{i,Y}^{\text{check}} \cup P_{i,Y}^{\text{check}'}$. Those in $P_{i,X}^{\text{present}}$ test the presence of the symbols in X in the sentential form, while those in $P_{i,X}^{\text{absent}}$ test the absence of all symbols in $\mathrm{dom}(P_i) \setminus X$. By alternating productions from the other components, some productions from P_i with left-hand sides in X are applied (using a coding to prime nonterminals) and in between we test if P_i is still k-competent. In detail, productions in $P_{i,Y}^{\text{check}}$ test the presence of the symbols in Y in the sentential form, while those in $P_{i,Y}^{\text{check}'}$ test the absence of all symbols in $\mathrm{dom}(P_i) \setminus Y$. For notational convenience, we identify rules and their labels, and rule sets and sets of labels of rules.

Let $X = \{A_1, \ldots, A_k\}$ and let $\mathrm{dom}(P_i) \setminus X = \{C_1, \ldots, C_m\}$. Then $P_{i,X}^{\text{present}}$ contains the following k productions, with $1 \leq j \leq k-1$:

$$r_{i,X,A_j}^{\text{present}} : (A_j \to A_j, \{r_{i,X,A_j}^{\text{present}}, r_{i,X,A_{j+1}}^{\text{present}}\}, \emptyset) \text{ and}$$
$$r_{i,X,A_k}^{\text{present}} : (A_k \to A_k, \{r_{i,X,A_k}^{\text{present}}\} \cup \{r_{i,X,C_1}^{\text{absent}}\}, \emptyset).$$

In case of a successful derivation, these productions check the presence of all symbols A_j, with $1 \leq j \leq k$, after which eventually production $r_{i,X,C_1}^{\text{absent}}$ is to be applied next. This production is part of $P_{i,X}^{\text{absent}}$, which contains the following $m = |\mathrm{dom}(P_i) \setminus X|$ productions, with $1 \leq \ell \leq m-1$:

$$r_{i,X,C_\ell}^{\text{absent}} : (C_\ell \to F, \{r_{i,X,C_\ell}^{\text{absent}}, r_{i,X,C_{\ell+1}}^{\text{absent}}\}, \{r_{i,X,C_\ell}^{\text{absent}}, r_{i,X,C_{\ell+1}}^{\text{absent}}\}) \text{ and}$$
$$r_{i,X,C_m}^{\text{absent}} : (C_m \to F, \{r_{i,X,C_m}^{\text{absent}}\} \cup P_{i,X}^{\text{apply}}, \{r_{i,X,C_m}^{\text{absent}}\} \cup P_{i,X}^{\text{apply}}).$$

Let F be a failure symbol (that cannot be rewritten): no successful derivation exists if F is introduced in the sentential form. In case of a successful derivation, we thus checked the absence of all symbols from $\mathrm{dom}(P_i) \setminus X$ and eventually one of the productions from $P_{i,X}^{\text{apply}}$ is to be applied next.

Productions in $P_{i,X}^{\text{apply}}$ simulate the applications of those productions in P_i that have A_j, with $1 \leq j \leq k$, as their left-hand side (recall that $X = \{A_1, \ldots, A_k\}$). However, as G works in $=k$-comp.dni-mode, we need to prime nonterminals to distinguish those occurrences present in the sentential form before rewriting the A_j from those introduced by rewriting the A_j. The unpriming is later done by productions from $P_{i,X}^{\text{decode}}$. As G rewrites in parallel, we also need to test that all occurrences of A_j are primed, and eventually unprimed. This is done by productions from $P_{i,X}^{\text{test}}$ and $P_{i,X}^{\text{test}'}$, respectively. Finally, after each application of a production that rewrites A_j, we need to test if P_i is still k-competent. Productions from $P_{i,X}^{\text{check}}$ and $P_{i,X}^{\text{check}'}$ do so in the way described above. Note that we use the fact that directly after the first production from $P_{i,X}^{\text{apply}}$ with A_j as its left-hand side has been applied, we know the symbols from $X \setminus \{A_j\}$ are still present in the sentential form — as this was tested earlier by $P_{i,X}^{\text{present}}$.

The set $P_{i,X}^{\text{apply}}$ is the union $\bigcup_{A_j \in X} P_{i,X,A_j}^{\text{apply}}$ of the following sets constructed for each symbol in X, with $1 \leq j \leq k$:

$$P_{i,X,A_j}^{\text{apply}} = \{(A_j \to z', P_{i,X}^{\text{apply}} \cup \{r_{i,X,A_1}^{\text{test}}\}, P_{i,X}^{\text{apply}} \cup \{r_{i,X,A_1}^{\text{test}}\}) \mid (A_j \to z) \in P_i\},$$

where z' is obtained from z by priming *all* (and only) its nonterminals.

Rule set $P_{i,X}^{\text{test}}$ contains the following k productions, with $1 \leq j \leq k-1$:

$$r_{i,X,A_j}^{\text{test}} : (A_j \to F, \{r_{i,X,A_j}^{\text{test}}, r_{i,X,A_{j+1}}^{\text{test}}\}, \{r_{i,X,A_j}^{\text{test}}, r_{i,X,A_{j+1}}^{\text{test}}\}) \text{ and}$$
$$r_{i,X,A_k}^{\text{test}} : (A_k \to F, \{r_{i,X,A_k}^{\text{test}}\} \cup P_{i,X}^{\text{decode}}, \{r_{i,X,A_k}^{\text{test}}\} \cup P_{i,X}^{\text{decode}}).$$

Let $P_{i,X}^{\text{decode}}$, used for unpriming the sentential form, be the set of rules

$$P_{i,X}^{\text{decode}} = \{\, (Y' \to Y, P_{i,X}^{\text{decode}} \cup \{r_{i,X,A_1'}^{\text{test}'}\}, P_{i,X}^{\text{decode}} \cup \{r_{i,X,A_1'}^{\text{test}'}\}) \mid Y \in N' \,\}.$$

Let $A_1', \ldots, A_p'$ be primed versions of all nonterminals in N. Then $P_{i,X}^{\text{test}'}$ contains the following $|N|$ productions, with $1 \leq j \leq p-1$:

$$r_{i,X,A_j}^{\text{test}'} : (A_j' \to F, \{r_{i,X,A_j}^{\text{test}'}, r_{i,X,A_{j+1}}^{\text{test}'}\}, \{r_{i,X,A_j}^{\text{test}'}, r_{i,X,A_{j+1}}^{\text{test}'}\}) \text{ and}$$
$$\begin{aligned} r_{i,X,A_p}^{\text{test}'} : (A_p' \to F, &\{r_{i,X,A_p}^{\text{test}'}\} \cup \{\, r_{i,X',B_1}^{\text{present}} \mid X' = \{B_1, \ldots, B_k\} \subseteq \text{dom}(P_i) \,\} \\ &\cup \{\, r_{i,Y,B_1}^{\text{check}} \mid Y = \{B_1, \ldots, B_j\} \subseteq \text{dom}(P_i),\ |Y| < k \,\}, \\ &\{r_{i,X,A_p}^{\text{test}'}\} \cup \{\, r_{i,X',B_1}^{\text{present}} \mid X' = \{B_1, \ldots, B_k\} \subseteq \text{dom}(P_i) \,\} \cup \\ &\{\, r_{i,Y,B_1}^{\text{check}} \mid Y = \{B_1, \ldots, B_j\} \subseteq \text{dom}(P_i),\ |Y| < k \,\}). \end{aligned}$$

In case of a sucessful derivation, it is possible to continue the derivation with component i using some set $X' \subset \text{dom}(P_i)$ of nonterminals of size k for the k-competent derivation, or to drop the derivation of the ith component. In the latter case, we must verify that the level of competence has not increased and dropped to some subset $Y \subseteq \text{dom}(P_i)$ of size strictly less than k. Therefore, let $Y = \{B_1, \ldots, B_j\}$, with $j < k$, and let $\text{dom}(P_i) \setminus Y = \{D_1, \ldots, D_q\}$. Then $P_{i,Y}^{\text{check}}$ contains these j productions, with $1 \leq h \leq j-1$:

$$r_{i,Y,B_h}^{\text{check}} : (B_h \to B_h, \{r_{i,Y,B_h}^{\text{check}}, r_{i,Y,B_{h+1}}^{\text{check}}\}, \emptyset) \text{ and}$$
$$r_{i,Y,B_j}^{\text{check}} : (B_j \to B_j, \{r_{i,Y,B_j}^{\text{check}}\} \cup \{r_{i,Y,D_1}^{\text{check}'}\}, \emptyset).$$

In case of a successful derivation, these productions check the presence of all symbols B_h, with $1 \leq h \leq j$, after which eventually production $r_{i,Y,D_1}^{\text{check}'}$ is to be applied next. This production is part of $P_{i,Y}^{\text{check}'}$, which contains the following $q = |\text{dom}(P_i) \setminus Y|$ productions, with $1 \leq \ell \leq q-1$:

$$r_{i,Y,D_\ell}^{\text{check}'} : (D_\ell \to F, \{r_{i,Y,D_\ell}^{\text{check}'}, r_{i,Y,D_{\ell+1}}^{\text{check}'}\}, \{r_{i,Y,D_\ell}^{\text{check}'}, r_{i,Y,D_{\ell+1}}^{\text{check}'}\}) \text{ and}$$
$$\begin{aligned} r_{i,Y,D_q}^{\text{check}'} : (D_q \to F,\ &\{r_{i,Y,D_q}^{\text{check}'}\} \cup \{r_{j,X,A_1}^{\text{present}} \mid j \neq i, X = \{A_1, \ldots, A_k\} \subseteq \text{dom}(P_j)\}, \\ &\{r_{i,Y,D_q}^{\text{check}'}\} \cup \{r_{j,X,A_1}^{\text{present}} \mid j \neq i, X = \{A_1, \ldots, A_k\} \subseteq \text{dom}(P_j)\}). \end{aligned}$$

Another competent enough component can be chosen for further rewriting.

Finally, note that G has a word α as axiom, whereas G' has a symbol S as axiom. Therefore G' moreover contains the following production:

$$s : (S \to \alpha, \{s\} \cup \{\, r^{\text{present}}_{i,X,A_1} \mid 1 \leq i \leq n,\ X = \{A_1, \ldots, A_k\} \subseteq \mathrm{dom}(P_i) \,\}, \emptyset).$$

Given axiom S, the application of this production s results in the sentential form α, after which the simulation of applying a k-competent component P_i, with $1 \leq i \leq n$, is started by applying a production $r^{\text{present}}_{i,X,A_1}$ from $P^{\text{present}}_{i,X}$, for a subset X of the domain of P_i with cardinality k.

We described the recurrent programmed grammar G' with appearance checking. The reader can verify that no successful derivation exists if productions are applied in different orders. This is ensured by productions guaranteeing, when needed, the introduction of failure symbol F that cannot be rewritten. The recurrent programmed grammar G' with appearance checking simulates the CDGS G working in $=k$-comp.dni-mode and rewriting in parallel, and generates $L(G)$. This proves the inclusion.

In contrast, for CDGSs working in $=k$-comp.-mode, for some $k \geq 2$, and rewriting sequentially, we do not know whether the two definitions of nonincreasing competence lead to different language classes. What we do know is that, for $k \geq 2$, sni-CDGSs working in $=k$-comp.-mode and rewriting sequentially can generate all languages that can be generated by context-free recurrent programmed grammars with appearance checking, while CDGSs working in $=k$-comp.dni-mode and rewriting sequentially can generate all languages that can be generated by context-free programmed grammars with appearance checking.

Theorem 4. $\mathcal{L}(\mathrm{rPR}, \mathrm{CF}, \mathrm{ac}) \subseteq \mathcal{L}(\text{sni-CD}, \mathrm{CF}, =k\text{-comp.}) \subseteq \mathcal{L}(\mathrm{PR}, \mathrm{CF}, \mathrm{ac}) = \mathcal{L}(\mathrm{CD}, \mathrm{CF}, =k\text{-comp.dni})$, *for* $k \geq 2$.

Proof. We prove the first inclusion (the second is trivial) and the equality, both for $k = 2$. The generalizations of the proofs to the cases that $k > 2$ are rather straightforward and left to the reader. Hence, let $k = 2$.

[$\mathcal{L}(\mathrm{rPR}, \mathrm{CF}, \mathrm{ac}) \subseteq \mathcal{L}(\text{sni-CD}, \mathrm{CF}, =k\text{-comp.})$] This follows directly from the proof of the inclusion $\mathcal{L}(\mathrm{rPR}, \mathrm{CF}, \mathrm{ac}) \subseteq \mathcal{L}(\text{sni-CD}, \mathrm{parCF}, =k\text{-comp.})$ in Theorem 3. It is easy to see that the exact same construction suffices.

Now consider the equality in the statement above. We only prove the inclusion from left to right, as the reverse direction is obvious.

[$\mathcal{L}(\mathrm{PR}, \mathrm{CF}, \mathrm{ac}) \subseteq \mathcal{L}(\mathrm{CD}, \mathrm{CF}, =k\text{-comp.dni})$] Let $G = (N, T, P, S, \Lambda, \sigma, \phi)$ be a programmed grammar with appearance checking. We assume its productions are of the form $p : (A \to z, \sigma(p), \phi(p))$, with success field $\sigma(p)$ and failure field $\phi(p)$ so that $p \notin \sigma(p) \cup \phi(p)$. Without loss of generality, assume p_1 is its only production able to rewrite S. To simulate G, we construct a CDGS G' with the disjoint union of sets of nonterminals

$$N' = N \cup \{F, Z\} \cup \{\, p, p' \mid p \in \Lambda \,\} \cup \bigcup_{p \in \Lambda} N_p, \text{ with } N_p = \{\, A_p, A'_p, \tilde{A}_p \mid A \in N \,\},$$

terminals T disjoint from N', axiom Sp_1Z and the components defined next. For each production $p : (A \to z, \sigma(p), \phi(p))$, we construct the components

$$P_{\text{p,code}} = \{A \to A_p, A \to A'_p\} \cup \{p \to p\} \text{ and } P_{\text{p,uncode}} = \{A'_p \to A\} \cup \{p \to p\}.$$

Now a successful application of production p is simulated by the components

$$\begin{aligned} P_{\text{p,present}} &= \{A_p \to \tilde{A}_p, \tilde{A}_p \to \tilde{A}_p\} \cup \{p \to p'\}, \\ P_{\text{p,apply}} &= \{\tilde{A}_p \to z\} \cup \{p' \to p'\} \text{ and} \\ P_{\text{p,finish}} &= \{\, B \to F \mid B \in N_p \,\} \cup \{Z \to Z\} \cup \{\, p' \to q \mid q \in \sigma(p), q \neq p \,\}, \end{aligned}$$

while a "failed" application of production p is simulated by the component

$$P_{\text{p,absent}} = \{B \to F \mid B \in N_q, q \in \Lambda\} \cup \{A \to F, Z \to Z\} \cup \{p \to q \mid q \in \phi(p), q \neq p\}.$$

Two types of markers are present in any sentential form: A general one (Z) to guarantee that components are 2-competent and a specific one (p, or its primed version) to indicate the context-free production being simulated.

Let $p : (A \to z, \sigma(p), \phi(p))$ be the production to be applied to the current sentential form, which we assume to contain one or more occurrences of A. The only 2-competent component which does not introduce a trap symbol F is $P_{\text{p,code}}$. The moment $P_{\text{p,code}}$ replaces the last nonterminal A, it is no longer 2-competent. All nonterminals A are then replaced by A_p or A'_P, respectively. Now $P_{\text{p,uncode}}$ is 2-competent if in the previous step nonterminals A'_p were introduced. In this case, all these nonterminals are rewritten to A. There may thus be nonterminals A_p left in the sentential form — if there is no A_p present, we are left with the sentential form we started with.

Now assume that the current sentential form has at least one A_p. Then $P_{\text{p,present}}$ is 2-competent on this sentential form; other components that may be competent under certain further assumptions, like $P_{\text{p,finish}}$ or $P_{\text{p,absent}}$, do not lead to a successful derivation. Also, no successful derivation exists if $p \to p'$ is applied before $A_p \to \tilde{A}_p$ by $P_{\text{p,present}}$. If one occurrence of A_p is replaced by $\tilde{A}_p$, then either $P_{\text{p,present}}$ becomes 3-competent, in case another nonterminal A_p is present in the sentential form, or it remains 2-competent and production $p \to p'$ is applied, while the level of competence drops to 1. Thus $P_{\text{p,present}}$ is applicable iff exactly one occurrence of nonterminal A_p is present in the sentential form under consideration, which is then replaced by $\tilde{A}_p$ (and p by p'). Here the dynamically nonincreasing feature is used to ensure that the sentential from contains exactly one nonterminal A_p for rewriting. The derivation continues with $P_{\text{p,apply}}$ followed by $P_{\text{p,finish}}$, leading to a sentential form in which the former $\tilde{A}_p$ is replaced by z and the label is changed to $q \in \sigma(p)$. Note that a production $Z \to Z$ is used to force the 2-competence of $P_{\text{p,finish}}$, and that it blocks shortcuts whenever p' and $\tilde{A}_p$ are present simultaneously, in case $P_{\text{p,apply}}$ was *not* applied after $P_{\text{p,present}}$. This shows how to successfully apply production p of G.

Now assume no A occurs in the sentential form. Then $P_{\text{p,absent}}$ is 2-competent on this sentential form. If $P_{\text{p,absent}}$ replaces p by q, for $q \in \phi(p)$, it is no longer 2-competent and the sentential form can simulate the production labelled q, thus simulating the "failed" application of p.

Finally, a derivation can only finish when no more nonterminals other than Z or p, for some $p \in \Lambda$, appear in the sentential form, in which case

$$P_{p,\text{terminate}} = \{Z \to \lambda, p \to \lambda\} \cup \{\, B \to F \mid B \in N' \,\}$$

is 2-competent and it removes all remaining nonterminals from the sentential form until a terminal word is obtained. An earlier application of component $P_{p,\text{terminate}}$ is not possible, even in case a primed label appears, as then also a nonterminal $\tilde{A}_p$ occurs and $P_{p,\text{terminate}}$ is at least 3-competent. If, on the other hand, nonterminal p is removed and the derivation of $P_{p,\text{terminate}}$ is stopped because it became 1-competent, then a successful derivation is blocked because no other component can become 2-competent anymore. A similar reasoning can be applied if nonterminal Z is removed first. Whenever $P_{p,\text{terminate}}$ is applied, it thus has to remove both nonterminals p and Z.

We described the CDGS G' working in $=2\text{-comp.dni}$-mode and rewriting sequentially. The reader can verify that no successful derivation exists if productions are applied in a different order. G' thus simulates the programmed grammar G with appearance checking, and generates $L(G)$.

6 Conclusion

In this paper, we introduced and examined both a static and a dynamic definition of nonincreasing competence in CDGSs working in $=k\text{-comp.}$-mode of derivation, for some $k \geq 1$, and rewriting in a context-free sequential or parallel manner. We obtained the following chain of inclusions, with $k \geq 2$:

$$\begin{aligned}
&\mathcal{L}(\text{ET0L}) = \mathcal{L}(\text{sni-CD}, \text{parCF}, =1\text{-comp.}) = \mathcal{L}(\text{CD}, \text{parCF}, =1\text{-comp.dni}) \\
&= \mathcal{L}(\text{sni-CD}, \text{CF}, =1\text{-comp.}) \subset \mathcal{L}(\text{CD}, \text{CF}, =1\text{-comp.dni}) = \mathcal{L}(\text{fRC}, \text{CF}) \\
&\subseteq \mathcal{L}(\text{sni-CD}, \text{parCF}, =k\text{-comp.}) = \mathcal{L}(\text{CD}, \text{parCF}, =k\text{-comp.dni}) = \mathcal{L}(\text{RC}, \text{ET0L}) \\
&\subseteq \mathcal{L}(\text{sni-CD}, \text{CF}, =k\text{-comp.}) \subseteq \mathcal{L}(\text{CD}, \text{CF}, =k\text{-comp.dni}) = \mathcal{L}(\text{RE}).
\end{aligned}$$

Our results might help solve a longstanding open problem in regulated rewriting: is $\mathcal{L}(\text{RC}, \text{ET0L}) = \mathcal{L}(\text{rPR}, \text{CF}, \text{ac}) \subseteq \mathcal{L}(\text{PR}, \text{CF}, \text{ac}) = \mathcal{L}(\text{RE})$ strict?

In Table 1 we compare the results we obtained in this paper with the results we obtained in [6,7]. This table should be read as follows: the entry at the intersection of the ith row, marked by a mode f, and the jth column, is either a language family $\mathcal{L}$ which coincides with the family X at the top of the column having components working in the appropriate mode or an expression of the form $\mathcal{L}_1 \subset \cdot = \mathcal{L}_2$, which means that $\mathcal{L}_1 \subset X = \mathcal{L}_2$ holds. For instance, for $\mathcal{L}(\text{CD}, \text{CF}, =1\text{-comp.dni})$ we have $\mathcal{L}(\text{ET0L}) \subset \cdot = \mathcal{L}(\text{fRC})$, which means $\mathcal{L}(\text{ET0L}) \subset \mathcal{L}(\text{CD}, \text{CF}, =1\text{-comp.dni}) = \mathcal{L}(\text{fRC})$.

This table shows some interesting open problems. It would, e.g., be interesting to know the generative power of CDGSs working in the ordinary $=1\text{-comp.}$-mode of derivation and rewriting sequentially. All we know is this:
$\mathcal{L}(\text{ET0L}) = \mathcal{L}(\text{sni-CD}, \text{CF}, =1\text{-comp.}) \subset \mathcal{L}(\text{CD}, \text{CF}, =1\text{-comp.dni}) = \mathcal{L}(\text{fRC}, \text{CF})$
$\subseteq \mathcal{L}(\text{CD}, \text{CF}, =1\text{-comp.}) \subseteq \mathcal{L}(\text{CD}, \text{CF}, =2\text{-comp.}) = \mathcal{L}(\text{RE})$.

Table 1. Generative power CDGS in (static/dynamic nonincreasing) $=k$-comp.-mode

CDGS f, with $k \geq 2$	$\mathcal{L}(\text{CDGS}, \text{CF}, f)$	$\mathcal{L}(\text{CDGS}, \text{parCF}, f)$
sni-CD=1-comp.	$\mathcal{L}(\text{ET0L})$	$\mathcal{L}(\text{ET0L})$
CD=1-comp.dni	$\mathcal{L}(\text{ET0L}) \subset \cdot = \mathcal{L}(\text{fRC})$	$\mathcal{L}(\text{ET0L})$
CD=1-comp.	$\mathcal{L}(\text{ET0L}) \subset \mathcal{L}(\text{fRC}) \subseteq \cdot \subseteq \mathcal{L}(\text{RE})$	$\mathcal{L}(\text{ET0L}) \subseteq \cdot \subseteq \mathcal{L}(\text{RC}, \text{ET0L})$
sni-CD=k-comp.	$\mathcal{L}(\text{RC}, \text{ET0L}) \subseteq \cdot \subseteq \mathcal{L}(\text{RE})$	$\mathcal{L}(\text{RC}, \text{ET0L})$
CD=k-comp.dni	$\mathcal{L}(\text{RE})$	$\mathcal{L}(\text{RC}, \text{ET0L})$
CD=k-comp.	$\mathcal{L}(\text{RE})$	$\mathcal{L}(\text{RC}, \text{ET0L})$

Acknowledgements

We thank an anonymous referee of [7] for suggesting to consider also a dynamic definition of nonincreasing competence-based derivation.

References

1. Meersman, R., Rozenberg, G.: Cooperating Grammar Systems. In: Winkowski, J. (ed.) MFCS 1978. LNCS, vol. 64, pp. 364–374. Springer, Heidelberg (1978)
2. Csuhaj–Varjú, E., Dassow, J.: On Cooperating Distributed Grammar Systems. Journal of Information Processing and Cybernetics EIK 26, 49–63 (1990)
3. Csuhaj–Varjú, E., Dassow, J., Kelemen, J., Păun, G.: Grammar Systems: A Grammatical Approach to Distribution and Cooperation. Topics in Computer Mathematics, vol. 5. Gordon and Breach, London (1994)
4. Dassow, J., Păun, G., Rozenberg, G.: Grammar Systems. In: [13], vol. 2, ch. 4, pp. 155–213 (1997)
5. Bordihn, H., Csuhaj–Varjú, E.: On Competence and Completeness in CD Grammar Systems. Acta Cybernetica 12, 347–361 (1996)
6. ter Beek, M.H., Csuhaj–Varjú, E., Holzer, M., Vaszil, G.: On Competence in CD Grammar Systems. In: Calude, C.S., Calude, E., Dinneen, M.J. (eds.) DLT 2004. LNCS, vol. 3340, pp. 76–88. Springer, Heidelberg (2004)
7. ter Beek, M.H., Csuhaj–Varjú, E., Holzer, M., Vaszil, G.: On Competence in CD Grammar Systems with Parallel Rewriting. International Journal of Foundations of Computer Science 18, 1425–1439 (2007)
8. Csuhaj–Varjú, E., Dassow, J., Holzer, M.: CD Grammar Systems with Competence Based Entry Conditions in their Cooperation Protocols. International Journal of Computer Mathematics 83, 159–169 (2006)
9. von Solms, S.H.: Some Notes on ET0L-languages. International Journal of Computer Mathematics 5, 285–296 (1975)
10. Bordihn, H., Holzer, M.: Grammar Systems with Negated Conditions in their Cooperation Protocols. Journal of Universal Computer Science 6, 1165–1184 (2000)
11. Fernau, H., Wätjen, D.: Remarks on Regulated Limited ET0L Systems and Regulated Context-Free Grammars. Theoretical Computer Science 194, 35–55 (1998)
12. Dassow, J., Păun, G.: Regulated Rewriting in Formal Language Theory. EATCS Monographs in Theoretical Computer Science, vol. 18. Springer, Berlin (1989)
13. Rozenberg, G., Salomaa, A. (eds.): Handbook of Formal Languages, vol. 1-3. Springer, Berlin (1997)
14. Bordihn, H., Holzer, M.: Personal communication