Ogden's Lemma, Multiple Context-Free Grammars, and the Control Language Hierarchy

Makoto Kanazawa[1]

Department of Advanced Sciences, Faculty of Science and Engineering, Hosei University, Koganei-shi, Tokyo, Japan

Abstract

I present a simple example of a multiple context-free language for which a very weak variant of generalized Ogden's lemma fails. This language is generated by a non-branching (and hence well-nested) 3-MCFG as well as by a (non-well-nested) binary-branching 2-MCFG; it follows that neither the class of well-nested 3-MCFLs nor the class of 2-MCFLs is included in Weir's control language hierarchy, for which Palis and Shende proved an Ogden-like iteration theorem. I then give a simple sufficient condition for an MCFG to satisfy a natural analogue of Ogden's lemma, and show that the corresponding class of languages is a substitution-closed full AFL which includes Weir's control language hierarchy. My variant of generalized Ogden's lemma is incomparable in strength to Palis and Shende's variant and is arguably a more natural generalization of Ogden's original lemma. I also prove a strengthening of my earlier pumping lemma for well-nested MCFLs which places a bound on the combined length of the substrings that can be iterated.

Keywords: grammars, Ogden's lemma, multiple context-free grammars, control languages, pumping lemma

1. Introduction

A *multiple context-free grammar* [1] is a context-free grammar on tuples of strings (of varying length). It has widely been believed that MCFGs provide an adequate formalization of Joshi's [2] informal concept of *mildly context-sensitive* grammars, but some recent work has cast doubt on this identification [3, 4]. For this reason, it is always interesting to ask to what extent a given prominent property of context-free grammars is either shared by or suitably generalizes to MCFGs.

An analogue of the pumping lemma, which asserts the existence of a certain number of substrings that can be simultaneously iterated, has been established for *well-nested* MCFGs and (non-well-nested) MCFGs of dimension 2 [5]. So far, it has been unknown whether an analogue of Ogden's [6] strengthening of the pumping lemma holds of these classes. This paper negatively answers the question for both classes, and moreover proves a generalized Ogden's lemma for the class of MCFGs satisfying a certain simple property. The class of languages generated by the grammars in this class includes Weir's [7] control language hierarchy, the only non-trivial subclass of MCFLs for which an Ogden-style iteration theorem has been proved so far [8].

The paper also gives a strengthened version of the pumping lemma of [5] that is more in line with the original statement of the pumping lemma for context-free languages [9].

2. Preliminaries

The power set of a set X is denoted $\mathscr{P}(X)$. If X and Y are sets, we write Y^X for the set of (total) functions from X to Y. The set of natural numbers is denoted $\mathbb{N}$. If i and j are natural numbers, we write $[i, j]$ for the set

Email address: kanazawa@hosei.ac.jp (Makoto Kanazawa)

[1]This work was supported by JSPS KAKENHI Grant Number 25330020.

$\{ n \in \mathbb{N} \mid i \leq n \leq j \}$. We write $|w|$ for the length of a string w and $|S|$ for the cardinality of a set S; the context should make it clear which is intended. If u, v, w are strings, we write $(u[v]w)$ for the subinterval $[|u| + 1, |uv|]$ of $[1, |uvw|]$. If w is a string, w^R denotes the reversal of w.

Given a positive integer k, a function ρ from $\{1, \dots, k\}$ to $\{1, R\}$, and homomorphisms $h_1, \dots, h_k$ from Σ^* to Γ^*, we define a function $\langle \rho, h_1, \dots, h_k \rangle$ from Σ^* to Γ^* by

$$\langle \rho, h_1, \dots, h_k \rangle(w) = h'_1(w) \dots h'_k(w),$$

where for $i = 1, \dots, k$, $h'_i(w)$ is either $h_i(w)$ or $(h_i(w))^R$ depending on whether $\rho(i)$ is 1 or R. Such a function is called a *homomorphic replication of type* ρ [10, 11]. In this paper, we sometimes represent $\rho \in \{1, R\}^{\{1,\dots,k\}}$ as a sequence $(\rho(1), \dots, \rho(k))$. A homomorphic replication is extended to a function from $\mathscr{P}(\Sigma^*)$ to $\mathscr{P}(\Gamma^*)$ in a familiar way. For example, if $L \subseteq \Sigma^*$ and h_1 and h_2 are homomorphisms from Σ^* to Γ^*, then $\langle (1, R), h_1, h_2 \rangle(L) = \{ h_1(w)(h_2(w))^R \mid w \in L \}$.

If Σ and Γ are finite alphabets, a function σ from Σ to $\mathscr{P}(\Gamma^*)$ is called a *substitution*. A substitution σ is extended to a function from Σ^* to $\mathscr{P}(\Gamma^*)$ and then to a function from $\mathscr{P}(\Sigma^*)$ to $\mathscr{P}(\Gamma^*)$ by

$$\begin{aligned} \sigma(\varepsilon) &= \varepsilon, \\ \sigma(aw) &= \sigma(a)\sigma(w) \quad \text{for } a \in \Sigma, w \in \Sigma^*, \\ \sigma(L) &= \bigcup_{w \in L} \sigma(w) \quad \text{for } L \subseteq \Sigma^*. \end{aligned}$$

If $\Sigma' \subseteq \Sigma$ and for each $c \in \Sigma'$, $L_c \subseteq \Gamma^*$, we write $[c \leftarrow L_c]_{c \in \Sigma'}$ for the substitution σ such that

$$\sigma(c) = \begin{cases} L_c & \text{if } c \in \Sigma', \\ \{c\} & \text{otherwise}, \end{cases}$$

and write $L[c \leftarrow L_c]_{c \in \Sigma'}$ for $\sigma(L)$.

If σ is a substitution from Σ to $\mathscr{P}(\Sigma^*)$, then we let

$$\begin{aligned} \sigma^0(L) &= L, \\ \sigma^{n+1}(L) &= \sigma(\sigma^n(L)), \\ \sigma^\infty(L) &= \bigcup_{n \in \mathbb{N}} \sigma^n(L). \end{aligned}$$

The operation σ^∞ is called an *iterated substitution*; it is a *nested iterated substitution* [12] if $c \in \sigma(c)$ for each $c \in \Sigma$.

A family $\mathscr{L}$ of languages is *closed under substitution* if whenever $L \in \mathscr{L} \cap \mathscr{P}(\Sigma^*)$ and $L_c \in \mathscr{L} \cap \mathscr{P}(\Gamma^*)$ for each $c \in \Sigma$, we have $L[c \leftarrow L_c]_{c \in \Sigma} \in \mathscr{L}$. We say that $\mathscr{L}$ is *closed under nested iterated substitution* if whenever $L \in \mathscr{L} \cap \mathscr{P}(\Sigma^*)$ and $c \in L_c \in \mathscr{L} \cap \mathscr{P}(\Sigma^*)$ for each $c \in \Sigma$, we have $\sigma^\infty(L) \in \mathscr{L}$, where $\sigma = [c \leftarrow L_c]_{c \in \Sigma}$. It is known that the family of context-free languages is closed under substitution and nested iterated substitution [13].

2.1. Multiple Context-Free Grammars

A *multiple context-free grammar* (MCFG) [1] is a quadruple $G = (N, \Sigma, P, S)$, where N is a finite set of *nonterminals*, each with a fixed *dimension* ≥ 1, Σ is a finite alphabet of *terminals*, P is a set of *rules*, and S is the distinguished *initial nonterminal* of dimension 1. We write $N^{(q)}$ for the set of nonterminals in N of dimension q. A nonterminal in $N^{(q)}$ is interpreted as a q-ary predicate over Σ^*. A rule is stated with the help of *variables* interpreted as ranging over Σ^*. Let $\mathcal{X}$ be a denumerable set of variables. We use boldface lower-case letters as elements of $\mathcal{X}$. A rule is a *definite clause* (in the sense of logic programming) constructed with *atoms* of the form $A(\alpha_1, \dots, \alpha_q)$, with $A \in N^{(q)}$ and $\alpha_1, \dots, \alpha_q$ *patterns*, i.e., strings over $\Sigma \cup \mathcal{X}$. An MCFG rule is of the form

$$A(\alpha_1, \dots, \alpha_q) \leftarrow B_1(\boldsymbol{x}_{1,1}, \dots, \boldsymbol{x}_{1,q_1}), \dots, B_n(\boldsymbol{x}_{n,1}, \dots, \boldsymbol{x}_{n,q_n}),$$

where $n \geq 0$, $A, B_1, \dots, B_n$ are nonterminals of dimensions $q, q_1, \dots, q_n$, respectively, the $\boldsymbol{x}_{i,j}$ are pairwise distinct variables, and each α_i is a string over $\Sigma \cup \{ \boldsymbol{x}_{i,j} \mid i \in [1, n], j \in [1, q_i] \}$, such that $(\alpha_1, \dots, \alpha_q)$ contains at most one

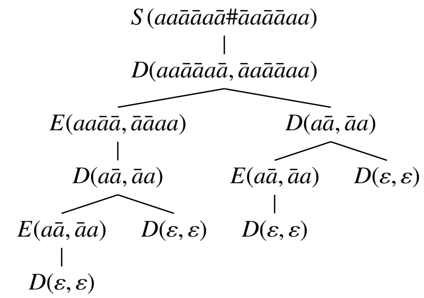

Figure 1: A derivation tree of a 2-MCFG.

occurrence of each $\boldsymbol{x}_{i,j}$. An MCFG is an *m-MCFG* if the dimensions of its nonterminals do not exceed m; it is *r-ary branching* if each rule has no more than r occurrences of nonterminals in its *body* (i.e., the part that follows the symbol $\leftarrow$). We call a unary branching grammar *non-branching*.[2]

An *instance* of a rule is the result of substituting a pattern for each variable in the rule. An atom or a rule instance is *ground* if it contains no variables. Given an MCFG $G = (N, \Sigma, P, S)$, a ground atom $A(w_1, \ldots, w_q)$ *directly follows* from a sequence of ground atoms $B_1(v_{1,1}, \ldots, v_{1,q_1}), \ldots, B_n(v_{n,1}, \ldots, v_{n,q_n})$ if

$$A(w_1, \ldots, w_q) \leftarrow B_1(v_{1,1}, \ldots, v_{1,q_1}), \ldots, B_n(v_{n,1}, \ldots, v_{n,q_n})$$

is a ground instance of some rule in P. A ground atom $A(w_1, \ldots, w_q)$ is *derivable*, written $\vdash_G A(w_1, \ldots, w_q)$, if it directly follows from some sequence of derivable ground atoms. In particular, if $A(w_1, \ldots, w_q) \leftarrow$ is a rule in P, we have $\vdash_G A(w_1, \ldots, w_q)$.

A derivable ground atom is naturally associated with a *derivation tree* whose nodes are labeled by derivable ground atoms. A derivation tree τ for a ground atom $A(w_1, \ldots, w_q)$ is a tree such that

- the root of τ is labeled by $A(w_1, \ldots, w_q)$,
- for each node v of τ, the ground atom labeling v directly follows from the sequence of ground atoms labeling its children.

When τ is a derivation tree of G for $A(w_1, \ldots, w_q)$, we sometimes write

$$\vdash_G \tau : A(w_1, \ldots, w_q).$$

The language generated by G is defined as $L(G) = \{\, w \in \Sigma^* \mid\, \vdash_G S(w) \,\}$, or equivalently, $L(G) = \{\, w \in \Sigma^* \mid G$ has a derivation tree for $S(w)\,\}$. The class of languages generated by m-MCFGs is denoted m-MCFL, and the class of languages generated by r-ary branching m-MCFGs is denoted m-MCFL(r).

Example 1. Consider the following 2-MCFG:

$$\begin{aligned} S(\boldsymbol{x}_1\#\boldsymbol{x}_2) &\leftarrow D(\boldsymbol{x}_1, \boldsymbol{x}_2) & \qquad D(\boldsymbol{x}_1\boldsymbol{y}_1, \boldsymbol{y}_2\boldsymbol{x}_2) &\leftarrow E(\boldsymbol{x}_1, \boldsymbol{x}_2), D(\boldsymbol{y}_1, \boldsymbol{y}_2) \\ D(\varepsilon, \varepsilon) &\leftarrow & E(a\boldsymbol{x}_1\bar{a}, \bar{a}\boldsymbol{x}_2 a) &\leftarrow D(\boldsymbol{x}_1, \boldsymbol{x}_2) \end{aligned}$$

Here, S is the initial nonterminal and D and E are both nonterminals of dimension 2. This grammar is binary branching and generates the language $\{\, w\#w^R \mid w \in D_1^* \,\}$, where D_1^* is the (one-sided) *Dyck language* over the alphabet $\{a, \bar{a}\}$. Figure 1 shows the derivation tree for $aa\bar{a}\bar{a}a\bar{a}\#\bar{a}a\bar{a}\bar{a}aa$.

It is also useful to define the notion of a derivation of an atom $A(\alpha_1, \ldots, \alpha_q)$ from an assumption $C(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r)$, where $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r$ are pairwise distinct variables. An atom $A(\alpha_1, \ldots, \alpha_q)$ is *derivable from an assumption* $C(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r)$, written $C(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r) \vdash_G A(\alpha_1, \ldots, \alpha_q)$, if either

[2]Non-branching MCFGs were called *linear* in [14].

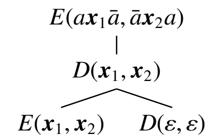

Figure 2: A derivation tree context for $E(a\boldsymbol{x}_1\bar{a}, \bar{a}\boldsymbol{x}_2 a)$ with an assumption $E(\boldsymbol{x}_1, \boldsymbol{x}_2)$.

1. $A = C$ and $(\alpha_1, \dots, \alpha_q) = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_r)$, or
2. there are some atom $B_i(\beta_1, \dots, \beta_{q_i})$ and ground atoms $B_j(v_{j,1}, \dots, v_{j,q_j})$ for $j \in [1, i-1] \cup [i+1, n]$ such that $C(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r) \vdash_G B_i(\beta_1, \dots, \beta_{q_i})$, $\vdash_G B_j(v_{j,1}, \dots, v_{j,q_j})$ ($j \in [1, i-1] \cup [i+1, n]$), and

$$A(\alpha_1, \dots, \alpha_q) \leftarrow B_1(v_{1,1}, \dots, v_{1,q_1}), \dots, B_{i-1}(v_{i-1,1}, \dots, v_{i-1,q_{i-1}}),$$
$$B_i(\beta_1, \dots, \beta_{q_i}), B_{i+1}(v_{i+1,1}, \dots, v_{i+1,q_{i+1}}), \dots, B_n(v_{n,1}, \dots, v_{n,q_n})$$

is an instance of some rule in P.

Analogously to the case of a derivation without an assumption, when an atom $A(\alpha_1, \dots, \alpha_q)$ is derivable from an assumption $C(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r)$, there is an associated tree witnessing this fact. In such a tree, there is a unique leaf labeled by $C(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q)$, and the nodes along the path from the root to that leaf are labeled by non-ground atoms, while all other nodes are labeled by ground atoms. We call such a tree *derivation tree context* [5] and the atom $C(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r)$ its *assumption*. We write $C(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r) \vdash_G \upsilon : A(\alpha_1, \dots, \alpha_q)$ to mean that υ is a derivation tree context witnessing $C(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r) \vdash_G A(\alpha_1, \dots, \alpha_q)$.

Let us write $[v_1/\boldsymbol{x}_1, \dots, v_r/\boldsymbol{x}_r]$ for the simultaneous substitution of strings $v_1, \dots, v_r$ for variables $\boldsymbol{x}_1, \dots, \boldsymbol{x}_r$. Evidently, when we have $\vdash_G \tau : C(v_1, \dots, v_r)$ and $C(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r) \vdash_G \upsilon : A(\alpha_1, \dots, \alpha_q)$, we can combine τ and υ into a derivation tree for $A(\alpha_1, \dots, \alpha_q)[v_1/\boldsymbol{x}_1, \dots, v_r/\boldsymbol{x}_r]$. This derivation tree, which we write $\upsilon[\tau]$, is the result of inserting τ in place of the assumption $C(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r)$ of υ and then applying the substitution $[v_1/\boldsymbol{x}_1, \dots, v_r/\boldsymbol{x}_r]$ to the remaining non-ground atoms. Thus, we have

$$\vdash_G \upsilon[\tau] : A(\alpha_1, \dots, \alpha_q)[v_1/\boldsymbol{x}_1, \dots, v_r/\boldsymbol{x}_r]$$

whenever $\vdash_G \tau : C(v_1, \dots, v_r)$ and $C(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r) \vdash_G \upsilon : A(\alpha_1, \dots, \alpha_q)$.

The following lemma says that when $B(v_1, \dots, v_r)$ is derived in the course of a derivation of $A(w_1, \dots, w_q)$, the derivation can be decomposed into one for $B(v_1, \dots, v_r)$ and a derivation tree context with an assumption $B(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r)$:

Lemma 2. *Let τ be a derivation tree of an MCFG G for some ground atom $A(w_1, \dots, w_q)$, and let τ' be a subtree of τ consisting of a node labeled by $B(v_1, \dots, v_r)$ and the nodes that lie below it. Then there is a derivation tree context υ with an assumption $B(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r)$ such that $\tau = \upsilon[\tau']$. In particular, we have*

$$B(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r) \vdash_G \upsilon : A(\alpha_1, \dots, \alpha_q),$$
$$(w_1, \dots, w_q) = (\alpha_1, \dots, \alpha_q)[v_1/\boldsymbol{x}_1, \dots, v_r/\boldsymbol{x}_r],$$

for some patterns $\alpha_1, \dots, \alpha_q$.

Example 3. Consider the derivation tree in Figure 1 and the node ν labeled by $E(aa\bar{a}\bar{a}, \bar{a}\bar{a}aa)$. Let τ be the subtree of this derivation tree consisting of ν and the nodes that lie below it. Consider the node ν_1 labeled by $E(a\bar{a}, \bar{a}a)$ in τ. The rules used in the portion of τ that remains after removing the nodes below ν_1 determine a derivation tree context witnessing $E(\boldsymbol{x}_1, \boldsymbol{x}_2) \vdash_G E(a\boldsymbol{x}_1\bar{a}, \bar{a}\boldsymbol{x}_2 a)$, which is depicted in Figure 2. Note that substituting $a\bar{a}, \bar{a}a$ for $\boldsymbol{x}_1, \boldsymbol{x}_2$ in $E(a\boldsymbol{x}_1\bar{a}, \bar{a}\boldsymbol{x}_2 a)$ gives back $E(aa\bar{a}\bar{a}, \bar{a}\bar{a}aa)$.

An MCFG rule $A(\alpha_1, \dots, \alpha_q) \leftarrow B_1(\boldsymbol{x}_{1,1}, \dots, \boldsymbol{x}_{1,q_1}), \dots, B_n(\boldsymbol{x}_{n,1}, \dots, \boldsymbol{x}_{n,q_n})$ is said to be

- *non-deleting* if all variables $\boldsymbol{x}_{i,j}$ in its body occur in $(\alpha_1, \dots, \alpha_q)$;
- *non-permuting* if for each $i \in [1, n]$, the variables $\boldsymbol{x}_{i,1}, \dots, \boldsymbol{x}_{i,q_i}$ occur in $(\alpha_1, \dots, \alpha_q)$ in this order;

- *well-nested* if it is non-deleting and non-permuting and there are no $i, j \in [1, n], k \in [1, q_i - 1], l \in [1, q_l - 1]$ such that $\boldsymbol{x}_{i,k}, \boldsymbol{x}_{j,l}, \boldsymbol{x}_{i,k+1}, \boldsymbol{x}_{j,l+1}$ occur in $(\alpha_1, \ldots, \alpha_q)$ in this order.

Every r-ary branching m-MCFG has an equivalent r-ary branching m-MCFG whose rules are all non-deleting and non-permuting, and henceforth we will always assume that these conditions are satisfied. An MCFG whose rules are all well-nested is a *well-nested MCFG* [5]. The 2-MCFG in Example 1 is well-nested. It is known that there is no well-nested MCFG for the language $\{ w\#w \mid w \in D_1^* \}$ [15], although it is easy to write a non-well-nested 2-MCFG for this language.

Every (non-deleting and non-permuting) non-branching MCFG is by definition well-nested. The class $\bigcup_m m$-MCFL(1) coincides with the class of output languages of *deterministic two-way finite-state transducers* (see [14]).

2.2. The Control Language Hierarchy

Weir's [7] *control language hierarchy* is defined in terms of the notion of a *labeled distinguished grammar*, which is a 5-tuple $G = (N, \Sigma, P, S, f)$, where $\overline{G} = (N, \Sigma, P, S)$ is an ordinary context-free grammar and $f\colon P \to \mathbb{N}$ is a function such that if $\pi \in P$ is a context-free production with n occurrences of nonterminals on its right-hand side, then $f(\pi) \in [0, n]$. We view P as a finite alphabet, and use a language $C \in P^*$ to restrict the derivations of G. The pair (G, C) is a *control grammar*. To define the language of (G, C), we first define the rewriting of a nonterminal induced by a nonempty string $\xi \in P^+$ inductively as follows:

- $A \overset{\pi}{\Longrightarrow}_G \alpha$ if $\pi = A \to \alpha$ is a production in P and $f(\pi) = 0$,
- $A \overset{\pi\xi}{\Longrightarrow}_G w_0 B_1 w_1 \ldots B_{i-1} w_{i-1} \beta w_i B_{i+1} w_{i+1} \ldots B_n w_n$ if $\pi = A \to w_0 B_1 w_1 \ldots B_n w_n$ is a production in P, $f(\pi) = i \geq 1$, and $B_i \overset{\xi}{\Longrightarrow}_G \beta$.

If $A \overset{\xi}{\Longrightarrow}_G \alpha$ for some $\xi \in C$, we write $A \overset{C}{\Longrightarrow}_G \alpha$. A controlled derivation of (G, C) starting from a nonterminal is defined inductively as follows:

- $A \Rightarrow^*_{(G,C)} A$,
- $A \Rightarrow^*_{(G,C)} \alpha\beta\gamma$ if $A \Rightarrow^*_{(G,C)} \alpha B \gamma$ and $B \overset{C}{\Longrightarrow}_G \beta$.

Clearly, if $A \Rightarrow^*_{(G,C)} \alpha$, then $A \Rightarrow^*_{\overline{G}} \alpha$. The language of (G, C) is

$$L(G, C) = \{ w \in \Sigma^* \mid S \Rightarrow^*_{(G,C)} w \}.$$

The first level of the control language hierarchy is C_1 = CFL, the family of context-free languages, and for $k \geq 1$,

$$C_{k+1} = \{ L(G, C) \mid (G, C) \text{ is a control grammar and } C \in C_k \}.$$

The second level C_2 is known to coincide with the family of languages generated by well-nested 2-MCFGs, or equivalently, the family of *tree-adjoining languages* [7].

Example 4. Let $G = (N, \Sigma, P, S, f)$ be a labeled distinguished grammar consisting of the following productions:

$$\pi_1 : S \to aS\bar{a}S, \qquad \pi_2 : S \to bS\bar{b}S, \qquad \pi_3 : S \to \varepsilon,$$

where $f(\pi_1) = 1, f(\pi_2) = 1, f(\pi_3) = 0$. Note that $\overline{G}$ is the well-known context-free grammar for D_2^*, the Dyck language over $\{a, \bar{a}, b, \bar{b}\}$. Let $C = \{ \pi_1^n \pi_2^n \pi_3 \mid n \in \mathbb{N} \}$. Then we have

$$\begin{aligned}
&S \overset{\pi_3}{\Longrightarrow}_G \varepsilon,\\
&S \overset{\pi_1\pi_2\pi_3}{\Longrightarrow}_G ab\bar{b}S\bar{a}S,\\
&S \overset{\pi_1^2\pi_2^2\pi_3}{\Longrightarrow}_G aabb\bar{b}S\bar{b}S\bar{a}S\bar{a}S,
\end{aligned}$$

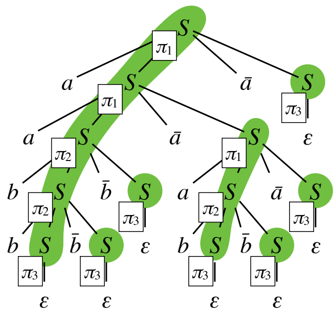

Figure 3: A derivation tree of a control grammar. In this tree, each node labeled by a nonterminal is accompanied by the label of the rule applied at that node, and patches of green connect each node with rule label π to the child corresponding to the value of $f(\pi)$.

and hence

$$S \Rightarrow^*_{(G,C)} \varepsilon,$$
$$S \Rightarrow^*_{(G,C)} ab\bar{b}\bar{a},$$
$$S \Rightarrow^*_{(G,C)} aabb\bar{b}\bar{b}\bar{a}ab\bar{b}\bar{a}\bar{a}.$$

The controlled derivation of the last string $aabb\bar{b}\bar{b}\bar{a}ab\bar{b}\bar{a}\bar{a}$ is shown in the form of a derivation tree in Figure 3. We have $L(G, C) = D_2^* \cap (\{ a^n b^n \mid n \in \mathbb{N} \}\{\bar{a}, \bar{b}\}^*)^*$. Since C is a context-free language, this language belongs to C_2.

Palis and Shende [8] proved the following Ogden-like theorem for C_k:

Theorem 5 (Palis and Shende)**.** *Let $L \in C_k$. There is a number p such that for all $z \in L$ and $D \subseteq [1, |z|]$, if $|D| \geq p$, there are $u_1, \dots, u_{2^k+1}, v_1, \dots, v_{2^k} \in \Sigma^*$ that satisfy the following conditions:*

(i) $z = u_1 v_1 u_2 v_2 \dots u_{2^k} v_{2^k} u_{2^k+1}$.

(ii) *for some* $j \in [1, 2^k]$,

$$D \cap (u_1 v_1 \dots [u_j] v_j u_{j+1} v_{j+1} \dots u_{2^k} v_{2^k} u_{2^k+1}) \neq \varnothing,$$
$$D \cap (u_1 v_1 \dots u_j [v_j] u_{j+1} v_{j+1} \dots u_{2^k} v_{2^k} u_{2^k+1}) \neq \varnothing,$$
$$D \cap (u_1 v_1 \dots u_j v_j [u_{j+1}] v_{j+1} \dots u_{2^k} v_{2^k} u_{2^k+1}) \neq \varnothing.$$

(iii) $|D \cap (u_1 v_1 \dots u_{2^{k-1}} [v_{2^{k-1}} u_{2^{k-1}+1} v_{2^{k-1}+1}] \dots u_{2^k} v_{2^k} u_{2^k+1})| \leq p$.

(iv) $u_1 v_1^n u_2 v_2^n \dots u_{2^k} v_{2^k}^n u_{2^k+1} \in L$ *for all* $n \in \mathbb{N}$.

Kanazawa and Salvati [16] proved the inclusion $C_k \subsetneq 2^{k-1}$-MCFL, while using Theorem 5 to show that the language $\mathrm{RESP}_{2^{k-1}}$ belongs to 2^{k-1}-MCFL $- C_k$ for $k \geq 2$, where $\mathrm{RESP}_l = \{ a_1^m a_2^m b_1^n b_2^n \dots a_{2l-1}^m a_{2l}^m b_{2l-1}^n b_{2l}^n \mid m, n \in \mathbb{N} \}$.

3. The Failure of Ogden's Lemma for Well-Nested MCFGs and 2-MCFGs

Let G be an MCFG, and consider a derivation tree τ for an element z of $L(G)$. When a node of τ and one of its descendants are labeled by ground atoms $B(w_1, \dots, w_r)$ and $B(v_1, \dots, v_r)$ sharing the same nonterminal B, the portion of τ consisting of the nodes that are neither above the first node nor below the second node determines a derivation tree context υ witnessing $B(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r) \vdash_G B(\beta_1, \dots, \beta_r)$ (called a *pump* in [5]), where $(\beta_1, \dots, \beta_r)[v_1/\boldsymbol{x}_1, \dots, v_r/\boldsymbol{x}_r] = (w_1, \dots, w_r)$. This was illustrated by Example 3. When each $\boldsymbol{x}_i$ occurs in β_i, i.e., $\beta_i = v_{2i-1}\boldsymbol{x}_i v_{2i}$ for some $v_{2i-1}, v_{2i} \in \Sigma^*$ (in which case υ is an *even pump* [5]), iterating υ gives a derivation tree context witnessing $B(\boldsymbol{x}_1, \dots, \boldsymbol{x}_r) \vdash_G B(v_1^n \boldsymbol{x}_1 v_2^n, \dots, v_{2r-1}^n \boldsymbol{x}_r v_{2r}^n)$. Combining this with the rest of τ gives a derivation tree for $z(n) = u_1 v_1^n u_2 v_2^n \dots u_{2r} v_{2r}^n u_{2r+1} \in L(G)$ for every $n \in \mathbb{N}$, where $z(1) = z$. When some $\boldsymbol{x}_i$ occurs in β_j with $j \neq i$ (υ is an *uneven pump*), however, the result of iterating υ exhibits a complicated pattern that is not easy to describe.

A language L is said to be *k-iterative* if all but finitely many elements of L can be written in the form $u_1 v_1 u_2 v_2 \dots u_k v_k u_{k+1}$ so that $v_1 \dots v_k \neq \varepsilon$ and $u_1 v_1^n u_2 v_2^n \dots u_k v_k^n u_{k+1} \in L$ for all $n \in \mathbb{N}$. A language that is either finite or includes an infinite

$$
\begin{array}{rcl}
A(\varepsilon) & \leftarrow & \\
A(b\boldsymbol{x}_1) & \leftarrow & A(\boldsymbol{x}_1) \\
B(\boldsymbol{x}_1, \varepsilon) & \leftarrow & A(\boldsymbol{x}_1) \\
B(a\boldsymbol{x}_1, b\boldsymbol{x}_2) & \leftarrow & B(\boldsymbol{x}_1, \boldsymbol{x}_2) \\
C(\boldsymbol{x}_1, \boldsymbol{x}_2, \varepsilon) & \leftarrow & B(\boldsymbol{x}_1, \boldsymbol{x}_2) \\
C(\boldsymbol{x}_1, a\boldsymbol{x}_2, b\boldsymbol{x}_3) & \leftarrow & C(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3) \\
C(\boldsymbol{x}_1\$\boldsymbol{x}_2, \boldsymbol{x}_3, \varepsilon) & \leftarrow & C(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3) \\
D(\boldsymbol{x}_1\$\boldsymbol{x}_2, \boldsymbol{x}_3) & \leftarrow & C(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3) \\
D(\boldsymbol{x}_1, a\boldsymbol{x}_2) & \leftarrow & D(\boldsymbol{x}_1, \boldsymbol{x}_2) \\
S(\boldsymbol{x}_1\$\boldsymbol{x}_2) & \leftarrow & D(\boldsymbol{x}_1, \boldsymbol{x}_2)
\end{array}
\qquad
\begin{array}{rcl}
A(\varepsilon) & \leftarrow & \\
A(b\boldsymbol{x}_1) & \leftarrow & A(\boldsymbol{x}_1) \\
B(\boldsymbol{x}_1, \varepsilon) & \leftarrow & A(\boldsymbol{x}_1) \\
B(a\boldsymbol{x}_1, b\boldsymbol{x}_2) & \leftarrow & B(\boldsymbol{x}_1, \boldsymbol{x}_2) \\
C(\varepsilon, \varepsilon) & \leftarrow & \\
C(a\boldsymbol{x}_1, b\boldsymbol{x}_2) & \leftarrow & C(\boldsymbol{x}_1, \boldsymbol{x}_2) \\
D(\boldsymbol{x}_1\$\boldsymbol{y}_1\boldsymbol{x}_2, \boldsymbol{y}_2) & \leftarrow & B(\boldsymbol{x}_1, \boldsymbol{x}_2), C(\boldsymbol{y}_1, \boldsymbol{y}_2) \\
D(\boldsymbol{x}_1\$\boldsymbol{y}_1\boldsymbol{x}_2, \boldsymbol{y}_2) & \leftarrow & D(\boldsymbol{x}_1, \boldsymbol{x}_2), C(\boldsymbol{y}_1, \boldsymbol{y}_2) \\
E(\boldsymbol{x}_1, \boldsymbol{x}_2) & \leftarrow & D(\boldsymbol{x}_1, \boldsymbol{x}_2) \\
E(\boldsymbol{x}_1, a\boldsymbol{x}_2) & \leftarrow & E(\boldsymbol{x}_1, \boldsymbol{x}_2) \\
S(\boldsymbol{x}_1\$\boldsymbol{x}_2) & \leftarrow & E(\boldsymbol{x}_1, \boldsymbol{x}_2)
\end{array}
$$

Figure 4: Two grammars generating the same language.

k-iterative subset is said to be *weakly k-iterative*. (These terms are from [17, 18].) The possibility of an uneven pump explains the difficulty of establishing $2m$-iterativity of an m-MCFL. In 1991, Seki et al. [1] proved that every m-MCFL is weakly $2m$-iterative, but whether every m-MCFL is $2m$-iterative remained an open question for a long time, until Kanazawa et al. [19] negatively settled it in 2014 by exhibiting a (non-well-nested) 3-MCFL that is not k-iterative for any k. Earlier, Kanazawa [5] had shown that the language of a well-nested m-MCFG is always $2m$-iterative, and moreover that a 2-MCFL is always 4-iterative. The proof of this last pair of results was much more indirect than the proof of the pumping lemma for the context-free languages, and did not suggest a way of strengthening them to an Ogden-style theorem. Below, we show that there is indeed no reasonable way of doing so.

Let us say that a language L has the *weak Ogden property* if there is a natural number p such that for every $z \in L$ and $D \subseteq [1, |z|]$ with $|D| \geq p$, there are strings $u_1, \dots, u_{k+1}, v_1, \dots, v_k$ ($k \geq 1$) satisfying the following conditions:

1. $z = u_1 v_1 \dots u_k v_k u_{k+1}$,
2. $D \cap (u_1 v_1 \dots u_i [v_i] \dots u_k v_k u_{k+1}) \neq \varnothing$ for some $i \in [1, k]$, and
3. $u_1 v_1^n \dots u_k v_k^n u_{k+1} \in L$ for all $n \geq 0$.

The elements of D are referred to as *distinguished positions* in z.

Theorem 6. *There is an $L \in$ 3-MCFL(1) $\cap$ 2-MCFL(2) that does not satisfy the weak Ogden property.*

Proof. Let L be the set of all strings over the alphabet $\{a, b, \$\}$ that are of the form

$$a^{i_1} b^{i_0} \$ a^{i_2} b^{i_1} \$ a^{i_3} b^{i_2} \$ \dots \$ a^{i_n} b^{i_{n-1}} \tag{†}$$

for some $n \geq 3$ and $i_0, \dots, i_n \geq 0$. This language is generated by the non-branching 3-MCFG (left) as well as by the binary branching 2-MCFG (right) in Figure 4. Now suppose L has the weak Ogden property, and let p be the number satisfying the required conditions. Let

$$z = a\$a^2 b\$a^3 b^2 \$ \dots \$a^{p+1} b^p,$$

and let D consist of the positions in z occupied by \$. Note that $|D| = p$. By the weak Ogden property, there must be strings $u_1, \dots, u_{k+1}, v_1, \dots, v_k$ ($k \geq 1$) such that $z = u_1 v_1 \dots u_k v_k u_{k+1}$, at least one of $v_1, \dots, v_k$ contains an occurrence of \$, and $u_1 v_1^n \dots u_k v_k^n u_{k+1} \in L$ for all n. Without loss of generality, we may assume that $v_1, \dots, v_k$ are all nonempty strings. Let us write $z(n)$ for $u_1 v_1^n \dots u_k v_k^n u_{k+1}$. First note that none of $v_1, \dots, v_k$ can start in a and end in b, since otherwise $z(2)$ would contain ba as a factor and not be of the form (†). Let i be the greatest number such that v_i contains an occurrence of \$. Since none of $v_{i+1}, \dots, v_k$ contains an occurrence of \$, it is easy to see that $v_{i+1}, \dots, v_k$ are all in $a^+ \cup b^+$. We consider two cases, depending on the number of occurrences of \$ in v_i. Each case leads to a contradiction.

Case 1. v_i contains just one occurrence of \$. Then $v_i = x\$y$, where x is a suffix of $a^{j+1} b^j$ and y is a prefix of $a^{j+2} b^{j+1}$ for some $j \in [0, p-1]$. Note that $z(3)$ contains $\$yx\$yx\$$ as a factor. Since $z(3)$ is of the form (†), this means that $yx = a^l b^l$ for some $l \geq 0$.

Case 1.1 $l \leq j+1$. Then y must be a prefix of a^{j+1} and since x is a suffix of $a^{j+1}b^j$, it follows that $l \leq j$. Since $yu_{i+1}v_{i+1}\ldots u_k v_k u_{k+1}$ has $a^{j+2}b^{j+1}$ as a prefix and $v_{i+1},\ldots,v_k \in a^+ \cup b^+$, $\$yx\$yu_{i+1}v_{i+1}^2 \ldots u_k v_k^2 u_{k+1}$ has $\$a^l b^l \$ a^q b^r$ as a prefix for some $q \geq j+2$ and $r \geq j+1$. The string $\$a^l b^l \$ a^q b^r$ is a factor of $z(2)$ and since $z(2)$ is of the form (†), we must have $l \geq r$, but this contradicts $l \leq j$.

Case 1.2. $l \geq j+2$ In this case x must be a suffix of b^j and y must have $a^{j+2}b^2$ as a prefix, so $l = j+2$. Note that

$$\$yx\$yu_{i+1}v_{i+1}^2 \ldots u_k v_k^2 u_{k+1} = \$a^l b^l \$yu_{i+1}v_{i+1}^2 \ldots u_k v_k^2 u_{k+1}$$

is a suffix of $z(2)$, so either $yu_{i+1}v_{i+1}^2 \ldots u_k v_k^2 u_{k+1}$ equals $a^q b^l$ or has $a^q b^l \$$ as a prefix for some q. Since $l = j+2$ and $yu_{i+1}v_{i+1}\ldots u_k v_k u_{k+1}$ either equals $a^{j+2}b^{j+1}$ or has $a^{j+2}b^{j+1}\$$ as a prefix, it follows that there is some $h > i$ such that $v_h = b$ and $v_{i+1},\ldots,v_{h-1}$ are all in a^+. But then $z(3)$ will contain

$$\$yx\$yu_{i+1}v_{i+1}^3 \ldots u_k v_k^3 u_{k+1},$$

which must have

$$\$a^{j+2}b^{j+2}\$a^{q'}b^{j+3}$$

as a prefix for some q', contradicting the fact that $z(3)$ is of the form (†).

Case 2. v_i contains at least two occurrences of \$. Then we can write

$$v_i = x\$a^{l+1}b^l\$\ldots\$a^{m+1}b^m\$y,$$

where $1 \leq l \leq m \leq p-1$, x is a suffix of $a^l b^{l-1}$, and y is a prefix of $a^{m+2}b^{m+1}$. Since

$$\$a^{m+1}b^m\$yx\$a^{l+1}b^l\$$$

is a factor of $z(2)$, we must have

$$yx = a^l b^{m+1}.$$

Since y is a prefix of $a^{m+2}b^{m+1}$ and $l < m+2$, y must be a prefix of a^l. It follows that x has b^{m+1} as a suffix. But then b^{m+1} must be a suffix of $a^l b^{l-1}$, contradicting the fact that $l-1 < m+1$. □

Since Theorem 5 above implies that every language in Weir's control language hierarchy satisfies the weak Ogden property, we obtain the following corollary:[3]

Corollary 7. *There is a language in* 3-MCFL(1) ∩ 2-MCFL(2) *that lies outside of Weir's control language hierarchy.*

Previously, Kanazawa et al. [19] showed that Weir's control language hierarchy does not include 3-MCFL(2), but left open the question of whether the former includes the languages of well-nested MCFGs. The above corollary settles this question in the negative.

4. A Generalized Ogden's Lemma for a Subclass of the MCFGs

An easy way of ensuring that an m-MCFG G satisfies a generalized Ogden's lemma is to demand that whenever $B(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_r) \vdash_G B(\beta_1,\ldots,\beta_r)$, each $\boldsymbol{x}_i$ occurs in β_i. For example, the grammar in Example 1 satisfies this property. This is a rather strict requirement, however, and the resulting class of grammars does not seem to cover even the second level C_2 of the control language hierarchy. In this section, we show that a weaker condition implies a natural analogue of Ogden's [6] condition; we prove in the next section that the result covers the entire control language hierarchy.

[3] The language L in the proof of Theorem 6 was inspired by Lemma 5.4 of Greibach [11], where a much more complicated language was used to show that the range of a deterministic two-way finite-state transducer need not be *strongly iterative*. One can see that the language Greibach used is an 8-MCFL(1). In her proof, Greibach essentially relied on a stronger requirement imposed by her notion of strong iterativity, namely that in the factorization $z = u_1 v_1 \ldots u_k v_k u_{k+1}$, there must be some i such that u_i and u_{i+1} contain at least one distinguished position and v_i contains at least *two* distinguished positions. Strong iterativity is not implied by the condition in Theorem 5, so Greibach's lemma fell short of providing an example of a language in $\bigcup_m$ m-MCFL(1) that does not belong to Weir's hierarchy.

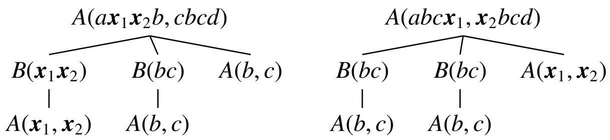

Figure 5: Decreasing and non-decreasing derivation tree contexts.

Let us say that a derivation tree context υ witnessing $A(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q) \vdash_G B(\beta_1, \ldots, \beta_r)$ is *decreasing* if there is a node labeled by an atom $C(\gamma_1, \ldots, \gamma_s)$ with $s < q$ along the path from the root of υ to the leaf labeled by $A(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)$; otherwise it is *non-decreasing*. (If $q > r$, there can be no non-decreasing derivation tree context witnessing $A(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q) \vdash_G B(\beta_1, \ldots, \beta_r)$.) An m-MCFG $G = (N, \Sigma, P, S)$ is *proper* if for each $A \in N^{(q)}$, whenever $A(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q) \vdash_G \upsilon : A(\alpha_1, \ldots, \alpha_q)$ for some non-decreasing derivation tree context υ, each $\boldsymbol{x}_i$ occurs in α_i.

Example 8. Consider the following 2-MCFG G:

$$S(\boldsymbol{x}_1\boldsymbol{x}_2) \leftarrow A(\boldsymbol{x}_1, \boldsymbol{x}_2) \qquad A(a\boldsymbol{y}\boldsymbol{x}_1, \boldsymbol{x}_2\boldsymbol{z}d) \leftarrow B(\boldsymbol{y}), B(\boldsymbol{z}), A(\boldsymbol{x}_1, \boldsymbol{x}_2)$$
$$A(b, c) \leftarrow \qquad B(\boldsymbol{x}_1\boldsymbol{x}_2) \leftarrow A(\boldsymbol{x}_1, \boldsymbol{x}_2)$$

We have, for example,

$$A(\boldsymbol{x}_1, \boldsymbol{x}_2) \vdash_G A(a\boldsymbol{x}_1\boldsymbol{x}_2b, cbcd), \qquad A(\boldsymbol{x}_1, \boldsymbol{x}_2) \vdash_G A(abc\boldsymbol{x}_1, \boldsymbol{x}_2bcd)$$

as witnessed by derivation tree contexts in Figure 5. The former is decreasing, while the latter is non-decreasing. It is easy to see that this grammar is proper.

Proposition 9. *The question of whether a given MCFG is proper is decidable.*

Proof. Given an m-MCFG, we first remove all useless nonterminals (i.e., nonterminals A such that there is no derivable ground atom $A(w_1, \ldots, w_q)$) by a standard technique. Let $G = (N, \Sigma, P, S)$ be the resulting m-MCFG without useless nonterminals.

Define a family of sets of functions $\mathcal{F}_{A,B} \subseteq \{1, \ldots, r\}^{\{1,\ldots,q\}}$ for $A \in N^{(q)}, B \in N^{(r)}$:

$$\begin{aligned}\mathcal{F}_{A,B} = \{ f \in \{1, \ldots, r\}^{\{1,\ldots,q\}} \mid\ & A(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q) \vdash_G \upsilon : B(\beta_1, \ldots, \beta_r), \\ & \boldsymbol{x}_i \text{ occurs in } \beta_{f(i)} \text{ for } i = 1, \ldots, q, \text{ and} \\ & \upsilon \text{ is a non-decreasing derivation tree context} \}.\end{aligned}$$

Clearly, the given MCFG is proper if and only if $\mathcal{F}_{A,A}$ contains just the identity function on $\{1, \ldots, q\}$ for all $A \in N^{(q)}$ and $q \leq m$.

The sets $\mathcal{F}_{A,B}$ form the least family of sets that satisfy the following closure conditions. For $A \in N^{(q)}$,

- $\mathcal{F}_{A,A}$ contains the identity function on $\{1, \ldots, q\}$, and
- if $r \geq q$, $C(\gamma_1, \ldots, \gamma_r) \leftarrow B_1(\boldsymbol{x}_{1,1}, \ldots, \boldsymbol{x}_{1,q_1}), \ldots, B_n(\boldsymbol{x}_{n,1}, \ldots, \boldsymbol{x}_{n,q_n})$ is a rule of G, $f \in \mathcal{F}_{A,B_i}$, and $\boldsymbol{x}_{i,j}$ occurs in $\gamma_{g(j)}$ for $j = 1, \ldots, q_i$, then $\mathcal{F}_{A,C}$ contains the composition $g \circ f$ defined by $(g \circ f)(k) = g(f(k))$.

Since there are only finitely many functions in $\{1, \ldots, r\}^{\{1,\ldots,q\}}$, the sets $\mathcal{F}_{A,B}$ are clearly computable. □

Theorem 10. *Let L be the language of a proper m-MCFG. There is a natural number p such that for every $z \in L$ and $D \subseteq [1, |z|]$ with $|D| \geq p$, there are strings $u_1, \ldots, u_{2m+1}, v_1, \ldots, v_{2m}$ satisfying the following conditions:*

(i) $z = u_1v_1 \ldots u_{2m}v_{2m}u_{2m+1}$.
(ii) *for some* $j \in [1, 2m]$,

$$D \cap (u_1v_1 \ldots [u_j]v_ju_{j+1}v_{j+1} \ldots u_{2m}v_{2m}u_{2m+1}) \neq \varnothing,$$
$$D \cap (u_1v_1 \ldots u_j[v_j]u_{j+1}v_{j+1} \ldots u_{2m}v_{2m}u_{2m+1}) \neq \varnothing,$$
$$D \cap (u_1v_1 \ldots u_jv_j[u_{j+1}]v_{j+1} \ldots u_{2m}v_{2m}u_{2m+1}) \neq \varnothing.$$

(iii) $|D \cap \bigcup_{i=1}^{m}(u_1 v_1 \dots u_{2i-1}[v_{2i-1} u_{2i} v_{2i}] \dots u_{2m} v_{2m} u_{2m+1})| \leq p$.

(iv) $u_1 v_1^n u_2 v_2^n \dots u_{2m} v_{2m}^n u_{2m+1} \in L$ *for all* $n \in \mathbb{N}$.

The third clause says that the m substrings $v_1 u_2 v_2, v_3 u_4 v_4, \dots, v_{2m-1} u_{2m} v_{2m}$ of z together contain at most p distinguished positions. The case $m = 1$ of Theorem 10 exactly matches the statement of Ogden's [6] original lemma (as does the case $k = 1$ of Theorem 5).

Proof of Theorem 10. Let $G = (N, \Sigma, P, S)$ be a proper m-MCFG. For a rule

$$A(\alpha_1, \dots, \alpha_q) \leftarrow B_1(\boldsymbol{x}_{1,1}, \dots, \boldsymbol{x}_{1,q_1}), \dots, B_n(\boldsymbol{x}_{n,1}, \dots, \boldsymbol{x}_{n,q_n}),$$

let its *weight* be the number of occurrences of terminal symbols in $\alpha_1, \dots, \alpha_q$ plus n, and let d be the maximal weight of a rule in P.

Let $z \in L$, $D \subseteq [1, |z|]$, and τ be a derivation tree for z. We refer to elements of D as *distinguished positions*. Note that it makes sense to ask whether a particular symbol occurrence in the atom $A(w_1, \dots, w_q)$ labeling a node ν of τ is in a distinguished position or not. This is because by Lemma 2, there are strings $z_1, \dots, z_{q+1}$ such that ν determines a derivation tree context witnessing $A(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q) \vdash_G S(z_1 \boldsymbol{x}_1 z_2 \boldsymbol{x}_2 \dots z_q \boldsymbol{x}_q z_{q+1})$, which tells us where in z each argument of $A(w_1, \dots, w_q)$ ends up. Henceforth, when the ground atom labeling a node ν contains a symbol occurrence in a distinguished position, we simply say that ν contains a distinguished position. We call a node ν a *B-node* (cf. [6]) if at least one of its children contains a distinguished position and ν contains more distinguished positions than any of its children. The *B-height* of a node ν is defined as the maximal B-height h of its children if ν is not a B-node, and $h + 1$ if ν is a B-node. (When ν has no children, its B-height is 0.)

Claim. A node ν of τ whose B-height is h can contain no more than d^{h+1} distinguished positions.

The proof of the claim is by induction on the (ordinary) height of ν. We distinguish two cases according to whether ν is a B-node.

Case 1. ν is not a B-node.

Case 1.1. ν has no children that contain a distinguished position. (This covers the case where ν is a leaf.) Then $h = 0$. If the rule used at ν has k occurrences of terminal symbols in its left-hand side, then ν can contain no more than $k \leq d = d^{h+1}$ distinguished positions.

Case 1.2. ν has exactly one child ν' that contains a distinguished position. Then the B-height of ν' is also h and ν contains the same number of distinguished positions as ν' does. By induction hypothesis, ν' contains no more than d^{h+1} distinguished positions.

Case 2. ν is a B-node. Then $h \geq 1$. Each of the children of ν has B-height $\leq h - 1$, and by induction hypothesis contains no more than d^h distinguished positions. If $A(\alpha_1, \dots, \alpha_q) \leftarrow B_1(\boldsymbol{x}_{1,1}, \dots, \boldsymbol{x}_{1,q_1}), \dots, B_n(\boldsymbol{x}_{n,1}, \dots, \boldsymbol{x}_{n,q_n})$ is the rule used at ν and k is the number of occurrences of terminal symbols in $(\alpha_1, \dots, \alpha_q)$, then ν can contain no more than $k + n \cdot d^h$ distinguished positions. By the definition of d, this number does not exceed $d \cdot d^h = d^{h+1}$.

This completes the proof of the claim.

Our goal is to find an h such that, when $|D| \geq d^{h+1}$, we can locate four nodes $\mu_1, \mu_2, \mu_3, \mu_4$, all of B-height $\leq h$, on the same path of τ that together decompose τ into $\upsilon_1, \upsilon_2, \upsilon_3, \upsilon_4, \tau'$ such that

$$A(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q) \vdash_G \upsilon_1 : S(z_1 \boldsymbol{x}_1 z_2 \boldsymbol{x}_2 \dots z_q \boldsymbol{x}_q z_{q+1}), \tag{1}$$

$$B(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q) \vdash_G \upsilon_2 : A(y_1 \boldsymbol{x}_1 y_2, \dots, y_{2q-1} \boldsymbol{x}_q y_{2q}), \tag{2}$$

$$B(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q) \vdash_G \upsilon_3 : B(v_1 \boldsymbol{x}_1 v_2, \dots, v_{2q-1} \boldsymbol{x}_q v_{2q}), \tag{3}$$

$$C(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q) \vdash_G \upsilon_4 : B(x_1 \boldsymbol{x}_1 x_2, \dots, x_{2q-1} \boldsymbol{x}_q x_{2q}), \tag{4}$$

$$\vdash_G \tau' : C(w_1, \dots, w_q), \tag{5}$$

where for some $j \in [1, 2q]$, each of x_j, v_j, y_j contains at least one distinguished position. Since

$$y_1 v_1 x_1 w_1 x_2 v_2 y_2, \dots, y_{2q-1} v_{2q-1} x_{2q-1} w_q x_{2q} v_{2q} y_{2q}$$

together can contain no more than d^{h+1} distinguished positions, this establishes the theorem, with $p = d^{h+1}$ and $u_1 = z_1y_1, u_2 = x_1w_1x_2, u_3 = y_2z_2y_3$, etc.

Let $M = \max\{\, |N^{(q)}| \mid 1 \leq q \leq m \,\}$ and let

$$\begin{aligned} g(1) &= 1, \\ g(q+1) &= h(q) + g(q) \qquad\qquad \text{for } 1 \leq q < m, \\ h(q) &= g(q) \cdot (2q \cdot (M+1) + 1). \end{aligned}$$

We show that

$$h = \sum_{q=1}^{m} h(q)$$

is the desired value for h.

By the "dimension" of a node, we mean the dimension of the nonterminal in the label of that node. Assume $|D| \geq d^{h+1}$. Then the root of τ has B-height $\geq h$, and τ must have a path that contains a node of each B-height $\leq h$. For each $i = 0, \dots, h$, from among the nodes of B-height i on that path, pick a node v_i of the lowest dimension.

By a *q-stretch*, we mean a contiguous subsequence of $v_0, v_1, \dots, v_h$ consisting entirely of nodes of dimension $\geq q$. We claim that some q-stretch contains more than $2q \cdot (M+1) + 1$ nodes of dimension q. For, suppose not. Then we can show by induction on q that the sequence of $h+1$ nodes $v_0, v_1, \dots, v_h$ contains no more than $g(q)$ maximal q-stretches and no more than $h(q)$ nodes of dimension q, which contradicts $h = \sum_{q=1}^{m} h(q)$. Since the entire sequence $v_0, v_1, \dots, v_h$ is a 1-stretch, the number of maximal 1-stretches is $g(1) = 1$, and the number of nodes of dimension 1 is $\leq 2 \cdot (M+1) + 1 = h(1)$. For $q < m$, each maximal $(q+1)$-stretch is contained in a maximal q-stretch, and if the last node of the former is not identical to the last node of the latter, then the former must be followed by a node of dimension q. By induction hypothesis, this means that the number of maximal $(q+1)$-stretches is $\leq g(q) + h(q) = g(q+1)$. Since each node of dimension $q+1$ belongs to a maximal $(q+1)$-stretch, the number of such nodes is $\leq g(q+1) \cdot (2(q+1) \cdot (M+1) + 1) = h(q+1)$.

So we have a q-stretch that contains nodes $v_{i_0}, \dots, v_{i_k}$ of dimension q for some $q \in [1, m]$, where $k = 2q \cdot (M+1) + 1$. Let A_n be the nonterminal in the label of v_{i_n}. By the definition of a q-stretch and the way the original sequence $v_0, \dots, v_h$ is defined, the nodes of τ that are neither below $v_{i_{n-1}}$ nor above v_{i_n} determine a non-decreasing derivation tree context witnessing $A_{n-1}(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q) \vdash_G A_n(x_{n,1}\boldsymbol{x}_1x_{n,2}, \dots, x_{n,2q-1}\boldsymbol{x}_qx_{n,2q})$ for some strings $x_{n,1}, \dots, x_{n,2q}$. Since there must be a B-node lying above $v_{i_{n-1}}$ and below or at v_{i_n}, at least one of $x_{n,1}, \dots, x_{n,2q}$ must contain a distinguished position. By the pigeon-hole principle, there is a $j \in [1, 2q]$ such that $\{\, n \in [1, k] \mid x_{n,j}$ contains a distinguished position $\}$ has at least $M+2$ elements. This means that we can pick three elements n_1, n_2, n_3 from this set so that $n_1 < n_2 < n_3$ and $A_{n_1} = A_{n_2}$. Letting $\mu_1 = v_{i_0}, \mu_1 = v_{i_{n_1}}, \mu_2 = v_{i_{n_2}}, \mu_3 = v_{i_{n_3}}$, we see that (2), (3), (4) hold with $C = A_{i_0}, B = A_{i_{n_1}} = A_{i_{n_2}}, A = A_{i_{n_3}}$ and x_j, v_j, y_j all containing a distinguished position, as desired. □

Let us write m-MCFL$_{\text{prop}}$ for the family of languages generated by proper m-MCFGs. Using standard techniques (cf. Theorem 3.9 of [1]), we can easily establish the following:

Proposition 11. *For each $m \geq 1$, m-MCFL$_{\text{prop}}$ is a substitution-closed full abstract family of languages.*

5. Relation to the Control Language Hierarchy

Kanazawa and Salvati [16] showed $C_k \subseteq 2^{k-1}$-MCFL for each k through a tree grammar generating the derivation trees of a level k control grammar (G, C), noting that the tree language in question can be obtained from the monadic tree representation of C by linear tree homomorphism, the tree analogue of the Kleene star operation, and intersection with regular tree language. In fact, detour through tree languages is not necessary—a level k control language can be obtained from a level $k-1$ control language by certain string language operations. It is easy to see that the family $\bigcup_m m$-MCFL$_{\text{prop}}$ is closed under those operations.

Let us sketch the idea using Example 4. Under the tree language approach, monadic trees representing strings of the form $\pi_1^n \pi_2^n \pi_3$ undergo a linear and nondeleting tree homomorphism:

$$\pi_1(x) \mapsto S^{(4)}(a, x, \bar{a}, \square), \qquad \pi_2(x) \mapsto S^{(4)}(b, x, \bar{b}, \square), \qquad \pi_3 \mapsto S^{(1)}\varepsilon,$$

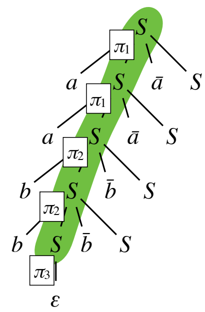

Figure 6: A fragment of a derivation tree of a control grammar, with rule labels.

where $S^{(4)}$ and $S^{(1)}$ are rank 4 and rank 1 symbols, respectively, and $a, \bar{a}, b, \bar{b}$ are rank 1 symbols. The set of derivation trees of the control grammar is then obtained from the set L consisting of the output of this tree homomorphism by iterating substitution $\Box \leftarrow L$ on the set $\{\Box\}$ (resulting in $L^{\Box,*}$, the $\Box$-*iteration* [20] or *closure* [21] of L) and throwing away trees that contain $\Box$.

Staying inside the realm of string languages, we can start by applying a homomorphic replication $\langle(1,R), h_1, h_2\rangle$ to the control set $C = \{\, \pi_1^n \pi_2^n \pi_3 \mid n \in \mathbb{N} \,\}$, obtaining

$$\langle(1,R), h_1, h_2\rangle(C) = \{\, h_1(w)(h_2(w))^R \mid w \in C \,\},$$

where h_1 and h_2 are homomorphisms defined by

$$\begin{aligned} h_1(\pi_1) &= a, & h_2(\pi_1) &= S\,\bar{a}, \\ h_1(\pi_2) &= b, & h_2(\pi_2) &= S\,\bar{b}, \\ h_1(\pi_3) &= \varepsilon, & h_2(\pi_3) &= \varepsilon. \end{aligned}$$

For instance, $\pi_1^2\pi_2^2\pi_3$ is mapped to $aabb\bar{b}S\,\bar{b}S\,\bar{a}S\,\bar{a}S$. (This string is the yield of the tree in Figure 6.) Iterating the substitution $S \leftarrow \langle(1,R), h_1, h_2\rangle(C)$ on $\{S\}$ and then throwing away strings that contain S gives the language of the control grammar of this example.

In general, let π be a production $A \to w_0 B_1 w_1 \ldots B_n w_n$ of a labeled distinguished grammar $G = (N, \Sigma, P, S, f)$. If $f(\pi) = i \in [1, n]$, we let

$$h_1(\pi) = w_0 B_1 w_1 \ldots B_{i-1} w_{i-1}, \qquad h_2(\pi) = w_n B_n \ldots w_{i+1} B_{i+1} w_i,$$

and if $f(\pi) = 0$, we let

$$h_1(\pi) = w_0 B_1 w_1 \ldots B_n w_n, \qquad h_2(\pi) = \varepsilon.$$

The control set $C \subseteq P^*$ is first intersected with a local set so as to ensure consistency of nonterminals in adjacent productions, and then partitioned into sets C_A indexed by nonterminals, with C_A holding only those strings whose first symbol is a production that has A on its left-hand side. More precisely, if π and π' are productions in P, let us say that π' *can follow* π if $f(\pi) = j \geq 1$ and the jth nonterminal on the right-hand side of π coincides with the left-hand side nonterminal of π'. Define

$$\begin{aligned} C_A = \{\, \pi_1 \ldots \pi_n \in C \mid\ & n \geq 1, \text{the left-hand side of } \pi_1 \text{ is } A, \\ & \pi_{i+1} \text{ can follow } \pi_i \text{ for each } i = 1, \ldots, n-1, \text{ and} \\ & f(\pi_n) = 0 \,\}. \end{aligned} \tag{6}$$

For each $A \in N$, the set C_A can be obtained from C by intersection with a local (and hence regular) set. Let

$$L_A = \langle(1,R), h_1, h_2\rangle(C_A) \tag{7}$$

for each $A \in N$. The final operation is iterating the substitution $[A \leftarrow L_A]_{A \in N}$ on $\{S\}$ and throwing away strings containing nonterminals.[4] This can be expressed equivalently in terms of a nested iterated substitution:

$$L(G, C) = \sigma^{\infty}(\{S\}) \cap \Sigma^*, \quad \text{where } \sigma = [A \leftarrow L_A \cup \{A\}]_{A \in N}. \tag{8}$$

That equation (8) should hold is easy to see. For each $A \in N$, we have the following equivalences:

- $A \overset{\xi}{\Longrightarrow}_G \alpha$ if and only if $\xi \in C_A$ and $\alpha = \langle (1, R), h_1, h_2 \rangle(\xi)$,
- $A \overset{C}{\Longrightarrow}_G \alpha$ if and only if $\alpha \in L_A$,
- $A \Rightarrow^*_{(G,C)} \alpha$ if and only if $\alpha \in \sigma^{\infty}(\{A\})$.

Thus, $L(G, C)$ can be obtained from C by intersection with regular sets, homomorphic replication, and nested iterated substitution.

Lemma 12. *If $L \in m$-MCFL$_{\text{prop}}$, $k \geq 1$, $\rho \in \{1, R\}^{\{1,\dots,k\}}$, and $h_1, \dots, h_k$ are homomorphisms, then the language $\langle \rho, h_1, \dots, h_k \rangle(L)$ belongs to km-MCFL$_{\text{prop}}$.*

Example 1 in Section 2.1 illustrates Lemma 12 with $m = 1$, $L = D_1^*$, $\rho = (1, R)$, and h_1, h_2 both equal to the identity function.

of Lemma 12. We only prove the lemma for $\rho = (1, R)$. The general case is similar.

Let $G = (N, \Sigma, P, S)$ be a proper m-MCFG for L and h_1, h_2 be homomorphisms from Σ^* to Γ^*. Define a $2m$-MCFG $G' = (N', \Gamma, P', S')$ as follows:

- $N' = \{S'\} \cup \bigcup_{i \leq m} (N')^{(2i)}$, where $S' \in (N')^{(1)}$ and for each $i \leq m$, $(N')^{(2i)} = N^{(i)}$.
- P' contains the rule

  $$S'(\boldsymbol{x}_1, \boldsymbol{x}_2) \leftarrow S(\boldsymbol{x}_1, \boldsymbol{x}_2),$$

  and for each rule π in P of the form

  $$A(\alpha_1, \dots, \alpha_q) \leftarrow B_1(\boldsymbol{x}_{1,1}, \dots, \boldsymbol{x}_{1,q_1}), \dots, B_n(\boldsymbol{x}_{n,1}, \dots, \boldsymbol{x}_{n,q_n}),$$

  the rule π' of the form

  $$A(h_1(\alpha_1), \dots, h_1(\alpha_q), (h_2(\alpha'_q))^R, \dots, (h_2(\alpha'_1))^R) \leftarrow \\ B_1(\boldsymbol{x}_{1,1}, \dots, \boldsymbol{x}_{1,q_1}, \boldsymbol{x}'_{1,q_1}, \dots, \boldsymbol{x}'_{1,1}), \dots, B_n(\boldsymbol{x}_{n,1}, \dots, \boldsymbol{x}_{n,q_n}, \boldsymbol{x}'_{n,q_n}, \dots, \boldsymbol{x}'_{n,1}),$$

  where for $l = 1, \dots, q$, α'_l is the result of replacing each $\boldsymbol{x}_{i,j}$ by $\boldsymbol{x}'_{i,j}$ in α_l, and the homomorphisms h_1, h_2 are extended to homomorphisms from $(\Sigma \cup \mathcal{X})^*$ to $(\Gamma \cup \mathcal{X})^*$ by $h_1(\boldsymbol{x}) = h_2(\boldsymbol{x}) = \boldsymbol{x}$ for all variables $\boldsymbol{x} \in \mathcal{X}$.

It is clear that the bijection $\pi \mapsto \pi'$ from P to $P' - \{S'(\boldsymbol{x}_1\boldsymbol{x}_2) \leftarrow S(\boldsymbol{x}_1, \boldsymbol{x}_2)\}$ puts the derivation trees of G and those of G' in one-to-one correspondence, and $L(G') = \langle (1, R), h_1, h_2 \rangle(L)$.

To see that G' is proper, note that for each $A \in N$, a non-decreasing derivation tree context of G' for $A(\gamma_1, \dots, \gamma_{2q})$ (with an assumption $A(\boldsymbol{x}_1, \dots, \boldsymbol{x}_{2q})$) is mapped to a non-decreasing derivation tree context of G for $A(\alpha_1, \dots, \alpha_q)$ (with an assumption $A(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q)$) such that for $i = 1, \dots, q$, $\gamma_i = h_1(\alpha_i)$ and $\gamma_{2q-i+1} = (h_2(\alpha'_i))^R$, where α'_i is the result of replacing $\boldsymbol{x}_j$ by $\boldsymbol{x}_{2q-j+1}$ for each $j = 1, \dots, q$. Since G is proper, $\boldsymbol{x}_i$ must occur in α_i for $i = 1, \dots, q$, which implies that $\boldsymbol{x}_i$ occurs in γ_i and $\boldsymbol{x}_{2q-i+1}$ occurs in γ_{2q-i+1}. □

The proof of the next lemma is similar to that of closure under substitution.

[4] This last step may be thought of as the fixed point computation of a "context-free grammar" with an infinite set of rules $\{ A \to \alpha \mid A \in N, \alpha \in L_A \}$.

Lemma 13. *The family m-MCFL$_{\text{prop}}$ is closed under nested iterated substitution.*

Proof. Let $G = (N, \Sigma, P, S)$ and $G_c = (N_c, \Sigma, P_c, S_c)$ for each $c \in \Sigma$ be proper m-MCFGs. We may assume without loss of generality that no two of these grammars share any nonterminals. Let $L = L(G)$ and $L_c = L(G_c)$. Assume that $c \in L_c$ for each $c \in \Sigma$ and let $\sigma = [c \leftarrow L_c]_{c\in\Sigma}$. Then σ^∞ is a nested iterated substitution. Our goal is to show that $\sigma^\infty(L)$ is in m-MCFL$_{\text{prop}}$.

We first modify G and G_c ($c \in \Sigma$) slightly without changing the generated languages. For each $d \in \Sigma$, introduce a new nonterminal A_d of dimension 1. For each rule π of G and G_c ($c \in \Sigma$), let π' be the result of replacing the occurrences of terminals in the left-hand side of π by distinct variables and adding appropriate atoms of the form $A_d(\boldsymbol{x})$ to the right-hand side of π, where $\boldsymbol{x}$ is a variable that replaced an occurrence of d. For example, if π is

$$A(a\boldsymbol{x}_1 b, \bar{b}c\boldsymbol{x}_2\bar{a}c) \leftarrow A(\boldsymbol{x}_1, \boldsymbol{x}_2),$$

then π' is

$$A(\boldsymbol{y}\boldsymbol{x}_1\boldsymbol{z}, \boldsymbol{w}\boldsymbol{v}\boldsymbol{x}_2\boldsymbol{u}\boldsymbol{t}) \leftarrow A(\boldsymbol{x}_1, \boldsymbol{x}_2), A_a(\boldsymbol{y}), A_b(\boldsymbol{z}), A_{\bar{b}}(\boldsymbol{w}), A_c(\boldsymbol{v}), A_{\bar{a}}(\boldsymbol{u}), A_c(\boldsymbol{t}).$$

Let

$$G' = (N \cup \{ A_d \mid d \in \Sigma \}, \Sigma, \{ \pi' \mid \pi \in P \} \cup \{ A_d(d) \leftarrow \mid d \in \Sigma \}, S),$$
$$G'_c = (N_c \cup \{ A_d \mid d \in \Sigma \}, \Sigma, \{ \pi' \mid \pi \in P_c \} \cup \{ A_d(d) \leftarrow \mid d \in \Sigma \}, S_c).$$

It is clear that G' and G'_c ($c \in \Sigma$) are proper m-MCFGs, $L(G') = L(G)$, and $L(G'_c) = L(G_c)$ for $c \in \Sigma$.

Now define

$$\widehat{G} = (\widehat{N}, \Sigma, \widehat{P}, S),$$
$$\widehat{N} = N \cup \bigcup_{c\in\Sigma} N_c \cup \{ A_d \mid d \in \Sigma \},$$
$$\widehat{P} = \{ \pi' \mid \pi \in P \cup \bigcup_{c\in\Sigma} P_c \} \cup \{ A_d(d) \leftarrow \mid d \in \Sigma \} \cup \{ A_d(\boldsymbol{x}) \leftarrow S_d(\boldsymbol{x}) \mid d \in \Sigma \}.$$

The grammar $\widehat{G}$ is the result of combining G', G'_c ($c \in \Sigma$) into one grammar and adding new rules $A_d(\boldsymbol{x}) \leftarrow S_d(\boldsymbol{x})$ for $d \in \Sigma$. We show that $\widehat{G}$ is proper and $L(\widehat{G}) = \sigma^\infty(L)$.

To show that $\widehat{G}$ is proper, consider a non-decreasing derivation tree context υ witnessing $A(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q) \vdash_{\widehat{G}} A(\alpha_1, \dots, \alpha_q)$, where $q \geq 2$. Since υ is non-decreasing, it cannot be decomposed into two derivation tree contexts witnessing

$$A(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q) \vdash_{\widehat{G}} A_d(\delta),$$
$$A_d(\boldsymbol{z}) \vdash_{\widehat{G}} A(\alpha'_1, \dots, \alpha'_q)$$

for any $d \in \Sigma$. It follows that every fact of the form $A_d(\gamma)$ derived in the course of υ is ground. Replace every such fact in υ by $A_d(d)$, deleting all facts used to derive it. The result must be a non-decreasing derivation tree context υ' witnessing $A(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q) \vdash_{\widehat{G}} A(\gamma_1, \dots, \gamma_q)$ such that $\boldsymbol{x}_i$ occurs in γ_j if and only if $\boldsymbol{x}_i$ occurs in α_j. Since no rule of the form $A_c(\boldsymbol{x}) \leftarrow S_c(\boldsymbol{x})$ is used in υ', it is a derivation tree context in G' or in some G'_c. Since G' and G'_c are proper, $\boldsymbol{x}_i$ must occur in γ_i, and hence in α_i, for $i = 1, \dots, q$. This shows that $\widehat{G}$ is proper.

To show that $L(\widehat{G}) = \sigma^\infty(L)$, we first note

$$u_1 c u_2 \in L(\widehat{G}) \text{ and } v \in L_c \text{ imply } u_1 v u_2 \in L(\widehat{G}). \tag{9}$$

For, suppose $\vdash_{\widehat{G}} S(u_1 c u_2)$ and $v \in L_c$. Since the only way c can be introduced into a derivation is by the rule $A_c(c) \leftarrow$, we get

$$A_c(\boldsymbol{x}) \vdash_{\widehat{G}} S(u_1 \boldsymbol{x} u_2)$$

by Lemma 2. Since $\vdash_{\widehat{G}} S_c(v)$ and $A_c(\boldsymbol{x}) \leftarrow S_c(\boldsymbol{x})$ is a rule of $\widehat{G}$, we obtain $\vdash_{\widehat{G}} S(uvu_2)$.

We can easily deduce $\sigma^\infty(L) \subseteq L(\widehat{G})$ from (9). It suffices to prove that $w \in \sigma^n(L)$ implies $w \in L(\widehat{G})$ by induction on n. If $w \in \sigma^0(L) = L$, then $\vdash_{G'} S(w)$ and so $\vdash_{\widehat{G}} S(w)$. If $w \in \sigma^{n+1}(L) = \sigma(\sigma^n(L))$, then there are $c_1, \dots, c_l \in \Sigma$ and $w_1, \dots, w_l \in \Sigma^*$ such that

- $c_1 \dots c_l \in \sigma^n(L)$,
- $w_i \in L_{c_i}$ for $i = 1, \dots, l$,
- $w = w_1 \dots w_l$.

By induction hypothesis, $c_1 \dots c_l \in L(\widehat{G})$. Then $w = w_1 \dots w_l \in L(\widehat{G})$ follows from (9).

It remains to show $L(\widehat{G}) \subseteq \sigma^\infty(L)$. For this, we prove that $\vdash_{\widehat{G}} S(w)$ implies $w \in \sigma^\infty(L)$ by induction on the number k of times rules of the form $A_d(\boldsymbol{x}) \leftarrow S_d(\boldsymbol{x})$ ($d \in \Sigma$) are used in the derivation tree τ of $\widehat{G}$ for $S(w)$. If $k = 0$, then τ is a derivation tree of G', and so $w \in L \subseteq \sigma^\infty(L)$. If $k > 0$, pick one of the lowest nodes ν of τ which is derived using a rule of the form $A_d(\boldsymbol{x}) \leftarrow S_d(\boldsymbol{x})$. Then there are strings u_1, u_2, v such that

- $w = u_1 v u_2$,
- $\vdash_{G'_d} S_d(v)$, and
- the part of τ that remains after deleting all nodes below ν determines a derivation tree context υ witnessing $A_d(\boldsymbol{x}) \vdash_{\widehat{G}} S(u_1 \boldsymbol{x} u_2)$.

The derivation tree context υ together with the rule $A_d(d) \leftarrow$ forms a derivation tree for $\vdash_{G'} S(u_1 d u_2)$ containing $k-1$ instances of rules of the form $A_c(\boldsymbol{x}) \leftarrow S_c(\boldsymbol{x})$. By induction hypothesis, $u_1 d u_2 \in \sigma^n(L)$ for some n. Since $v \in L_d$ and $c \in L_c$ for all $c \in \Sigma$, $u_1 v u_2 \in \sigma(\sigma^n(L)) = \sigma^{n+1}(L) \subseteq \sigma^\infty(L)$. □

Theorem 14. *For each $k \geq 2$, $C_k \subsetneq 2^{k-1}$-MCFL$_{\text{prop}}$.*

Proof. The inclusion $C_k \subseteq 2^{k-1}$-MCFL$_{\text{prop}}$ for each $k \geq 1$ is proved by induction on k. For the induction basis, we have $C_1 = \text{CFL}$, which clearly equals 1-MCFL$_{\text{prop}}$. Now let $k \geq 1$ and $L \in C_{k+1}$. Then $L = L(G, C)$ for some labeled distinguished grammar $G = (N, \Sigma, P, S, f)$ and some $C \in \mathscr{P}(P^*) \cap C_k$. For each nonterminal A of G, let C_A and L_A be as defined by (6) and (7). By induction hypothesis, $C \in 2^{k-1}$-MCFL$_{\text{prop}}$, and since 2^{k-1}-MCFL$_{\text{prop}}$ is closed under intersection with regular sets, each C_A belongs to 2^{k-1}-MCFL$_{\text{prop}}$ as well. By Lemma 12, then, $L_A \in 2^k$-MCFL$_{\text{prop}}$. Given the equation (8), Lemma 13 and closure under intersection with regular sets (again) imply $L \in 2^k$-MCFL$_{\text{prop}}$.

The properness of the inclusion for $k \geq 2$ is again witnessed by the language RESP$_{2^{k-1}}$, since Palis and Shende's [8] theorem (Theorem 5 above) implies that RESP$_{2^{k-1}} = \{\, a_1^m a_2^m b_1^n b_2^n \dots a_{2^k-1}^m a_{2^k}^m b_{2^k-1}^n b_{2^k}^n \mid m, n \in \mathbb{N} \,\}$ does not belong to C_k, while RESP$_{2^{k-1}}$ is generated by the following proper 2^{k-1}-MCFG:

$$\begin{aligned}
S(\boldsymbol{x}_1 \boldsymbol{y}_1 \cdots \boldsymbol{x}_{2^{k-1}} \boldsymbol{y}_{2^{k-1}}) &\leftarrow A(\boldsymbol{x}_1, \dots, \boldsymbol{x}_{2^{k-1}}), B(\boldsymbol{y}_1, \dots, \boldsymbol{y}_{2^{k-1}}),\\
A(\varepsilon, \dots, \varepsilon) &\leftarrow ,\\
A(a_1 \boldsymbol{x}_1 a_2, \dots, a_{2^k-1} \boldsymbol{x}_{2^{k-1}} a_{2^k}) &\leftarrow A(\boldsymbol{x}_1, \dots, \boldsymbol{x}_{2^{k-1}}),\\
B(\varepsilon, \dots, \varepsilon) &\leftarrow ,\\
B(b_1 \boldsymbol{x}_1 b_2, \dots, b_{2^k-1} \boldsymbol{x}_{2^{k-1}} b_{2^k}) &\leftarrow B(\boldsymbol{x}_1, \dots, \boldsymbol{x}_{2^{k-1}}).
\end{aligned}$$

(Here, A and B are nonterminals of dimension 2^{k-1}.) □

For $k = 2$, the language $\{\, w\#w \mid w \in D_1^* \,\}$ (mentioned in Section 2.1) also witnesses the separation of 2-MCFL$_{\text{prop}}$ from C_2, since it is known that every language in C_2 has a well-nested 2-MCFG. I currently do not see how to settle the question of whether the inclusion of $\bigcup_k C_k$ in $\bigcup_m m$-MCFL$_{\text{prop}}$ is strict.

6. A Refined Pumping Lemma for Well-Nested MCFGs

The pumping lemma of [5] simply stated that every well-nested m-MCFL L is $2m$-iterative, which means that every sufficiently long string z in L can be written in the form $z = u_1 v_1 u_2 v_2 \dots u_{2m} v_{2m} u_{2m+1}$ so that $u_1 v_1^n u_2 v_2^n \dots u_{2m} v_{2m}^n u_{2m+1} \in L$ for all $n \in \mathbb{N}$. From Theorem 6, we know that for $m \geq 3$, we cannot constrain the locations of the substrings $v_1, v_2, \dots, v_{2m}$ within z so that they include at least one of arbitrarily picked positions. Given Theorem 10, a question that naturally arises is whether the lemma can be augmented with a bound on the combined length of the strings

$v_{2i-1}u_{2i}v_{2i}$ ($i = 1, \dots, m$).[5] This would make a natural generalization of the pumping lemma for context-free languages as originally stated by Bar-Hillel et al. [9]. Below, we give a proof of such a strengthening of the pumping lemma for well-nested m-MCFLs.

The proof is a modification of the proof in [5], which proceeded by induction on m. In the modified proof, we need to work with a statement that is stronger than the theorem we wish to prove, which necessitates a slightly generalized notion of a (well-nested) m-MCFG. By a *non-strict* m-MCFG, we mean an MCFG in which

- the initial nonterminal may have an arbitrary dimension, but cannot appear in the right-hand side of a rule, and
- the dimension of every other nonterminal is less than or equal to m.

A non-strict m-MCFG is just an ordinary MCFG except for the choice of the initial nonterminal, so we can continue to use notions like derivation trees, derivation tree contexts, well-nestedness of rules, etc., with their meanings unchanged. If $G = (N, \Sigma, P, S)$ is a non-strict m-MCFG and the dimension of its initial nonterminal S is l, then its language is a set of l-tuples of strings defined by $L(G) = \{ (w_1, \dots, w_l) \mid \vdash_G S(w_1, \dots, w_l) \}$.

Given a non-strict m-MCFG G, an *even m-pump* is a derivation tree context υ such that

- υ contains more than one node,
- $B(\boldsymbol{x}_1, \dots, \boldsymbol{x}_m) \vdash_G \upsilon : B(\beta_1, \dots, \beta_m)$, and
- $\boldsymbol{x}_i$ occurs in β_i for $i = 1, \dots, m$.

We say that a derivation tree τ of G *contains* an even m-pump υ if $\tau = \upsilon'[\upsilon[\tau']]$ for some derivation tree context υ' and derivation tree τ'. An even m-pump υ of G is *redundant* if $B(\boldsymbol{x}_1, \dots, \boldsymbol{x}_m) \vdash_G \upsilon : B(\boldsymbol{x}_1, \dots, \boldsymbol{x}_m)$. Clearly, every tuple $(w_1, \dots, w_l) \in L(G)$ has a derivation tree that does not contain any redundant even m-pumps.

A key fact about well-nested MCFGs used by [5] can be generalized to the following lemma, which now applies to non-strict well-nested MCFGs:

Lemma 15. *Let $m \geq 2$ and let G be a non-strict well-nested m-MCFG with initial nonterminal of dimension l. There is a non-strict well-nested $(m-1)$-MCFG $\widetilde{G}$ such that*

$$L(\widetilde{G}) = \{ (z_1, \dots, z_l) \in L(G) \mid G \text{ has a derivation tree for } (z_1, \dots, z_l) \text{ containing no even } m\text{-pump} \}.$$

This can be established by a series of lemmas, exactly like in [5]. The only difference is that in [5], in the last transformation used to obtain $\widetilde{G}$, all nonterminals of dimension $\geq m$ become useless and get eliminated, whereas here, the initial nonterminal is always retained.

Lemma 16. *Let $m \geq 1$ and let G be a non-strict well-nested m-MCFG with initial nonterminal of dimension l. There is a natural number p such that for every $(z_1, \dots, z_l) \in L(G)$ with $|z_1 \dots z_l| \geq p$, there exist an l-tuple of patterns $(\gamma_1, \dots, \gamma_l)$ with variables $\boldsymbol{x}_1, \dots, \boldsymbol{x}_m$ and strings $w_1, \dots, w_m, v_1, v_2, \dots, v_{2m}$ satisfying the following conditions:*

(i) $(z_1, \dots, z_l) = (\gamma_1, \dots, \gamma_l)[v_1 w_1 v_2/\boldsymbol{x}_1, \dots, v_{2m-1} w_m v_{2m}/\boldsymbol{x}_m]$.
(ii) $|v_1 v_2 \dots v_{2m}| > 0$.
(iii) $\sum_{i=1}^{m} |v_{2i-1} w_i v_{2i}| \leq p$.
(iv) $(\gamma_1, \dots, \gamma_l)[v_1^n w_1 v_2^n/\boldsymbol{x}_1, \dots, v_{2m-1}^n w_m v_{2m}^n/\boldsymbol{x}_m] \in L$ *for all* $n \in \mathbb{N}$.

Proof. The theorem is proved by induction on m.

Induction basis. $m = 1$. Let $G = (N, \Sigma, P, S)$ be a non-strict well-nested 1-MCFG. There must be numbers n_1 and n_2 such that whenever

$$S(\beta_1, \dots, \beta_l) \leftarrow B_1(\boldsymbol{x}_1), \dots, B_n(\boldsymbol{x}_n) \tag{10}$$

[5]Compare Groenink's [22] notion of *k-pumpability*, which included a bound on the length of v_i.

is a rule of G, the number of occurrences of terminal symbols in $(\beta_1, \ldots, \beta_l)$ does not exceed n_1, and $n \leq n_2$. For every non-initial nonterminal B of G, the set of strings z such that $\vdash_G B(z)$ is a context-free language. By the pumping lemma for context-free languages [9], for each B, there is a number p_B such that whenever $\vdash_G B(z)$ and $|z| \geq p_B$, there are strings u_1, u_2, u_3, v_1, v_2 such that $z = u_1 v_1 u_2 v_2 u_3$, $|v_1 v_2| > 0$, $|v_1 u_2 v_3| \leq p_B$, and $\vdash_G B(u_1 v_1^n u_2 v_2^n u_3)$ for all $n \in \mathbb{N}$. Let

$$p = n_1 + n_2 \cdot \max\{\, p_B \mid B \in N^{(1)} \,\}.$$

Now let $\vdash_G S(z_1, \ldots, z_l)$ and $|z_1 \ldots z_l| \geq p$, and suppose that the last step of the derivation of $S(z_1, \ldots, z_l)$ is by a rule of the form (10). Then there must be $y_1, \ldots, y_n \in \Sigma^*$ and $i \in [1, n]$ such that

$$\begin{aligned} &\vdash_G B_j(y_j) \quad \text{for } j = 1, \ldots, n, \\ &(z_1, \ldots, z_l) = (\beta_1, \ldots, \beta_l)[y_1/\boldsymbol{x}_1, \ldots, y_n/\boldsymbol{x}_n] \\ &|y_i| \geq p_{B_i}. \end{aligned}$$

There are strings u_1, u_2, u_3, v_1, v_2 such that $y_i = u_1 v_1 u_2 v_2 u_3$, $|v_1 v_2| > 0$, $|v_1 u_2 v_2| \leq p_{B_i}$, and $\vdash_G B_i(u_1 v_1^n u_2 v_2^n u_2)$ for all $n \in \mathbb{N}$. Let

$$(\gamma_1, \ldots, \gamma_l) = (\beta_1, \ldots, \beta_l)[y_1/\boldsymbol{x}_1, \ldots, y_{i-1}/\boldsymbol{x}_{i-1}, u_1 \boldsymbol{x}_1 u_3/\boldsymbol{x}_i, y_{i+1}/\boldsymbol{x}_{i+1}, \ldots, y_n/\boldsymbol{x}_n].$$

The tuple $(\gamma_1, \ldots, \gamma_l)$ and the strings u_2, v_1, v_2 satisfy the conditions (i)–(iv).

Induction step. Assume $m \geq 2$ and let $G = (N, \Sigma, P, S)$ be a non-strict well-nested m-MCFG. For each nonterminal $A \in N^{(m)}$, let $G_A = ((N - \{S\}) \cup \{S_A\}, \Sigma, P_A, S_A)$, where S_A is a new initial nonterminal of dimension m and P_A consists of all rules of P not involving S together with new rules of the form

$$S_A(\alpha_1, \ldots, \alpha_m) \leftarrow B_1(\boldsymbol{x}_{1,1}, \ldots, \boldsymbol{x}_{1,q_1}), \ldots, B_n(\boldsymbol{x}_{n,1}, \ldots, \boldsymbol{x}_{n,q_n}) \tag{11}$$

such that

$$A(\alpha_1, \ldots, \alpha_m) \leftarrow B_1(\boldsymbol{x}_{1,1}, \ldots, \boldsymbol{x}_{1,q_1}), \ldots, B_n(\boldsymbol{x}_{n,1}, \ldots, \boldsymbol{x}_{n,q_n}) \tag{12}$$

is a rule of P. By Lemma 15, there is a non-strict well-nested $(m-1)$-MCFG $\widetilde{G}$ generating all l-tuples of strings for which G has a derivation tree containing no even m-pump. Likewise, for each $A \in N^{(m)}$, we have a non-strict well-nested $(m-1)$-MCFG $\widetilde{G}_A$ generating all m-tuples of strings for which G_A has a derivation tree containing no even m-pump. By induction hypothesis, these non-strict well-nested $(m-1)$-MCFGs satisfy the conditions of the theorem, with $m-1$ in place of m. Let $\tilde{p}$ and $\tilde{p}_A$ $(A \in N^{(m)})$ be the natural numbers associated with these grammars by the theorem. Let $p = \max(\{\tilde{p}\} \cup \{\, \tilde{p}_A \mid A \in N^{(m)} \,\})$.

Now take an arbitrary tuple $(z_1, \ldots, z_l) \in L(G)$ with $|z_1 \ldots z_l| \geq p$. We distinguish two cases according to whether G has a derivation tree for $(z_1, \ldots, z_l)$ containing no even m-pump.

Case 1. G has a derivation tree for $(z_1, \ldots, z_l)$ containing no even m-pump. Then $(z_1, \ldots, z_l)$ belongs to $L(\widetilde{G})$. Since $|z_1 \ldots z_l| \geq \tilde{p}$, there are $(\gamma_1, \ldots, \gamma_l)$ and $w_1, \ldots, w_{m-1}, v_1, v_2, \ldots, v_{2(m-1)}$ satisfying (i)–(iv) with $m-1$ in place of m. Let $w_m = v_{2m-1} = v_{2m} = \varepsilon$. Then the l-tuple $(\gamma_1, \ldots, \gamma_{l-1}, \gamma_l \boldsymbol{x}_m)$ and the strings $w_1, \ldots, w_m, v_1, v_2, \ldots, v_{2m}$ satisfy (i)–(iv).

Case 2. Every derivation tree of G for $(z_1, \ldots, z_l)$ contains an even m-pump. Take a derivation tree τ for $(z_1, \ldots, z_l)$ that contains no redundant even m-pump and let υ be one of the lowest even m-pumps contained in τ. That is to say, $\tau = \upsilon'[\upsilon[\tau']]$ for some derivation tree context υ' and derivation tree τ' such that $\upsilon[\tau']$ contains no even m-pump other than υ. We have

$$\begin{aligned} A(\boldsymbol{x}_1, , \ldots, \boldsymbol{x}_m) &\vdash_G \upsilon' : S(\gamma_1, \ldots, \gamma_l), \\ A(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) &\vdash_G \upsilon : A(v_1 \boldsymbol{x}_1 v_2, \ldots, v_{2m-1} \boldsymbol{x}_m v_{2m}), \\ &\vdash_G \tau' : A(w_1, \ldots, w_m) \end{aligned}$$

for some nonterminal $A \in N^{(m)}$, patterns $\gamma_1, \ldots, \gamma_l$, and strings $w_1, \ldots, w_m, v_1, v_2, \ldots, v_{2m}$. Since υ is not redundant, $|v_1 v_2 \ldots v_{2m}| > 0$.

Case 2.1. $\sum_{i=1}^{m} |v_{2i-1} w_i v_{2i}| \leq p$. Then the conditions (i)–(iv) are clearly satisfied.

Case 2.2. $\sum_{i=1}^{m} |v_{2i-1} w_i v_{2i}| > p$. We have

$$\vdash_{G_A} \tau'' : S_A(v_1 w_1 v_2, \ldots, v_{2m-1} w_m v_{2m}),$$

where τ'' is a derivation tree that is just like $\upsilon[\tau']$ except that the last rule applied is changed from a rule of the form (12) to a rule of the form (11). By the choice of υ, it is clear that τ'' contains no even m-pump. Hence

$$(v_1 w_1 v_2, \ldots, v_{2m-1} w_m v_{2m}) \in L(\widetilde{G}_A).$$

Since $\sum_{i=1}^{m} |v_{2i-1} w_i v_{2i}| > \tilde{p}_A$, there are patterns $\delta_1, \ldots, \delta_m$ and strings $y_1, \ldots, y_{m-1}, x_1, x_2, \ldots, x_{2(m-1)}$ such that

$$(v_1 w_1 v_2, \ldots, v_{2m-1} w_m v_{2m}) = (\delta_1, \ldots, \delta_m)[x_1 y_1 x_2/\boldsymbol{x}_1, \ldots, x_{2(m-1)-1} y_{m-1} x_{2(m-1)}/\boldsymbol{x}_{m-1}],$$
$$|x_1 x_2 \ldots x_{2(m-1)}| > 0,$$
$$\sum_{i=1}^{m-1} |x_{2i-1} y_i x_{2i}| \leq \tilde{p}_A,$$
$$(\delta_1, \ldots, \delta_m)[x_1^n y_1 x_2^n/\boldsymbol{x}_1, \ldots, x_{2(m-1)-1}^n y_{m-1} x_{2(m-1)}^n/\boldsymbol{x}_{m-1}] \in L(\widetilde{G}_A) \quad \text{for all } n \in \mathbb{N}.$$

By the construction of G_A, the last condition implies

$$\vdash_G A(\delta_1, \ldots, \delta_m)[x_1^n y_1 x_2^n/\boldsymbol{x}_1, \ldots, x_{2(m-1)}^n y_{m-1} x_{2(m-1)}^n/\boldsymbol{x}_{m-1}],$$

which in turn implies

$$\vdash_G S(\gamma_1', \ldots, \gamma_l')[x_1^n y_1 x_2^n/\boldsymbol{x}_1, \ldots, x_{2(m-1)-1}^n y_{m-1} x_{2(m-1)}^n/\boldsymbol{x}_{m-1}],$$

where

$$(\gamma_1', \ldots, \gamma_l') = (\gamma_1, \ldots, \gamma_l)[\delta_1/\boldsymbol{x}_1, \ldots, \delta_m/\boldsymbol{x}_m].$$

Let $y_m = x_{2m-1} = x_{2m} = \varepsilon$. Then the l-tuple $(\gamma_1', \ldots, \gamma_l' \boldsymbol{x}_m)$ and the strings $y_1, \ldots, y_m, x_1, x_2, \ldots, x_{2m}$ satisfy (i)–(iv). □

Theorem 17. *Let L be the language of a well-nested m-MCFG. There is a natural number p such that for every $z \in L$ with $|z| \geq p$, there exists strings $u_1, u_2, \ldots, u_{2m+1}, v_1, v_2, \ldots, v_{2m}$ satisfying the following conditions:*

(i) $z = u_1 v_1 u_2 v_2 \ldots u_{2m} v_{2m} u_{2m+1}$.

(ii) $|v_1 v_2 \ldots v_{2m}| > 0$.

(iii) $\sum_{i=1}^{m} |v_{2i-1} u_{2i} v_{2i}| \leq p$.

(iv) $u_1 v_1^n u_2 v_2^n \ldots u_{2n} v_{2n}^n u_{2n+1} \in L$ *for all* $n \in \mathbb{N}$.

Since Lemma 15 can be proved for non-strict (not necessarily well-nested) 2-MCFGs, the above theorem also holds of 2-MCFGs.[6]

7. Conclusion

We have proved a natural generalization of Ogden's [6] lemma to what we call proper m-MCFGs. We have shown that the pumping lemma of [5] can be strengthened to include a bound the combined length of the substrings that can be simultaneously iterated, but there is no way of adding a further Ogden-like restriction on the positions of these substrings.

Since $C_k \subseteq 2^{k-1}$-MCFL$_{\text{prop}}$, Palis and Shende's [8] theorem (Theorem 5), on the one hand, and Theorem 10 with $m = 2^{k-1}$, on the other, both apply to languages in C_k, but they place incomparable requirements on the factorization $z = u_1 v_1 \ldots u_{2^k} v_{2^k} u_{2^k+1}$. For $k \geq 2$, Theorem 10 does not require $v_{2^{k-1}} u_{2^{k-1}} v_{2^{k-1}+1}$ to contain $\leq p$ distinguished positions. On the other hand, it does not seem easy to derive additional restrictions on $v_{2i-1} u_{2i} v_{2i}$ from Palis and Shende's [8] proof. From the point of view of MCFGs, the conditions in Theorem 10 are very natural: the substrings that can be simultaneously iterated should contain only a small number of distinguished positions.

[6] A referee pointed out that Theorem 17 was already stated by Sorokin [23] (Theorem 3 of his paper). I find this paper poorly written and difficult to understand in general, and one of his main results (his Theorem 4) in particular contradicts Theorem 6 of the present paper and therefore is in error (as acknowledged in [24]). As far as his Theorem 3 is concerned, his proof is based on a grammar transformation similar to that in my 2009 paper [5] but takes advantage of a Chomsky-like normal form for well-nested MCFGs (cf. [25]). I find his proof wanting in rigor and perspicuity, but the general ideas appear to be correct.

References

[1] H. Seki, T. Matsumura, M. Fujii, T. Kasami, On multiple context-free grammars, Theoretical Computer Science 88 (2) (1991) 191–229.

[2] A. K. Joshi, Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions?, in: D. R. Dowty, L. Karttunen, A. M. Zwicky (Eds.), Natural Language Parsing: Psychological, Computational and Theoretical Perspectives, Cambridge University Press, Cambridge, 206–250, 1985.

[3] M. Kanazawa, The convergence of well-nested mildly context-sensitive grammar formalisms, an invited talk given at the 14th Conference on Formal Grammar, Bordeaux, France. Slides available at `https://makotokanazawa.ws.hosei.ac.jp/talks/fg2009_talk.pdf`, 2009.

[4] M. Kanazawa, S. Salvati, MIX is not a tree-adjoining language, in: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 666–674, 2012.

[5] M. Kanazawa, The pumping lemma for well-nested multiple context-free languages, in: V. Diekert, D. Nowotka (Eds.), Developments in Language Theory: 13th International Conference, DLT 2009, vol. 5583 of *Lecture Notes in Computer Science*, Springer, Berlin, 312–325, 2009.

[6] W. Ogden, A helpful result for proving inherent ambiguity, Mathematical Systems Theory 2 (3) (1968) 191–194.

[7] D. J. Weir, A geometric hierarchy beyond context-free languages, Theoretical Computer Science 104 (2) (1992) 235–261.

[8] M. A. Palis, S. M. Shende, Pumping lemmas for the control language hierarchy, Mathematical Systems Theory 28 (3) (1995) 199–213.

[9] Y. Bar-Hillel, M. Perles, E. Shamir, On formal properties of simple phrase structure grammars, Zeitschrift für Phonetik, Sprachwissenschaft und Kommunikationsforschung 14 (2) (1961) 143–172.

[10] S. Ginsburg, E. H. Spanier, AFL with the semilinear property, Journal of Computer and System Sciences 5 (4) (1971) 365–396.

[11] S. A. Greibach, The strong independence of substitution and homomorphic replication, R.A.I.R.O. Informatique théorique 12 (3) (1978) 213–234.

[12] S. A. Greibach, Full AFLs and nested iterated substitution, Information and Control 16 (1970) 7–35.

[13] J. Král, A modification of a substitution theorem and some necessary and sufficient conditions for sets to be context-free, Mathematical Systems Theory 4 (2) (1970) 129–139.

[14] J. Engelfriet, Context-free graph grammars, in: Handbook of Formal Languages, Volume 3: Beyond Words, Springer, Berlin, 125–213, 1997.

[15] M. Kanazawa, S. Salvati, The copying power of well-nested multiple context-free grammars, in: A.-H. Dediu, H. Fernau, C. Martín-Vide (Eds.), Language and Automata Theory and Applications, Fourth International Conference, LATA 2010, vol. 6031 of *Lecture Notes in Computer Science*, Springer, Berlin, 344–355, 2010.

[16] M. Kanazawa, S. Salvati, Generating control languages with abstract categorial grammars, in: Preliminary Proceedings of FG-2007: The 12th Conference on Formal Grammar, 2007.

[17] S. A. Greibach, One-way finite visit automata, Theoretical Computer Science 6 (1978) 175–221.

[18] S. A. Greibach, Hierarchy theorems for two-way finite state transducers, Acta Informatica 11 (1978) 89–101.

[19] M. Kanazawa, G. M. Kobele, J. Michaelis, S. Salvati, R. Yoshinaka, The failure of the strong pumping lemma for multiple context-free languages, Theory of Computing Systems 55 (1) (2014) 250–278.

[20] F. Gécseg, M. Steinby, Tree Automata, ArXiv e-prints .

[21] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, C. Löding, S. Tison, M. Tommasi, *Tree Automata Techniques and Applications*, Available on `http://tata.gforge.inria.fr/`, release November 18, 2008, 2008.

[22] A. V. Groenink, Mild context-sensitivity and tuple-based generalization of context-free grammar, Linguistics and Philosophy 20 (1997) 607–636.

[23] A. Sorokin, Pumping lemma and Ogden lemma for displacement context-free grammars, in: A. M. Shur, M. V. Volkov (Eds.), DLT 2014: Developments in Language Theory, vol. 8633 of *Lecture Notes in Computer Science*, Springer, Cham, 154–165, 2014.

[24] A. Sorokin, Ogden property for linear displacement context-free grammars, in: S. Artemov, A. Nerode (Eds.), LFCS 2016: Logical Foundations for Computer Science, vol. 9537 of *Lecture Notes in Computer Science 9537*, Springer, Cham, 376–391, 2016.

[25] C. Gómez-Rodríguez, M. Kuhlmann, G. Satta, Efficient Parsing of Well-Nested Linear Context-Free Rewriting Systems, in: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL, Association for Computational Linguistics, 276–284, 2010.