The Pumping Lemma for Well-Nested Multiple Context-Free Languages

Makoto Kanazawa*

National Institute of Informatics

Abstract. Seki et al. (1991) proved a rather weak pumping lemma for multiple context-free languages, which says that any infinite m-multiple context-free language contains a string that is pumpable at some $2m$ substrings. We prove a pumping lemma of the usual universal form for the subclass consisting of *well-nested* multiple context-free languages. This is the same class of languages generated by non-duplicating macro grammars and by coupled-context-free grammars.

1 Introduction

Call a string z in a language L *k-pumpable* (in L) if there are strings $u_0, \dots, u_k$ and $v_1, \dots, v_k$ that satisfy the following conditions:

$$z = u_0 v_1 u_1 v_2 u_2 \dots u_{k-1} v_k u_k,$$
$$v_1 v_2 \dots v_k \neq \epsilon,$$
$$u_0 v_1^i u_1 v_2^i u_2 \dots u_{k-1} v_k^i u_k \in L \quad \text{for every } i \geq 0.$$

In their paper introducing *multiple context-free grammars*, Seki et al. (1991) proved a rather weak pumping lemma for multiple context-free languages, which says that if L is an infinite m-MCFL, then *some* string in L is $2m$-pumpable. Despite its peculiarly weak existential form, this lemma is quite useful; for example, it can be used to show that the language $\{\, \mathtt{a}_1^n \dots \mathtt{a}_{2m+1}^n \mid n \geq 0 \,\}$ over the alphabet $\{\mathtt{a}_1, \dots, \mathtt{a}_{2m+1}\}$ separates the $(m+1)$-MCFLs from m-MCFLs.

As it happens, Seki et al.'s (1991) proof of their lemma was quite complex, and by some quirk of fate a number of people were led to believe erroneously that a pumping lemma of the more usual universal form has been established for MCFLs: if L is an m-MCFL, *all but finitely many* strings in L are $2m$-pumpable. Radzinski (1991) appealed to it in his attempt to prove that the language $\{\, \mathtt{ab}^{k_1} \mathtt{ab}^{k_2} \dots \mathtt{ab}^{k_n} \mid n \geq 1, k_1 > k_2 > \dots > k_n > 0 \,\}$ is not an MCFL.[1] As a matter of fact, it remains an open question to this day whether the universal pumping lemma holds of all m-MCFLs (Kanazawa and Salvati 2007).

* I am grateful to Shoh Yamashita and Makoto Tatsuta for helpful discussions.

[1] This language, considered by Radzinski (1991) in connection with the system of Chinese number names, was shown to be non-semilinear by Michaelis and Kracht (1997), so Radzinski's (1991) claim that it is not an MCFL was correct, even though his appeal to the universal pumping lemma for general MCFLs was not justified.

This question is especially interesting in view of the fact that the class of m-MCFLs is captured by a large number of different formalisms, including among many others *hyperedge replacement grammars* (Engelfriet and Heyker 1991) and *deterministic tree-walking transducers* (Weir 1992).

In this paper, we show that a universal pumping lemma holds for the subclass of MCFLs generated by *well-nested* multiple context-free grammars. It is not difficult to prove that well-nested MCFGs are equivalent to *coupled-context-free grammars* (Hotz and Pitsch 1996) and to *non-duplicating macro grammars* (Fischer 1968).

The principal difficulty in proving a pumping lemma for MCFLs lies in the fact that pumpability of a derivation tree does not translate into pumpability of its string yield. Consider a derivation tree that contains inside it a "pump", i.e., a tree whose frontier consists of terminal nodes plus a single node marked by the same nonterminal as the root. In the case of a CFG, the function of a pump is to take a string and wrap two strings around it; its contribution to the string yield of the whole derivation tree is simply insertion of two substrings into the yield. Iterating the pump i times in the derivation tree results in insertion of i consecutive copies of those substrings, leading to 2-pumpability. In the case of an m-MCFG, in contrast, the function of a pump is to take a k-tuple of strings ($k \leq m$) and return another k-tuple. Each component of the original k-tuple appears somewhere in the resulting k-tuple, but it may move into a different component. If that happens, the effect of the presence of a pump is not merely insertion of $2k$ substrings, but also involves shuffling of the substrings contributed by the subtree below the pump.

Call a pump of an MCFG an *even k-pump* if it maps a k-tuple of strings $(x_1, \ldots, x_k)$ to a k-tuple of the form $(v_1 x_1 v_2, \ldots, v_{2k-1} x_k v_{2k})$. The presence of an even k-pump inside a derivation tree leads to $2k$-pumpability of its string yield, but it is not generally true, even for well-nested MCFGs, that all but finitely many derivation trees contain an even pump. Our strategy for proving the universal pumping lemma for well-nested m-MCFGs is as follows. We show by a series of transformations that for every well-nested m-MCFG G, the set of strings that are the yields of derivation trees of G without even m-pumps is the language of some well-nested $(m-1)$-MCFG. Since well-nested 1-MCFGs are just CFGs, this implies, by induction, that all but finitely many strings in the language of G are $2m$-pumpable.

2 Preliminaries

For $m, n \in \mathbb{N}$ (the set of natural numbers), we use the notation $[m, n]$ to denote $\{\, i \in \mathbb{N} \mid m \leq i \leq n \,\}$.

A *ranked alphabet* is a finite set $\Delta = \bigcup_{n \in \mathbb{N}} \Delta^{(n)}$ such that $\Delta^{(i)} \cap \Delta^{(j)} = \varnothing$ for $i \neq j$. The rank of $f \in \Delta$ is the unique r such that $f \in \Delta^{(r)}$. The set $\mathbb{T}_\Delta$ of *trees* over a ranked alphabet Δ is the smallest set closed under the rule: $f \in \Delta^{(n)}$ and $T_1, \ldots, T_n \in \mathbb{T}_\Delta$ imply $(f T_1 \ldots T_n) \in \mathbb{T}_\Delta$. We also use trees with holes, which are represented by trees over a ranked alphabet Δ augmented with a set

$\mathbf{Y}$ of variables. The notation $\mathbb{T}_\Delta(\mathbf{Y})$ denotes the set of trees over $\Delta \cup \mathbf{Y}$, where the variables in $\mathbf{Y}$ have rank 0. We use expressions like $T[\mathbf{y}_1, \ldots, \mathbf{y}_m]$ to denote trees in $\mathbb{T}_\Delta(\mathbf{Y})$ whose variables are among $\mathbf{y}_1, \ldots, \mathbf{y}_m$, and write $T[T_1, \ldots, T_m]$ for the result of substituting $T_1, \ldots, T_m$ for $\mathbf{y}_1, \ldots, \mathbf{y}_m$ in $T[\mathbf{y}_1, \ldots, \mathbf{y}_m]$. A tree $T \in \mathbb{T}_\Delta(\mathbf{Y})$ is a *simple tree* if each variable in $\mathbf{Y}$ occurs in T at most once.

We assume that the reader is familiar with the notion of *recognizable* set of trees (see Comon et al. 2007 or Gecseg and Steinby 1997). The family of recognizable sets is closed under Boolean operations.

3 Multiple Context-Free Grammars

A *multiple context-free grammar* (Seki et al. 1991) is a context-free grammar on tuples of strings, and is a special kind of *parallel multiple context-free grammar*, which in turn is a special kind of *elementary formal system* (Smullyan 1961, Groenink 1997), a logic program on strings. We use the notation of elementary formal systems, rather than Seki et al.'s (1991), to represent rules of MCFGs.

Let N be a ranked alphabet and Σ be an unranked alphabet. We assume that we are given a fixed infinite supply X of variables ranging over strings. An expression of the form $B(t_1, \ldots, t_r)$, where $B \in N^{(r)}$ and $t_1, \ldots, t_r$ are strings over $\Sigma \cup X$, is called an *atom* over N, Σ. A *rule* over N, Σ is an expression

$$B(t_1, \ldots, t_r) :- B_1(x_{1,1}, \ldots, x_{1,r_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,r_n}),$$

where

1. $n \geq 0$,
2. $B \in N^{(r)}$ and $B_i \in N^{(r_i)}$ for all $i \in [1, n]$,
3. $x_{i,j}$ are pairwise distinct variables in X,
4. $t_1, \ldots, t_r$ are strings over $\Sigma \cup \{ x_{i,j} \mid i \in [1, n], j \in [1, r_i] \}$ such that each $x_{i,j}$ occurs at most once in $t_1 \ldots t_r$.

In a rule

$$B(t_1, \ldots, t_r) :- B_1(x_{1,1}, \ldots, x_{1,r_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,r_n}),$$

the atom to the left of :− is called the *head* of the rule, and the sequence of atoms to the right of :− is called the *body*. Each atom in the body is called a *subgoal*. The symbol :− is omitted from a rule whose body is empty. Such a rule is called a *terminating rule*.

A *multiple context-free grammar* (MCFG) is a 4-tuple $G = (N, \Sigma, P, S)$, where

1. N is a ranked alphabet of *nonterminals*,
2. Σ is an (unranked) alphabet of *terminals*, disjoint from N,
3. P is a finite set of *rules* over N, Σ, and
4. $S \in N^{(1)}$.

We say that G is an m-MCFG if the rank of nonterminals does not exceed m.[2]

The language of G is $L(G) = \{\, w \in \Sigma^* \mid \vdash_G S(w) \,\}$, where $\vdash_G$ is defined by the following inference schema:

$$\frac{\vdash_G B_1(w_{1,1}, \ldots, w_{1,r_1}) \quad \ldots \quad \vdash_G B_n(w_{n,1}, \ldots, w_{n,r_n})}{\vdash_G B(t_1, \ldots, t_r)\sigma}$$

where $w_{i,j} \in \Sigma^*$, $B(t_1, \ldots, t_r) :- B_1(x_{1,1}, \ldots, x_{1,r_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,r_n})$ is a rule in P, and σ is the substitution that sends $x_{i,j}$ to $w_{i,j}$. If G is an m-MCFG, $L(G)$ is called an *m-MCFL*.

It is also convenient to extend the definition of $\vdash_G$ as follows:

$$\frac{}{B(x_1, \ldots, x_r) \vdash_G B(x_1, \ldots, x_r)}$$

$$\frac{\Gamma_1 \vdash_G B_1(x_{1,1}, \ldots, x_{1,r_1})\sigma \quad \ldots \quad \Gamma_n \vdash_G B_n(x_{n,1}, \ldots, x_{n,r_n})\sigma}{\Gamma_1, \ldots, \Gamma_n \vdash_G B(t_1, \ldots, t_r)\sigma}$$

In the second schema, $B(t_1, \ldots, t_n) :- B_1(x_{1,1}, \ldots, x_{1,r_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,r_n})$ is a rule in P, and $\Gamma_1, \ldots, \Gamma_n$ is a sequence of atoms with no repeated variables.

We call an MCFG $G = (N, \Sigma, P, S)$ *reduced* if the following conditions hold for each nonterminal $B \in N^{(r)}$:

- $\vdash_G B(w_1, \ldots, w_r)$ for some $w_1, \ldots, w_r \in \Sigma^*$, and
- $B(x_1, \ldots, x_r) \vdash_G S(t)$ for some $t \in (\Sigma \cup \{x_1, \ldots, x_r\})^*$.

The following lemma can be shown by a familiar method:

Lemma 1. *For every m-MCFG $G = (N, \Sigma, P, S)$ such that $L(G) \neq \varnothing$, there exists a reduced m-MCFG $G' = (N', \Sigma, P', S)$ such that $L(G) = L(G')$, $N' \subseteq N$, and $P' \subseteq P$.*

A rule is *non-deleting* if it satisfies the strengthened form of the fourth condition on rules:

4′. $t_1, \ldots, t_r$ are strings over $\Sigma \cup \{\, x_{i,j} \mid i \in [1, n], j \in [1, r_i] \,\}$ such that each $x_{i,j}$ occurs exactly once in $t_1 \ldots t_r$.

A *non-deleting* MCFG is one whose rules are all non-deleting. Non-deleting (m-)MCFGs are also known as *(string-based) (m-)linear context-free rewriting systems* (LCFRSs) (Vijay-Shanker et al. 1987, Weir 1988). It is known that for every m-MCFG G, there is a non-deleting m-MCFG G' such that $L(G) = L(G')$ (Seki et al. 1991).

We call a rule $B(t_1, \ldots, t_r) :- B_1(x_{1,1}, \ldots, x_{1,r_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,r_n})$ *non-permuting* if it satisfies the following condition:

- there are no i, j, k such that $1 \leq i \leq n$, $1 \leq j < k \leq r_i$, and $t_1 \ldots t_r \in (\Sigma \cup X)^* x_{i,k} (\Sigma \cup X)^* x_{i,j} (\Sigma \cup X)^*$.

[2] The rank of a nonterminal is called its *dimension* by Seki et al. (1991).

A *non-permuting* MCFG is one whose rules are all non-permuting. Non-deleting non-permuting MCFGs correspond to what Villemonte de la Clergerie (2002a, 2002b) called *ordered simple RCG* and Kracht (2003) called *monotone* LCFRSs.

Theorem 2 (Kracht 2003). *For every m-MCFG G, there is a non-deleting non-permuting m-MCFG G' such that $L(G) = L(G')$.*

Example 3. The following (non-deleting non-permuting) 2-MCFG generates $\text{RESP} = \{\, \mathtt{a}_1^m \mathtt{a}_2^m \mathtt{b}_1^n \mathtt{b}_2^n \mathtt{a}_3^m \mathtt{a}_4^m \mathtt{b}_3^n \mathtt{b}_4^n \mid m, n \geq 0 \,\}$:

$$S(x_1 y_1 x_2 y_2) :- P(x_1, x_2), Q(y_1, y_2).$$
$$P(\epsilon, \epsilon). \qquad P(\mathtt{a}_1 x_1 \mathtt{a}_2, \mathtt{a}_3 x_2 \mathtt{a}_4) :- P(x_1, x_2).$$
$$Q(\epsilon, \epsilon). \qquad Q(\mathtt{b}_1 y_1 \mathtt{b}_2, \mathtt{b}_3 y_2 \mathtt{b}_4) :- Q(y_1, y_2).$$

We call a rule $B(t_1, \ldots, t_r) :- B_1(x_{1,1}, \ldots, x_{1,r_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,r_n})$ *well-nested* if it is non-deleting and non-permuting and satisfies the following condition:

- there are no i, j, k, i', j', k' such that $1 \leq i, i' \leq n$, $i \neq i'$, $1 \leq j < k \leq r_i$, $1 \leq j' < k' \leq r_{i'}$, and $t_1 \ldots t_r \in (\Sigma \cup X)^* x_{ij} (\Sigma \cup X)^* x_{i'j'} (\Sigma \cup X)^* x_{ik} (\Sigma \cup X)^* x_{i'k'} (\Sigma \cup X)^*$.

We say that an MCFG G is *well-nested* if every rule of G is well-nested. A *well-nested MCFL* is the language of some well-nested MCFG. Note that the grammar in Example 3 is not well-nested, because the first rule is not well-nested.

Example 4. The following is an example of a well-nested 2-MCFG:

$$S(x_1 x_2) :- A(x_1, x_2). \qquad A(\epsilon, \epsilon). \qquad A(\mathtt{a} x_1 \mathtt{b} x_2 \mathtt{c}, \mathtt{d}) :- A(x_1, x_2).$$

The well-nestedness constraint has been studied by Kuhlmann and Möhl (2007a, 2007b) in the context of *dependency grammars*. Well-nested (m-)MCFGs are essentially the same as *coupled-context-free grammars* (of rank m) (Hotz and Pitsch 1996), and it can be shown that they are equivalent to *non-duplicating macro grammars* (of rank $m - 1$) (Fischer 1968).[3] It is known that well-nested 2-MCFGs are equivalent to *tree-adjoining grammars* (Joshi and Schabes 1997).

Although well-nestedness restricts the generative power of m-MCFGs for each m (Seki and Kato 2008), almost all examples of m-MCFGs that have appeared

[3] The equivalence between well-nested MCFGs and coupled-context-free grammars is a special case of the equivalence between MCFGs and *local unordered scattered context grammars* (Rambow and Satta 1999). Seki and Kato (2008) prove that all non-duplicating macro grammars—which they call *variable-linear* macro grammars—have equivalent MCFGs. The MCFGs constructed by their proof are well-nested.

The languages generated by non-duplicating macro grammars are the same as the yield images of the tree languages generated by *linear context-free tree grammars* (Kepser and Mönnich 2006).

in the literature have an equivalent well-nested m'-MCFG for some $m' \geq m$. The only exception we are aware of is the 3-MCFG G_{ex} given by Michaelis (2009):

$$S(x_1x_2x_3) :- B(x_1, x_2, x_3). \qquad A(\mathtt{a}, \mathtt{a}, \mathtt{a}). \qquad A(\mathtt{b}, \mathtt{b}, \mathtt{b}).$$
$$A(x_1\mathtt{a}, x_2\mathtt{a}, \mathtt{a}x_3) :- A(x_1, x_2, x_3). \qquad A(x_1\mathtt{b}, x_2\mathtt{b}, \mathtt{b}x_2) :- A(x_1, x_2, x_3).$$
$$A(x_1y_1, y_2x_2, y_3x_3) :- B(x_1, x_2, x_3), B(y_1, y_2, y_3).$$
$$B(\epsilon, \mathtt{[]}, \epsilon). \qquad B(x_1, x_2, x_3) :- A(x_1, x_2, x_3). \qquad B(x_1, \mathtt{[}x_2\mathtt{]}, x_3) :- B(x_1, x_2, x_3).$$

This non-well-nested 3-MCFG generates the following language:[4]

$$L(G_{\text{ex}}) = \{\, w_1 \dots w_n z_n w_n \dots z_1 w_1 z_0 w_n^R \dots w_1^R \mid n \geq 1,\ w_i \in \{\mathtt{a}, \mathtt{b}\}^+ \text{ for } 1 \leq i \leq n, \text{ and } z_n \dots z_0 \in D^*_{\{\mathtt{[},\mathtt{]}\}} \,\}.$$

According to Staudacher (1993), this language is not an indexed language, which implies that it does not have a non-duplicating macro grammar and hence is not a well-nested MCFL.

In what follows, we will only consider MCFGs that are non-deleting and non-permuting. By a "rule", we will mean a rule that is non-deleting and non-permuting.

4 Derivation Trees of Multiple Context-Free Grammars

We now give our definition of the notion of a *derivation tree* for an MCFG $G = (N, \Sigma, P, S)$. For this purpose, we view the set P of rules as a ranked alphabet. A rule π of the form

$$B(t_1, \dots, t_r) :- B_1(x_{1,1}, \dots, x_{1,r_1}), \dots, B_n(x_{n,1}, \dots, x_{n,r_n})$$

belongs to $P^{(n)}$ (i.e., is treated as a symbol of rank n).

Let $\mathbf{Y}$ be a countably infinite set of variables. We use boldface letters $\mathbf{y}, \mathbf{y}_i$, to represent variables in $\mathbf{Y}$, to distinguish them from variables in MCFG atoms, for which we use $x_{i,j}, z_{i,j}, y_i$, etc. We use simple trees in $\mathbb{T}_P(\mathbf{Y})$ as terms in statements of the form

$$\Gamma \vdash_G T : B(t_1, \dots, t_r),$$

where Γ is an expression of the form

$$\mathbf{y}_1 : C_1(z_{1,1}, \dots, z_{1,r_1}), \dots, \mathbf{y}_p : C_p(z_{p,1}, \dots, z_{p,r_p}).$$

The earlier system for deriving statements of the form $\Gamma \vdash_G B(t_1, \dots, t_r)$, where Γ is a sequence of atoms, is now augmented with trees from $\mathbb{T}_P(\mathbf{Y})$ as follows:

$$\frac{}{\mathbf{y} : B(x_1, \dots, x_n) \vdash_G \mathbf{y} : B(x_1, \dots, x_n)}$$

$$\frac{\Gamma_1 \vdash_G T_1 : B_1(x_{1,1}, \dots, x_{1,r_1})\sigma \quad \dots \quad \Gamma_n \vdash_G T_n : B_n(x_{n,1}, \dots, x_{n,r_n})\sigma}{\Gamma_1, \dots, \Gamma_n \vdash_G \pi T_1 \dots T_n : B(x_1, \dots, x_r)\sigma}$$

[4] Here, $D^*_{\{\mathtt{[},\mathtt{]}\}}$ refers to the (one-sided) Dyck language over a single pair of brackets $\mathtt{[},\mathtt{]}$.

In the second schema, π is the rule

$$B(x_1, \ldots, x_r) :- B_1(x_{1,1}, \ldots, x_{1,r_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,r_n}),$$

and the variables (including boldface ones) in $\Gamma_1, \ldots, \Gamma_n$ have no repeated occurrences.

A tree $T \in \mathbb{T}_P$ is called a *derivation tree* if

$$\vdash_G T : B(w_1, \ldots, w_r)$$

holds for some $B \in N$ and $w_1, \ldots, w_r \in \Sigma^*$. The nonterminal B is called the *type* of T, and the tuple $(w_1, \ldots, w_r)$ is called its *yield*. Any derivation tree T has a unique type and a unique yield, for which we write $\mathrm{type}(T)$ and $\mathrm{yield}(T)$, respectively. A derivation tree is *complete* if it is of type S. We denote the set of derivation trees of type B by $\mathcal{D}_G^B$. Note that $\mathcal{D}_G^B$ is a recognizable (in fact, local) subset of $\mathbb{T}_P$.

We call a simple tree $T \in \mathbb{T}_P(\mathbf{Y})$ a *non-terminating derivation tree* if

$$\Gamma \vdash_G T : B(t_1, \ldots, t_r)$$

holds for some $\Gamma, B, t_1, \ldots, t_r$. Note that derivation trees are special cases of non-terminating derivation trees. We call a non-terminating derivation tree T a *derivation tree context* if

$$\mathbf{y} : C(y_1, \ldots, y_p) \vdash_G T : B(t_1, \ldots, t_r)$$

holds of some $\mathbf{y} \in \mathbf{Y}$, $C \in N^{(p)}$ and $B \in N^{(r)}$.

It is easy to see that if T is a derivation tree and T' is a subtree of T, then T' is also a derivation tree. The same goes with non-terminating derivation trees.

Lemma 5. *If* $\Gamma \vdash_G T : B(t_1, \ldots, t_r)$ *and* $\mathbf{y} : B(x_1, \ldots, x_r) \vdash_G U[\mathbf{y}] : C(u_1, \ldots, u_p)$, *then* $\Gamma \vdash_G U[T] : C(u_1, \ldots, u_p)\sigma$, *where* σ *is the substitution that sends* x_i *to* t_i *for all* $i \in [1, r]$.

Lemma 6. *Let* T *be a derivation tree of type* B *with yield* $(w_1, \ldots, w_r)$. *Suppose that* T' *is a subtree of* T *such that* $\mathrm{type}(T') = C$ *and* $\mathrm{yield}(T') = (v_1, \ldots, v_p)$. *Then there is a derivation tree context* $U[\mathbf{y}]$ *and some* $t_1, \ldots, t_r$ *such that:*

$$\begin{gathered} T = U[T'], \\ \mathbf{y} : C(y_1, \ldots, y_p) \vdash_G U[\mathbf{y}] : B(t_1, \ldots, t_r), \\ (w_1, \ldots, w_r) = (t_1, \ldots, t_r)\sigma, \end{gathered}$$

where σ *is the substitution that maps* y_i *to* v_i *for* $i \in [1, p]$.

We call a derivation tree context $U[\mathbf{y}]$ a *k-pump* if $U[\mathbf{y}] \neq \mathbf{y}$ and there exist $B \in N^{(k)}$ and $t_1, \ldots, t_k \in (\Sigma \cup \{x_1, \ldots, x_k\})^*$ such that

$$\mathbf{y} : B(x_1, \ldots, x_k) \vdash_G U[\mathbf{y}] : B(t_1, \ldots, t_k).$$

We say that a derivation tree T *contains* a k-pump $U[\mathbf{y}]$ if $T = U'[U[T']]$ for some derivation tree T' and derivation tree context $U'[\mathbf{y}]$.

A k-pump $U[\mathbf{y}]$ is *even* if there are $v_1, \dots, v_{2k} \in \Sigma^*$ such that

$$\mathbf{y} : B(x_1, \dots, x_k) \vdash_G U[\mathbf{y}] : B(v_1 x_1 v_2, \dots, v_{2k-1} x_k v_{2k}),$$

and it is a *proper* even k-pump if it moreover holds that

$$v_1 \dots v_{2k} \neq \epsilon.$$

Lemma 7. *Let G be an MCFG and T be a complete derivation tree of G with yield z. If T contains a proper even k-pump, then z is $2k$-pumpable in $L(G)$.*

Example 4 (continued). Let π_1, π_2, π_3 name the three rules of the grammar G in Example 4. We have $\mathcal{D}_G^S = \{\, T_i \mid i \geq 0 \,\}$, where

$$T_i = \pi_1 \underbrace{(\pi_2 \dots (\pi_2}_{i \text{ times}} \; \pi_3 \underbrace{) \dots)}_{i \text{ times}},$$

and

$$\text{yield}(T_i) = \begin{cases} \epsilon & \text{if } i = 0, \\ \mathtt{a}^{i-1}\mathtt{abc}(\mathtt{bdc})^{i-1}\mathtt{d} & \text{if } i \geq 1. \end{cases}$$

Note that the derivation tree T_1 contains an (uneven) 2-pump, but $\text{yield}(T_1) = \mathtt{abcd}$ is not 4-pumpable. For $i \geq 2$, $\text{yield}(T_i) = \mathtt{a}^{i-1}\mathtt{abc}(\mathtt{bdc})^{i-1}\mathtt{d}$ is 2-pumpable, but no matter which 2-pump one picks in T_i, the occurrences of symbols that come from the 2-pump do not form contiguous substrings of $\text{yield}(T_i)$ that can be repeated.

Lemma 8. *Let $G = (N, \Sigma, P, S)$ be an MCFG. Then the set*

$$\{\, T \in \mathcal{D}_G^S \mid T \textit{ contains an even } k\textit{-pump} \,\}$$

is recognizable.

Let $G = (N, \Sigma, P, S)$ and $G' = (N', \Sigma, P', S')$ be m-MCFGs. A mapping $h \colon N' \to N$ is a *homomorphism* from G' to G if the following conditions hold:

- $h(S') = S$.
- For every $B' \in N'^{(r)}$, $h(B') \in N^{(r)}$.
- For every $\pi' \in P'$ of the form

  $$B'(t_1, \dots, t_r) :- B'_1(x_{1,1}, \dots, x_{1,r_1}), \dots, B'_n(x_{n,1}, \dots, x_{n,r_n}),$$

  if $B = h(B')$ and $B_i = h(B'_i)$ for $i \in [1, n]$, then

  $$B(t_1, \dots, t_r) :- B_1(x_{1,1}, \dots, x_{1,r_1}), \dots, B_n(x_{n,1}, \dots, x_{n,r_n})$$

  is a rule in P. (We refer to this rule as $h(\pi')$.)

Lemma 9. *Let G and G' be MCFGs such that there is a homomorphism from G' to G. If G is well-nested, then so is G'.*

If h is a homomorphism from G' to G and $T' \in \mathcal{D}_{G'}^{B'}$, then we write $h(T')$ for the result of replacing occurrences of each rule π' in T' by $h(\pi')$. Note that $h(T')$ must be a derivation tree in $\mathcal{D}_G^{h(B')}$ that has the same yield as T'.

Lemma 10. *Let $G = (N, \Sigma, P, S)$ be an m-MCFG. If K is a non-empty recognizable subset of $\mathcal{D}_G^S$, then there are an m-MCFG $G' = (N', \Sigma, P', S')$ and a homomorphism h from G' to G such that*

1. *$K = \{\, h(T') \mid T \in \mathcal{D}_{G'}^{S'} \,\}$.*
2. *If G is reduced, then G' is reduced.*

5 Unfolding

Let $G = (N, \Sigma, P, S)$ be an m-MCFG. A rule

$$B(t_1, \ldots, t_r) :- B_1(x_{1,1}, \ldots, x_{1,r_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,r_n})$$

over N, Σ is a *derivable rule* of G if it holds that

$$B_1(x_{1,1}, \ldots, x_{1,r_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,r_n}) \vdash_G B(t_1, \ldots, t_r).$$

We call an MCFG $G' = (N', \Sigma, P', S')$ *conservative over* $G = (N, \Sigma, P, S)$ if $N' \subseteq N$, $S' = S$, and every rule in P' is a derivable rule of G. Clearly, "is conservative over" is a transitive relation.

Lemma 11. *Let G and G' be MCFGs such that G' is conservative over G.*

1. *If G is well-nested, then so is G'.*
2. *If G' has an even m-pump, so does G.*

Let π, π' denote the following two rules:

$$B(t_1, \ldots, t_r) :- B_1(x_{1,1}, \ldots, x_{1,r_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,r_n})$$
$$C(u_1, \ldots, u_s) :- C_1(y_{1,1}, \ldots, y_{1,s_1}), \ldots, C_p(y_{p,1}, \ldots, y_{p,s_p}),$$

where $B_i = C$ (which implies $r_i = s$). Then we denote by $\pi \circ_i \pi'$ the rule

$$\begin{aligned} B(t_1, \ldots, t_r)\sigma :- \; & B_1(x_{1,1}, \ldots, x_{1,r_1}), \ldots, B_{i-1}(x_{i-1,1}, \ldots, x_{i-1,r_{i-1}}), \\ & C_1(y_{1,1}, \ldots, y_{1,s_1}), \ldots, C_p(y_{p,1}, \ldots, y_{p,s_p}), \\ & B_{i+1}(x_{i+1,1}, \ldots, x_{i+1,r_{i+1}}), \ldots, B_n(x_{n,1}, \ldots, x_{n,r_n}), \end{aligned}$$

where σ is the substitution that sends $x_{i,j}$ to u_j. Note that if π and π' are rules of an MCFG G, then $\pi \circ_i \pi'$ is a derivable rule of G.

Let $G = (N, \Sigma, P, S)$ be an MCFG. Let $\pi \in P$ be as above, and let $\pi_1, \ldots, \pi_k$ be all the rules in P that have B_i in their head. The result of *unfolding* π in G (at the i-th subgoal) is defined to be $G' = (N, \Sigma, P', S)$, where $P' = (P - \{\pi\}) \cup \{\, \pi \circ_i \pi_j \mid j \in [1, k] \,\}$. Clearly, G' is conservative over G. The following is familiar from logic programming (Tamaki and Sato 1984):

Lemma 12. *Let $G = (N, \Sigma, P, S)$ be an MCFG and π be a rule in P. If G' is the result of unfolding π in G at some subgoal, then $L(G) = L(G')$.*

6 Proof of the Main Theorem

We call a rule $B(t_1,\dots,t_m) :- B_1(x_{1,1},\dots,x_{1,r_1}),\dots,B_n(x_{n,1},\dots,x_{n,r_n})$ of an m-MCFG *m-proper* if there exists an $i \in [1,n]$ such that $r_i = m$ and for all $j \in [1,m]$,

$$t_j \in (\Sigma \cup X)^* x_{i,j} (\Sigma \cup X)^*.$$

In this case we call this rule *m-proper on the i-th subgoal*.

Lemma 13. *Let G be an m-MCFG that has no even m-pump. Then there is an m-MCFG G' that satisfies the following conditions:*

- *G' is conservative over G,*
- *G' has no m-proper rules, and*
- *$L(G) = L(G')$.*

Proof. The desired grammar G' may be obtained from G by repeatedly unfolding m-proper rules. We omit the details. □

Lemma 14. *Let $m \geq 2$ and let G be a well-nested m-MCFG without m-proper rules. Then there is a well-nested $(m-1)$-MCFG G' such that $L(G) = L(G')$.*

Proof. Define the *m-degree* of a rule to be the number of subgoals whose nonterminal is of rank m if the nonterminal in the head is of rank m, and 0 otherwise. We repeatedly apply the following transformation to eliminate from G rules that have m-degree > 0. Pick a rule π with m-degree > 0, if there is one. Modulo the order of the subgoals, π is of the following form:

$$B(t_1,\dots,t_m) :- C(y_1,\dots,y_m), B_1(x_{1,1},\dots,x_{1,r_1}),\dots,B_n(x_{n,1},\dots,x_{n,r_n}).$$

Let $f\colon \{1,\dots,m\} \to \{1,\dots,m\}$ be the function such that y_i occurs in $t_{f(i)}$ for all $i \in [1,m]$. Since π is not m-proper, at least one of the following possibilities must obtain:

Case 1. Either $1 < f(1)$ or $f(m) < m$. Suppose $t_{f(1)} = uy_1v$ and $t_{f(m)} = u'y_mv'$. Since π is well-nested,

$$B_1(x_{1,1},\dots,x_{1,r_1}),\dots,B_n(x_{n,1},\dots,x_{n,r_n})$$

can be partitioned into Γ_1, Γ_2 so that $x_{i,j}$ occurs in

$$t_1 \dots t_{f(1)-1} uv' t_{f(m)+1} \dots t_m$$

if and only if $B_i(x_{i,1},\dots,x_{i,r_i})$ is in Γ_1. Let $p = f(m) - f(1) + 1$, and let D be a new nonterminal of rank p. Note $p < m$.

Case 1a. $f(1) < f(m)$. We replace π by the following two well-nested rules:

$$B(t_1,\dots,t_{f(1)-1},uz_1,z_2,\dots,z_{p-1},z_pv',t_{f(m)+1},\dots,t_m) :- \Gamma_1, D(z_1,\dots,z_p).$$
$$D(y_1v,t_{f(1)+1},\dots,t_{f(m)-1},u'y_m) :- C(y_1,\dots,y_m),\Gamma_2.$$

Case 1b. $f(1) = f(m)$. Then $t_{f(1)} = t_{f(m)} = uy_1wy_mv'$ for some w. We replace π by the following two well-nested rules:

$$B(t_1, \ldots, t_{f(1)-1}, uz_1v', t_{f(m)+1}, \ldots, t_m) :- \Gamma_1, D(z_1).$$
$$D(y_1wy_m) :- C(y_1, \ldots, y_m), \Gamma_2.$$

Case 2. There is a $k \in [1, m-1]$ such that $f(k+1) - f(k) > 1$. Suppose $t_{f(k)} = uy_kv$ and $t_{f(k+1)} = u'y_{k+1}v'$. Since π is well-nested,

$$B_1(x_{1,1}, \ldots, x_{1,r_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,r_n})$$

can be partitioned into Γ_1, Γ_2 so that $x_{i,j}$ occurs in

$$vt_{f(k)+1} \ldots t_{f(k+1)-1}u'$$

if and only if $B_i(x_{i,1}, \ldots, x_{i,r_i})$ is in Γ_1. Let $p = f(k)+(m-f(k+1)+1)$, and let D be a new nonterminal of rank p. Note $p < m$. We replace π by the following two well-nested rules:

$$B(z_1, \ldots, z_{f(k)-1}, z_{f(k)}v, t_{f(k)+1}, \ldots, t_{f(k+1)-1}, u'z_{f(k)+1}, z_{f(k)+2}, \ldots, z_p) :- D(z_1, \ldots, z_p), \Gamma_1.$$
$$D(t_1, \ldots, t_{f(k)-1}, uy_k, y_{k+1}v', t_{f(k+1)+1}, \ldots, t_m) :- C(y_1, \ldots, y_m), \Gamma_2.$$

In all cases, π is replaced by two new rules, and the m-degree of the first rule is less than that of π and the m-degree of the second rule is 0 (since the rank of D is $p < m$), so the transformation reduces the sum of the m-degrees of the rules. It is also clear that the first rule is not m-proper, so the grammar continues to be without m-proper rules. The generated language remains the same because the original grammar can be obtained from the new grammar by unfolding (Lemma 12).

The process of repeatedly applying this transformation must terminate after a finite number of steps, and every rule in the final grammar has rank m nonterminals only in the head or only in the body (if any).

We can now eliminate all occurrences of rank m nonterminals in rule bodies by unfolding. If a rule π has a rank m nonterminal C in the i-th subgoal, we unfold π at that subgoal. Since any rule π' that has C in the head has no rank m nonterminal in the body, $\pi \circ_i \pi'$ has one fewer rank m nonterminals in the body than π does. Thus, we can repeatedly apply this transformation, which will terminate in a finite number of steps.

After this procedure, rank m nonterminals become useless, and we can simply delete rules with rank m nonterminals in the head to obtain an $(m-1)$-MCFG. □

Example 4 (continued). Applying the procedure of Lemma 14 to the grammar G in Example 4 gives the following 1-MCFG:

$$S(\epsilon). \qquad S(\mathtt{a}z\mathtt{cd}) :- D(z). \qquad D(\mathtt{b}). \qquad D(\mathtt{a}z\mathtt{cbd}) :- D(z).$$

Theorem 15. *Let $m \geq 1$. For every well-nested m-MCFG G, all but finitely many strings $z \in L(G)$ are $2m$-pumpable in $L(G)$.*

Proof. Induction on m. The case $m = 1$ is just the pumping lemma for context-free languages. Let $m \geq 2$ and assume that the theorem holds for $m - 1$.

Let $G = (N, \Sigma, P, S)$ be a well-nested m-MCFG. Without loss of generality, we can assume that G is reduced. Let

$$K = \{\, T \in \mathcal{D}_G^S \mid T \text{ does not contain an even } m\text{-pump} \,\}.$$

By Lemma 8, K is a recognizable subset of $\mathcal{D}_G^S$. By Lemma 10, there is a reduced well-nested m-MCFG $G' = (N', \Sigma, P', S')$ such that $L(G') = \{\, w \in \Sigma^* \mid \vdash_G T : S(w) \text{ for some } T \in K \,\}$ and no derivation tree in $\mathcal{D}_{G'}^{S'}$ contains an even m-pump. Since G' is reduced, it follows that G' has no even m-pump. By Lemmas 13 and 14, there is a well-nested $(m-1)$-MCFG G'' such that $L(G') = L(G'')$. By induction hypothesis, there is a number p such that all strings z in $L(G'')$ with $|z| \geq p$ are $2(m-1)$-pumpable.

Now assume $z \in L(G)$ and $|z| \geq p$. We show that z is $2m$-pumpable. If $z \in L(G'')$, then z is $2(m-1)$-pumpable, so a fortiori z is $2m$-pumpable. Now suppose $z \notin L(G'')$ and consider a smallest complete derivation tree T of G with yield z. Since $z \notin L(G'')$, T contains an even m-pump $U[\mathbf{y}]$:

$$T = U'[U[T']].$$

Because of the minimality of T, the even m-pump $U[\mathbf{y}]$ must be proper (otherwise $U'[T']$ is a smaller complete derivation tree for z). By Lemma 7, we conclude that z is $2m$-pumpable. □

Example 16. Let $D^*_{\{a,\bar{a}\}}$ be the Dyck language over $\{a, \bar{a}\}$ generated by the following context-free grammar:

$$S \to \epsilon \mid TS \qquad T \to aS\bar{a}$$

Define $\text{Shuffle}_3(L_1, L_2, L_3)$ to be

$$\{\, u_1v_1w_1 \dots u_nv_nw_n \mid n \geq 1, u_1 \dots u_n \in L_1, v_1 \dots v_n \in L_2, w_1 \dots w_n \in L_3 \,\},$$

and consider the language $L = \text{Shuffle}_3(D^*_{\{\mathtt{a},\bar{\mathtt{a}}\}}, D^*_{\{\mathtt{b},\bar{\mathtt{b}}\}}, D^*_{\{\mathtt{c},\bar{\mathtt{c}}\}})$. Note that L is semiliniear and satisfies Seki et al.'s pumping condition for 3-MCFLs. We do not know whether L is a 3-MCFL, but we can use Theorem 15 to show that L is not a *well-nested* 3-MCFL. Suppose that L is a well-nested 3-MCFL. Let

$$\begin{aligned} K &= L \cap \mathtt{a}^*(\bar{\mathtt{a}}\mathtt{b})^*(\bar{\mathtt{b}}\mathtt{c})^*(\bar{\mathtt{c}}\mathtt{a})^*(\bar{\mathtt{a}}\mathtt{b})^*(\bar{\mathtt{b}}\mathtt{c})^*\bar{\mathtt{c}}^* \\ &= \{\, \mathtt{a}^i(\bar{\mathtt{a}}\mathtt{b})^j(\bar{\mathtt{b}}\mathtt{c})^k(\bar{\mathtt{c}}\mathtt{a})^l(\bar{\mathtt{a}}\mathtt{b})^m(\bar{\mathtt{b}}\mathtt{c})^n\bar{\mathtt{c}}^q \mid \\ &\qquad i \geq j \geq k \geq l \leq m \leq n \leq q = i \text{ and } i + l = j + m = k + n \,\}. \end{aligned}$$

From known facts about equivalent formalisms (Fischer 1968, Kepser and Mönnich 2006, Seki and Kato 2008), the class of well-nested m-MCFLs is closed

under intersection with regular sets, so K must be a well-nested 3-MCFL. Note that K still satisfies Seki et al.'s pumping condition for 3-MCFLs, and is also semilinear. By Theorem 15, there is a number p such that all strings in K of length $\geq p$ are 6-pumpable. Take

$$w = \mathtt{a}^p(\bar{\mathtt{a}}\mathtt{b})^p(\bar{\mathtt{b}}\mathtt{c})^p(\bar{\mathtt{c}}\mathtt{a})^p(\bar{\mathtt{a}}\mathtt{b})^p(\bar{\mathtt{b}}\mathtt{c})^p\bar{\mathtt{c}}^p \in K,$$

which must have six substrings that can be pumped up and down. It is not hard to see that each of the six substrings must lie entirely inside one of the seven intervals $[pi + 1, p(i + 1)]$ consisting of the $(pi + 1)$-th through the $p(i + 1)$-th symbols of w $(i = 0, \ldots, 6)$, and yet each of the seven intervals must contain one of the six substrings, a contradiction.

7 Conclusion

We have proved a pumping lemma for *well-nested* m-MCFGs, which, unlike Seki et al.'s (1991) pumping lemma for general MCFGs, has the usual universal form. The special case of this for $m = 2$ is already known (Palis and Shende 1995), but the result is new for $m \geq 3$. The only place in our proof where well-nestedness is used is Lemma 14. While it is an open question whether this lemma holds of m-MCFGs in general, it is easy to see that it holds of (not necessarily well-nested) 2-MCFGs. Thus we have

Theorem 17. *For every 2-MCFG G, all but finitely many strings $z \in L(G)$ are 4-pumpable in $L(G)$.*

References

Comon, Hubert, Max Dauchet, Rémi Gilleron, Florent Jacquemard, Denis Lugiez, Christof Löding, Sophie Tison, and Marc Tommasi. 2007. *Tree Automata Techniques and Applications.* Available on http://tata.gforge.inria.fr/. Release October 12, 2007.

Engelfriet, Joost and Linda Heyker. 1991. The string generating power of context-free hypergraph grammars. *Journa of Computer and System Sciences* 43:328–360.

Fisher, Michael J. 1968. *Grammars with Macro-Like Productions*. Ph.D. thesis, Harvard University.

Gécseg, Ferenc and Magnus Steinby. 1997. Tree languages. In G. Rozenberg and A. Salomaa, eds., *Handbook of Formal Languages, Vol. 3: Beyond Words*, pages 1–68. Berlin: Springer.

Groenink, Annius. 1997. *Surface without Structure*. Ph.D. thesis, Utrecht University.

Hotz, Günter and Gisela Pitsch. 1996. On parsing coupled-context-free languages. *Thoretical Computer Science* 161:205–253.

Joshi, Aravind K. and Yves Schabes. 1997. Tree-adjoining grammars. In G. Rozenberg and A. Salomaa, eds., *Handbook of Formal Languages, Vol. 3: Beyond Words*, pages 69–123. Berlin: Springer.

Kanazawa, Makoto and Sylvain Salvati. 2007. Generating control languages with abstract categorial grammars. In *the preliminary proceedings of FG-2007: The 12th Conference on Formal Grammar*.

Kepser, Stephan and Uwe Mönnich. 2006. Closure properties of linear context-free tree languages with an application to optimality theory. *Theoretical Computer Science* 354(1):82–97.

Kracht, Marcus. 2003. *The Mathematics of Language*. Berlin: Mouton de Gruyter.

Kuhlman, Marco and Mathias Möhl. 2007a. Mildly context-sensitive dependency languages. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

Kuhlman, Marco and Mathias Möhl. 2007b. The string-generative capacity of regular dependency languages. In *FG-2007*.

Michaelis, Jens. 2009. An additional observation on strict derivational minimalism. In J. Rogers, ed., *Proceedings of FG-MoL 2005: The 10th conference on Formal Grammar and the 9th Meeting on Mathematics of Language*, pages 101–111. Stanford, CA: CSLI Publications.

Michaelis, Jens and Marcus Kracht. 1997. Semilinearity as a syntactic invariant. In C. Retoré, ed., *Logical Aspects of Computational Linguistics*, pages 329–345. Berlin: Springer.

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

Radzinski, Daniel. 1991. Chinese number-names, tree adjoining languages, and mild context-sensitivity. *Computational Linguistics* 17(3):277–299.

Rambow, Owen and Giorgio Satta. 1999. Independent parallelism in finite copying parallel rewriting systems. *Theoretical Computer Science* 223:87–120.

Seki, Hiroyuki and Yuki Kato. 2008. On the generative power of multiple context-free grammars and macro grammars. *IEICE Transactions on Information and Systems* E91–D(2):209–221.

Seki, Hiroyuki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science* 88(2):191–229.

Smullyan, Raymond M. 1961. *Theory of Formal Systems*. Princeton, N.J.: Princeton University Press.

Staudacher, Peter. 1993. New frontiers beyond context-freeness: DI-grammars and DI-automata. In *6th Conference of the European Chapter of the Association for Computational Linguistics (EACL '93)*, pages 358–367.

Tamaki, H. and T. Sato. 1984. Unfold/fold transformation of logic programs. In *Proceedings of the Second International Conference on Logic Programming*, pages 127–138.

Vijay-Shanker, K., David J. Weir, and Aravind K. Joshi. 1987. Characterizing structural descriptions produced by various grammatical formalisms. In *25th Annual Meeting of the Association for Computational Linguistics*, pages 104–111.

Villemonte de la Clergerie, Éric. 2002a. Parsing MCS languages with thread automata. In *Proceedings of the Sixth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+6)*, pages 101–108.

Villemonte de la Clergerie, Éric. 2002b. Parsing mildly context-sensitive languages with thread automata. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7.

Weir, David. 1992. Linear context-free rewriting systems and deterministic tree-walking transducers. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 136–143.

Weir, David J. 1988. *Characterizing Mildly Context-Sensitive Grammar Formalisms*. Ph.D. thesis, University of Pennsylvania.