A revised version is to appear in: S. Pogodalla and J.-P. Prost (eds.), *Logical Aspects of Computational Linguistics (LACL 2011)*, Lecture Notes in Artificial Intelligence Vol. 6736, pp. 112-128, Springer, Berlin, Heidelberg, 2011.

Well-Nestedness Properly Subsumes Strict Derivational Minimalism*

Makoto Kanazawa[1], Jens Michaelis[2], Sylvain Salvati[3], and Ryo Yoshinaka[4]

[1] National Institute of Informatics, Tokyo, Japan
[2] Bielefeld University, Bielefeld, Germany
[3] INRIA Bordeaux – Sud-Ouest, Talence, France
[4] Japan Science and Technology Agency, ERATO MINATO Project, Sapporo, Japan

Abstract. *Minimalist grammars (MGs)* constitute a mildly context-sensitive formalism when being equipped with a particular *locality condition (LC)*, the *shortest move condition*. In this format MGs define the same class of derivable string languages as *multiple context-free grammars (MCFGs)*. Adding another LC to MGs, the *specifier island condition (SPIC)*, results in a proper subclass of derivable languages. It is rather straightforward to see this class is embedded within the class of languages derivable by some *well-nested MCFG (MCFG$_{wn}$)*. In this paper we show that the embedding is even proper. We partially do so adapting the methods used in [13] to characterize the separation of MCFG$_{\text{wn}}$-languages from MCFG-languages by means of a "simple copying" theorem. The separation of strict derivational minimalism from well-nested MCFGs is then characterized by means of a "simple reverse copying" theorem. Since for MGs, *well-nestedness* seems to be a rather *ad hoc* restriction, whereas for MCFGs, this holds regarding the SPIC, our result may suggest we are concerned here with a structural difference between MGs and MCFGs which cannot immediately be overcome in a non-stipulated manner.

1 Introduction

Inspired by the work originating in [1], the formal type of a *minimalist grammar (MG)* has been introduced in [28] as an attempt at a rigorous algebraic formalization of the corresponding perspectives adopted within the linguistic framework of transformational grammar. MGs have been shown to be capable of integrating, if needed, a variety of arguably "odd" items from the syntactician's toolbox such as *head movement* [28, 30], *(strict) remnant movement* [28, 29], *affix hopping* [30], *copy-movement* [14] and *relativized minimality* [31], to mention some.

Interestingly, the formal MG-setting can also be seen as having anticipated some of the crucial developments and changes within the theoretical setting of the minimalist branch of generative grammar since the mid of the 1990s (see e.g. [2, 3]). Maybe the most prominent deviance from at least the original linguistic setting

* This work has essentially been carried out within the joint research project "Open Problems on Multiple Context-Free Grammars" funded by the National Institute of Informatics, Tokyo, Japan.

was that MGs never incorporated any so-called *transderivational constraints*. However, *locality conditions (LCs)* applying to the *move*-operator have always been of decisive nature within the formal MG-framework. A particular LC, the *shortest move condition (SMC)*, played a crucial role in showing that each MG satisfying the definition in [28], and thus, obeying the SMC can be constructively transformed into a *multiple context-free grammar (MCFG)* in the sense of [27] deriving the same string language. The construction presented in [18] has not only proven the corresponding MG-class to be mildly context-sensitive in the sense of [11], but also led to a succinct, "chain-based" reformulation of MGs reducing them to their "bare essentials," cf. [32]. By means of this reformulation, MGs can be straightforwardly interpreted as a proper subtype of MCFGs. In particular, all corresponding MCFGs are of rank 2, i.e., the righthand side of each rule consists of at most two nonterminals. Nevertheless, in terms of derivable string languages the generative power of MCFGs is not reduced as shown independently in [10] and [20].

In particular building on the work in [16], in [29] a revised MG-type has been introduced. Throughout that paper this type is not distinguished by name from the type introduced earlier in [28], although beside the SMC, the revised version implicitly implements a second LC, which has been explicitly referred to as *specifier island condition (SPIC)* in [5] and later work. Closely in keeping with further theoretical linguistic considerations, in [29] also a particular type of a *strict minimalist grammar (SMG)* has been introduced, implementing the SPIC with somewhat more "strictness," and leading to *heavy pied-piping* constructions. In [19, 21] it has been shown that the SMG-class and the MG-class of revised type define identical classes of derivable languages. From this point of view we consider the MG-class of revised type an instance of *strict derivational minimalism*.

With emphasis on particular linguistic aspects, the combinatory power of the SMC and the SPIC within the MG-framework has been discussed in [6], formal results have been proven in different other places: we already mentioned [18] and [10, 20] as the sources showing that the class of MCFGs and the class of MGs obeying the SMC, but not necessarily the SPIC give rise to the same class of derivable string languages. [15] proves MGs obeying the SPIC, but not necessarily the SMC to be Turing complete. [26] shows that the decision problem for MGs neither obeying the SPIC nor the SMC is as hard the the one for proof search in *multiplicative exponential linear logic (MELL)* as introduced in [8].[5]

In [19, 21] it is shown that MGs obeying both the SMC and the SPIC derive the same class of string language as a particular subtype of MCFGs of rank 2, referred to in [19, 21] as the type of an $MCFG_{1,2}$. Here, we refer to this subtype as the type of a *monadic branching MCFG* ($MCFG_{mb}$). It plays the central role in our paper: an $\mathrm{MCFG_{mb}}$ is an MCFG of rank 2 such that for each rule with two nonterminals appearing on the righthand side it holds that from the first nonterminal only simple strings of terminals can be derived, i.e., only 1-tuples of terminal strings can be derived from the first nonterminal instead of k-tuples for

[5] The latter result provides a negative answer to the question, whether all languages generated by such an MG are *semilinear*.

an arbitrary, but fixed $k \geq 1$ as in the general MCFG-case. In fact, it can be shown that the MCFG_{mb}-class constitutes a proper subclass of the full MCFG-class also in terms of derivable string languages, cf. [22]. Figure 1 is summing up the complexity results mentioned so far concerning MGs and the interaction of SMC and SPIC, where $\mathcal{MCFL}$ and $\mathcal{MCFL}_{\text{mb}}$ denote the classes of derivable string languages determined the MCFG-class and the MCFG_{mb}-class, respectively.

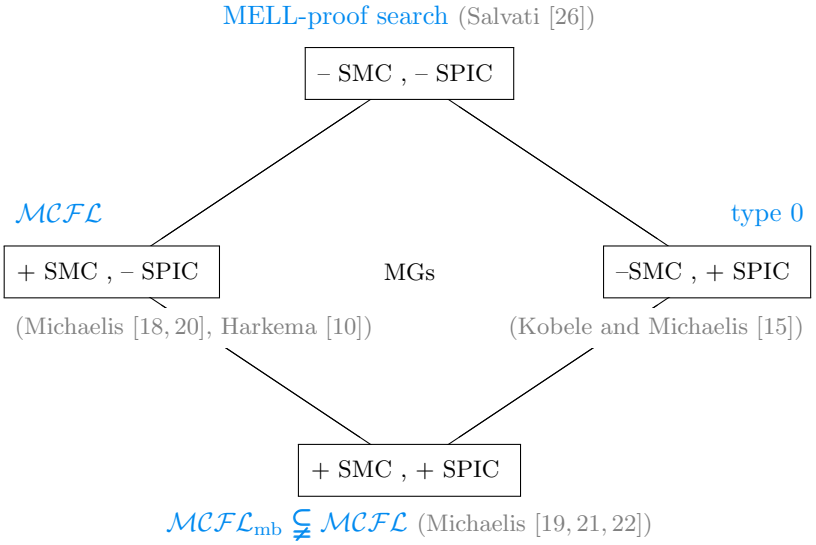

Fig. 1. The interaction of the SMC and the SPIC with the MG-framework.

The proof presented in [22] to separate $\mathcal{MCFL}_{\text{mb}}$ from $\mathcal{MCFL}$ builds on the inclusion of $\mathcal{MCFL}_{\text{mb}}$ within $\mathcal{MCFL}_{\text{wn}}$, the latter being the class of string languages derivable by some *well-nested MCFG (MCFG$_{wn}$)*. $\mathcal{MCFL}_{\text{wn}}$ in its turn constitutes a proper subclass of $\mathcal{MCFL}$. As pointed out, e.g., in [23], the latter was (at least implicitly) known for quite a while. [13], however, crucially presents a separation theorem relying on arguments on "simple copying" which were not available in that form before. Whether the inclusion of $\mathcal{MCFL}_{\text{mb}}$ within $\mathcal{MCFL}_{\text{wn}}$ is proper, has, to the best of our knowledge, been generally open so far. We show here that the answer is positive. We partially do so adapting the methods used in [13]. The separation of $\mathcal{MCFL}_{\text{mb}}$ from $\mathcal{MCFL}_{\text{wn}}$, and thus, of strict derivational minimalism from well-nested MCFGs, is then characterized by means of a "simple reverse copying" theorem.

2 Multiple Context-Free Grammars

A *ranked alphabet* is a finite set Δ of the form $\Delta = \bigcup_{k \geq 0} \Delta^{(k)}$, where $\langle \Delta^{(k)} \mid k \geq 0 \rangle$ is an indexed family of pairwise disjoint sets. The set of *trees (over Δ)*, $\mathcal{T}(\Delta)$, is built up recursively in the following way: If for some $k \geq 0$, we have $d \in \Delta^{(k)}$

and $T_1, \ldots, T_k \in \mathcal{T}(\Delta)$ then $(dT_1 \cdots T_k) \in \mathcal{T}(\Delta)$. In writing trees, we adopt the abbreviatory convention of dropping the outermost parentheses. Note that in case $k = 0$ the string $T_1 \cdots T_k$ is the empty string, ε, Therefore we generally omit the parentheses in this case.

Let N and Σ be a ranked and an unranked alphabet, respectively, with $N^{(0)} = \emptyset$, and assume $X = \{x_i \mid i \geq 0\}$ to be a countably infinite set of variables ranging over Σ^*. A *rule over* $\langle N, \Sigma \rangle$ (or, simply a *rule*, if $\langle N, \Sigma \rangle$ is understood from context) is an expression of the form

$$B_0(\alpha_1, \ldots, \alpha_{k_0}) \leftarrow B_1(x_{1,1}, \ldots, x_{1,k_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,k_n}) \tag{1}$$

for some $n \geq 0$ and $k_i \geq 1$ for $i \in [0, n]$ such that for $i \in [0, n]$, $B_i \in N^{(k_i)}$, and such that for $i \in [1, k_0]$, α_i is a string over $\Sigma \cup \{x_{i,j} \mid i \in [1, n], j \in [1, k_i]\}$, where $\{x_{i,j} \mid i \in [1, n], j \in [1, k_i]\}$ is a set of pairwise distinct variables from X. In addition, each $x_{i,j}$ occurs at most once in $\alpha_1 \cdots \alpha_{k_0}$, the concatenation of all α_i "from left to right." In case $n = 0$ such a rule is *terminating*, otherwise it is *non-terminating*.

Definition 1. A *multiple context-free grammar (MCFG)*, G, is a quadruple $\langle N, \Sigma, P, S \rangle$, where N is a ranked alphabet of *nonterminals* with $N^{(0)} = \emptyset$, where Σ is an unranked alphabet of *terminals*, where P is a finite set of rules over $\langle N, \Sigma \rangle$, and where $S \in N^{(1)}$.

Let $G = \langle N, \Sigma, P, S \rangle$ be an MCFG.

For $k_0 \geq 1$, and corresponding $B_0 \in N^{(k_0)}$ and $w_1, \ldots, w_{k_0} \in \Sigma^*$, we write $\vdash_G B_0(w_1, \ldots, w_{k_0})$ to mean that $B_0(w_1, \ldots, w_{k_0})$ is *derivable (in G)* according to the following inference scheme:

$$\frac{\vdash_G B_1(w_{1,1}, \ldots, w_{1,k_1}) \ \ldots \ \vdash_G B_n(w_{n,1}, \ldots, w_{n,k_n})}{\vdash_G B_0(\alpha_1, \ldots, \alpha_{k_0})\sigma} \tag{2}$$

where $B_0(\alpha_1, \ldots, \alpha_{k_0}) \leftarrow B_1(x_{1,1}, \ldots, x_{1,k_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,k_n})$ is a rule in P according to (1), where $w_{i,j} \in \Sigma^*$, and where σ is the substitution which maps each variable $x_{i,j}$ to $w_{i,j}$.

The *language derivable by* G, $L(G)$, is the set $\{w \in \Sigma^* \mid \vdash_G S(w)\}$.

Definition 2. A *multiple context-free language (MCFL)* is a set (of strings), L, such that there is an MCFG, G, with $L(G) = L$.

Let $G = \langle N, \Sigma, P, S \rangle$ be an MCFG.

If $A \in N^{(k)}$ for some $k \geq 1$ then k is the *arity of* A, denoted by $arity(A)$. The *dimension of* G is defined as the maximum of $\{arity(A) \mid A \in N\}$.

If $B_0(\alpha_1, \ldots, \alpha_{k_0}) \leftarrow B_1(x_{1,1}, \ldots, x_{1,k_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,k_n})$ is some rule $p \in P$ according to (1) then the number n is the *rank of* p, denoted $rank(p)$. Thus, p is terminating iff $rank(p) = 0$. The *rank of* G, denoted $rank(G)$, is defined as the maximum of $\{rank(p) \mid p \in P\}$.

For $m, r \geq 1$, an *m-MCFG(r)* is an MCFG, G, of dimension at most m and rank at most r. An *m-MCFL(r)* is an MCFL, L, such that there is an m-MCFG(r), G, with $L(G) = L$ for some $m \geq 1$ and $r \geq 1$.

We denote by m-$\mathcal{MCFG}(r)$ the class of all MCFGs of dimension at most m and rank at most r, and by m-$\mathcal{MCFL}(r)$ the class of all MCFLs generated by some m-MCFG(r).

We let m-$\mathcal{MCFG}$, $\mathcal{MCFG}(r)$, $\mathcal{MCFG}$, m-$\mathcal{MCFL}$, $\mathcal{MCFL}(r)$ and $\mathcal{MCFL}$ denote the classes $\bigcup_{r\geq 1} m\text{-}\mathcal{MCFG}(r)$, $\bigcup_{m\geq 1} m\text{-}\mathcal{MCFG}(r)$, $\bigcup_{m,r\geq 1} m\text{-}\mathcal{MCFG}(r)$, $\bigcup_{r\geq 1} m\text{-}\mathcal{MCFL}(r)$, $\bigcup_{m\geq 1} m\text{-}\mathcal{MCFL}(r)$ and $\bigcup_{m,r\geq 1} m\text{-}\mathcal{MCFL}(r)$, respectively.

Theorem 1. *Let $L = \{w\#w^R \mid w \in L_0\}$ for some set of strings L_0.*

(i) If for some $m, r \geq 1$, $L_0 \in m$-$\mathcal{MCFL}(r)$ then $L \in 2m$-$\mathcal{MCFL}(r)$.
(ii) If for some $m, r \geq 1$, $L \in m$-$\mathcal{MCFL}(r)$ then $L_0 \in m$-$\mathcal{MCFL}(r)$.

Proof (sketch). Let $m, r \geq 1$. (i): constructing an $2m$-MCFG(r) G with $L(G) = L$, from a given m-MCFG(r) G_0 with $L(G_0) = L_0$, is straightforward. (ii): the class of m-$\mathcal{MCFL}(r)$ is closed under rational transductions. □

Let $G = \langle N, \Sigma, P, S\rangle$ be an MCFG.

In order to be able to talk about derivation trees of derivable facts, we will identify P with a ranked alphabet Δ_P relying on a bijection $f : P \to \Delta_P$ such that for each $p \in P$, $f(p) \in \Delta^{(n)}$ iff $rank(p) = n$. *Derivation trees* are trees over the ranked alphabet Δ_P. *Derivation trees contexts* are trees over the ranked alphabet $\Delta_P(Y)$, where $\Delta_P(Y)^{(n)} = \Delta_P^{(n)}$ for $n \geq 1$, and $\Delta_P(Y)^{(0)} = \Delta_P^{(0)} \cup Y$ with $Y = \{y_i \mid i \geq 0\}$ being a countably infinite set of variables disjoint from Δ_P. The following inference system associates derivation trees with derivable facts and derivation tree contexts with facts derivable from some premises:

$$\frac{}{y : B(x'_1, \ldots, x'_k) \vdash y : B(x'_1, \ldots, x'_k)} \tag{3}$$

$$\frac{\Gamma_1 \vdash_G T_1 : B_1(\beta_{1,1}, \ldots, \beta_{1,k_1}) \ \ldots \ \Gamma_n \vdash_G T_n : B_n(\beta_{n,1}, \ldots, \beta_{n,k_n})}{\Gamma_1, \ldots, \Gamma_n \vdash_G pT_1 \cdots T_n : B_0(\alpha_1, \ldots, \alpha_{k_0})\sigma} \tag{4}$$

In the first scheme, (3), it holds that $y \in Y$, $B \in N^{(k)}$ for some $k \geq 1$ and $x'_i \in X$. In the second scheme, (4), it holds that p is a rule from P of the form

$$B_0(\alpha_1, \ldots, \alpha_{k_0}) \leftarrow B_1(x_{1,1}, \ldots, x_{1,k_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,k_n})$$

according to (1), $\beta_{i,j} \in (\Sigma \cup X)^*$, and σ is a substitution mapping each $x_{i,j}$ to $\beta_{i,j}$. Each Γ_i is a finite sequence of premises of the form $z : C(x'_1, \ldots, x'_k)$ with $z \in Y$, and with $C \in N^{(k)}$ for some $k \geq 1$ and $x'_i \in X$. It is also understood that Γ_i and Γ_j do not share any variables if $i \neq j$. Each T_i is a derivation tree context over $\Delta_P(Y)$. For $k \geq 1$, $A \in N^{(k)}$ and $w_i \in \Sigma^*$ for $i \in [1, k]$, it clearly holds that $\vdash_G A(w_1, \ldots, w_k)$ iff $\vdash_G T : A(w_1, \ldots, w_k)$ for some derivation tree T over Δ_P.

For each $k \geq 1$, $A \in N^{(k)}$ is *useful* if there are $w_i \in \Sigma^*$ for $i \in [1, k]$ such that $\vdash_G A(w_1, \ldots, w_k)$, and if there are $y \in Y$, $x'_i \in X$ for $i \in [1, k]$, $\alpha \in (\Sigma \cup X)^*$ and some derivation tree T over $\Delta_P(Y)$ such that $y : A(x'_1, \ldots, x'_k) \vdash_G T : S(\alpha)$. $A \in N^{(k)}$ is *useless* if it is not useful.

Let $B_0(\alpha_1, \ldots, \alpha_{k_0}) \leftarrow B_1(x_{1,1}, \ldots, x_{1,k_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,k_n})$ be some rule $p \in P$ according to (1).

- p is *non-deleting* if for $i \in [1,n]$ and $j \in [1,k_i]$, $x_{i,j}$ occurs in $\alpha_1 \cdots \alpha_{k_0}$. (5)

- p is *non-permuting* if for $i \in [1,n]$ and $j,k \in [1,k_i]$, $j < k$ implies that the occurence (if any) of $x_{i,j}$ in $\alpha_1 \cdots \alpha_{k_0}$ precedes the occurence (if any) of $x_{i,k}$ in $\alpha_1 \cdots \alpha_{k_0}$. (6)

- p is *well-nested* if it is non-deleting and non-permuting, and for every $i, i' \in [1,n]$ with $i \neq i'$, $j \in [1, k_i - 1]$ and $j' \in [1, k_{i'} - 1]$, it additionally satisfies: (7)
$\alpha_1 \cdots \alpha_{k_0}$
$\notin (\Sigma \cup X)^* x_{i,j} (\Sigma \cup X)^* x_{i',j'} (\Sigma \cup X)^* x_{i,j+1} (\Sigma \cup X)^* x_{i',j'+1} (\Sigma \cup X)^*$.

- p is *monadic branching* if $n \leq 2$, and $n = 2$ implies $k_1 = 1$. (8)

Note that, if each $p \in P$ is non-deleting then G is a *linear context-free rewriting system (LCFRS)* in the sense of [33]; if each $p \in P$ is non-permuting then G is an MCFG in *monotone function form* in the sense of [19]; and if each $p \in P$ is non-deleting and non-permuting then G is an *ordered simple RCG* in the sense of [34] as well as a *monotone LCFRS* in the sense of [17].

Definition 3. An MCFG $G = \langle N, \Sigma, P, S \rangle$ is *well-nested* if each rule $p \in P$ is well-nested in the sense of (7).

Definition 4. An MCFG $G = \langle N, \Sigma, P, S \rangle$ is *monadic branching* if each rule $p \in P$ is monadic branching in the sense of (8).

We attach the subscripts "wn" and/or "mb" to "MCFG" and "MCFL" in order to refer to a well-nested and/or monadic branching MCFG and MCFL of corresponding type. More concretely, we write "MCFG_x," "m-MCFG_x," "$\mathrm{MCFG}_x(r)$" and "m-$\mathrm{MCFG}_x(r)$" as well as "MCFL_x," "m-MCFL_x," "$\mathrm{MCFL}_x(r)$" and "m-$\mathrm{MCFL}_x(r)$" with x being of the form "wn", "mb" or "wn,mb."

We likewise do so with regard to "$\mathcal{MCFG}$" and "$\mathcal{MCFL}$" and the corresponding (sub-)classes of MCFGs and MCFLs.

Corollary 1. *For $m \geq 1$, m-$\mathcal{MCFL}(1) = m$-$\mathcal{MCFL}_{mb}(1) = m$-$\mathcal{MCFL}_{wn}(1)$.*

Proof. For $m \geq 1$, m-$\mathcal{MCFL}(1)$ and m-$\mathcal{MCFL}_{\mathrm{mb}}(1)$ are identical by definition. The identity of m-$\mathcal{MCFL}_{\mathrm{mb}}(1)$ and m-$\mathcal{MCFL}_{\mathrm{wn}}(1)$ is a special case of Proposition 1, because we have m-$\mathcal{MCFL}_{\mathrm{wn,mb}}(1)$=$m$-$\mathcal{MCFL}_{\mathrm{wn}}(1)$. □

Theorem 2. $\mathcal{MCFL} = \mathcal{MCFL}(2)$

Theorem 3. *For $m \geq 1$, m-$\mathcal{MCFL}_{wn} = m$-$\mathcal{MCFL}_{wn}(2)$.*

Theorem 2 is a corollary of Theorem 11 of [24]. Theorem 3 is Lemma 5 of [13].

Theorem 4. *For any $m, r \geq 1$ let G be an m-MCFG(r).*

(i) There is a non-deleting m-MCFG(r), G', such that $L(G) = L(G')$.
(ii) If $G \in \mathcal{MCFG}_{mb}$ then G' from (i) can also be chosen from $\mathcal{MCFG}_{mb}$.

Proof. This is Corollary 2.2.10 of [19] which essentially follows from both Lemma 2.2 and its concrete proof in [27]. □

Proposition 1. *For $m \geq 1$, $r \in [1, 2]$, m-$\mathcal{MCFL}_{mb}(r) = m$-$\mathcal{MCFL}_{wn,mb}(r)$.*

Proof. Let $G \in m$-$\mathcal{MCFG}_{\mathrm{mb}}(r)$ for some $m \geq 1$ and $r \in [1, 2]$. By (ii) of the last theorem we can w.l.o.g. assume G to be non-deleting. Now, transform G into its non-permuting "closure", i.e. a non-deleting and non-permuting m-$\mathrm{MCFG}_{\mathrm{mb}}(r)$, G', deriving the same language (cf. Construction 2.4.3 and Corollary 2.4.4 in [19]). Since each rule in G' is not only non-deleting and non-permuting, but also monadic branching, well-nestedness of such a rule holds straightforwardly. □

Proposition 2. $\mathcal{MCFL}(1) \subsetneqq \mathcal{MCFL}_{mb}$.

Proof. $\mathcal{MCFL}(1) \subseteq \mathcal{MCFL}_{\mathrm{mb}}$ is an immediate consequence of the corresponding definitions. Because of

- $\mathcal{MCFL}(1) = \mathcal{ET0L}_{\mathrm{fin}}$ and $\mathcal{ET0L}_{\mathrm{fin}} \subseteq \mathcal{EDT0L}$, cf. [4] and [24],[6]
- $\mathcal{CFL} - \mathcal{EDT0L} \neq \emptyset$, cf. [4], and
- $\mathcal{CFL} = 1$-$\mathcal{MCFL}(2)$ and 1-$\mathcal{MCFL}(2) \subseteq \mathcal{MCFL}_{\mathrm{mb}}$

even proper inclusion of $\mathcal{MCFL}(1)$ within $\mathcal{MCFL}_{\mathrm{mb}}$ holds. □

3 Separating $\mathcal{MCFL}_{\mathbf{wn}}$ from $\mathcal{MCFL}$

In this section we briefly recapitulate the main results from [13], in order to emphasize the analogies and differences to the way of separating $\mathcal{MCFL}_{\mathrm{mb}}$ from $\mathcal{MCFL}_{\mathrm{wn}}$ presented in the next section. The first theorem is Theorem 8 of [13].

Theorem 5 (copying theorem for $\mathcal{MCFL}_{\mathbf{wn}}$). *If for some set of strings L_0, $L = \{w\#w \mid w \in L_0\}$ holds then for each $m \geq 1$, the following are equivalent:*

(i) $L \in m$-$\mathcal{MCFL}_{wn}$. *(ii) $L \in m$-$\mathcal{MCFL}(1)$.*

Corollary 2. *If for some set of strings L_0, $L = \{w\#w \mid w \in L_0\}$ holds then the following are equivalent:*

(i) $L \in \mathcal{MCFL}_{wn}$. *(ii) $L \in \mathcal{MCFL}(1)$.* *(iii) $L_0 \in \mathcal{MCFL}(1)$.*

This is Corollary 9 of [13]. It is proven there relying on two theorems, namely, the one presented here as Theorem 5 and the equivalent version of our Theorem 1 taking into account the language $\{w\#w \mid w \in L_0\}$ instead of $\{w\#w^R \mid w \in L_0\}$.

Theorem 6 (separation theorem for $\mathcal{MCFL}_{\mathbf{wn}}$). $\mathcal{MCFL}_{wn} \subsetneqq \mathcal{MCFL}$.

This is Corollary 10 of [13] following from the last corollary combined with the facts that $\mathcal{CFL} - \mathcal{MCFL}(1) \neq \emptyset$, and that for $L_0 \in \mathcal{CFL}$, $\{w\#w \mid w \in L_0\} \in \mathcal{MCFL}$.

[6] $\mathcal{CFL}$ denotes the class of all context-free languages. For definitions of the language classes $\mathcal{EDT0L}$ and $\mathcal{ET0L}_{\mathrm{fin}}$ as well as their origins see [4].

4 Separating $\mathcal{MCFL}_{\mathbf{mb}}$ from $\mathcal{MCFL}_{\mathbf{wn}}$

We start by presenting an analog to the copying theorem for $\mathcal{MCFL}_{\mathrm{wn}}$.

Theorem 7 (reverse copying theorem for $\mathcal{MCFL}_{\mathbf{mb}}$). *If for some set of strings L_0, $L = \{w\#w^R \mid w \in L_0\}$ holds then for each $m \geq 1$, (i') implies (ii'):*

(i') $L \in m\text{-}\mathcal{MCFL}_{mb}$.
(ii') $L \in m+1\text{-}\mathcal{MCFL}_{wn}(1)$ and $L_0 \in m+1\text{-}\mathcal{MCFL}_{wn}(1)$.

Corollary 3. *If for some set of strings L_0, $L = \{w\#w^R \mid w \in L_0\}$ holds then the following are equivalent:*

(i) $L \in \mathcal{MCFL}_{mb}$. *(ii) $L \in \mathcal{MCFL}(1)$.* *(iii) $L_0 \in \mathcal{MCFL}(1)$.*

Proof. "(iii)⇒(ii)": special case of Theorem 1. "(ii)⇒(i)": cf. Proposition 2. "(i)⇒(iii)": this is a corollary of Theorem 7. □

Lemma 1. *For each $L_0 \in \mathcal{CFL}$, $L = \{w\#w^R \mid w \in L_0\} \in 2\text{-}\mathcal{MCFL}_{wn}$.*

Proof. Starting, e.g., with a CFG in Chomsky normal form generating L_0, the construction of an 2-$\mathrm{MCFG}_{\mathrm{wn}}(2)$ generating L is straightforward. □

Theorem 8 (separation theorem for $\mathcal{MCFL}_{\mathbf{mb}}$). $\mathcal{MCFL}_{mb} \subsetneqq \mathcal{MCFL}_{wn}$.

Proof. Choose existing $L_0 \in \mathcal{CFL} - \mathcal{MCFL}(1)$. Then, by Theorem 7 and 1, $L = \{w\#w^R \mid w \in L_0\} \in 2\text{-}\mathcal{MCFL}_{\mathrm{wn}} - \mathcal{MCFL}_{\mathrm{mb}}$. □

The remaining part of this section is devoted to a detailed description of the crucial points underlying a proof of Theorem 7.

Proof (sketch) of Theorem 7. For some $m \geq 1$, let $L \in m\text{-}\mathcal{MCFL}_{\mathrm{mb}}$. When having shown that $L \in m+1\text{-}\mathcal{MCFL}_{\mathrm{wn}}(1)$ holds, $L_0 \in m+1\text{-}\mathcal{MCFL}_{\mathrm{wn}}(1)$ follows from Theorem 1 and Corollary 1(ii).

Let $G = \langle N, \Sigma \cup \{\#\}, P, S\rangle$ be an m-$\mathrm{MCFG}_{\mathrm{mb}}$ with $L(G) = L$. W.l.o.g. G is well-nested by Proposition 1, thus, in particular, each $p \in P$ is non-deleting. Moreover, we can w.l.o.g. assume that each $A \in N$ is useful and derives an infinite set of tuples of strings over Σ, i.e., $\{\langle w_1, \ldots, w_k\rangle \in (\Sigma^*)^k \mid \vdash_G A(w_1, \ldots, w_k)\}$ is infinite for $k = \mathit{arity}(A)$. Trivially, Σ can be chosen such that $\Sigma \cap \{\#\} = \emptyset$.

Depending on G, in (21)-(24) we construct a $G' \in m+1\text{-}\mathcal{MCFG}_{\mathrm{wn}}(1)$ with $L(G') = L$. Before doing so, the crucial properties of G virtually employed by G' are carefully revealed step by step, and the presented technical details providing a precise characterization of those properties are summed up in Fig. 4. In a nutshell, we are concerned with the following situation as to G:

> if $\vdash_G T : S(\widehat{w}\,\#\,\widehat{w}^R)$ for some derivation tree T over Δ_P and $\widehat{w} \in \Sigma^*$, then looking at T from a bottom-up perspective, the unique instance of "#" appearing in the derived string $\widehat{w}\#\widehat{w}^R$ is successively passed on upward from the leftmost leaf of the tree to the root, i.e. along the tree's leftmost path, and within no other node of the tree any instance of # is created or manipulated in another way. (9)

Consider $p \in P$ with $rank(p) = 2$. Because it is monadic branching, p is of the form $B_0(\alpha_1, \ldots, \alpha_{k_0}) \leftarrow B_1(x_{1,1}, \ldots, x_{1,k_1}), \ldots, B_n(x_{n,1}, \ldots, x_{n,k_n})$ according to (1) such that $n = 2$ and $k_1 = 1$. We set $A = B_0$, $B = B_1$ and $C = B_2$, and also $k = k_0$, $l = k_2$, $x'_0 = x_{1,1}$ and $x'_i = x_{2,i}$ for $i \in [1, k_2]$. Thus, p is of the form

$$A(\alpha_1, \ldots, \alpha_k) \leftarrow B(x'_0), C(x'_1, \ldots, x'_l) \tag{10}$$

Since A, B and C are useful, there are $v_i \in (\Sigma \cup \{\#\})^*$ for $i \in [1, k]$, $u_i \in (\Sigma \cup \{\#\})^*$ for $i \in [0, l]$ and derivation trees T_B and T_C over Δ_P, and there are $y \in Y$, $x''_i \in X$ for $i \in [1, k]$, $\alpha \in (\Sigma \cup \{x''_i \mid i \in [1, k]\})^*$ and a derivation tree context $\widetilde{T}_S$ over $\Delta_P(Y)$ such that

$$\vdash_G pT_BT_C : A(v_1, \ldots, v_k) \quad \text{and} \quad y : A(x''_1, \ldots, x''_k) \vdash_G \widetilde{T}_S : S(\alpha) \tag{11}$$

and

$$\vdash_G T_B : B(u_0) \quad \text{and} \quad \vdash_G T_C : C(u_1, \ldots, u_l) \tag{12}$$

We will crucially show that (16) and (18) and, therefore, (20) hold, i.e., we will show a) that u_0 contains exactly one instance of $\#$, while for $i \in [1, l]$, u_i does not contain any such instance, b) that $l > 1$ and c) that, therefore, in case $k > 1$, A cannot appear itself on the righthand side of any strictly binary rule from P. These properties essentially imply (9).

a) Since by choice of G each rule is non-deleting, and because each $\widetilde{w} \in L(G)$ is of the form $w\#w^R$ for some $w \in \Sigma^*$, from (11) and (12), it follows that

$$\begin{array}{l} u_i \in \Sigma^*\{\#, \varepsilon\}\Sigma^* \text{ holds for each } i \in [0, l], \text{ but } u_i \in \Sigma^*\{\#\}\Sigma^* \text{ is true} \\ \text{for at most one } i \in [0, l], \end{array} \tag{13}$$

and, in particular, there exist a unique $j_0 \in [1, k]$ and $v, \overline{v} \in (\Sigma \cup \{\#\})^*$ with

$$v_{j_0} = v\, u_0\, \overline{v} \tag{14}$$

Suppose, $u_0 \in \Sigma^*$. Then again, because of (11) and (12), and since by choice of G each of its rules is non-deleting, there are w_1, $w_2 \in \Sigma^*$ such that, w.l.o.g.,

$$\vdash_G S(w_1u_0w_2\#(w_1u_0w_2)^R) \quad \text{and thus,} \quad \vdash_G S(w_1u'w_2\#(w_1u_0w_2)^R) \tag{15}$$

whenever $\vdash_G T'_B : B(u')$ for some $u' \in \Sigma^*$ and some derivation tree T'_B over Δ_P.

Figure 2 depicts the situation as fixed in (11)-(15). However, having chosen G such that, in particular, the nonterminal B derives an infinite set of strings over Σ, (15) yields a contradiction to the fact that each element in $L(G)$ is of the form $w\#w$ for some $w \in \Sigma^*$. In other words, in combination with (13), we have

$$u_0 \in \Sigma^*\{\#\}\Sigma^* \quad \text{and} \quad u_i \in \Sigma^* \text{ for } i \in [1, l] \tag{16}$$

b) Suppose, $l = 1$. Then, we can again derive a contradiction. We can do so analogously to the case resulting from the assumption that $u_0 \in \Sigma^*$: because $u_1 \in \Sigma^*$ by (16), we can conclude that there are w_1, $w_2 \in \Sigma^*$ such that, w.l.o.g.,

$$\vdash_G S(w_1u_1w_2\#(w_1u_1w_2)^R) \quad \text{and thus,} \quad \vdash_G S(w_1u'w_2\#(w_1u_1w_2)^R) \tag{17}$$

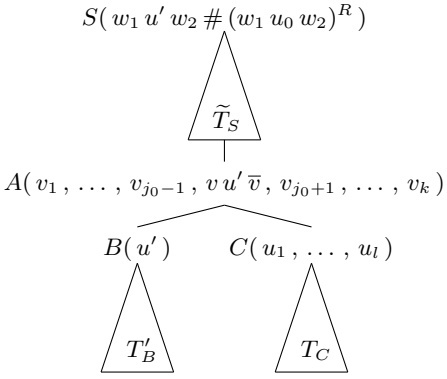

Fig. 2. Derivation tree according to (11)-(15).

whenever $\vdash_G C(u')$ for some $u' \in \Sigma^*$. By choice of G, the nonterminal C derives an infinite set of strings over Σ, and therefore the assumption $l = 1$ allows us to derive strings from S which are not in $L(G)$. Thus, it must hold that

$$l > 1 \tag{18}$$

c) Let T_S be the derivation tree over Δ_P which results from substituting the variable $y \in Y$ within the derivation context $\widetilde{T}_S$ over $\Delta_P(Y)$ by pT_BT_C. Recall, once more, that each rule of G is non-deleting. Thus, from (11)-(14) and (16) it, moreover, follows that there are $u, \overline{u}, w, \overline{w} \in \Sigma^*$ such that

$$u_0 = u \,\#\, \overline{u} \quad , \quad v_i \in \Sigma^* \text{ for } i \neq j_0 \quad \text{and} \quad \vdash_G T_S : S(w\, v\, u \,\#\, \overline{u}\, \overline{v}\, \overline{w}) \tag{19}$$

The situation as fixed in (11)-(14), (16) and (19) is displayed in Fig. 3. Taking into account the above considerations on the nonterminal C, in particular, the properties expressed in (16) and (18), it becomes clear that in case $k > 1$,

$$A \text{ cannot appear on the righthand side of any } p' \in P \text{ with } \mathit{rank}(p') = 2. \tag{20}$$

If A did so, $L(G)$ would, contradicting its definition, include a string consisting of more than one instance of #. Recall that $v_{j_0} \in \Sigma^*\{\#\}\Sigma^*$ by (14) and (16).

Thus, T_S and $wvvu$ are in fact respective instances of a derivation tree T and a string $\widehat{w}$ in the sense of the above "nutshell" (9). More concretely, for some $m(S) \geq 0$ there is a finite sequence of derivation tree contexts over $\Delta_P(Y)$, $\langle V_j \rangle_{0 \leq j \leq m(S)}$, such that V_0 is a tree over Δ_P with no occurrence of variables, and such that for $j \in [1, m(S)]$, V_j is a tree over $\Delta_P(\{y_j\})$ with exactly on instance of y_j occurring in V_j.

Furthermore, if $W_0 = V_0$, and if for $j \in [1, m(S)]$, W_j is the result of substituting y_j within V_j by W_{j-1} then $W_{m(S)} = T_S$.

Each V_j is built up in the following way:

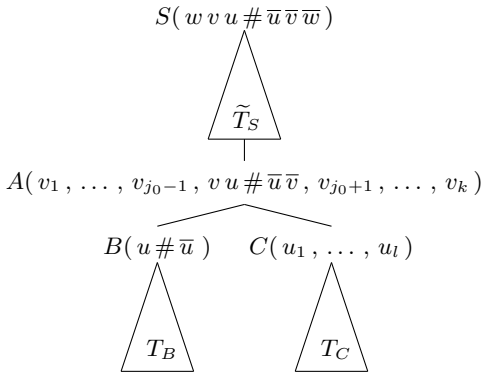

Fig. 3. Derivation tree T_S and intermediately derived "objects."

- There are particular numbers $n(j) = n \geq 0$ and $s_i \geq 0$ for $i \in [0, n]$.
- There are nonterminals $B^{(i)} \in N$ and $C^{(i,i')} \in N$ for $i \in [0, n]$ and $i' \in [0, s_i]$ with $\mathit{arity}(B^{(i)}) = 1$ for $i \in [1, n]$, $\mathit{arity}(C^{(0,0)}) = 1$, and $C^{(0,s_0)} = S$ if $s_0 = 0$.
- We let
$$\begin{aligned} r_i &:= \mathit{arity}(B^{(i)}) \quad \text{for} \quad i \in [0, n] \\ l(i, i') &:= \mathit{arity}(C^{(i,i')}) \quad \text{for} \quad i \in [0, n]\ ,\ i' \in [0, s_i] \end{aligned}$$
- Then, there is a set of pairwise distinct variables
$$X_j = \{x_{i''}^{(i)}, x_{i'''}^{(i,i')} \mid i \in [0, n], i'' \in [1, r_i], i' \in [0, s_i], i''' \in [1, l(i, i')]\} \subseteq X$$

and there are

$$\begin{aligned} \alpha_{i''}^{(i)} &\in (\Sigma \cup \{\#\} \cup X_j)^* \quad \text{for} \quad i \in [0, n]\ ,\ i'' \in [0, r_i] \\ \alpha_{i''}^{(0)} &\in (\Sigma \cup \{\#\})^* \quad \text{for} \quad i'' \in [0, r_0] \quad \text{in case} \quad n = 0 \\ \beta_{i'''}^{(0,i')} &\in (\Sigma \cup \{\#\} \cup X_j)^* \quad \text{for} \quad i' \in [0, s_0]\ ,\ i''' \in [1, l(0, i')] \\ \beta_{i'''}^{(i,i')} &\in (\Sigma \cup X_j)^* \quad \text{for} \quad i \in [1, n]\ ,\ i' \in [0, s_i - 1]\ ,\ i''' \in [1, l(i, i')] \\ u_{i'''}^{(i,s_i)} &\in \Sigma^* \quad \text{for} \quad i \in [1, n]\ ,\ i''' \in [1, l(i, s_i)] \end{aligned}$$

- such that for $i \in [0, n-1]$, there are non-terminating rules from P of the form
$$p^{(i)} = B^{(i)}(\alpha_1^{(i)}, \ldots, \alpha_{r_i}^{(i)}) \leftarrow B^{(i+1)}(x_1^{(i+1)})\, , C^{(i+1,0)}(x_1^{(i+1,0)}, \ldots, x_{l(i+1,0)}^{(i+1,0)})$$

and such that in case $n = 0$,[7] there is a terminating rule from P of the form

$$p^{(0)} = B^{(0)}(\alpha_1^{(0)}, \ldots, \alpha_{r_0}^{(0)}) \leftarrow$$

- Furthermore, for $i \in [1, n]$, there are terminating rules from P of the form
$$q^{(i,s_i)} = C^{(i,s_i)}(u_1^{(i,s_i)}, \ldots, u_{l(i,s_i)}^{(i,s_i)}) \leftarrow$$

while $\quad q^{(0,s_0)} = p^{(0)} \quad$, $\quad$ implying that $\quad C^{(0,s_0)} = B^{(0)} \quad$, $\quad$ and

[7] Note that $n = 0$ implies $\{p^{(i)} \mid i \in [0, n-1]\} = \emptyset$.

for $i \in [0,n]$, $i' \in [0, s_i - 1]$, there are unary branching rules from P of the form

$$q^{(i,i')} = C^{(i,i')}(\beta_1^{(i,i')}, \ldots, \beta_{l(i,i')}^{(i,i')}) \leftarrow C^{(i,i'+1)}(x_1^{(i,i'+1)}, \ldots, x_{l(i,i'+1)}^{(i,i'+1)})$$

- For $i \in [1,n]$, we now define derivation trees over Δ_P by

$$T^{(i,s_i)} := q^{(i,s_i)} \quad \text{and} \quad T^{(i,i')} := q^{(i,i')} T^{(i,i'+1)} \quad \text{for} \quad i' \in [0, s_i - 1]$$

and for $y_j \in Y$, we define derivation tree contexts over $\Delta_P(\{y_j\})$ by

$$\begin{array}{rll} U^{(n-1)} := p^{(n-1)} y_j T^{(n,0)} & & \text{in case} \quad n > 0 \\ U^{(i)} := p^{(i)} U^{(i+1)} T^{(i+1,0)} & & \text{for} \quad i \in [0, n-2] \\ U^{(0)} := p^{(0)} & & \text{in case} \quad n = 0 \end{array}$$

Finally, we set

$$T^{(0,s_0)} := U^{(0)} \ , \ T^{(0,i')} := q^{(0,i')} T^{(0,i'+1)} \ \text{ for } \ i' \in [0, s_0 - 1] \ \text{ and } \ V_j := T^{(0,0)}$$

- Thus,

$$y_j : B^{(n)}(x_1^{(n)}) \vdash_G V_j : C^{(0,0)}(\beta_>) \ \text{ if } \ n > 0 \quad \text{and} \quad \vdash_G V_j : C^{(0,0)}(\beta_=) \ \text{ if } \ n = 0$$

for some $\beta_> \in (\Sigma \cup \{\#, x_1^{(n)}\})^*$ if $n > 0$, and $\beta_= \in (\Sigma \cup \{\#\})^*$ if $n = 0$. Recall that $arity(C^{(0,0)}) = 1$ in general, and that $arity(B^{(n)}) = 1$ in case $n > 0$.

Figure 4 aims at making the formal setting as it regards the derivation tree context V_j somewhat more accessible. Note that for $i \in [0, n-1]$, the respective calculation of the contributions of $B^{(i+1)}$ and $C^{(i+1,0)}$ to $B^{(i)}$ are independent of each other. Crucially, from a bottom-up perspective, the calculation of the contribution of $B^{(i+1)}$ can be done first and can be stored in a buffer, while calculating the contribution of C^{i+1}. In terms of the arity of a nonterminal the buffer size needed is 1, since for each $i \in [0, n-1]$, $B^{(i+1)}$ has arity 1. Exactly this property is used below in order to define the $m+1$-MCFG(1), G', based on the given m-MCFG$_{\rm mb}$, G, with $L(G') = L(G)$: in terms of the transformed grammar, the subderivation trees $T(i,0)$ for $i \in [1,n]$, i.e. the "ⓘ-parts" of the original derivation tree context V_j (cf. Fig. 4), become integral parts of the leftmost path resulting in a completely unary branching derivation tree (cf. Fig. 5).

For $\{[A/B] \mid A, B \in N\}$ being a set of pairwise distinct new symbols, define now $G' = \langle N', \Sigma, P', S \rangle \in m+1\text{-}\mathcal{MCFG}(1)$ depending on G with $L(G') = L$.

- The set of nonterminals $N' = \bigcup_{k \geq 0} N'^{(k)}$ is defined by $N'^{(0)} = \emptyset$ and

$$N'^{(k+1)} = N^{(k+1)} \cup \{[A/B] \mid A \in N^{(k)}, B \in N^{(1)}\} \quad \text{for } k \geq 0 \qquad (21)$$

- In order to define the rule set P', we distinguish three types of rules in P.

– A binary branching $p \in P$ is of the form $A(\alpha_1, \ldots, \alpha_k) \leftarrow B(x_0'), C(x_1', \ldots, x_l')$ in accordance with (10). For each such $p \in P$ we let

$$A(\alpha_1, \ldots, \alpha_k) \leftarrow [C/B](x_0', x_1', \ldots, x_l') \in P' \qquad (22)$$

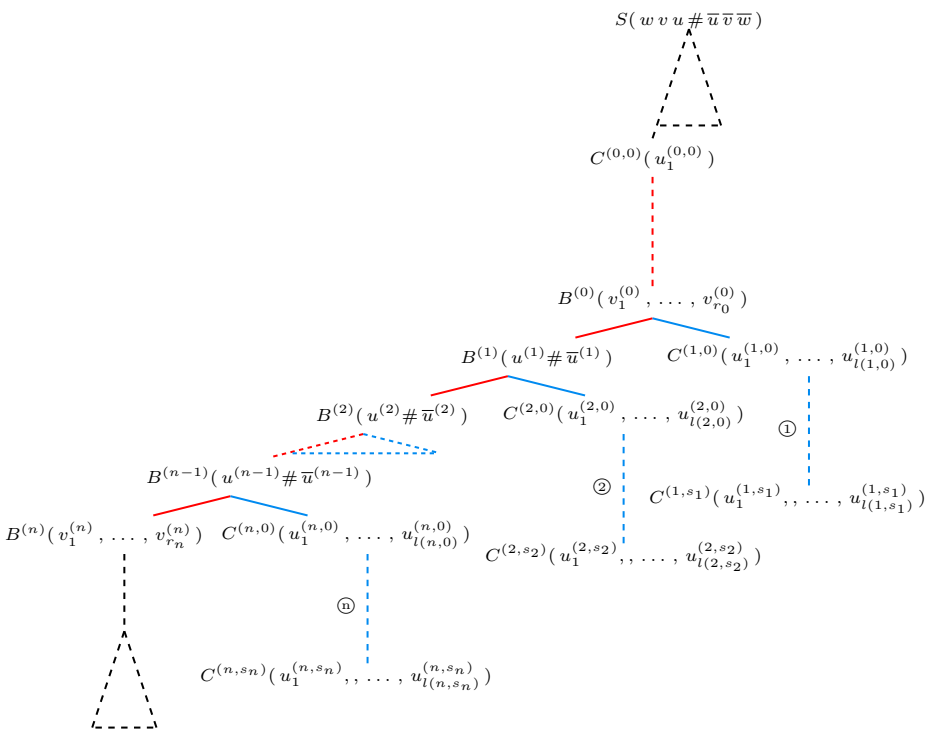

Fig. 4. Typical configuration within derivation tree T_S corresponding to derivation tree context V_j over $\Delta_P(\{y_j\})$ for arbitrary $j \in [0, m(S)]$.

– A unary branching $p \in P$ is of the form $A(\alpha_1, \ldots, \alpha_k) \leftarrow C(x'_1, \ldots, x'_l)$ according to (1), where $k = k_0$, $l = k_1$, $A = B_0$, $C = B_1$, $x'_i = x_{1,i}$ for $i \in [1, l]$. For each such $p \in P$, each $B \in N^{(1)}$, and some $x'_0 \in X - \{x'_i \,|\, i \in [1, l]\}$ let

$$p \in P' \quad \text{and} \quad [A/B](x'_0, \alpha_1, \ldots, \alpha_k) \leftarrow [C/B](x'_0, x'_1, \ldots, x'_l) \in P' \tag{23}$$

– A terminating $p \in P$ is of the form $A(w_1, \ldots, w_k) \leftarrow$ in accordance with (1), where $k = k_0$, $A = B_0$ and $w_i = \alpha_i$ for $i \in [1, k]$. For each such $p \in P$, each $B \in N^{(1)}$, and some $x'_0 \in X - \{x'_i \,|\, i \in [1, l]\}$ let

$$p \in P' \quad \text{and} \quad [A/B](x'_0, w_1, \ldots, w_k) \leftarrow B(x'_0) \in P' \tag{24}$$

An induction on the length of a derivation showed that $L(G') = L(G)$. Due to (20), in G' we do not have to "lift" by means of "$[\cdot/B]$" over the lefthand side of a binary branching rule from G, cf. (22). Rather, the "lifting" instantiated in (24) and inherited in (23) is validated in (22). Instead of giving more details we refer back to the considerations above and point to Fig. 4 and 5 depicting how a derivation tree context V_j is transformed to a corresponding one in terms of G'.

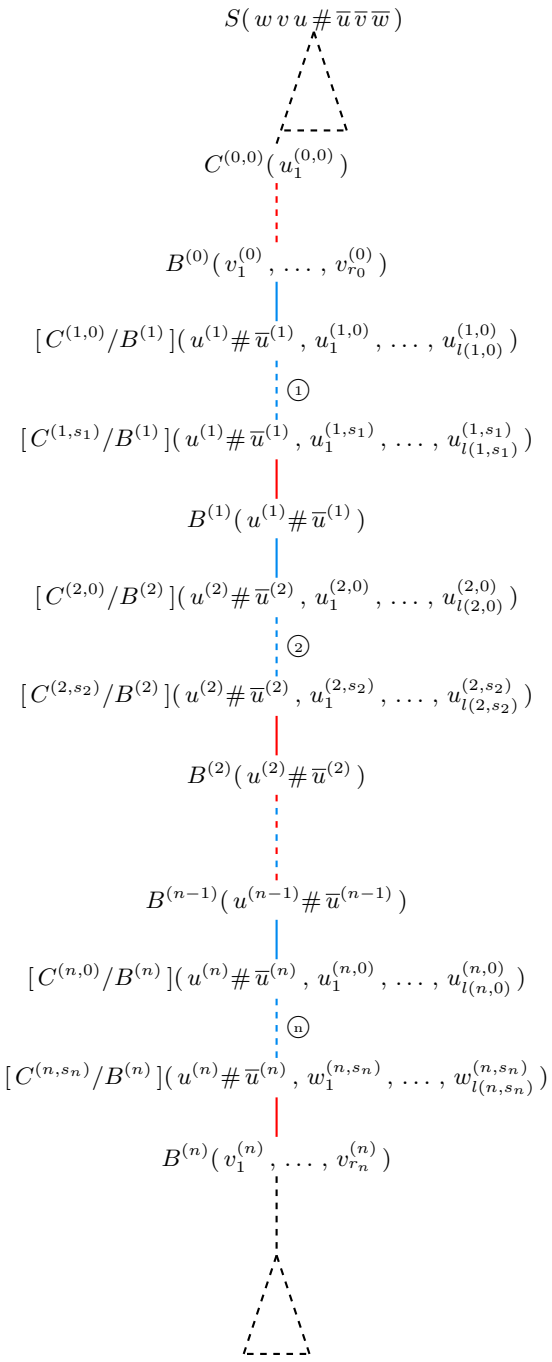

Fig. 5. Typical configuration within derivation tree of G' corresponding to the transformed derivation tree context V_j over $\Delta_P(\{y_j\})$ for arbitrary $j \in [0, m(S)]$.

5 Conclusion

We have characterized the separation of monadic branching MCFGs, and thus, MGs obeying the shortest move condition (SMC) and the specifier island condition (SPIC), from well-nested MCFGs by means of a "simple reverse copying" theorem concerning the derivable languages. Solving a generally open problem, the result also provides a direct comparison to the separation of well-nested MCFGs from MCFGs, and thus, MGs only obeying the SMC, by means of an already known "simple copying" theorem.

The SPIC provides a rather canonical restriction within the MG-setting.[8] *Well-nestedness* provides a rather canonical restriction on MCFGs, or reversing the perspective, within the MCFG-framework well-nested MCFGs constitute a natural generalization of, e.g., *tree adjoining grammars*, the former crucially preserving the well-nestedness property of the latter.[9] Since on the other hand, in terms of MGs, *well-nestedness* seems to be a rather *ad hoc* restriction, whereas for MCFGs, this seems to hold with regard to the SPIC, our result may suggest that we are concerned here with a structural difference between MGs and MCFGs which cannot immediately be overcome in a non-stipulated manner.

References

1. Chomsky, N.: The Minimalist Program. MIT Press, Cambridge, MA (1995)
2. Chomsky, N.: Derivation by phase. In: Kenstowicz, M. (ed.) Ken Hale. A Life in Language, pp. 1–52. MIT Press, Cambridge, MA (2001)
3. Chomsky, N.: On phases. In: Freidin, R., Otero, C., Zubizaretta, M.L. (eds.) Foundational Issues in Linguistic Theory, pp. 133–166. MIT Press, Cambridge, MA (2008)
4. Engelfriet, J., Rozenberg, G., Slutzki, G.: Tree transducers, L systems, and two-way machines. Journal of Computer and System Sciences 20, 150–202 (1980)
5. Gärtner, H.M., Michaelis, J.: A note on the complexity of constraint interaction. Locality conditions and minimalist grammars. In: Blache, P., Stabler, E., Busquets, J., Moot, R. (eds.) LACL 2005, LNAI, Vol. 3492, pp. 114–130. Springer, Berlin, Heidelberg (2005)
6. Gärtner, H.M., Michaelis, J.: Some remarks on locality conditions and minimalist grammars. In: Sauerland, U., Gärtner, H.M. (eds.) Interfaces + Recursion = Language?, pp. 161–195. Mouton de Gruyter, Berlin (2007)
7. Gazdar, G.: Unbounded dependencies and coordinate structure. Linguistic Inquiry 12, 155–184 (1981)
8. Girard, J.Y.: Linear logic. Theoretical Computer Science 50, 1–102 (1987)
9. de Groote, P., Morrill, G., Retoré, C. (eds.): LACL 2001, LNAI, Vol. 2099. Springer, Berlin, Heidelberg (2001)
10. Harkema, H.: A characterization of minimalist languages. In: de Groote et al. [9], pp. 193–211

[8] Independently of possible, linguistically motivated objections, the SPIC might, e.g., be interpreted as a "generalized" *generalized left branch condition* in the sense of [7].

[9] It has even be argued that there are good reasons to think that well-nestedness should be an essential property of the concept of *mild context-sensitivity*, cf. [12].

11. Joshi, A.K.: Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions? In: Dowty, D.R., Karttunen, L., Zwicky, A.M. (eds.) Natural Language Parsing, pp. 206–250. Cambridge University Press, New York, NY (1985)
12. Kanazawa, M.: The convergence of well-nested mildly context-sensitive grammar formalisms (2009), invited talk held at FG-2009, Bordeaux.
13. Kanazawa, M., Salvati, S.: The copying power of well-nested multiple context-free grammars. In: Dediu, A.H., Fernau, H., Martín-Vide, C. (eds.) LATA 2010, LNCS, Vol. 6031, pp. 344–355. Springer, Berlin, Heidelberg (2010)
14. Kobele, G.M.: Generating Copies. An investigation into structural identity in language and grammar. Ph.D. thesis, University of California, Los Angeles (2006)
15. Kobele, G.M., Michaelis, J.: Two type-0 variants of minimalist grammars. In: Rogers [25], pp. 81–91
16. Koopman, H., Szabolcsi, A.: Verbal Complexes. MIT Press, Cambridge, MA (2000)
17. Kracht, M.: The Mathematics of Language. Mouton de Gruyter, Berlin (2003)
18. Michaelis, J.: Derivational minimalism is mildly context-sensitive. In: Moortgat, M. (ed.) LACL '98, LNAI, Vol. 2014, pp. 179–198. Springer, Berlin, Heidelberg (2001)
19. Michaelis, J.: On Formal Properties of Minimalist Grammars. Linguistics in Potsdam 13, Universitätsbibliothek, Publikationsstelle, Potsdam (2001), Ph.D. thesis
20. Michaelis, J.: Transforming linear context-free rewriting systems into minimalist grammars. In: de Groote et al. [9], pp. 228–244
21. Michaelis, J.: Observations on strict derivational minimalism. Electronic Notes in Theoretical Computer Science 53, 192–209 (2004)
22. Michaelis, J.: An additional observation on strict derivational minimalism. In: Rogers [25], pp. 101–111
23. Mönnich, U.: Some remarks on mildly context-sensitive copying. In: Hanneforth, T., Fanselow, G. (eds.): Language and Logos, pp. 367–389, Akad. Verlag, Berlin (2010)
24. Rambow, O., Satta, G.: Independent parallelism in finite copying parallel rewriting systems. Theoretical Computer Science 223, 87–120 (1999)
25. Rogers, J. (ed.): Proceedings of FG-MoL 2005. CSLI Publications, Stanford (2009)
26. Salvati, S.: Minimalist grammars in the light of logic. Research Report, INRIA Bordeaux (2011), available at http://hal.inria.fr/inria-00563807/en/
27. Seki, H., Matsumura, T., Fujii, M., Kasami, T.: On multiple context-free grammars. Theoretical Computer Science 88, 191–229 (1991)
28. Stabler, E.P.: Derivational minimalism. In: Retoré, C. (ed.) LACL '96, LNAI, Vol. 1328, pp. 68–95. Springer, Berlin, Heidelberg (1997)
29. Stabler, E.P.: Remnant movement and complexity. In: Bouma, G., Kruijff, G.J.M., Hinrichs, E., Oehrle, R.T. (eds.) Constraints and Resources in Natural Language Syntax and Semantics, pp. 299–326. CSLI Publications, Stanford, CA (1999)
30. Stabler, E.P.: Recognizing head movement. In: de Groote et al. [9], pp. 245–260
31. Stabler, E.P.: Computational perspectives on minimalism. In: Boeckx, C. (ed.) Oxford Handbook of Linguistic Minimalism, pp. 616–641. Oxford University Press, New York, NY (2011)
32. Stabler, E.P., Keenan, E.L.: Structural similarity within and among languages. Theoretical Computer Science 293, 345–363 (2003)
33. Vijay-Shanker, K., Weir, D.J., Joshi, A.K.: Characterizing structural descriptions produced by various grammatical formalisms. In: 25th Annual Meeting of the Association for Computational Linguistics, Stanford, CA. pp. 104–111. ACL (1987)
34. Villemonte de la Clergerie, É.: Parsing mcs languages with thread automata. In: Proceedings of the Sixth International Workshop on Tree Adjoining Grammars and Related Formalisms, Venezia. pp. 101–108 (2002)