10

An Additional Observation on Strict Derivational Minimalism

Jens Michaelis

Abstract

We answer a question which, so far, was left an open problem: does—in terms of derivable string languages—the type of a *minimalist grammar (MG)* as originally introduced in Stabler 1997 defines a proper superclass of the revised type of an MG and, therefore, the type of a *strict MG* both introduced in Stabler 1999, and known to be weakly equivalent? For both the revised as well as the strict MG-type, the essential difference to the original MG-definition consists in imposing—in addition to the corresponding implementation of the *shortest move condition*—a second condition on the move-operator providing a formulation of the *specifier island condition*, and as such, restricting (further) the domain to which the operator can apply. It has been known already that this additional condition, in fact, ensures that—in terms of derivable string languages—the revised and, therefore, the strict MG-type both constitute a subclass of the original MG-type. We here present a string language proving that the inclusion is proper.

Keywords (strict) minimalist grammars, specifier island condition, multiple context-free grammars/linear context-free rewriting systems, (linear) context-free tree grammars

10.1 Introduction

The *minimalist grammar (MG)* formalism introduced in Stabler 1997 provides an attempt at a rigorous algebraic formalization of the perspectives currently adopted within the linguistic framework of transformational grammar. As has been shown (Michaelis 2001a, 2001b, Harkema 2001), this MG-type determines the same class of derivable string languages as *linear context-free rewriting systems (LCFRSs)* (Vijay-Shanker et al. 1987, Weir 1988).

FG-MoL 2005.
James Rogers (ed.).

Inspired, i.a., by the linguistic work presented in Koopman and Szabolcsi 2000, in Stabler 1999 a revised MG-type has been proposed whose essential departure from the version in Stabler 1997 can be seen as the following: in addition to the *shortest move constraint (SMC)*, a second locality condition, the *specifier island constraint (SPIC)*, is imposed on the move-operator regulating which maximal projection may move *overtly* into the highest specifier position. Deviating from the operator *move* as originally defined in Stabler 1997, a constituent has to belong to the transitive complement closure of a given tree or to be a specifier of such a constituent in order to be movable. An MG of this type, henceforth, is referred to as MG^{+SPIC}.

Closely in keeping with some further suggestions in Koopman and Szabolcsi 2000, a certain type of a *strict minimalist grammar (SMG)* has been introduced in Stabler 1999 as well: implementing the SPIC with somewhat more "strictness," leading to *heavy pied-piping* constructions, the SMG-type allows only movement of constituents belonging to the transitive complement closure of a tree. But in contrast to the $\mathrm{MG}^{+\mathrm{SPIC}}$-type, the triggering licensee feature may head the head-label of any constituent within the reflexive-transitive specifier closure of a moving constituent.

$\mathrm{MG}^{+\mathrm{SPIC}}$s and SMGs have been shown to be weakly equivalent by Michaelis (2004, 2002) confirming a conjecture explicitly stated in Stabler 1999. The equivalence turned out proving that, in terms of derivable languages, $\mathrm{MG}^{+\mathrm{SPIC}}$s and SMGs not only are subsumed by LCFRSs, but both are equivalent to a particular subclass of the latter, referred to as $LCFRS_{1,2}$-type: the righthand side of each rewriting rule of a corresponding LCFRS involves at most two nonterminals, and if two nonterminals appear on the righthand side then only simple strings of terminals are derivable from the first one.[1] It was, however, left unsolved, whether the respective classes of string languages derivable by $\mathrm{LCFRS}_{1,2}$s and unrestricted LCFRSs—and thus the respective classes of string languages derivable by $\mathrm{MG}^{+\mathrm{SPIC}}$s (or, likewise, SMGs) as defined in Stabler 1999 and MGs as defined in Stabler 1997—are identical.[2]

In this paper we show that the inclusion is in fact proper. We implicitly do so by expressing the reduced structural generative capacity

[1] Exactly this condition expresses the strict opacity of specifiers within the $\mathrm{MG}^{+\mathrm{SPIC}}$-version.

[2] Note that, instead of adding the SPIC to the original MG-formalism, using it to simply replace the SMC does not lead to a reduction of the class of derivable string languages. Quite the opposite, the resulting type of MG even allows derivation of every type 0-language (Kobele and Michaelis 2005).

in terms of a homomorphism mapping trees to strings. Explicitly, we show that, although a particular language (combining "string reversal," "simple copying" and "intervening balanced bracketing in terms of the context-free Dyck language") is derivable by an LCFRS, it's not derivable by an $\text{LCFRS}_{1,2}$. The most crucial part of our proof consists in showing that for each $\text{LCFRS}_{1,2}$, there is a linear *context-free tree grammar* (cf. Rounds 1970a,b, Fischer 1968, Engelfriet and Schmidt 1977) deriving, modulo a homomorphism, the same string language in *inside-out mode*.

10.2 Context-Free Tree Grammars

Giving our definition of a *context-free tree grammar (CFTG)* as it goes back to the work Rounds (1970a,b) and Fischer (1968), we mainly lean on the presentation in Engelfriet and Schmidt 1977.

Definition 25 A *ranked alphabet*, Σ, is an indexed family $\langle \Sigma_n \mid n \in \mathbb{N} \rangle$ of pairwise disjoint sets.[3] For $n \in \mathbb{N}$, a $\sigma \in \Sigma_n$ is an *operator of rank* n, whose rank is denoted by $\mathit{rank}(\sigma)$. The *set of trees (over* Σ*)*, $T(\Sigma)$, is built up recursively using the operators in the usual way: if for some $n \in \mathbb{N}$, we have $\sigma \in \Sigma_n$ and $t_1, \ldots, t_n \in T(\Sigma)$ then $t = \sigma(t_1, \ldots, t_n)$ is a tree. The *yield of* t, $\mathit{yield}(t) \in \Sigma_0^*$, is defined by $\mathit{yield}(t) = \sigma$ if $n = 0$, and $\mathit{yield}(t) = \mathit{yield}(t_1) \cdots \mathit{yield}(t_n)$ otherwise. A tree $t' \in T(\Sigma)$ is a *subtree (of* t*)* if $t' = t$, or if t' is a subtree of t_i for some $1 \leq i \leq n$.

Throughout we let $X = \{x_1, x_2, x_3, \ldots\}$ be a countable set of variables, and for $k \in \mathbb{N}$, we define $X_k \subseteq X$ as $\{x_1, \ldots, x_k\}$. Then for a ranked alphabet Σ, the set of *k-ary trees (over* Σ*)*, $T(\Sigma, X_k)$, is the set of trees $T(\Sigma')$ over the ranked alphabet $\Sigma' = \langle \Sigma'_n \mid n \in \mathbb{N} \rangle$, where $\Sigma'_0 = \Sigma_0 \cup X_k$, and $\Sigma'_n = \Sigma_n$ for $n > 0$. Let $T(\Sigma, X) = \bigcup_{k \in \mathbb{N}} T(\Sigma, X_k)$.

Definition 26 A *context-free tree grammar (CFTG)*, Γ, is a 5-tuple $\langle \Sigma, \mathcal{F}, \mathcal{S}, X, P \rangle$, where Σ and $\mathcal{F}$ are finite ranked alphabets of *inoperatives* and *operatives*, respectively. $\mathcal{S}$ is a distinguished element in $\mathcal{F}_n$ for some $n \in \mathbb{N}$, the *start symbol*. P is a finite set of productions. Each $p \in P$ is of the form $F(x_1, \ldots, x_n) \to t$ for some $n \in \mathbb{N}$, where $F \in \mathcal{F}_n$, $x_1, \ldots, x_n \in X$, and $t \in T(\Sigma \cup \mathcal{F}, X_n)$. If, in addition, for each such $p \in P$, no x_i occurs more than once in t then Γ is *linear*.

For $t, t' \in T(\Sigma \cup \mathcal{F}, X)$, t' is *directly derivable* from t ($t \Rightarrow t'$) if for some m and $n \in \mathbb{N}$, there are a $t_0 \in T(\Sigma \cup \mathcal{F}, X_{n+1})$ containing exactly *one* occurrence of x_{n+1}, a production $F(x_1, \ldots, x_m) \to t'' \in P$, and $t_1, \ldots, t_m \in T(\Sigma \cup \mathcal{F}, X)$ such that $t = t_0[x_1, \ldots, x_n, F(t_1, \ldots, t_m)]$

[3] Throughout the paper the following conventions apply: $\mathbb{N}$ is the set of all non-negative integers. For any set M, M^* denotes the Kleene closure of M, including ϵ, the empty string. M_ϵ is the set $M \cup \{\epsilon\}$.

and $t' = t_0[x_1, \ldots, x_n, t''[t_1, \ldots, t_m]]$.[4] If x_{n+1} is not dominated in t_0 by an operative then t' is derived from t by an *outside-in (OI)* step ($t \Rightarrow_{\mathrm{OI}} t'$).[5] If $t_1, t_2, \ldots, t_n \in T(\Sigma, X)$ then t' is derived by an *inside-out (IO)* step ($t \Rightarrow_{\mathrm{IO}} t'$). $\Rightarrow^*$, $\Rightarrow^*_{\mathrm{OI}}$ and $\Rightarrow^*_{\mathrm{IO}}$ denote the reflexive-transitive closures of $\Rightarrow$, $\Rightarrow_{\mathrm{OI}}$ and $\Rightarrow_{\mathrm{IO}}$, respectively.

The *(tree) languages derivable by* Γ *in unrestricted, OI-* and *IO-mode* are $\mathcal{L}(\Gamma) = \{t \in T(\Sigma) \mid \mathcal{S} \Rightarrow^* t\}$, $\mathcal{L}_{\mathrm{OI}}(\Gamma) = \{t \in T(\Sigma) \mid \mathcal{S} \Rightarrow^*_{\mathrm{OI}} t\}$ and $\mathcal{L}_{\mathrm{IO}}(\Gamma) = \{t \in T(\Sigma) \mid \mathcal{S} \Rightarrow^*_{\mathrm{IO}} t\}$, respectively. The corresponding *string languages derivable by* Γ are the sets $L(\Gamma) = \{\mathit{yield}(t) \mid t \in \mathcal{L}(\Gamma)\}$, $L_{\mathrm{OI}}(\Gamma) = \{\mathit{yield}(t) \mid t \in \mathcal{L}_{\mathrm{OI}}(\Gamma)\}$ and $L_{\mathrm{IO}}(\Gamma) = \{\mathit{yield}(t) \mid t \in \mathcal{L}_{\mathrm{IO}}(\Gamma)\}$, each of which being a subset of Σ_0^*.[6]

10.3 Linear Context-Free Rewriting Systems

The formalism of a *linear context-free rewriting systems (LCFRSs)* in the sense of Vijay-Shanker et al. 1987 can be seen as presenting a subtype of the formalism a *multiple context-free grammar (MCFG)* in the sense of Seki et al. 1991, where in terms of derivable string languages the generative power of LCFRSs is identical to that of MCFGs.

Definition 27 (Seki et al. 1991, Vijay-Shanker et al. 1987) A *multiple context-free grammar (MCFG)*, G, is a 5-tuple $\langle N, T, F, R, S\rangle$, where N and T are the finite sets of *nonterminals* and *terminals*, respectively. Each $A \in N$ is associated with some $d_G(A) \in \mathbb{N} \setminus \{0\}$. S is a distinguished symbol from N, the *start symbol*, with $d_G(S) = 1$. F and R are the finite sets of *functions* and *(rewriting) rules*, respectively, such that each $r \in R$ is of the form $A_0 \to f(A_1, \ldots, A_n)$ for some $f \in F$ and $A_0, A_1, \ldots, A_n \in N$ for some $n \in \mathbb{N}$, where f is a linear regular function from $(T^*)^{d_G(A_1)} \times \cdots \times (T^*)^{d_G(A_n)}$ into $(T^*)^{d_G(A_0)}$, allowing deletion of single components. r is *nonterminating* in case $n > 0$, otherwise r is *terminating*. If the latter, we have $f(\emptyset) \in (T^*)^{d_G(A_0)}$, and we usually denote r in the form $A_0 \to f(\emptyset)$.

For $A \in N$ and $k \in \mathbb{N}$, $L^k_G(A) \subseteq (T^*)^{d_G(A)}$ is given recursively by means of $\theta \in L^0_G(A)$ for each terminating $A \to \theta \in R$, and for $k \in \mathbb{N}$, $\theta \in L^{k+1}_G(A)$ if $\theta \in L^k_G(A)$, or if there are $A \to f(A_1, \ldots, A_n) \in R$

[4] For each $k \in \mathbb{N}$, and given trees $\tau \in T(\Sigma \cup \mathcal{F}, X_k)$ and $\tau_1, \ldots, \tau_k \in T(\Sigma \cup \mathcal{F}, X)$, $\tau[\tau_1, \ldots, \tau_k]$ is the tree in $T(\Sigma \cup \mathcal{F}, X)$ resulting from substituting for $1 \leq i \leq k$, each occurrence of the (trivial) subtree x_i of τ by an instance of τ_i.

[5] x_{n+1} is said to be *dominated in* t_0 *by an operative* $A \in \mathcal{F}_k$ for some $k \in \mathbb{N}$, if there are $\tau_1, \ldots, \tau_k \in T(\Sigma \cup \mathcal{F}, X)$ such that $A(\tau_1, \ldots, \tau_k)$ is a subtree of t_0 which contains the unique occurrence of x_{n+1} in t_0.

[6] Note that $\mathcal{L}(\Gamma) = \mathcal{L}_{\mathrm{OI}}(\Gamma)$ holds for each CFTG Γ, but the class of context-free tree languages derivable in OI-mode and the one of those derivable in IO-mode are not comparable in full general (Fischer 1968).

and $\theta_i \in L^k_G(A_i)$ for $1 \leq i \leq n$ such that $f(\theta_1, \ldots, \theta_n) = \theta$. The set $L_G(A) = \bigcup_{k \in \mathbb{N}} L^k_G(A)$ is the *language derivable from A (by G)*.[7] $L_G(S)$, also denoted by $L(G)$, is the *multiple context-free language (MCFL) (derivable by G)*. We have $L(G) \subseteq T^*$, because $d_G(S) = 1$. The *rank of G*, *rank*(G), is the number $\max\{n \mid A_0 \to f(A_1, \ldots, A_n) \in R\}$, the *fan-out of G* is the number $\max\{d_G(A) \mid A \in N\}$.

If each $f \in F$ appearing in some $A_0 \to f(A_1, \ldots, A_n) \in R$, in addition, has the property that no component of the tuples of tuples of $(T^*)^{d_G(A_1)} \times \cdots \times (T^*)^{d_G(A_n)}$ is erased by mapping under f into $(T^*)^{d_G(A_0)}$, then G is a *(string based) linear context-free rewriting system (LCFRS)*, and $L(G)$ is a *(string based) linear context-free rewriting language (LCFRL)*.

Example An LCFRS which has rank 2 and fan-out 3 is the LCFRS $G_{\text{ex}} = \langle \{S, A, B\}, \{a, b, [,]\}, \{conc, id, e_{[]}, e_a, e_b, f_a, f_b, g, h\}, R, S\rangle$ with $d(A) = d(B) = 3$, where R consists of the following rules:

$$S \to conc(B),$$

$$A \to e_a(\emptyset) \mid e_b(\emptyset) \mid f_a(A) \mid f_b(A) \mid h(B, B) \text{ and}$$

$$B \to e_{[]}(\emptyset) \mid id(A) \mid g(B),$$

and where the functions are given by:

$$\begin{array}{lrcl}
conc: & \langle x_0, x_1, x_2 \rangle & \mapsto & x_0x_1x_2 \\
id: & \langle x_0, x_1, x_2 \rangle & \mapsto & \langle x_0, x_1, x_2 \rangle \\
e_{[]}: & \emptyset & \mapsto & \langle \epsilon, [\,], \epsilon \rangle \\
e_a: & \emptyset & \mapsto & \langle a, a, a \rangle \\
e_b: & \emptyset & \mapsto & \langle b, b, b \rangle \\
f_a: & \langle x_0, x_1, x_2 \rangle & \mapsto & \langle x_0a, x_1a, ax_2 \rangle \\
f_b: & \langle x_0, x_1, x_2 \rangle & \mapsto & \langle x_0b, x_1b, bx_2 \rangle \\
g: & \langle x_0, x_1, x_2 \rangle & \mapsto & \langle x_0, [x_1], x_2 \rangle \\
h: & \langle\langle x_0, x_1, x_2 \rangle, \langle y_0, y_1, y_2 \rangle\rangle & \mapsto & \langle x_0y_0, y_1x_1, y_2x_2 \rangle
\end{array}$$

The language derivable by G_{ex} is

$$L(G_{\text{ex}}) = \{w_1 \cdots w_n z_n w_n \cdots z_1 w_1 z_0 w_n^R \cdots w_1^R \mid n \in \mathbb{N} \setminus \{0\}, w_i \in \{a, b\}^+ \text{ for } 1 \leq i \leq n, z_n \cdots z_0 \in D\},$$

where D is the Dyck language of balanced parentheses, generated by the context free grammar $G_D = \langle \{S\}, \{[,]\}, \{S \to SS \mid [S] \mid \epsilon\}, S\rangle$, and where for each $w \in T^*$, w^R denotes the reversal of w.[8]

[7] Thus, employing the notion of the CFTG-derivation modes introduced above, an MCFG can be considered to derive a tuple of strings in IO-mode.

[8] Thus for each set M and $w \in M^*$, $w^R \in M^*$ is defined recursively by $\epsilon^R = \epsilon$, and $(av)^R = v^R a$ for $a \in M$ and $v \in M^*$.

The class of MCFLs and the class of LCFRLs are known to be identical (cf. Seki et al. 1991, Lemma 2.2). Theorem 11 in Rambow and Satta 1999, therefore, shows that for each MCFG G there is an LCFRS G' with $rank(G') \leq 2$ for which $L(G) = L(G')$ holds.

Definition 28 An *LCFRS$_{1,2}$* is an LCFRS G according to Definition 27 such that $rank(G) = 2$, and $d_G(A_1) = 1$ for each $A_0 \to f(A_1, A_2) \in R$. In this case $L(G)$ is an *LCFRL$_{1,2}$*.

Definition 29 A given $\mathrm{LCFRS}_{1,2}$ $G = \langle N, T, F, R, S\rangle$ is in *LCFRS$_{1,2}$-normalform (LCFRS$_{1,2}$-NF)* if each $f \in F$ is of one of the forms (i)–(iii) for some $m \in \mathbb{N} \setminus \{0\}$, or of the form (iv) for some $a \in T_\epsilon$.

$$\begin{array}{lrcl}
\text{(i)} & \langle\langle y_1\rangle, \langle x_1, x_2, \ldots, x_m\rangle\rangle & \mapsto & \langle y_1, x_1, x_2, \ldots, x_m\rangle \\
\text{(ii)} & \langle\langle y_1\rangle, \langle x_1, x_2, \ldots, x_m\rangle\rangle & \mapsto & \langle y_1 x_1, x_2, \ldots, x_m\rangle \\
\text{(iii)} & \langle x_1, x_2, \ldots, x_{m+1}\rangle & \mapsto & \langle x_{m+1} x_1, x_2, \ldots, x_m\rangle \\
\text{(iv)} & \emptyset & \mapsto & a
\end{array}$$

Proposition 32 *For every LCFRS$_{1,2}$ G, there exists an LCFRS$_{1,2}$ $G' = \langle N, T, F, R, S\rangle$ in LCFRS$_{1,2}$-NF such that $L(G) = L(G')$.*

Proof. The proposition can essentially be proven applying a "double transformation" to a given $\mathrm{LCFRS}_{1,2}$: first, using the construction presented in Michaelis 2004, the $\mathrm{LCFRS}_{1,2}$ is transformed into an $\mathrm{MG}^{+\mathrm{SPIC}}$ deriving the same string language. Then, using the construction presented in Michaelis 2002, the resulting $\mathrm{MG}^{+\mathrm{SPIC}}$ is transformed into an $\mathrm{LCFRS}_{1,2}$ of the corresponding normal form still deriving the same string language, but with (iv') instead of (iv).[9]

$$\text{(iv')} \qquad \emptyset \qquad \mapsto \quad w \ , \ w \in T^*$$

Verifying that (iv') can be strengthened to (iv) is straightforward.[10] □

10.4 Proper Inclusion within LCFRLs

Let $G = \langle N, T, F, R, S\rangle$ be an $\mathrm{LCFRS}_{1,2}$ in $\mathrm{LCFRS}_{1,2}$-NF.

Construction We now construct a linear CFTG $\Gamma = \langle \Sigma, \mathcal{F}, \mathcal{S}, X, P\rangle$ with $\Sigma_0 = T \cup \{\Lambda\}$ for a new symbol Λ, and with $h[L_{\mathrm{IO}}(\Gamma)] = L(G)$,[11] where h is the homomorphism from Σ_0^* to T^* determined by $h(a) = a$ for $a \in T$, and $h(\Lambda) = \epsilon$. In constructing Γ, we assume $\bullet$ and $\mathcal{S}$ to be two further, new distinct symbols and let

[9] Here, (i) and (ii) simulate the behavior of the *merge*-operator, (iii) simulates the behavior of the *move*-operator, and (iv') provides "lexical insertion."

[10] If need be, we simply add new nonterminals, functions and rules of the form $A \to a$ and $B \to f(C, D)$ to G', where $a \in T_\epsilon$ and f is in line with (ii), and successively replace all rules of the form (iv') not being in line with (iv).

[11] For any two sets M_1 and M_2, and any mapping g from M_1 into M_2, $g[M_1]$ denotes the image of M_1 under g, i.e., the set $\{g(m) \mid m \in M_1\} \subseteq M_2$.

$\Sigma_0 = T \cup \{\Lambda\}$
$\Sigma_2 = \{\bullet\} \cup \{A' \mid A \in N \text{ and } d_G(A) = 2\}$
$\Sigma_n = \{A' \mid A \in N \text{ and } d_G(A) = n\}$ for $n \in \mathbb{N} \setminus \{0, 2\}$
$\mathcal{F}_0 = \{\mathcal{S}\} \cup \{\widetilde{B} \mid B \in N\}$
$\mathcal{F}_n = \{\widetilde{A}_D \mid A, D \in N \text{ and } d_G(A) = n\}$ for $n \in \mathbb{N} \setminus \{0\}$

Defining the set of productions in Γ, we first let

(s) $\mathcal{S} \to \widetilde{S} \in P$.

For each terminating $A \to \theta \in R$ for some $A \in N$ and $\theta \in (T^*)^{d_G(A)}$, we have $d_G(A) = 1$ and $\theta \in T_\epsilon$, because of (iv).

If $\theta \neq \epsilon$ we let

(t.1) $\widetilde{B} \to \widetilde{A}_B(\theta) \in P$ for each $B \in N$, and

(t.2) $\widetilde{A} \to A'(\theta) \in P$.

If $\theta = \epsilon$ we let

(t.1) $\widetilde{B} \to \widetilde{A}_B(\Lambda) \in P$ for each $B \in N$, and

(t.2) $\widetilde{A} \to A'(\Lambda) \in P$.

For each nonterminating $A \to f(B) \in R$ for some $A, B \in N$ and $f \in F$, we have $f : \langle x_1, \ldots, x_{d_G(B)} \rangle \mapsto \langle x_{d_G(B)} x_1, x_2, \ldots, x_{d_G(B)-1} \rangle$, because of (iii).

We let

(iii.1) $\widetilde{B}_D(x_1, \ldots, x_{d_G(B)}) \to \widetilde{A}_D(\bullet(x_{d_G(B)}, x_1), x_2, \ldots, x_{d_G(B)-1}) \in P$ for each $D \in N$, and

(iii.2) $\widetilde{B}_A(x_1, \ldots, x_{d_G(B)}) \to A'(\bullet(x_{d_G(B)}, x_1), x_2, \ldots, x_{d_G(B)-1}) \in P$.

For each nonterminating $A \to f(B, C) \in R$ for some $A, B, C \in N$ and $f \in F$, because of (i) and (ii), we either have (i') or (ii').

(i') $f : \langle \langle y_{d_G(B)} \rangle, \langle x_1, \ldots, x_{d_G(C)} \rangle \rangle \mapsto \langle y_{d_G(B)}, x_1, \ldots, x_{d_G(C)} \rangle$

(ii') $f : \langle \langle y_{d_G(B)} \rangle, \langle x_1, x_2, \ldots, x_{d_G(C)} \rangle \rangle \mapsto \langle y_{d_G(B)} x_1, x_2, \ldots, x_{d_G(C)} \rangle$

If (i'), we let

(i.1) $\widetilde{C}_D(x_1, \ldots, x_{d_G(C)}) \to \widetilde{A}_D(\widetilde{B}, x_1, x_2, \ldots, x_{d_G(C)}) \in P$ for each $D \in N$, and

(i.2) $\widetilde{C}_A(x_1, \ldots, x_{d_G(C)}) \to A'(\widetilde{B}, x_1, x_2, \ldots, x_{d_G(C)}) \in P$.

If (ii'), we let

(ii.1) $\widetilde{C}_D(x_1, \ldots, x_{d_G(C)}) \to \widetilde{A}_D(\bullet(\widetilde{B}, x_1), x_2, \ldots, x_{d_G(C)}) \in P$ for each $D \in N$, and

(ii.2) $\widetilde{C}_A(x_1, \ldots, x_{d_G(C)}) \to A'(\bullet(\widetilde{B}, x_1), x_2, \ldots, x_{d_G(C)}) \in P$.

We will not provide a strictly formal proof, but briefly emphasize, why Γ does its job as desired: the way in which Γ simulates G crucially depends on the possibility to rewrite blindly a nonterminal $B \in N$ by starting with rewriting B as any $A \in N$ for which there is a terminating rule in R available, cf. (t.1). In terms of Γ, this A gets indexed by B storing the necessity that A has to be successively developed inside-out, finally creating a tree rooted in B. That is to say, in some later derivation step the blind rewriting of B simulated by Γ must get legitimated by an application of a rule of the sort (i.2), (ii.2) or (iii.2), in order to create a convergent derivation. This sort of simple information inheritance is possible because of the fact that the complex productions in G, those which are of rank 2, are still "simple enough," i.e., the contribution of at least one of the nonterminals on the righthand side consists in a simple string of terminals.

Proposition 33 *Each $LCFRL_{1,2}$ is, up to a homomorphism, the string language derivable by some linear CFTG in IO-mode.* □

An *indexed language (IL)* is the string language derived by an *indexed grammar (IG)* in the sense of Aho 1968.

Corollary 34 *Each $LCFRL_{1,2}$ is an IL.*

Proof. By Proposition 33, because for linear CFTGs, IO- and OI-mode derive identical (string) languages (cf. Kepser and Mönnich forthcoming). Furthermore the class of string languages derived by CFTGs in OI-mode is included in the class of ILs (Fischer 1968, Rounds 1970a,b),[12] and the class of ILs is closed under homomorphisms (Aho 1968). □

Proposition 35 *The class of $LCFRL_{1,2}$s is properly included in the class of LCFRLs.*

Proof. By Corollary 34, because the LCFRL from our example above, $L(G_{\mathrm{ex}})$, is known not to be an IL (Staudacher 1993). □

Corollary 36 *The class of languages derivable by MG^{+SPIC}s is properly included in the class of languages derivable by MGs.* □

10.5 Summary

We have shown that—in terms of derivable string languages—the type of a minimalist grammar (MG) as originally introduced in Stabler 1997 defines a proper superclass of the revised type of an MG and, therefore, the type of a strict MG both introduced in Stabler 1999, and known to be weakly equivalent. In order to achieve the result we have

[12] Note that, vice versa, for each IL L, $L \setminus \{\epsilon\}$ is the string language derived by some CFTG in OI-mode (cf. Fischer 1968, Rounds 1970a,b).

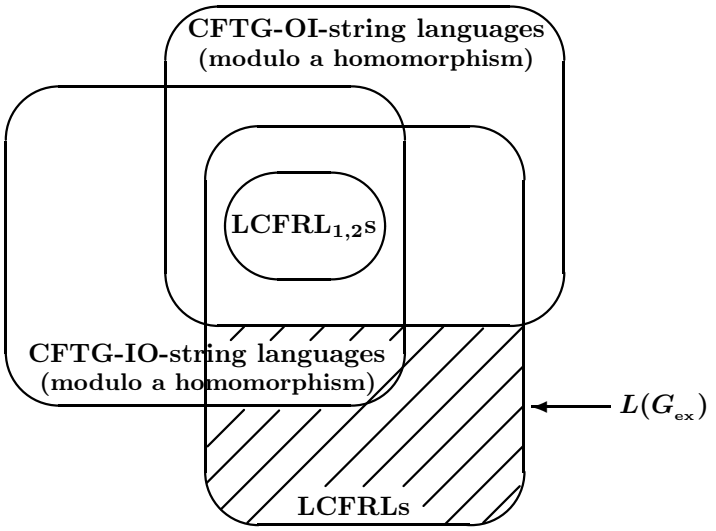

FIGURE 1: Proving the proper inclusion of $\mathrm{LCFRL}_{1,2}$s within LCFRLs.

proven that the class of *(string based) linear context-free rewriting languages (LCFRLs)* properly subsumes a particular subclass of LCFRLs, referred to as the class of *$LCFRL_{1,2}$s*. This was motivated by the fact that LCFRLs and $\mathrm{LCFRL}_{1,2}$s coincide with the two classes of derivable string languages defined by the original and the revised MG-type, respectively.

The most crucial part of our proof consists in showing that every $\mathrm{LCFRL}_{1,2}$ is, modulo a homomorphism, the string language derivable in *inside-out (IO)* mode by a linear *context-free tree grammar (CFTG)*. For linear CFTGs it holds that the class of languages derivable in IO-mode is not distinguishable from the class of languages derivable in *outside-in (OI)* mode; and the class of string languages generally derivable by CFTGs in OI-mode is known to be subsumed by the class of languages derivable by *indexed grammars (IGs)*. Since the latter class of languages is closed under homomorphisms, the intended result finally followed from presenting an LCFRL known not to be derivable by an IG, namely, the language $L(G_{\mathrm{ex}})$ (cf. Figure 1).

References

Aho, Alfred V. 1968. Indexed grammars—An extension of context-free grammars. *Journal of the Association for Computing Machinery* 15:647–671.

de Groote, Philippe, Glyn Morrill, and Christian Retoré, eds. 2001. *Logical Aspects of Computational Linguistics (LACL '01)*, Lecture Notes in Artificial Intelligence Vol. 2099. Berlin, Heidelberg: Springer.

Engelfriet, Joost and Erik M. Schmidt. 1977. IO and OI. I. *Journal of Computer and System Sciences* 15:328–353.

Fischer, Michael J. 1968. Grammars with macro-like productions. In *Conference Record of 1968 Ninth Annual Symposium on Switching and Automata Theory* , Schenectady, NY, pages 131–142. IEEE.

Harkema, Henk. 2001. A characterization of minimalist languages. In de Groote et al. (2001), pages 193–211.

Kepser, Stephan and Uwe Mönnich. forthcoming. Closure properties of linear context-free tree languages with an application to optimality theory. *Theoretical Computer Science*. Preprint available at `http://tcl.sfs.uni-tuebingen.de/~kepser/papers/pubs.html`.

Kobele, Gregory M. and Jens Michaelis. 2005. Two type-0 variants of minimalist grammars. In *FG-MoL 2005. The 10th conference on Formal Grammar and The 9th Meeting on Mathematics of Language,* Edinburgh. This volume.

Koopman, Hilda and Anna Szabolcsi. 2000. *Verbal Complexes*. Cambridge, MA: MIT Press.

Michaelis, Jens. 2001a. Derivational minimalism is mildly context–sensitive. In M. Moortgat, ed., *Logical Aspects of Computational Linguistics (LACL '98)*, Lecture Notes in Artificial Intelligence Vol. 2014, pages 179–198. Berlin, Heidelberg: Springer.

Michaelis, Jens. 2001b. Transforming linear context-free rewriting systems into minimalist grammars. In de Groote et al. (2001), pages 228–244.

Michaelis, Jens. 2002. Implications of a revised perspective on minimalist grammars. Draft, Potsdam University. Available at `http://www.ling.uni-potsdam.de/~michael/papers.html`.

Michaelis, Jens. 2004. Observations on strict derivational minimalism. *Electronic Notes in Theoretical Computer Science* 53:192–209. Proceedings of the joint meeting of the 6th Conference on Formal Grammar and the 7th Meeting on Mathematics of Language (FGMOL '01), Helsinki, 2001.

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

Rounds, William C. 1970a. Mappings and grammars on trees. *Mathematical Systems Theory* 4:257–287.

Rounds, William C. 1970b. Tree-oriented proofs of some theorems on context-free and indexed languages. In *Proceedings of the 2nd Annual ACM Symposium on Theory of Computing*, Northhampton, MA, pages 109–116. ACM.

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

Stabler, Edward P. 1997. Derivational minimalism. In C. Retoré, ed., *Logical Aspects of Computational Linguistics (LACL '96)*, Lecture Notes in Artificial Intelligence Vol. 1328, pages 68–95. Berlin, Heidelberg: Springer.

Stabler, Edward P. 1999. Remnant movement and complexity. In G. Bouma, G.-J. M. Kruijff, E. Hinrichs, and R. T. Oehrle, eds., *Constraints and Resources in Natural Language Syntax and Semantics*, pages 299–326. Stanford, CA: CSLI Publications.

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),* Utrecht, pages 358–367. ACL.

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 (ACL '87)*, Stanford, CA, pages 104–111. ACL.

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