Appeared in *Model-Theoretic Syntax at 10*, Proceedings of the
ESSLLI-Workshop, pp. 87-98. Dublin, August 13-17, 2007.

Locality Conditions and the Complexity of Minimalist Grammars: A Preliminary Survey

Hans-Martin Gärtner
ZAS
Berlin, Germany
gaertner@zas.gwz-berlin.de

Jens Michaelis
Universität Osnabrück
Osnabrück, Germany
jmichael@uos.de

Introduction

Among the well-established variety of formal grammar types providing a *mildly context-sensitive grammar (MCSG)* formalism in the sense of Joshi (1985), Stabler's *minimalist grammars (MGs)* (Stabler 1997, 1999) come closest to modeling the tools used in the Chomskyan branch of generative syntax known as "minimalism" (Chomsky 1995, 2000, 2001). Interestingly, without there being a rise in (at least weak) generative power, (extensions and variants of) MGs accommodate a wide variety of (arguably) "odd" items from the syntactician's toolbox, such as *head movement* (Stabler 1997, 2001), *affix hopping* (Stabler 2001), *(strict) remnant movement* (Stabler 1997, 1999), and (to some extent) *adjunction* and *scrambling* (Frey and Gärtner 2002; Gärtner and Michaelis 2003). As a descendant of *transformational grammar (TG)*, minimalism carries over the division of labor between a phrase-structural and a transformational component. These find their way into MGs as operations *merge* and *move*, respectively. As is well-known, the *Aspects*-vintage of TG (Chomsky 1965) was shown to be Turing equivalent by Peters and Ritchie 1973. This led to intensive investigation into *locality conditions (LCs)* (Chomsky 1973, 1977, 1986; Rizzi 1990; Cinque 1990; Manzini 1992; among others) in an attempt to restrict the power of transformations. However, complexity results for these grammars with LC-addons are largely absent.[1] This picture has changed with MGs, as a formalized version of minimalism, which were shown to belong among the MCSG-formalisms in Michaelis 2001a. On the basis of this result it was possible to begin an investigation into how the complexity of MGs is affected by the presence or absence of various LCs. Here we are going to review this work and explore some directions for further research.

In particular we are going to look at the behavior and interaction of the *shortest move condition (SMC)*, the *specifier island condition (SPIC)* and the *adjunct island condition (AIC)*. It will turn out that different LCs have different effects on complexity. The original complexity result has been shown to hold for standard MGs incorporating the SMC. Adding the SPIC to standard MGs has non-monotonic consequences: whether complexity goes up or down depends on the absence or copresence of the SMC, respectively (Section 2.1).[2]

[1] One notable exception is the work of Rogers (1998), who proves the (strong) context-freeness of a restricted *government and binding (GB)* formalism, which he develops in terms of a (monadic second order) logical approach. In the connection of a logical formalization of GB(-like) structures also Kracht 1995b and 1995a (as well as Kracht's follow-up work) deserve attention. Some further relevant discussion can be found in the literature on *constraint-* or *principle-based parsing* such as, e.g., Cornell 1992, Stabler 1992, or Berwick 1991.

[2] A more general picture of the MCSG landscape is given in Figure 11 within the Appendix, where, in particular, we have the following abbreviations: TAG = tree adjoining gram-

For the AIC, the picture is more complicated. First of all, the AIC only makes sense if base-adjunction and adjunction by scrambling/extraposition is added to MGs (as suggested in Frey and Gärtner 2002; Gärtner and Michaelis 2003). Even more specifically, the AIC seems to make a difference if adjunction is allowed to occur countercyclically or *late*, i.e. if it is allowed to target a non-root constituent. Under these conditions, adding the AIC together with the SMC guarantees that the resulting grammars stay within the class of MCSGs. Without the AIC there are configurations that appear to go beyond. In MGs without the SMC, on the other hand, it is plausible to assume that the AIC does not change complexity at all, i.e. it is void (Section 2.2).

Before we go into these particulars about LCs, we will provide a formal introduction to (the relevant variants of) MGs in Section 1. In a further outlook (Section 3), we sketch an MG-analysis of multiple wh-constructions and conclude with some general remarks about future research.

1 Minimalist Grammars

Throughout we let $\neg Syn$ and Syn be a finite set of *non-syntactic features* and a finite set of *syntactic features*, respectively, in accordance with (F1)–(F3) below. We take *Feat* to be the set $\neg Syn \cup Syn$.

(F1) $\neg Syn$ is disjoint from Syn and partitioned into the sets *Phon* and *Sem*, a set of *phonetic features* and a set *semantic features*, respectively.

(F2) Syn is partitioned into six sets:[3]

$$\begin{aligned} &Base, \\ &\textit{M-Select} &&= \{\, \texttt{=x} \mid \texttt{x} \in Base \,\}, \\ &\textit{A-Select} &&= \{\, \texttt{≈x} \mid \texttt{x} \in Base \,\}, \\ &\textit{M-Licensors} &&= \{\, \texttt{+x} \mid \texttt{x} \in Base \,\}, \\ &\textit{M-Licensees} &&= \{\, \texttt{-x} \mid \texttt{x} \in Base \,\}, \text{ and} \\ &\textit{S-Licensees} &&= \{\, \texttt{~x} \mid \texttt{x} \in Base \,\}, \end{aligned}$$

the sets of *(basic) categories*, *m(erge)-selectors*, *a(djoin)-selectors*, *m(ove)-licensors*, *m(ove)-licensees*, and *s(cramble)-licensees*, respectively.

(F3) *Base* includes at least the category `c`.

We use *Licensees* as a shorthand denoting the set $\textit{M-Licensees} \cup \textit{S-Licensees}$.

Definition 1.1 An *expression (over Feat)*, also referred to as a *minimalist tree (over Feat)*, is a five-tuple $\langle N_\tau, \lhd_\tau^*, \prec_\tau, <_\tau, label_\tau \rangle$ obeying (E1)–(E3).

(E1) $\langle N_\tau, \lhd_\tau^*, \prec_\tau \rangle$ is a finite, binary (ordered) tree defined in the usual sense: N_τ is the finite, non-empty set of *nodes*, and $\lhd_\tau^*$ and $\prec_\tau$ are the respective binary relations of *dominance* and *precedence* on N_τ.[4]

(E2) $<_\tau \subseteq N_\tau \times N_\tau$ is the asymmetric relation of *(immediate) projection* that holds for any two siblings, i.e., for each $x \in N_\tau$ different from the root of $\langle N_\tau, \lhd_\tau^*, \prec_\tau \rangle$ either $x <_\tau sibling_\tau(x)$ or $sibling_\tau(x) <_\tau x$ holds.[5]

(E3) $label_\tau$ is the *leaf-labeling function* from the set of leaves of $\langle N_\tau, \lhd_\tau^*, \prec_\tau \rangle$ into $Syn^* \{\#\} Syn^* Phon^* Sem^*$.[6]

We take $Exp(Feat)$ to denote the class of all expressions over *Feat*.

mars, LIG = linear indexed grammars, CCG = combinatory categorial grammars, HG = head grammars, LCFRS = linear context-free rewriting systems, MCTAG = (set local) multi-component tree adjoining grammars, IG = indexed grammars (cf. Joshi et al. 1991). An arrow always points to a class which is less powerful in generative capacity. If there is a double-arrow between two classes their generative capacity is equal.

[3] Elements from *Syn* will usually be typeset in `typewriter font`.

[4] Thus, $\lhd_\tau^*$ denotes the reflexive-transitive closure of $\lhd_\tau$, the binary relation of *immediate dominance* on N_τ.

[5] For each $x \in N_\tau$ different from the root of $\langle N_\tau, \lhd_\tau^*, \prec_\tau \rangle$, $sibling_\tau(x)$ denotes the (unique) sibling. If $x <_\tau y$ for some $x, y \in N_\tau$ then x is said to *(immediately) project over* y.

[6] For each set M, M^* is the Kleene closure of M, including ε, the empty string. For any two sets of strings, M and N, MN is the product of M and N w.r.t. string concatenation. Further, # denotes a new symbol not appearing in *Feat*.

Let $\tau = \langle N_\tau, \lhd_\tau^*, \prec_\tau, <_\tau, label_\tau \rangle \in Exp(Feat)$.

For each $x \in N_\tau$, the *head of x (in* τ*)*, denoted by $head_\tau(x)$, is the (unique) leaf of τ with $x \lhd_\tau^* head_\tau(x)$ such that each $y \in N_\tau$ on the path from x to $head_\tau(x)$ with $y \neq x$ projects over its sibling, i.e. $y <_\tau sibling_\tau(y)$. The *head of* τ is the head of τ's root. τ is said to be a *head* (or *simple*) if N_τ consists of exactly one node, otherwise τ is said to be a *non-head* (or *complex*).

A given expression $\phi = \langle N_\phi, \lhd_\phi^*, \prec_\phi, <_\phi, label_\phi \rangle$ belonging to $Exp(Feat)$ is a *subexpression of* τ in case $\langle N_\phi, \lhd_\phi^*, \prec_\phi \rangle$ is a subtree of $\langle N_\tau, \lhd_\tau^*, \prec_\tau \rangle$, $<_\phi = <_\tau \restriction_{N_\phi \times N_\phi}$, and $label_\phi = label_\tau \restriction_{N_\phi}$. Such a subexpression ϕ is a *maximal projection (in* τ*)* if its root is a node $x \in N_\tau$ such that x is the root of τ, or such that $sibling_\tau(x) <_\tau x$. $MaxProj(\tau)$ is the set of maximal projections in τ.

$comp_\tau \subseteq MaxProj(\tau) \times MaxProj(\tau)$ is the binary relation defined such that for all $\phi, \chi \in MaxProj(\tau)$ it holds that $\phi\, comp_\tau\, \chi$ iff $head_\tau(r_\phi) <_\tau r_\chi$, where r_ϕ and r_χ are the roots of ϕ and χ, respectively. If $\phi\, comp_\tau\, \chi$ holds for some $\phi, \chi \in MaxProj(\tau)$ then χ is a *complement of* ϕ *(in* τ*)*. $comp_\tau^+$ is the transitive closure of $comp_\tau$. $Comp^+(\tau)$ is the set $\{\phi \mid \tau\, comp_\tau^+\, \phi\}$.

$spec_\tau \subseteq MaxProj(\tau) \times MaxProj(\tau)$ is the binary relation defined such that for all $\phi, \chi \in MaxProj(\tau)$ it holds that $\phi\, spec_\tau\, \chi$ iff both $r_\chi = sibling_\tau(x)$ and $x <_\tau r_\chi$ for some $x \in N_\tau$ with $r_\phi \lhd_\tau^+ x \lhd_\tau^+ head_\tau(r_\phi)$, where r_ϕ and r_χ are the roots of ϕ and χ, respectively. If $\phi\, spec_\tau\, \chi$ for some $\phi, \chi \in MaxProj(\tau)$ then χ is a *specifier of* ϕ *(in* τ*)*. $Spec(\tau)$ is the set $\{\phi \mid \tau\, spec_\tau\, \phi\}$. Note that, if $Spec(\tau) \neq \emptyset$ then $Spec(\tau)$ is not necessarily a singleton set, but there is a unique specifier υ of τ, which we will refer to as the *highest specifier* of τ, such that the root of υ is immediately dominated by the root of τ.[7]

[7]Note that the leaf-labeling function $label_\tau$ can easily be extended to a total labeling function ℓ_τ from N_τ into $Feat^*\{\#\}Feat^* \cup \{<, >\}$, where $<$ and $>$ are two new distinct symbols: to each non-leaf $x \in N_\tau$ we can assign a label from $\{<, >\}$ by ℓ_τ such that $\ell_\tau(x) = <$ iff $y <_\tau z$ for $y, z \in N_\tau$ with $x \lhd_\tau y, z$, and $y \prec_\tau z$. In this sense a concrete $\tau \in Exp(Feat)$ is depictable in the way indicated in Figure 1.

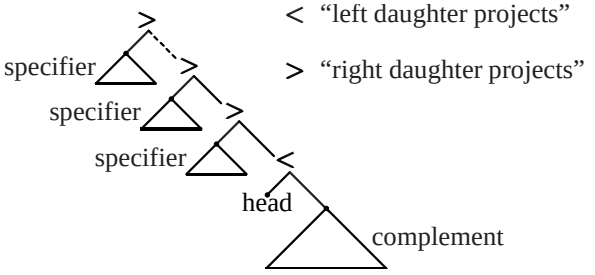


Figure 1: A typical minimalist tree.

A $\phi \in MaxProj(\tau)$ is said to *have*, or *display*, *(open) feature* f if the label assigned to ϕ's head by $label_\tau$ is of the form $\beta\#f\beta'$ for some $f \in Feat$ and some $\beta, \beta' \in Feat^*$.[8]

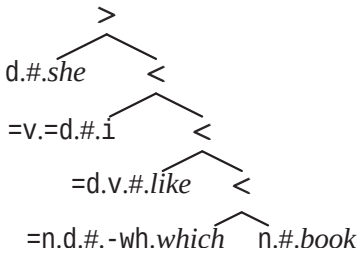


Figure 2: Example of a minimalist tree.

τ is *complete* if its head-label is in $Syn^*\{\#\}\{c\}Phon^*Sem^*$, and each of its other leaf-labels is in $Syn^*\{\#\}Phon^*Sem^*$. Hence, a complete expression over *Feat* is an expression that has category c, and this instance of c is the only instance of a syntactic feature which is preceded by an instance of # within its local leaf-label, i.e. the leaf-label it appears in.

The *phonetic yield of* τ, denoted by $Y_{Phon}(\tau)$, is the string which results from concatenating in "left-to-right-manner" the labels assigned via $label_\tau$ to the leaves of $\langle N_\tau, \lhd_\tau^*, \prec_\tau \rangle$, and replacing all instances of non-phonetic features with the empty string, afterwards.[9]

[8]Thus, e.g., the expression depicted in Figure 2 has feature i, while there is a maximal projection which has feature -wh. For the sake of simplicity, we assume *she*, *like*, *which*, and *book* to be strings of phonetic features.

[9]Tree in Figure 2 has phonetic yield *she like which book*.

For two expressions $\phi, \chi \in Exp(Feat)$, $[_{<}\phi, \chi]$ (respectively, $[_{>}\phi, \chi]$) denotes the complex expression $\psi = \langle N_\psi, \lhd_\psi^*, \prec_\psi, <_\psi, label_\psi \rangle \in Exp(Feat)$ for which ϕ and χ are those two subexpressions such that $r_\psi \lhd_\psi r_\phi$, $r_\psi \lhd_\psi r_\chi$ and $r_\phi \prec_\psi r_\chi$, and such that $r_\phi <_\psi r_\chi$ (respectively $r_\chi <_\psi r_\phi$), where r_ϕ, r_χ and r_ψ are the roots of ϕ, χ and ψ, respectively.

For any $\phi, \chi, \psi \in Exp(Feat)$ such that χ is a subexpression of ϕ, $\phi\{\chi/\psi\}$ is the expression which results from substituting ψ for χ in ϕ.

As before we use *MG* as a shorthand for *minimalist grammar*.

Definition 1.2 An *MG without both SMC and SPIC ($MG^{/-,-/}$)* is a 5-tuple of the form $\langle \neg Syn, Syn, Lex, \Omega, \mathtt{c} \rangle$ where Ω is the operator set consisting of the structure building functions $merge^{/-/}$ and $move^{/-,-/}$ defined as in (me$^{\text{-SPIC}}$) and (mo$^{\text{-SMC,-SPIC}}$) below, respectively, and where *Lex* is a *lexicon (over Feat)*, a finite set of simple expressions over *Feat*, with each lexical item $\tau \in Lex$ being of the form $\langle \{r_\tau\}, \lhd_\tau^*, \prec_\tau, <_\tau, label_\tau \rangle$ such that $label_\tau(r_\tau)$ is an element in $\{\#\}(\textit{M-Select} \cup \textit{M-Licensors})^* \textit{Base}\, \textit{M-Licensees}^* \textit{Phon}^* \textit{Sem}^*$.

The operators from Ω build larger structure from given expressions by successively checking "from left to right" the instances of syntactic features appearing within the leaf-labels of the expressions involved. The symbol # serves to mark which feature instances have already been checked by the application of some structure building operation.

(me$^{\text{-SPIC}}$) $merge^{/-/}$ is a partial mapping from $Exp(Feat) \times Exp(Feat)$ into $Exp(Feat)$. For any $\phi, \chi \in Exp(Feat)$, $\langle \phi, \chi \rangle$ is in $Dom(merge^{/-/})$ if for some category $\mathtt{x} \in Base$ and $\alpha, \alpha', \beta, \beta' \in Feat^*$, conditions (me.i) and (me.ii) are fulfilled:[10]

(me.i) the head-label of ϕ is $\alpha\#{=}\mathtt{x}\alpha'$ (i.e. ϕ displays m-selector $=\mathtt{x}$),

(me.ii) the head-label of χ is $\beta\#\mathtt{x}\beta'$ (i.e. χ displays category $\mathtt{x}$).

Then,

(me.1) $merge^{/-/}(\phi, \chi) = [_{<}\phi', \chi']$ if ϕ is simple,

(me.2) $merge^{/-/}(\phi, \chi) = [_{>}\chi', \phi']$ if ϕ is complex,

where ϕ' and χ' result from ϕ and χ, respectively, just by interchanging the instance of # and the instance of the feature directly following the instance of # within the respective head-label (cf. Figure 3).

$$merge : Exp(Feat) \times Exp(Feat) \longrightarrow Exp(Feat)$$

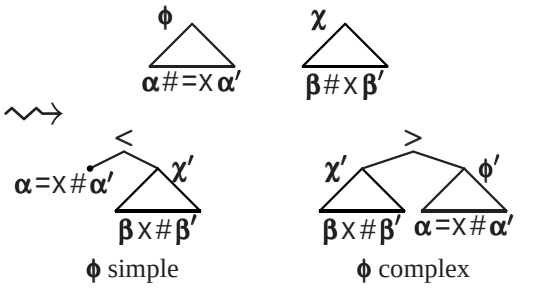


Figure 3: The merge-operator.

(mo$^{\text{-SMC,-SPIC}}$) $move^{/-,-/}$ is a partial mapping from $Exp(Feat)$ to $\mathcal{P}_{\text{fin}}(Exp(Feat))$.[11] An expression $\phi \in Exp(Feat)$ is in $Dom(move^{/-,-/})$ if $-\mathtt{x} \in \textit{M-Licensees}$ and $\alpha, \alpha' \in Feat^*$ exist such that (mo.i) and (mo.ii) are true:

(mo.i) the head-label of ϕ is $\alpha\#{+}\mathtt{x}\alpha'$ (i.e. ϕ displays licensor $+\mathtt{x}$),

(mo.ii) there is a $\chi \in MaxProj(\phi)$ with head-label $\beta\#{-}\mathtt{x}\beta'$ for some $\beta, \beta' \in Feat^*$ (i.e. $\chi \in MaxProj(\phi)$ exists displaying feature $-\mathtt{x}$).

Then,

[10] For a partial function f from a class A into a class B, $Dom(f)$ is the domain of f, i.e., the class of all $x \in A$ for which $f(x)$ is defined.

[11] $\mathcal{P}_{\text{fin}}(Exp(Feat))$ is the class of all finite subsets of $Exp(Feat)$.

$$move^{/-,-/}(\phi) = \left\{ [_{>}\chi', \phi'] \;\middle|\; \begin{array}{l} \chi \in MaxProj(\phi) \text{ with} \\ \text{head-label } \beta\#{-}x\beta' \text{ for} \\ \text{some } \beta, \beta' \in Feat^* \end{array} \right\},$$

where ϕ' results from ϕ by interchanging the instance of # and the instance of $+x$ directly following it within the head-label of ϕ, while the subtree χ is replaced by a single node labeled ε. χ' arises from χ by interchanging the instance of # and the instance of $-x$ immediately to its right within the head-label of χ (cf. Figure 4).

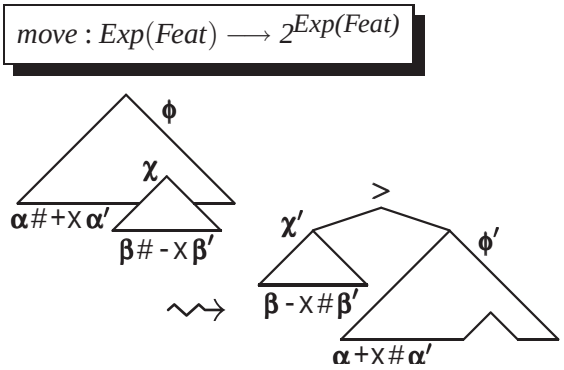


Figure 4: The move-operator.

Definition 1.3 An *MG without SMC, but with SPIC (MG$^{/-,+/}$)* is a five-tuple of the form $\langle \neg Syn, Syn, Lex, \Omega, c \rangle$ where Ω is the operator set consisting of the structure building functions $merge^{/+/}$ and $move^{/-,+/}$ defined as in (me$^{+\text{SPIC}}$) and (mo$^{-\text{SMC},+\text{SPIC}}$) below, respectively, and where *Lex* is a lexicon over *Feat* defined as in Definition 1.2.

(me$^{+\text{SPIC}}$) $merge^{/+/}$ is a partial mapping from $Exp(Feat) \times Exp(Feat)$ into $Exp(Feat)$. For any $\phi, \chi \in Exp(Feat)$, $\langle \phi, \chi \rangle$ is in $Dom(merge^{/+/})$ if for some category $x \in Base$ and $\alpha, \alpha', \beta, \beta' \in Feat^*$, conditions (me.i) and (me.ii) above and (me.spic) are fulfilled:

(me.spic) if ϕ is complex then there is no $\psi \in MaxProj(\chi)$ with head-label $\gamma\#y\gamma'$ for some $y \in Licensees$ and $\gamma, \gamma' \in Feat^*$ (i.e. the selected specifier does not properly contain a maximal projection with an unchecked licensee feature instance).

Then, $merge^{/+/}(\phi, \chi) = merge^{/-/}(\phi, \chi)$.

(mo$^{-\text{SMC},+\text{SPIC}}$) The operator $move^{/-,+/}$ is a partial mapping from $Exp(Feat)$ to $\mathcal{P}_{\text{fin}}(Exp(Feat))$. A $\phi \in Exp(Feat)$ is in $Dom(move^{/-,+/})$ if for some $-x \in M\text{-}Licensees$ and $\alpha, \alpha' \in Feat^*$, (mo.i) and (mo.ii) given above and (mo.spic) are true:

(mo.spic) there is no $\psi \in MaxProj(\chi)$ different from χ, and with head-label $\gamma\#y\gamma'$ for some $y \in Licensees$ and $\gamma, \gamma' \in Feat^*$ (i.e. the maximal projection moved to the specifier does not itself properly contain itself a maximal projection displaying an unchecked syntactic feature instance).

Then, $move^{/-,+/}(\phi) = move^{/-,-/}(\phi)$.

The formulation of the SPIC as presented here, could be seen as an "active" variant, preventing the creation of expressions which include specifiers from which proper extraction could potentially take place. The MG-version presented in Stabler 1999 allows derivation of such expressions, but prevents these expressions to enter a convergent derivation by explicitly stating a "passive" formulation of the SPIC, demanding that the maximal projection $\chi \in MaxProj(\phi)$ which has feature $-x$ can only move in order to check the licensee, if there exists a $\psi \in Comp^+(\phi)$ with $\chi = \psi$ or $\chi \in Spec(\psi)$.

Definition 1.4 An *MG with SMC, but without SPIC (MG$^{/+,-/}$)* is a five-tuple of the form $\langle \neg Syn, Syn, Lex, \Omega, c \rangle$ where Ω is the operator set consisting of the structure building functions $merge^{/-/}$ and $move^{/+,-/}$ defined as in (me$^{-\text{SPIC}}$) above and (mo$^{+\text{SMC},-\text{SPIC}}$) below, respectively, and

where *Lex* is a lexicon over *Feat* defined as in Definition 1.2.

(mo$^{+\text{SMC},-\text{SPIC}}$) The operator $move^{/+,-/}$ is a partial mapping from $Exp(Feat)$ to $\mathcal{P}_{\text{fin}}(Exp(Feat))$. A $\phi \in Exp(Feat)$ is in $Dom(move^{/+,-/})$ if for some $-\mathtt{x} \in$ *M-Licensees* and $\alpha, \alpha' \in Feat^*$, (mo.i) and (mo.ii) above and (mo.smc) are true:

(mo.smc) exactly one $\chi \in MaxProj(\phi)$ exists with head-label $\gamma\#-\mathtt{x}\gamma'$ for some $\gamma, \gamma' \in Feat^*$ (i.e. exactly one $\chi \in MaxProj(\phi)$ displays $-\mathtt{x}$).[12]

Then, $move^{/+,-/}(\phi) = move^{/-,-/}(\phi)$.

Definition 1.5 An *MG with both SMC and SPIC ($MG^{/+,+/}$)* is a five-tuple of the form $\langle \neg Syn, Syn, Lex, \Omega, \mathtt{c} \rangle$ where Ω is the operator set consisting of the structure building functions $merge^{/+/}$ and $move^{/+,+/}$ defined as in (me$^{+\text{SPIC}}$) above and (mo$^{+\text{SMC},+\text{SPIC}}$) below, respectively, and where *Lex* is a lexicon over *Feat* defined as in Definition 1.2.

(mo$^{+\text{SMC},+\text{SPIC}}$) The operator $move^{/+,+/}$ is a partial mapping from $Exp(Feat)$ to $\mathcal{P}_{\text{fin}}(Exp(Feat))$. A $\phi \in Exp(Feat)$ is in $Dom(move^{/+,+/})$ if for some $-\mathtt{x} \in$ *M-Licensees* and $\alpha, \alpha' \in Feat^*$, (mo.i), (mo.ii), (mo.spic) and (mo.smc) above are true.

Then, $move^{/+,+/}(\phi) = move^{/-,-/}(\phi)$.[13]

Let $G = \langle \neg Syn, Syn, Lex, \Omega, \mathtt{c} \rangle$ be an MG$^{/-,-/}$, MG$^{/-,+/}$, MG$^{/+,-/}$, respectively MG$^{/+,+/}$. For the sake of convenience, we refer to the corresponding merge- and move-operator in Ω by *merge* and *move*, respectively. Then the *closure of G*, $CL(G)$, is the set $\bigcup_{k \in \mathbb{N}} CL^k(G)$, where $CL^0(G) = Lex$, and for $k \in \mathbb{N}$,[14] $CL^{k+1}(G) \subseteq Exp(Feat)$ is recursively defined as the set

$$CL^k(G)$$
$$\cup \{merge(\phi, \chi) \mid \langle \phi, \chi \rangle \in Dom(merge) \cap CL^k(G) \times CL^k(G)\}$$
$$\cup \bigcup_{\phi \in Dom(move) \cap CL^k(G)} move(\phi).$$

The set $\{\tau \mid \tau \in CL(G) \text{ and } \tau \text{ complete}\}$, denoted $T(G)$, is the *minimalist tree language derivable by G*. The set $\{Y_{Phon}(\tau) \mid \tau \in T(G)\}$, denoted $L(G)$, is the *minimalist (string) language derivable by G*.

In the following we will use the notation $MG_{adj,ext}$ as a shorthand for *minimalist grammar with generalized adjunction and extraposition*.

Definition 1.6 An *$MG_{adj,ext}$ without both SMC and AIC ($MG^{/-,-/}_{adj,ext}$)* is a 5-tuple $G = \langle \neg Syn, Syn, Lex, \Omega, \mathtt{c} \rangle$ where Ω is the operator set consisting of the functions $merge^{/-/}$, $move^{/-,-/}$, $adjoin^{/-/}$ and $scramble^{/-,-/}$ defined as in (me$^{-\text{SPIC}}$) and (mo$^{-\text{SMC},-\text{SPIC}}$) above, and (ad$^{-\text{AIC}}$) and (sc$^{-\text{SMC},-\text{AIC}}$) below, respectively, and where *Lex* is a *lexicon (over Feat)*, a finite set of simple expressions over *Feat*, and each lexical item $\tau \in Lex$ is of the form $\langle \{r_\tau\}, \lhd_\tau^*, \prec_\tau, <_\tau, label_\tau \rangle$ such that $label_\tau(r_\tau)$ belongs to $\{\#\}(M\text{-}Select \cup M\text{-}Licensors)^*(Base \cup A\text{-}Select)Licensees^* Phon^* Sem^*$.

(ad$^{-\text{AIC}}$) $adjoin^{/-/}$ is a partial mapping from $Exp(Feat) \times Exp(Feat)$ into the class $\mathcal{P}_{\text{fin}}(Exp(Feat))$. A pair $\langle \phi, \chi \rangle$ with $\phi, \chi \in Exp(Feat)$ belongs to $Dom(adjoin^{/-/})$ if for some category $\mathtt{x} \in Base$ and $\alpha, \alpha' \in Feat^*$, conditions (ad.i) and (ad.ii) are fulfilled:

(ad.i) the head-label of ϕ is $\alpha\#\approx_{\mathtt{x}}\alpha'$ (i.e. ϕ displays a-selector $\approx_{\mathtt{x}}$), and

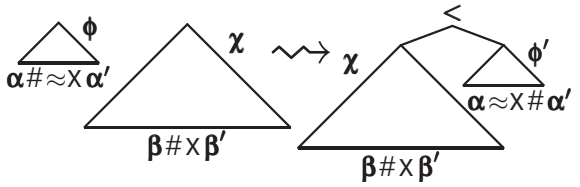
[12] Note that condition (mo.smc) implies (mo.ii).

[13] Note that the sets $move^{/+,-/}(\phi)$ and $move^{/+,+/}(\phi)$ in (mo$^{+\text{SMC},-\text{SPIC}}$) and (mo$^{+\text{SMC},+\text{SPIC}}$), respectively, both are singleton sets because of (mo.smc). Thus, the corresponding functions can easily be identified with one from $Exp(Feat)$ to $Exp(Feat)$.

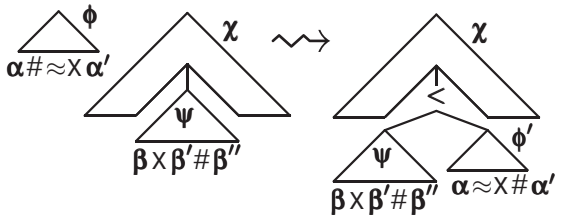
[14] $\mathbb{N}$ is the set of all non-negative integers.

(ad.ii) there exists some $\psi \in \mathit{MaxProj}(\chi)$ with head-label of the form $\beta\#\mathrm{x}\beta'$ or $\beta\mathrm{x}\beta'\#\beta''$ for some $\beta, \beta', \beta'' \in \mathit{Feat}^*$

$\mathit{adjoin} : \mathit{Exp}(\mathit{Feat}) \times \mathit{Exp}(\mathit{Feat}) \longrightarrow 2^{\mathit{Exp(Feat)}}$



cyclic adjunction (Frey and Gärtner 2002)



acyclic/late adjunction (Gärtner and Michaelis 2003)

Figure 5: The adjoin-operator.

Then,

$$\mathit{adjoin}^{/-/}(\phi,\chi) = \left\{ \chi\{\psi/[_{<}\psi,\phi']\} \;\middle|\; \begin{array}{l}\psi \in \mathit{MaxProj}(\chi) \text{ with head-label} \\ \beta\#\mathrm{x}\beta' \text{ or } \beta\mathrm{x}\beta'\#\beta'' \text{ for some} \\ \beta,\beta',\beta'' \in \mathit{Feat}^*\end{array} \right\},$$

where ϕ' results from ϕ by interchanging the instances of # and $\approx$x, the latter directly following the former in the head-label of ϕ (cf. Figure 5).

(sc$^{\text{-SMC,-AIC}}$) The function $\mathit{scramble}^{/-,-/}$ maps partially from $\mathit{Exp}(\mathit{Feat})$ into the class $\mathcal{P}_{\text{fin}}(\mathit{Exp}(\mathit{Feat}))$. A $\phi \in \mathit{Exp}(\mathit{Feat})$ is in $\mathit{Dom}(\mathit{scramble}^{/-,-/})$ if for some $\mathrm{x} \in \mathit{Base}$ and $\alpha, \alpha' \in \mathit{Feat}^*$, (sc.i) and (sc.ii) are true:

(sc.i) the head-label of ϕ is $\alpha\#\mathrm{x}\alpha'$ (i.e. ϕ displays category x), and

(sc.ii) there is a $\chi \in \mathit{MaxProj}(\phi)$ with head-label $\beta\#{\sim}\mathrm{x}\beta'$ for some $\beta, \beta' \in \mathit{Feat}^*$ (i.e. there is some $\chi \in \mathit{MaxProj}(\phi)$ displaying $\sim$x).

Then,

$$\mathit{scramble}^{/-,-/}(\phi) = \left\{ [_{<}\phi',\chi'] \;\middle|\; \begin{array}{l}\chi \in \mathit{MaxProj}(\phi) \text{ with} \\ \text{head-label } \beta\#{\sim}\mathrm{x}\beta' \text{ for} \\ \text{some } \beta,\beta' \in \mathit{Feat}^*\end{array} \right\},$$

where $\phi' \in \mathit{Exp}(\mathit{Feat})$ is identical to ϕ except for the fact that the subtree χ is replaced by a single node labeled ε. $\chi' \in \mathit{Exp}(\mathit{Feat})$ arises from χ by interchanging the instance of # and the instance of $\sim$x immediately to its right within the head-label of χ (cf. Figure 6).

$\mathit{scramble} : \mathit{Exp}(\mathit{Feat}) \longrightarrow 2^{\mathit{Exp(Feat)}}$

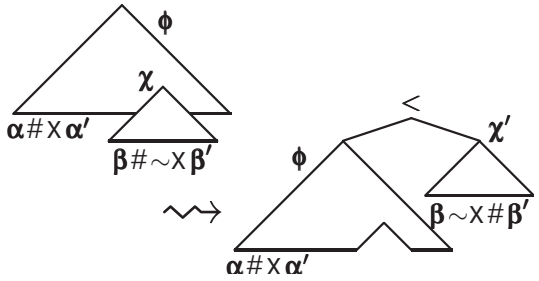


Figure 6: The scramble-operator.

Definition 1.7 An $\mathit{MG}_{\mathit{adj,ext}}$ *without SMC, but with AIC* ($\mathit{MG}^{/-,+/}_{\mathit{adj,ext}}$) is a five-tuple of the form $\langle \neg\mathit{Syn}, \mathit{Syn}, \mathit{Lex}, \Omega, \mathrm{c}\rangle$ where Ω is the operator set consisting of the structure building functions $\mathit{merge}^{/-/}$, $\mathit{move}^{/-,-/}$, $\mathit{adjoin}^{/+/}$ and $\mathit{scramble}^{/-,+/}$ defined as in (me$^{\text{-SPIC}}$) and (mo$^{\text{-SMC,-SPIC}}$) above, and (ad$^{\text{+AIC}}$) and (sc$^{\text{-SMC,+AIC}}$) below, respectively, and where *Lex* is a lexicon over *Feat* defined as in Definition 1.6.

(ad$^{\text{+AIC}}$) $\mathit{adjoin}^{/+/}$ is a partial mapping from $\mathit{Exp}(\mathit{Feat}) \times \mathit{Exp}(\mathit{Feat})$ into the class $\mathcal{P}_{\text{fin}}(\mathit{Exp}(\mathit{Feat}))$. A pair $\langle \phi, \chi\rangle$ with

$\phi, \chi \in Exp(Feat)$ belongs to $Dom(adjoin^{/+/})$ if for some category $\mathtt{x} \in Base$ and $\alpha, \alpha' \in Feat^*$, conditions (ad.i) and (ad.ii) above and (ad.aic) are fulfilled:

(ad.aic) there is no $\psi \in MaxProj(\phi)$ with head-label $\gamma\#y\gamma'$ for some $y \in Licensees$ and $\gamma, \gamma' \in Feat^*$ (i.e. the adjunct does not properly contain a maximal projection with an unchecked syntactic feature instance).

Then, $adjoin^{/+/}(\phi, \chi) = adjoin^{/-/}(\phi, \chi)$.

(sc$^{\text{-SMC,+AIC}}$) The function $scramble^{/-,+/}$ maps partially from $Exp(Feat)$ into the class $\mathcal{P}_{\text{fin}}(Exp(Feat))$. A $\phi \in Exp(Feat)$ is in $Dom(scramble^{/-,+/})$ if for some $\mathtt{x} \in Base$ and $\alpha, \alpha' \in Feat^*$, (sc.i) and (sc.ii) above and (sc.aic) are true:

(sc.aic) there is no $\psi \in MaxProj(\chi)$ different from χ, and with head-label $\gamma\#y\gamma'$ for some $y \in Licensees$ and $\gamma, \gamma' \in Feat^*$ (i.e. the maximal projection scrambled/extraposed to an adjunct position does not itself properly contain a maximal projection displaying an unchecked syntactic feature instance).

Then, $scramble^{/-,+/}(\phi) = scramble^{/-,-/}(\phi)$.

Definition 1.8 An *$MG_{adj,ext}$ with SMC, but without AIC ($MG^{/+,-/}_{adj,ext}$)* is a five-tuple of the form $\langle \neg Syn, Syn, Lex, \Omega, \mathtt{c} \rangle$ where Ω is the operator set consisting of the structure building functions $merge^{/-/}$, $move^{/+,-/}$, $adjoin^{/-/}$ and $scramble^{/+,-/}$ defined as in (me$^{\text{-SPIC}}$), (mo$^{\text{+SMC,-SPIC}}$) and (ad$^{\text{-AIC}}$) above and (sc$^{\text{+SMC,-AIC}}$) below, respectively, and where *Lex* is a lexicon over *Feat* defined as in Definition 1.6.

(sc$^{\text{+SMC,-AIC}}$) $scramble^{/+,-/}$ is a partial mapping from $Exp(Feat)$ to $\mathcal{P}_{\text{fin}}(Exp(Feat))$. A tree $\phi \in Exp(Feat)$ is in $Dom(scramble^{/+,-/})$ if for some $\mathtt{x} \in Base$ and $\alpha, \alpha' \in Feat^*$, (sc.i) and (sc.ii) above and (sc.smc) are true:

(sc.smc) exactly one $\chi \in MaxProj(\phi)$ exists with head-label $\gamma\#\sim\mathtt{x}\gamma'$ for some $\gamma, \gamma' \in Feat^*$ (i.e. exactly one $\chi \in MaxProj(\phi)$ displays $\sim\mathtt{x}$).[15]

Then, $scramble^{/+,-/}(\phi) = scramble^{/-,-/}(\phi)$.

Definition 1.9 An *$MG_{adj,ext}$ with both SMC and AIC ($MG^{/+,+/}_{adj,ext}$)* is a five-tuple of the form $\langle \neg Syn, Syn, Lex, \Omega, \mathtt{c} \rangle$ where Ω is the operator set consisting of the structure building functions $merge^{/-/}$, $move^{/+,-/}$, $adjoin^{/+/}$ and $scramble^{/+,+/}$ defined as in (me$^{\text{-SPIC}}$), (mo$^{\text{+SMC,-SPIC}}$) and (ad$^{\text{+AIC}}$) above and (sc$^{\text{+SMC,+AIC}}$) below, respectively, and where *Lex* is a lexicon over *Feat* defined as in Definition 1.6.

(sc$^{\text{+SMC,+AIC}}$) $scramble^{/+,+/}$ is a partial mapping from $Exp(Feat)$ to $\mathcal{P}_{\text{fin}}(Exp(Feat))$. A tree $\phi \in Exp(Feat)$ is in $Dom(scramble^{/+,+/})$ if for some $\mathtt{x} \in Base$ and $\alpha, \alpha' \in Feat^*$, (sc.i), (sc.ii), (sc.aic) and (sc.smc) above are true.

Then, $scramble^{/+,+/}(\phi) = scramble^{/-,-/}(\phi)$.[16]

Let $G = \langle \neg Syn, Syn, Lex, \Omega, \mathtt{c} \rangle$ be an $\text{MG}^{/-,-/}_{\text{adj,ext}}$, $\text{MG}^{/-,+/}_{\text{adj,ext}}$, $\text{MG}^{/+,-/}_{\text{adj,ext}}$, respectively $\text{MG}^{/+,+/}_{\text{adj,ext}}$. For the sake of convenience, we refer to the corresponding merge-, move-, adjoin- and scramble-operator in Ω by *merge*, *move*, *adjoin* and *scramble*, respectively. Let $CL^0(G) = Lex$, and for each $k \in \mathbb{N}$, let $CL^{k+1}(G) \subseteq Exp(Feat)$ be recursively defined as

$$
\begin{aligned}
& CL^k(G) \\
\cup\ & \{ merge(\phi, \chi) \mid \\
& \qquad \langle \phi, \chi \rangle \in Dom(merge) \cap CL^k(G) \times CL^k(G) \} \\
\cup\ & \bigcup\nolimits_{\phi \in Dom(move) \cap CL^k(G)} move(\phi) \\
\cup\ & \bigcup\nolimits_{\langle \phi, \chi \rangle \in Dom(adjoin) \cap CL^k(G) \times CL^k(G)} adjoin(\phi, \chi) \\
\cup\ & \bigcup\nolimits_{\phi \in Dom(scramble) \cap CL^k(G)} scramble(\phi)
\end{aligned}
$$

[15] Note that condition (sc.smc) implies (sc.ii).

[16] $scramble^{/+,-/}(\phi)$ and $scramble^{/+,+/}(\phi)$ in (sc$^{\text{+SMC,-AIC}}$) and (sc$^{\text{+SMC,+AIC}}$), respectively, both are singleton sets because of (sc.smc). Thus, the corresponding functions can easily be identified with one from $Exp(Feat)$ to $Exp(Feat)$.

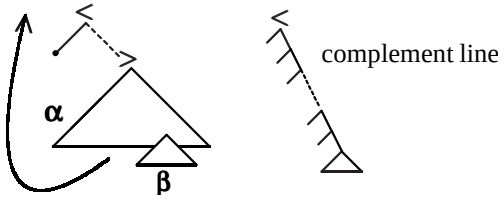
Then, $\bigcup_{k\in\mathbb{N}} CL^k(G)$ is the *closure of* G, denoted $CL(G)$. The set $\{\tau \mid \tau \in CL(G) \text{ and } \tau \text{ complete}\}$, denoted $T(G)$, is the *minimalist tree language derivable by* G. The set $\{Y_{Phon}(\tau) \mid \tau \in T(G)\}$, denoted $L(G)$, is the *minimalist (string) language derivable by* G.

2 Locality Conditions and Complexity Results

2.1 The Specifier Island Condition

Figure 7 presents an example of a non-mildly context-sensitive MG not fulfilling the SMC but the SPIC, and deriving a language without constant growth property, namely, $\{a^{2^n} \mid n \geq 0\}$. The central column shows the lexical items as they are drawn from the lexicon, i.e., with all features unchecked. Arrows show the possible orders of interaction among lexical items and resulting constituents in terms of *merge*. Intermediate steps of *move* are left implicit.

As shown by Kobele and Michaelis (2005), not only this language, but in fact every language of type 0 can be derived by some MG not fulfilling the SMC but the SPIC for essentially two reasons: a) because of the SPIC, movement of a constituent



α into a specifier position freezes every proper subconstituent β within α, and b) without the SMC, therefore, the complement line of a tree (in terms of the successively embedded complements) can technically be employed as a queue. As is well-known, systems able to simulate queues are able to generate arbitrary type 0-languages.

Starting the "outer" cycle of our example in Figure 7, the currently derived tree shows 2^n+1 successively embedded complements on the complement line, all with an unchecked instance of `-l`, except for the lowest one, which displays `-m`. (n equals the number of cycles already completed.) The initializing selecting head `#.=v.z.-l` introduces an additional licensee `-l` to create string a on a cycleless derivation. Going through the cycle provides a successive bottom-to-top "roll-up" of those complements in order to check the displayed features. Thereby, 2^{n+1}+1 successively embedded complements on the complement line are created, again all displaying feature `-l` except for the lowest, which displays feature `-m`. Leaving the cycle procedure after a cycle has been completed leads to a final checking of the displayed licensees, where for each checked `-l` an a is introduced in the structure. This is the only way to create a convergent derivation.[17]

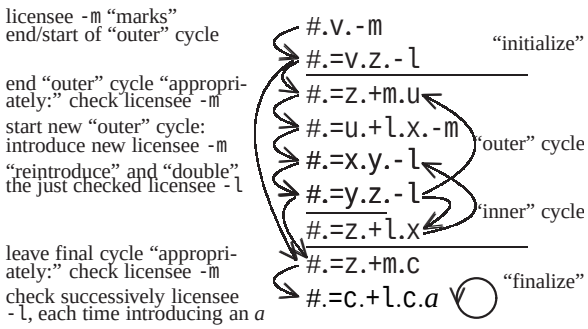


Figure 7: MG-example.

Figure 12 of the Appendix summarizes what we know about the interaction of SMC and SPIC,[18] where $\mathcal{L}_1 \searrow \mathcal{L}_2$, respectively $\mathcal{L}_2 \swarrow \mathcal{L}_1$, means "language class $\mathcal{L}_2$ is lower in generative capacity than language class $\mathcal{L}_1$" while $\mathcal{L}_1 \nearrow \mathcal{L}_2$, respectively $\mathcal{L}_2 \nwarrow \mathcal{L}_1$, means "language class $\mathcal{L}_2$ is higher in generative capacity than language class $\mathcal{L}_1$." Crucially, adding the SPIC can either properly reduce complexity (lower left side) or properly increase

[17] For further details see Gärtner and Michaelis 2005.

[18] The MIX language is the language of all finite strings consisting of an equal number of a's, b's, and c's appearing in arbitrary order.

complexity (upper right side). What the SPIC does depends on the presence or absence of SMC. Its behavior is thus non-monotonic.

The SPIC, in fact, strictly reduces the generative capacity, when the SMC is present. Michaelis 2005 presents a string language which is derivable by an MGs obeying the SMC, but falls outside the class of string languages derivable by MGs obeying both the SMC and SPIC.[19]

2.2 The Adjunct Island Condition

In this section we look at MGs with *(late) adjunction* and *scrambling/extraposition* and the effects of imposing the AIC in a situation where the SMC alone appears to be too weak to guarantee mild context-sensitivity. Figure 8 gives a schematic illustration of countercyclic or late adjunction, i.e. adjunction to a non-root position.

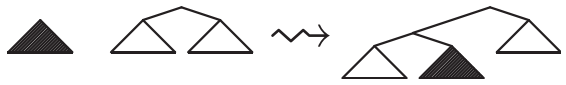
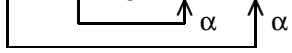


Figure 8: Countercyclic/late adjunction.

For the complexity issue we are interested in here it is important to note that late adjunction is capable of circumventing the SMC (cf. Gärtner and Michaelis 2003). (1) presents a case where this is actually welcome.

(1) [[[[Only those papers t_i]$_k$ did
[everyone t_j] read t_k]
[who was on the committee]$_j$]
[that deal with adjunction]$_i$]

We assume for simplicity that both of the relative clauses in (1) are extraposed by an application of rightward scrambling and are adjoined to CP. This is very roughly sketched in (2).

(2) * $[_{CP}$ CP_2 CP_1]


For standard bottom-up derivations, (2) violates the SMC, given the simultaneous presence of alpha (i.e. $\sim$c) on both CPs. However, as sketched in (3), a derivational sequence of (first) extraposition, late adjunction and (second) extraposition voids this problem.

(3)

$[_{CP}$ CP_1^{α}]		start here
$[_{CP}$ –]	$CP_1^{\not\alpha}$	move CP_1, check α
$[_{CP}CP_2^{\alpha}$ –]	$CP_1^{\not\alpha}$	late adjoin CP_2
$[_{CP}$ – –]	$CP_1^{\not\alpha}$ $CP_2^{\not\alpha}$	move CP_2, check α

The proof that MGs without late adjunction, but obeying the SMC are mildly context-sensitive rests on the technical possibility of removing checked features from the structures. Formally, late adjunction creates a situation where in order to locate the individual adjunction sites, an a priori not bounded amount of (categorial) information has to be stored during a derivation, i.e., adjunction sites have to be kept accessible. Therefore it is unclear whether, in general, MGs allowing late adjunction still belong to the same complexity class. If, however, the AIC is imposed, we can apply a specific reduction method in proving that for the resulting MGs the old complexity result holds. Under this reduction, however, late adjunction can only be simulated if the adjunct does not properly contain constituents bearing unchecked licensees. But, this is exactly the situation where the AIC comes in. From a linguistic point of view it is rather natural to exclude extraction from adjuncts as Huang (1982) argued. This means that the weak generative capacity of MGs with late adjunction and extraposition can be kept within the bounds of standard MGs, i.e. mild context-sensitivity, if the AIC is imposed in addition to the SMC. Figure 13 of the Appendix summarizes our results for SMC/AIC-interaction.

[19] More concretely, Michaelis 2005 proves the latter class to be (strictly) subsumed by the class of *indexed languages (ILs)* in the sense of Aho 1968, and the corresponding language presented as a case in point does, as shown in Staudacher 1993, not even belong to the class of ILs.

3 Further Outlook

3.1 Multiple Wh-Constructions and the SMC

One phenomenon appearing to challenge the SMC adopted here is multiple wh-fronting in Slavic languages. Take, e.g., (4) from Bulgarian (Richards 2001, p. 249).

(4) *Koj$_i$ kogo$_j$ kakvo$_k$* t$_i$ *e pital* t$_j$ t$_k$
Who whom what AUX ask
'Who asked whom what?'

On standard assumptions, (4) requires three licensee instances of type -wh, which are successively checked in the C-domain. The required pre-movement representation, (5), is ruled out by (the strictest version of) the SMC.

(5) [$_{IP}$ -wh.koj e [$_{VP}$ pital -wh.kogo -wh.kakvo]]

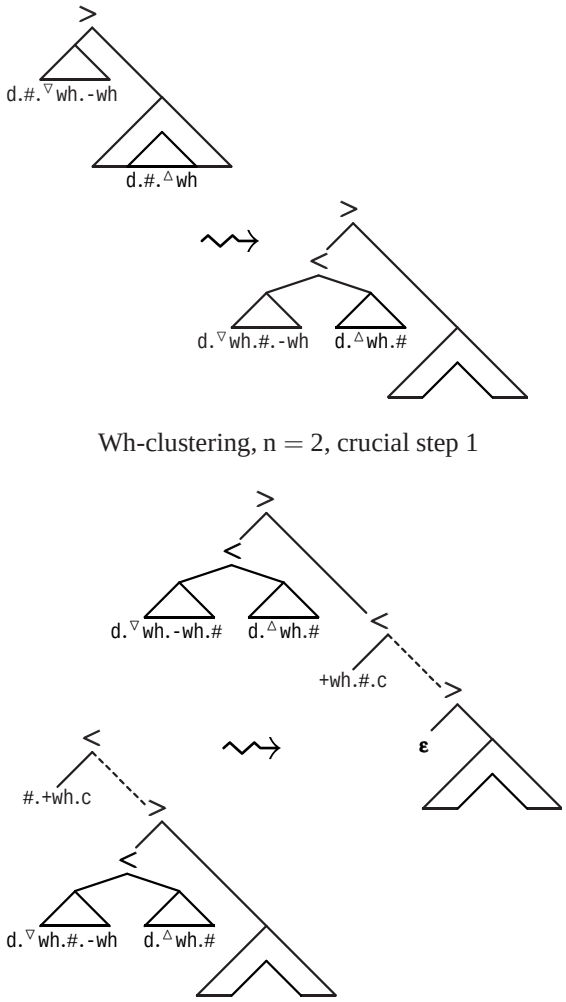
A corresponding SMC-violation can be circumvented, however, if we adopt the wh-cluster hypothesis as argued for by Sabel (1998; 2001) and Grewendorf (2001) going back to Rudin (1988). Under this perspective, wh-expressions undergo successive cluster-formation before the resulting cluster takes a single wh-movement step, in compliance with the SMC. For this we have to add the feature type of *c(luster)-licensees* and *-licensors* to MGs.

c(luster)-licensees: $^{\triangle}x, {}^{\triangle}y, {}^{\triangle}z, \ldots$
c(luster)-licensors: $^{\triangledown}x, {}^{\triangledown}y, {}^{\triangledown}z, \ldots$

In Figure 9 we show a derivation with two wh-phrases. For cases with three or more such phrases the intermediate ones have to be of type d.$^{\triangledown}$wh.$^{\triangle}$wh.

Note that additional word order variation can be found in Bulgarian, as shown in (6) (Richards 2001, p. 249).

(6) *Koj kakvo kogo e pital*



Wh-clustering, n = 2, crucial step 1

Wh-clustering, n = 2, crucial step 2

Figure 9: Wh-clustering involving c-licensors and c-licensees.

This can be derived if cluster-formation is preceded by a scrambling-step of *kakvo* across *kogo* to VP, which requires it to be of type d.∼v.$^{\triangledown}$wh. See Sabel (1998) for more discussion of wh- and focus-driven movements in multiple wh-configurations. A formal definition of the cluster-operator is given now.[20]

[20]Given the "specifier condition" (cl.i), it is clear that—in order to potentially end up with a convergent derivation—within a lexical item an instance of a c-licensor must be immediately preceded by an instance of a basic category, a-selector,

(cl^{+SMC}) The operator *cluster* is a partial mapping from $Exp(Feat)$ to $Exp(Feat)$. An expression $\phi \in Exp(Feat)$ is in $Dom(cluster)$ if there are a c-licensee ${}^{\triangle}x$ and $\alpha, \alpha' \in Feat^*$ such that (cl.i), (cl.ii) and (cl.smc) are true:

(cl.i) there is a $\chi \in MaxProj(\phi)$ such that χ is the highest specifier of ϕ, and the head-label of χ is $\alpha\#^{\triangledown}x\alpha'$ (i.e. ϕ displays the corresponding c-licensor ${}^{\triangledown}x$),

(cl.ii) there is a $\psi \in MaxProj(\phi)$ with head-label $\beta\#^{\triangle}x\beta'$ for some $\beta, \beta' \in Feat^*$ (i.e. $\psi \in MaxProj(\phi)$ exists displaying ${}^{\triangle}x$).

(cl.smc) the existing $\psi \in MaxProj(\phi)$ from (cl.ii) is unique (i.e. there is exactly one $\psi \in MaxProj(\phi)$ displaying ${}^{\triangle}x$).

Then, $cluster(\phi) = \phi'\{\chi/[_{<}\chi', \psi']\}$,

where ϕ' results from ϕ by replacing the subtree ψ by a single node labeled ε. χ' results from χ by interchanging the instances of # and ${}^{\triangledown}x$, the latter directly following the former in the head-label of χ, while ψ' results from ψ by interchanging the instances of # and ${}^{\triangle}x$, the latter directly following the former in the head-label of ψ (cf. Figure 10).[21]

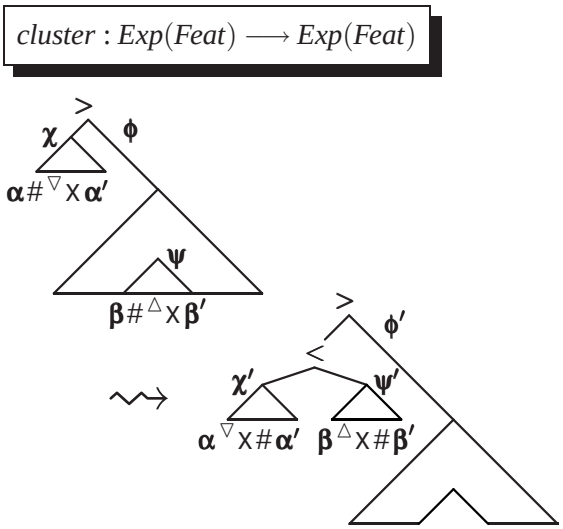


Figure 10: The cluster-operator.

Semantically, wh-cluster-formation can be interpreted as quantifier composition, a.k.a. "absorption" (Higginbotham and May 1981).

3.2 Future Research

There are two directions for future research that we consider of immediate relevance. First, it is necessary to develop a more systematic and complete combinatorics of LCs within and their complexity impact on MGs. Second, it is important to analyze the exact role LCs are playing in the other MCSG-frameworks (for TAGs see especially Frank 2002, for CCGs see Steedman 1996, for LIG-extensions see Wartena 1999), and try to establish the LCs' impact on complexity there. From the study of LCs within GB we already know that boundedness of chain overlap is crucial for $L^2_{K,P}$-definability of locality (Rogers 1998, p. 182, cf. the result on Scandinavian extraction in Miller 1991). This comes very close to the essence of what the SMC does within MGs. We also strongly suspect that it is the addition of *remnant movement (RM)* that puts MGs beyond context-freeness. A proof of the non-$L^2_{K,P}$-definability of recursively applicable RM would thus be a particularly interesting way of confirming this.

m-licensee, or s-licensee, i.e., in particular an instance of a c-licensor cannot be preceded by an instance of a c-licensee.

[21] As long as the SMC is obeyed, a proof showing that at least the weak generative capacity is unaffected seems to be straightforward by employing the "usual" reduction methods (cf. Michaelis 2001a).

References

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

Berwick, Robert C. (1991). Principle-based parsing. In Sells et al. 1991, pp. 115–226.

Chomsky, Noam (1965). *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA.

Chomsky, Noam (1973). Conditions on transformations. In S. Anderson and P. Kiparsky, eds., *A Festschrift for Morris Halle*, pp. 232–286. Holt, Rinehart and Winston, New York, NY.

Chomsky, Noam (1977). On wh-movement. In P. Culicover, T. Wasow, and A. Akmajian, eds., *Formal Syntax*, pp. 71–132. Academic Press, New York, NY.

Chomsky, Noam (1986). *Barriers*. MIT Press, Cambridge, MA.

Chomsky, Noam (1995). *The Minimalist Program*. MIT Press, Cambridge, MA.

Chomsky, Noam (2000). Minimalist inquiries. The framework. In R. Martin, D. Michaels, and J. Uriagereka, eds., *Step by Step. Essays on Minimalist Syntax in Honor of Howard Lasnik*, pp. 89–155. MIT Press, Cambridge, MA.

Chomsky, Noam (2001). Derivation by phase. In Michael Kenstowicz, ed., *Ken Hale. A Life in Language*, pp. 1–52. MIT Press, Cambridge, MA.

Cinque, Guglielmo (1990). *Types of A'-Dependencies*. MIT Press, Cambridge, MA.

Cornell, Thomas L. (1992). *Description Theory, Licensing Theory, and Principle-Based Grammars and Parsers*. Ph.D. thesis, University of California, Los Angeles, CA.

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

Frank, Robert (2002). *Phrase Structure Composition and Syntactic Dependencies*. MIT Press, Cambridge, MA.

Frey, Werner and Hans-Martin Gärtner (2002). On the treatment of scrambling and adjunction in minimalist grammars. In *Proceedings of the Conference on Formal Grammar (FGTrento),* Trento, pp. 41–52.

Gärtner, Hans-Martin and Jens Michaelis (2003). A note on countercyclicity and minimalist grammars. In *Proceedings of the Conference on Formal Grammar (FGVienna),* Vienna, pp. 103–114.

Gärtner, Hans-Martin and Jens Michaelis (2005). A note on the complexity of constraint interaction. Locality conditions and minimalist grammars. In P. Blache, E. Stabler, J. Busquets, and R. Moot, eds., *Logical Aspects of Computational Linguistics (LACL '05)*, LNAI Vol. 3492, pp. 114–130. Springer, Berlin, Heidelberg.

Grewendorf, Günther (2001). Multiple wh-fronting. *Linguistic Inquiry*, **32**:87–122.

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

Higginbotham, James and Robert May (1981). Questions, quantifiers, and crossing. *The Linguistic Review*, **1**:41–79.

Huang, James C.-T. (1982). *Logical Relations in Chinese and the Theory of Grammar*. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA.

Joshi, Aravind K. (1985). Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions? In D. R. Dowty, L. Karttunen, and A. M. Zwicky, eds., *Natural Language Parsing. Psychological, Computational, and Theoretical Perspectives*, pp. 206–250. Cambridge University Press, New York, NY.

Joshi, Aravind K. K. Vijay-Shanker, and David J. Weir (1991). The convergence of mildly context-sensitive grammar formalisms. In Sells et al. 1991, pp. 31–81.

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, pp. 83–93.

Kracht, Marcus (1995a). Is there a genuine perspective on feature structures? *Linguistics and Philosophy*, **18**:401–458.

Kracht, Marcus (1995b). Syntactic codes and grammar refinement. *Journal of Logic, Language and Information*, **4**:41–60.

Manzini, Rita (1992). *Locality*. MIT Press, Cambridge, MA.

Michaelis, Jens (2001a). Derivational minimalism is mildly context-sensitive. In M. Moortgat, ed., *Logical Aspects of Computational Linguistics (LACL '98)*, LNAI Vol. 2014, pp. 179–198. Springer, Berlin, Heidelberg.

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

Michaelis, Jens (2005). An additional observation on strict derivational minimalism. In *FG-MoL 2005. The 10th conference on Formal Grammar and The 9th Meeting on Mathematics of Language,* Edinburgh, pp. 103–113.

Miller, Philip H. (1991). Scandinavian extraction phenomena revisted. Weak and strong generative capacity. *Linguistics and Philosophy*, **14**:101–113.

Peters, Stanley and Graeme Ritchie (1973). On the generative power of transformational grammars. *Infor-*

mation Sciences, **6**:49–84.

Richards, Norvin (2001). *Movement in Language. Interactions and Architectures*. Oxford University Press, Oxford.

Rizzi, Luigi (1990). *Relativized Minimality*. MIT Press, Cambridge, MA.

Rogers, James (1998). *A Descriptive Approach to Language-Theoretic Complexity*. Studies in Logic, Language and Information. CSLI Publications, Stanford, CA.

Rudin, Catherine (1988). On multiple questions and multiple wh-fronting. *Natural Language and Linguistic Theory*, **6**:445–501.

Sabel, Joachim (1998). Principles and parameters of wh-movement. Habilitationsschrift, Universität Frankfurt.

Sabel, Joachim (2001). Deriving multiple head and phrasal movement. The cluster hypothesis. *Linguistic Inquiry*, **32**:532–547.

Sells, Peter, Stuart M. Shieber, and Thomas Wasow, eds. (1991). *Foundational Issues in Natural Language Processing*. MIT Press, Cambridge, MA.

Stabler, Edward P. (1992). *The Logical Approach to Syntax*. MIT Press, Cambridge, MA.

Stabler, Edward P. (1997). Derivational minimalism. In C. Retoré, ed., *Logical Aspects of Computational Linguistics (LACL '96)*, LNAI Vol. 1328, pp. 68–95. Springer, Berlin, Heidelberg.

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*, pp. 299–326. CSLI Publications, Stanford, CA.

Stabler, Edward P. (2001). Recognizing head movement. In de Groote et al. 2001, pp. 245–260.

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

Steedman, Mark (1996). *Surface Structure and Interpretation*. MIT Press, Cambridge, MA.

Wartena, Christian (1999). *Storage Structures and Conditions on Movement in Natural Language Syntax*. Ph.D. thesis, Potsdam University, Potsdam.

Appendix

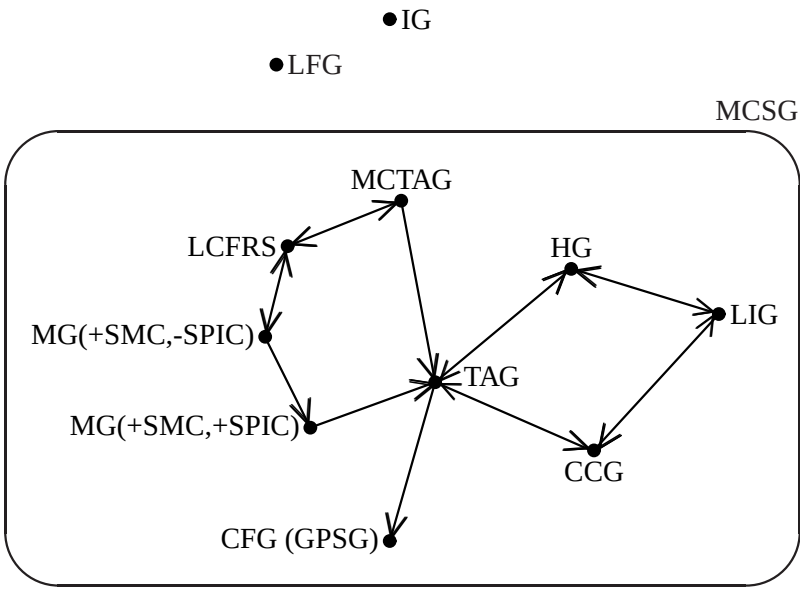


Figure 11: MCSG landscape.

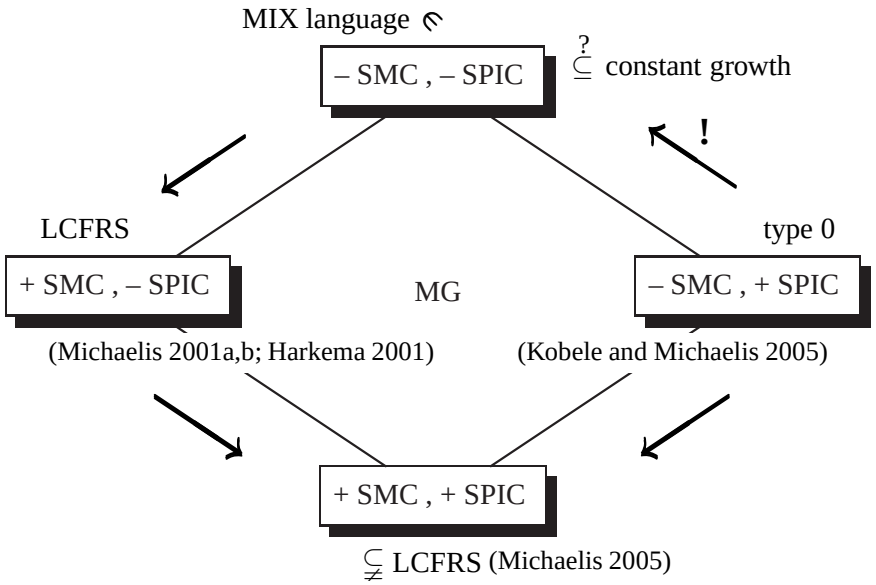


Figure 12: MG-diamond — Shortest move (SMC) and specifier islands (SPIC).

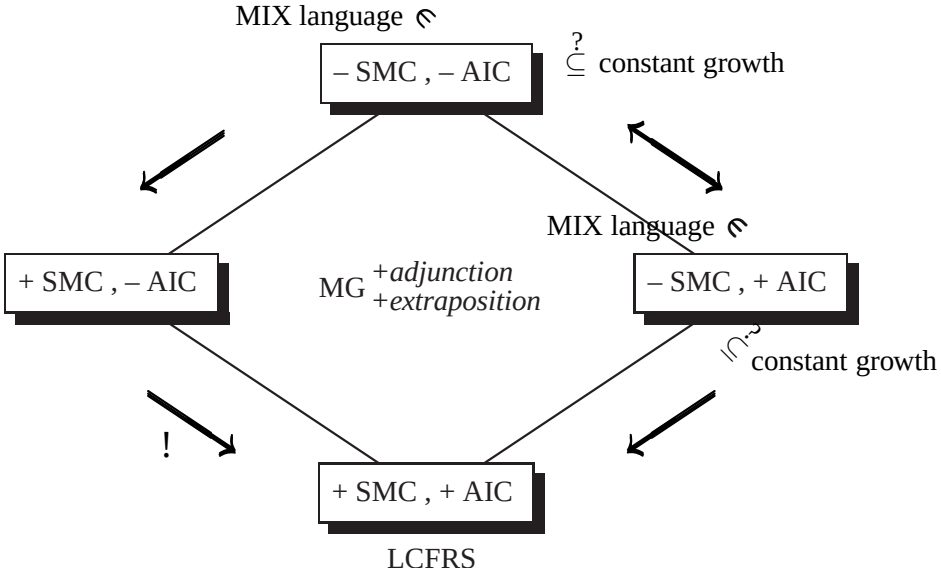


Figure 13: MG-diamond — Shortest Move (SMC) and Adjunct Islands (AIC).