Some Remarks on Locality Conditions and Minimalist Grammars*

Hans-Martin Gärtner and Jens Michaelis

Introduction

In this paper we undertake a study of syntactic *locality conditions (LCs)* within Stablerian *minimalist grammars (MGs)* (Stabler 1997, 1998, 1999 and elsewhere). We show that the "restrictiveness" of LCs measured in terms of weak generative capacity depends on how they are combined. Thus, standard MGs incorporating just the *shortest move condition (SMC)* are mildly context-sensitive. Adding the *specifier island condition (SPIC)* to such grammars either reduces complexity or, interestingly, it increases complexity. This depends on the co-presence or absence of the SMC, respectively. Likewise, the effect of adding the *adjunct island condition (AIC)* to an extended MG is either trivial (without co-presence of the SMC) or, apparently, crucial in preserving mild context-sensitivity. The point of this exercise is to demonstrate that LCs as such - intuitions to the contrary notwithstanding - are not automatically restrictive where a formal notion of restrictiveness is applied. Independent motivation for our work comes from a recent convergence of two research trends. On the one hand, appeal has been made to the formal complexity of natural languages in work on language evolution (Hauser et al. 2002, Piattelli-Palmarini and Uriagereka 2004) and to *computational efficiency* in mainstream minimalism (Chomsky 2005). On the other hand, the formally well-understood Stablerian MGs provide enough descriptive flexibility to be taken seriously as a syntactic theory by the working linguist. A more comprehensive study of the complexity of constraint interaction is still outstanding.

*For comments, criticisms and suggestions we are greatful to audiences at FG 2003 Vienna, ZAS Berlin, Symposium "Recursion + Interfaces = Language?," LACL 2005 Bordeaux, FGMoL 2005 Edinburgh, INRIA/LaBRI Bordeaux, SfS Tübingen. Special thanks go to an anonymous reviewer. Finally we would like to thank Tom Cornell, Ed Keenan, Greg Kobele, Hap Kolb, Marcus Kracht, Uwe Mönnich, Christian Retoré, Jim Rogers, Ed Stabler, Peter Staudacher, Wolfgang Sternefeld, Craig Thiersch for many hours of stimulating discussion. The usual disclaimers apply.

1 Locality Conditions

Generative grammar took one of its more important turns when *locality conditions (LCs)* were established in work by Ross (1967) and Chomsky (1973, 1977). As is well-known, this led to a period of intense research into the proper formulation of LCs, as documented in work by Huang (1982), Chomsky (1986), Rizzi (1990), Cinque (1990), Manzini (1992), Müller and Sternefeld (1993), and Szabolcsi and Zwarts (1997), among others.

Formally LCs can be separated into two types, *intervention-based LCs (ILCs)* and *containment-based LCs (CLCs)*. ILCs are often characterized in terms of minimality constraints, such as the minimal link, minimal chain, shortest move, or attract closest condition. In the framework of *minimalist grammars (MGs)* (Stabler 1997, 1999), which we are adopting in this paper, intervention-based locality is captured by the shortest move condition (SMC). CLCs are often characterized in terms of (generalized) grammatical functions. Familiar conditions define adjunct islands, subject islands, and specifier islands. MGs have integrated versions of a *specifier island condition (SPIC)* (Stabler 1999) and an *adjunct island condition (AIC)* (Frey and Gärtner 2002; Gärtner and Michaelis 2003). In (1) we schematically illustrate the structure of these LC-types.

(1) a. $[\ \ldots\ \alpha\ \ldots\ [\ \ldots\ \beta\ \ldots\ \gamma\ \ldots\ ]\ ]$

b. $[\ \ldots\ \alpha\ \ldots\ [_{\beta}\ \ \ldots\ \ \gamma\ \ \ldots\ \ ]\ ]$

An ILC, (1a), prevents establishing dependencies between constituents α and γ across an intervening β. Intervention is typically defined in terms of c-command or similar notions. A CLC, (1b), on the other hand, prevents establishing dependencies between constituents α and γ into or out of a containing β. Containment is usually defined in terms of dominance.

It is also well-known that LCs have been central in the quest for achieving the "Goals of Generative Linguistic Theory." Thus, consider the following statement by Chomsky (1973, p. 232):

> From the point of view that I adopt here, the fundamental empirical problem of linguistics is to explain how a person can acquire knowledge of language. [...] To approach the fundamental empirical problem, we attempt to restrict the class of potential human languages by setting various conditions on the form and function of grammars.

Quite uncontroversially, LCs have been taken to serve as restrictions in this sense. However, the important underlying notion of restrictiveness is much less easy to pin down in a principled manner. In particular, it is difficult to answer the following two questions in any satisfactory way.

Q1: How do we know that we have restricted the class of potential human languages?

Q2: Could we measure the degree of restriction, and if so, how?

Researchers are fundamentally divided over how to deal with these questions. Currently, at least two major approaches coexist. The first one, which we will call "formalist," is rooted in formal complexity theory as discussed in Chomsky (1956, 1959). The alternative, which we call "cognitivist," is built on the prospects of establishing a theory of "relevant cognitive complexity." For this distinction we rely on Berwick and Weinberg (1982, p. 187), who emphasized that "[t]here is a distinction to be drawn between *relevant cognitive complexity* and the *mathematical complexity* of a language."

Interestingly, Chomsky (1977) may be understood as having sided with the cognitivists, interpreting locality conditions as part of such a theory, as the following quote indicates.[1]

> Each of these conditions [subjacency, SSC, PIC] may be thought of as a limitation on the scope of the processes of mental computation. (Chomsky 1977, p. 111)

Now, a standard criticism raised by formalists against cognitivists concerns the inability of the latter of answering questions Q1 and Q2. In particular, cognitivist notions of restrictiveness have been found inadequate for defining classes of languages. Formalism, on the other hand, is typically criticized especially for employing the measure of weak generative capacity, which, it is felt, requires abstractions too far removed from the grammars found useful by linguists.

However, recent developments, taking their outset from "The Minimalist Program" (Chomsky 1995) have created a situation where formalism and cognitivism have begun to converge on common interests again.

In particular, work on language evolution by, i.a., Hauser et al. (2002) and Piattelli-Palmarini and Uriagereka (2004) has raised the interest of cognitivists in formalist concerns.[2]

[1] *SSC* stands for the *specified subject condition*, and *PIC* stands for the *propositional island condition*.

[2] According to Kolb (1997, p. 3) the same trend toward formalism characterizes Chomsky's minimalist revision of *principles and parameters (PP) theory*: "PP theory often gives the im-

At the same time, work on minimalist grammars (MGs), as defined by Stabler (1997), has led to a realignment of "grammars found 'useful' by linguists" and formal complexity theory. MGs are capable of integrating (if needed) mechanisms such as: head movement (Stabler 1997, 2001), (strict) remnant movement (Stabler 1997, 1999), affix hopping (Stabler 2001), adjunction and scrambling (Frey and Gärtner 2002), and late adjunction and extraposition (Gärtner and Michaelis 2003).

In addition to this descriptive flexibility, Michaelis (1998 [2001a]) has shown MGs to provide a *mildly context-sensitive grammar (MCSG)* formalism in the sense of Joshi (1985).[3] This class of formalisms, which is shown in Fig. 15 (Appendix C), has repeatedly been argued to be of exactly the right kind when it comes to characterizing the complexity of human languages. MCSGs combine conditions on weak generative capacity with the condition of polynomial time parsability[4] and the so-called *constant growth* property. Constant growth informally means that "if the strings of a language are arranged in increasing order of length, then two consecutive lengths do not differ in arbitrarily large amounts" (Joshi et al. 1991, p. 32).

Given the two properties just outlined, MGs are an ideal tool for studying the complexity and/or restrictiveness of LCs. Such a study is what the remainder of this paper is devoted to. Concretely we are going to look at the behavior and interaction of the SMC, the SPIC and the 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. Now, importantly, adding the SPIC to standard MGs has non-monotonic consequences in the sense that whether complexity goes up or down depends on the absence or (co-)presence of the SMC, respectively. Thus, if we interpret (and measure) growing restrictiveness in terms of complexity reduction, we must conclude that adding a constraint like the SPIC as such does not - intuitions to the contrary notwithstanding - lead to more restrictive grammars automatically.

pression of a mere collection of 'interesting' facts which is largely data driven and where every new phenomenon may lead to new (ad hoc) formal devices, often incompatible, and without a measure to compare and/or decide between conflicting analyses meaningfully—in short: As a formal system it looks largely incoherent. [...] In what amounts to just about a U-turn, [in] its latest version, chapter 4 of Chomsky (1995) [...] [c]omplexity considerations are reintroduced into theory formation, and the non-recursiveness assumption is (implicitly) retracted." The trend has gained full momentum in Chomsky's more recent writings, where *computational efficiency* is counted among the crucial (sub-)factors of *language design* (Chomsky 2005, p. 6).

[3]See also Michaelis (2001b, 2005) and references cited therein for further details.

[4]This is the dimension that underlies the formal study of island conditions in Berwick (1992). For psycholinguistic studies, see Pritchett (1992) and Gibson (1991).

For the AIC, the picture is slightly more complicated. First of all, the AIC only makes sense if (base-)adjunction and adjunction by scrambling/extraposition is added to MGs. 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 that boundary. 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. Again we can conclude that the restrictiveness of a constraint is not inherently given but depends on the structure it interacts with.

Before we can present these results, we give a brief introduction to standard MGs and the relevant extensions. This will be done in Section 2. Section 3 contains our main results. Section 3.1 illustrates how an MG including the SPIC but without the SMC goes beyond MCSGs. Section 3.2 shows a case where an MG with the SMC but without the AIC appears to lose its status as MCSG. Section 4 is devoted to conclusions and a further outlook. Appendix A provides formal definitions and Appendix B sketches our approach to multiple wh-movement. We show there how to remove a prima facie conflict between this phenomenon and the SMC.

2 Minimalist Grammars

The objects specified by a *minimalist grammar (MG)* are so-called *minimalist expressions* or *minimalist trees*, which straightforwardly translate into the usual aboreal picture from syntactic theory as depicted in Fig. 1.[5]

A *simple expression* is given as a list of feature instances (technically: a single-noded tree labeled by that list) to be checked from left to right, where the intervening marker # is used to separate the checked part of feature instances from the non-checked one. A minimalist tree is said to *have*, or likewise, *display feature* f if its head-label is of the form $\alpha\#f\alpha'$.

Starting from a finite set of simple expressions (the *lexicon*), minimalist expressions can be built up recursively from others by applying structure building functions. The applications of these functions are triggered by particular instances of syntactic features appearing in the trees to which the functions are applied. After having applied a structure building function, the triggering

[5] Stabler's minimalist expressions are closely related to but not to be confused with Chomskyan *linguistic* expressions, the latter defined as pairs, $\langle\pi,\lambda\rangle$, of PF- and LF-representations (Chomsky 1995, p. 170).

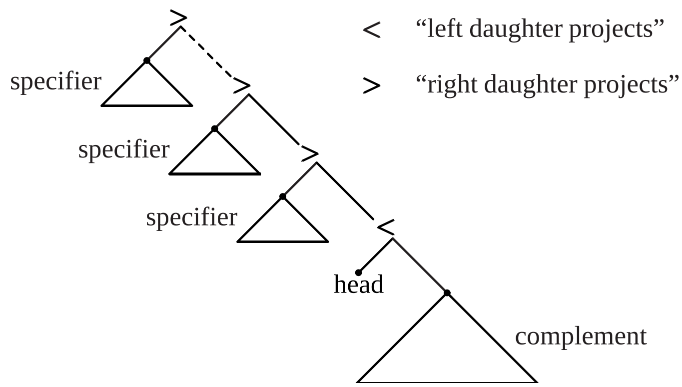


Figure 1. A typical minimalist expression

feature instances get marked as checked. Different structure building operations are triggered by different types of syntactic features. The standard ones are given by the following list:

(basic) categories: `x`, `y`, `z`, ...
m(erge)-selectors: `=x`, `=y`, `=z`, ...

m(ove)-licensees: `-x`, `-y`, `-z`, ...
m(ove)-licensors: `+x`, `+y`, `+z`, ...

Instances of (basic) category features and m-selectors trigger the *merge*-operator mapping a pair of trees to a single tree if the selecting tree ϕ displays m-selector `=x` and the selected tree χ displays the corresponding category `x`. χ is selected as a complement in case ϕ is simple, and as a specifier, otherwise. In both cases, the triggering feature instances get marked as checked in the resulting tree (see Fig. 2).

Instances of m-licensors and m-licensees trigger applications of the *move*-operator by which—without imposing the *shortest move condition (SMC)*—a single tree displaying m-licensor `+x` is mapped to a finite set of trees, consisting of every tree which results from moving a maximal projection displaying the corresponding m-licensee `-x` into a specifier position. Again the feature instances triggering the application of the operator get marked as checked in the resulting tree (see Fig. 3).

$\mathit{merge} : \mathit{Trees} \times \mathit{Trees} \longrightarrow \mathit{Trees}$

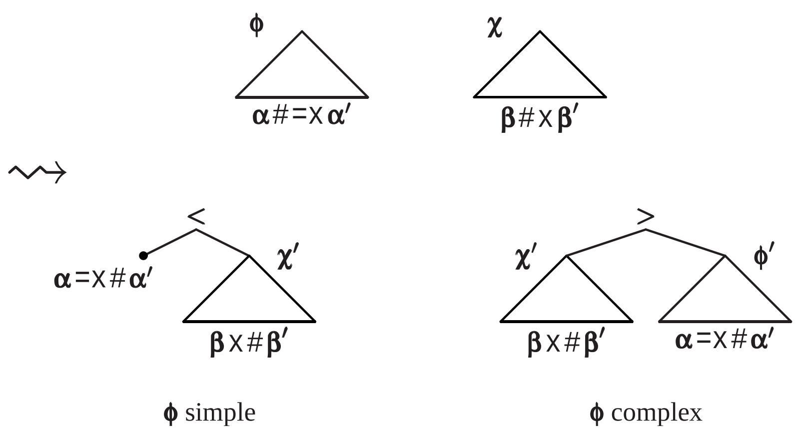


Figure 2. The merge-operator.

$\mathit{move} : \mathit{Trees} \longrightarrow 2^{\mathit{Trees}}$

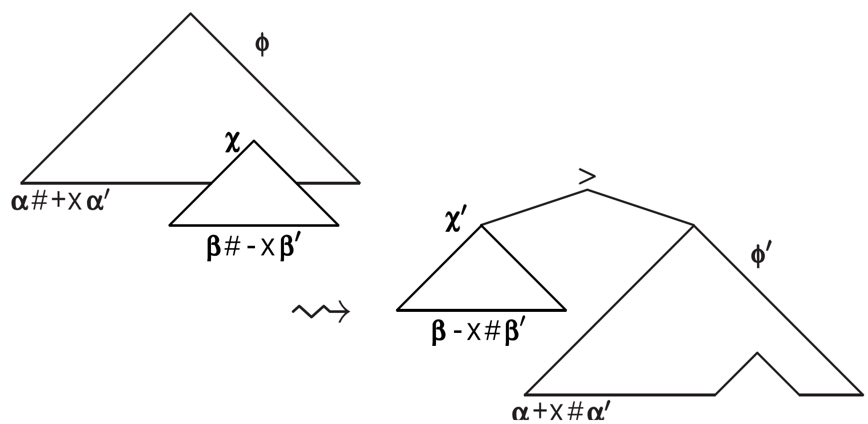


Figure 3. The move-operator.

The *tree language of an MG* is the set of trees of category c (the root category "complete" or "complementizer") each of which with essentially no unchecked syntactic features left after having been derived from a finite number of (possibly multiple) instances of lexical items by successively applying structure building operators. The *string language of an MG* is the set of strings each of which resulting from concatenating "left-to-right" the terminal leaf-labels of some tree belonging to the tree language.

Standard MGs usually come with a specific implementation of the *shortest move condition (SMC)*: for each MG there is an absolute (finite) upper bound n on the number of competing, i.e. simultaneously displayed, licensee features triggering an application of the move-operator. In the most radical version we have $n = 1$. As shown in Fig. 4, in the standard case this excludes both crossing and nesting dependencies involving multiple licensees of one and the same type.[6] Note also that, in this sense, absence of the SMC (– *SMC*, for short) means that no absolute upper bound on simultaneously displayed licensee features exists.

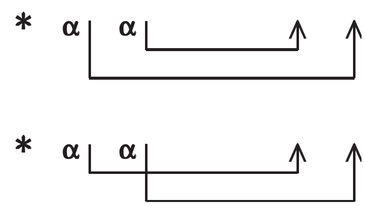


Figure 4. The shortest movement condition (SMC) (Stabler 1997, 1999)

The MG-variant proposed by Stabler (1999) also includes an implementation of the *specifier island condition (SPIC)* which essentially demands that proper extraction from specifiers is blocked (see Fig. 5).

Structurally similar to the SPIC, MGs can be endowed with an implementation of the *adjunct island condition (AIC)* demanding that, if at all, only full adjuncts but none of their proper subparts can extract (see Fig. 6).

Talk of adjuncts and the AIC presupposes extending MGs with additional syntactic features and structure building functions. To the list of syntactic

[6] See Section 3.2 for an exploitation of the dynamic character of the SMC. See Michaelis (2001b) and Stabler (1999) for the MG-treatment of cross-serial dependencies. See Appendix B for our approach to multiple wh-dependencies.

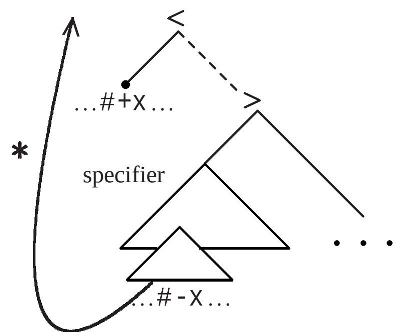


Figure 5. The specifier island condition (SPIC) (Stabler 1999).

features we add:

a(djoin)-selectors: $\approx$x , $\approx$y , $\approx$z , ...

s(cramble)-licensees: $\sim$x , $\sim$y , $\sim$z , ...

Then we introduce an *adjoin*-operator and *extraposition/scramble*-operator, which in contrast to the merge- and move-operator do not function as a bilateral checking mechanism but a unilateral one. This implements type-preservingness and iterability of adjunction, as is familiar from categorial grammar.

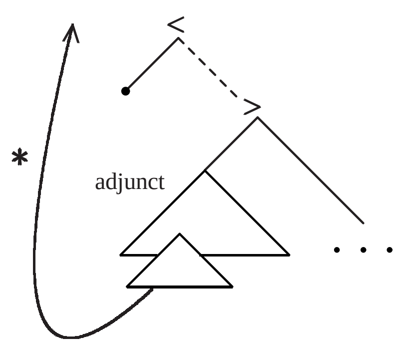
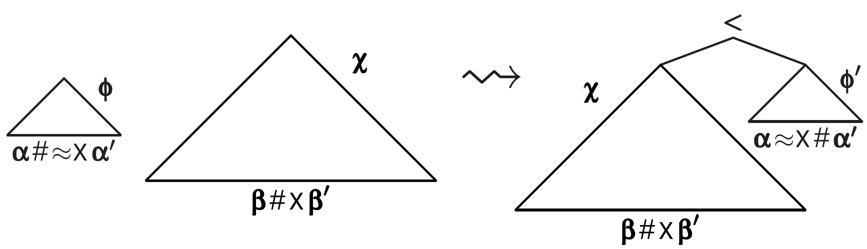


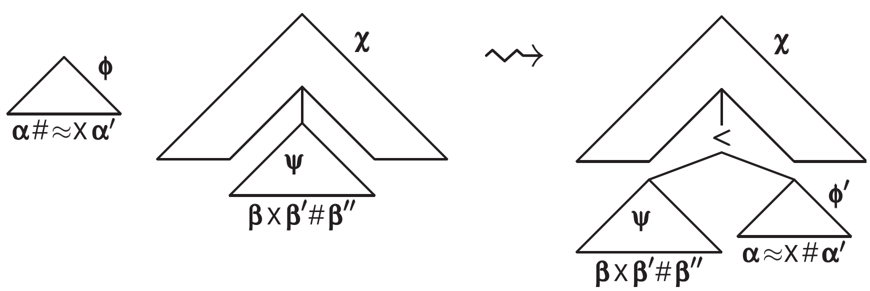
Figure 6. The adjunct island condition (AIC) (Frey and Gärtner 2002).

Instances of (basic) category features and a-selectors trigger the adjoin-operator mapping a pair of trees, $\langle\phi,\chi\rangle$, to a finite set of trees, consisting of every tree which results from adjoining the tree ϕ if it displays the a-selector $\approx\mathtt{x}$ to the tree χ: cyclically in case χ displays the corresponding category $\mathtt{x}$, or acyclically to a maximal projection ψ properly contained in χ in case the head-label of ψ contains a checked instance of the category $\mathtt{x}$. In both cases, the a-selector feature instance triggering the application of the operator gets marked as checked in the resulting tree, while the other head-label of χ, respectively ψ, remains unchanged (cf. Fig. 7).

$\boxed{\textit{adjoin} : \textit{Trees} \times \textit{Trees} \longrightarrow 2^{\textit{Trees}}}$



cyclic adjunction (Frey and Gärtner 2002)



acyclic/late adjunction (Gärtner and Michaelis 2003)

Figure 7. The operator *adjoin*.

Instances of (basic) categories and s-licensees trigger applications of the scramble-operator which—without imposing the SMC—maps a single tree displaying category x to a finite set of trees, consisting of every tree which results from extraposing/scrambling a maximal projection displaying the corresponding s-licensee ∼x into an adjoined position. Again, only the s-licensee feature instance triggering the application of the operator gets marked as checked in the resulting tree, while the corresponding head-label displaying category x remains unchanged (cf. Fig. 8).

$\boxed{\textit{scramble} : \textit{Trees} \longrightarrow 2^{\textit{Trees}}}$

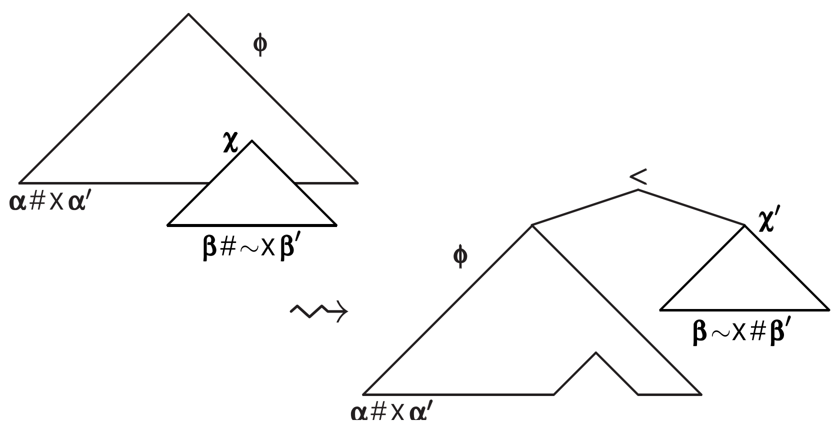


Figure 8. The operator *scramble*.

3 Locality Conditions and Complexity Results

As indicated in Section 1, our complexity results concern the interaction of locality constraints. In Section 3.1 we look at the interaction of the SPIC and the SMC within standard MGs. In Section 3.2 we introduce MGs with late adjunction and discuss the interaction of the AIC and the SMC within such extended grammars. The *MG-diamonds* in Fig. 9 provide a systematic picture for our study. The ultimate task is to establish complexity results for each corner and to reflect on their relation.

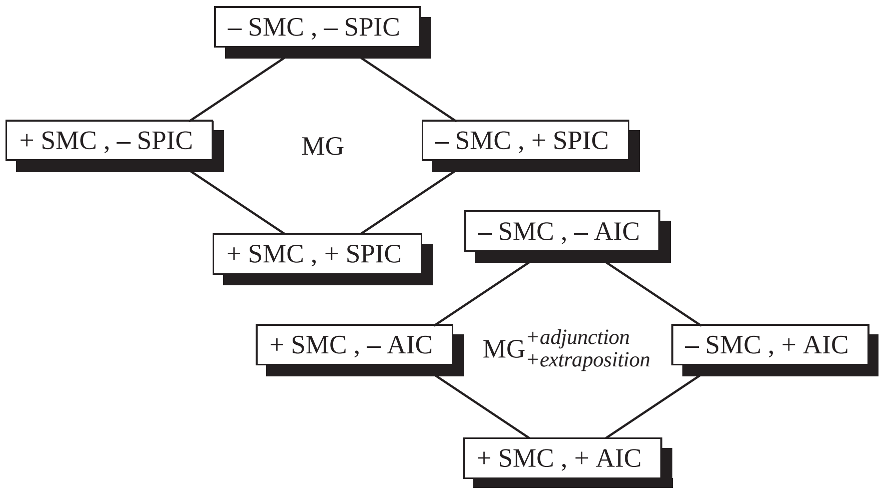
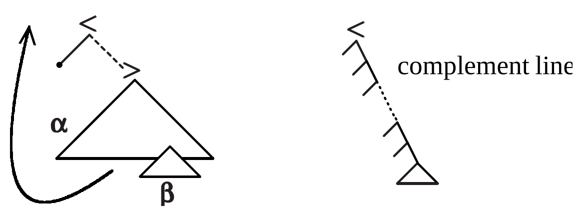


Figure 9. MG-diamonds — Towards complexity results concerning LCs

3.1 The Specifier Island Condition

Fig. 10 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, $\{a^{2^n} \mid n \geq 0\} = \{a, aa, aaaa, aaaaaaaa, \ldots\}$. 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.

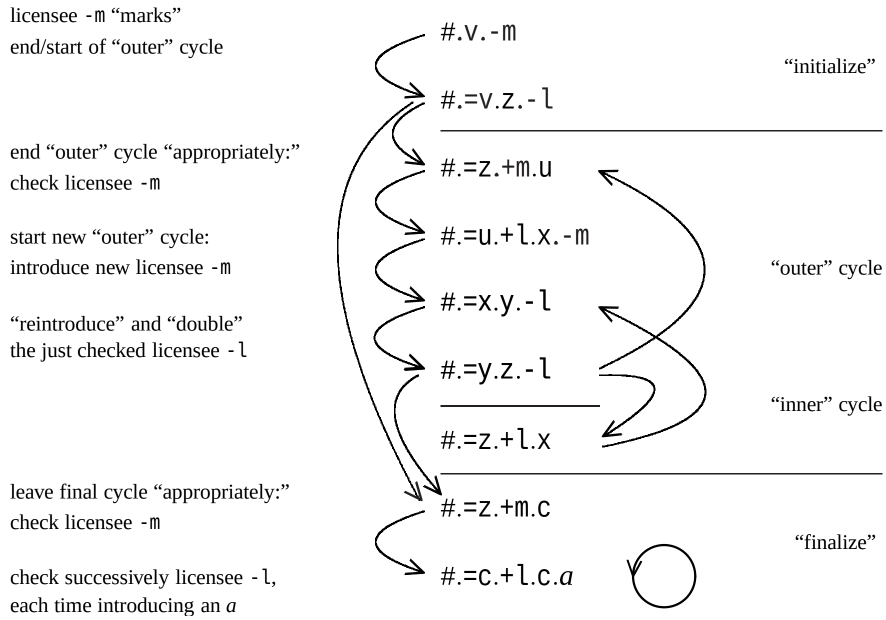


Figure 10. MG-example — Complexity results concerning LCs

Starting the "outer" cycle of our example in Fig. 10, 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 m-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.[7] Fig. 11 shows the result of a cycleless derivation creating string *a*, and a one-cycle derivation creating string *aa*.

[7] For further details see Gärtner and Michaelis (2005).

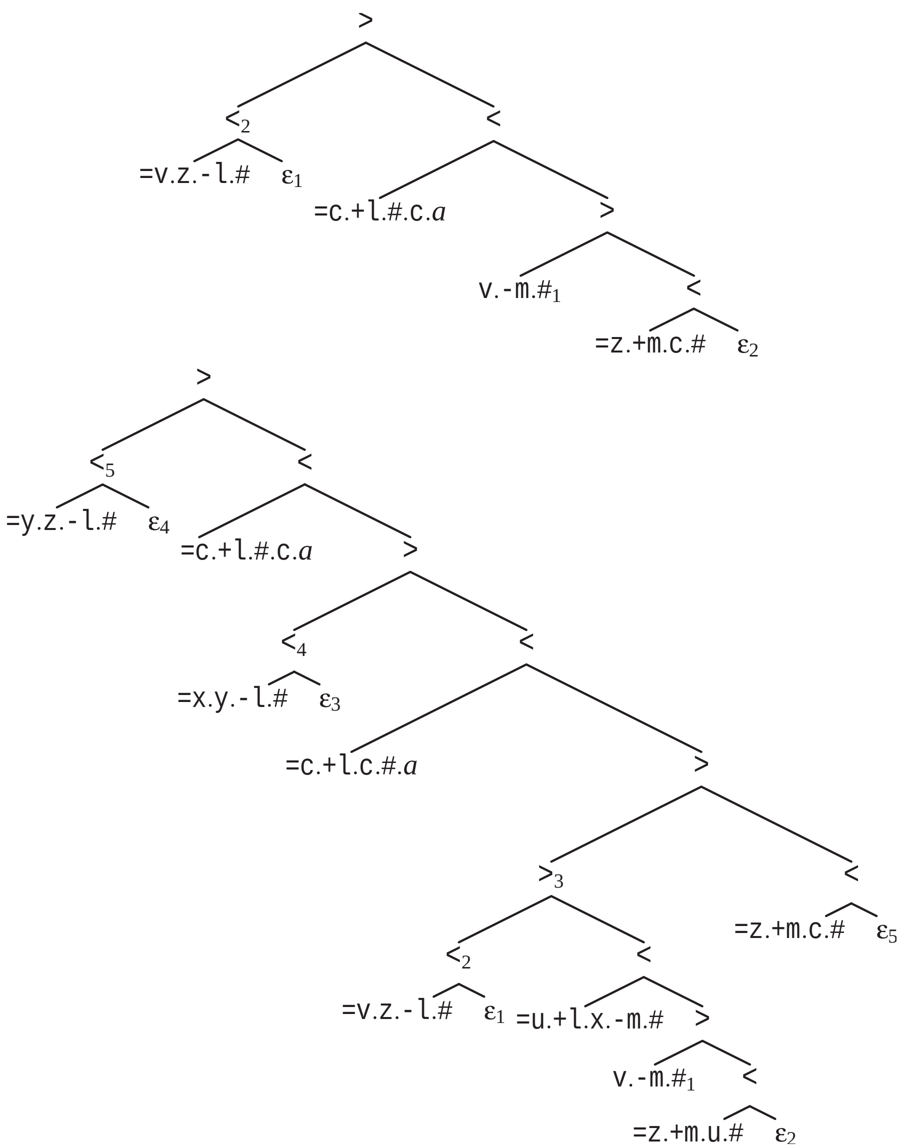


Figure 11. MG-example — Complexity results concerning LCs
(Numerical indices indicate antecedent-trace relations)

Fig. 12 summarizes what we know about the interaction of SMC and SPIC,[8] 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 complexity (upper right side). What the SPIC does depends on the presence or absence of SMC. Its behavior is thus non-monotonic.

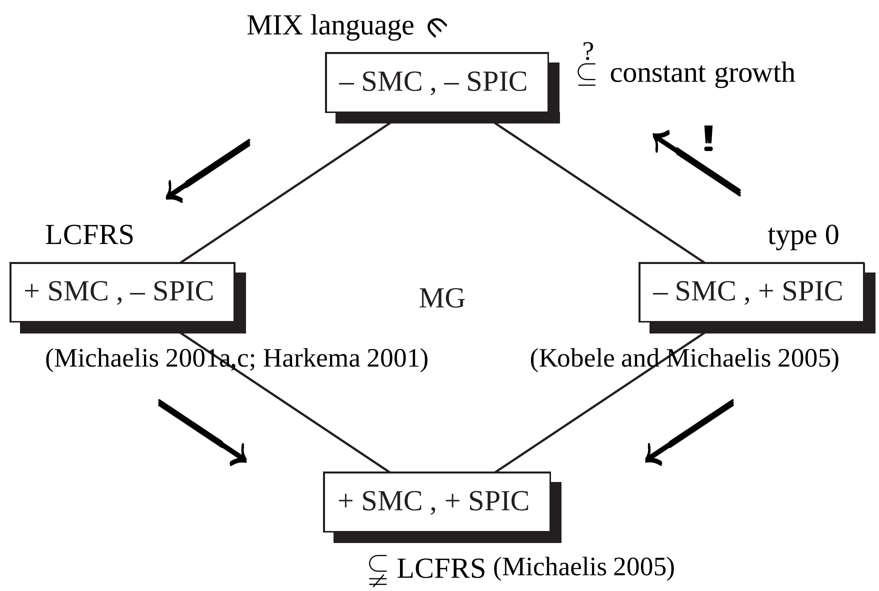


Figure 12. MG-diamond — Shortest move (SMC) and specifier islands (SPIC)

3.2 The Adjunct Island Condition

In this section we look at MGs with *(late) adjunction* and *scrambling/extraposition* and study the effects of imposing the AIC in a situation where the SMC alone appears to be too weak to guarantee mild context-sensitivity. As

[8] In Fig. 12 *LCFRS* stands for *Linear Context-Free Rewriting System*. For a more comprehensive picture of how these systems fit into the MCSG landscape see Appendix C. 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.

in Section 3.1, the task is to fill in complexity relations between the corners of our MG-diamond, shown with relevant changes made in Fig. 13.

Late or countercyclic adjunction has already been introduced in Section 2 (cf. Fig. 7). One of its main linguistic motivations, going back (at least) to Lebeaux (1991), has to do with the possibility of avoiding standardly predicted but unattested violations of binding principle C. This is done by adjoining a constituent containing an R-expression after the constituent adjoined to has moved out of the c-command domain of a potentially offensive binder for that R-expression. (2) gives an example with a modifying relative clause.

(2) $[_{DP}\ [_{DP}$ which book $]_j\ [_{CP}$ that Mary$_i$ read $]\ ]$ did she$_i$ like t_j

For the complexity issue we are interested in here it is important to note that, as already briefly indicated by Gärtner and Michaelis (2003), late adjunction is capable of circumventing the SMC. (3) presents a case where this is actually welcome.

(3) [[[[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 relative clauses in (3) are extraposed by an application of rightward scrambling and are adjoined to CP. This is very roughly sketched in (4).

(4) * $[_{CP}$ CP_2 CP_1] α α

This violates the SMC (see above) if α is instantiated as $\sim$c. However, as sketched in (5), a derivational sequence of (first) extraposition, late adjunction and (second) extraposition voids this problem.

(5)	$[_{CP}$		CP_1^{α}	]			start here
	$[_{CP}$		--	]	$CP_1^{\cancel{\alpha}}$		move CP_1, check α
	$[_{CP}$	CP_2^{α}	--	]	$CP_1^{\cancel{\alpha}}$		late adjoin CP_2
	$[_{CP}$	--	--	]	$CP_1^{\cancel{\alpha}}$	$CP_2^{\cancel{\alpha}}$	move CP_2, check α

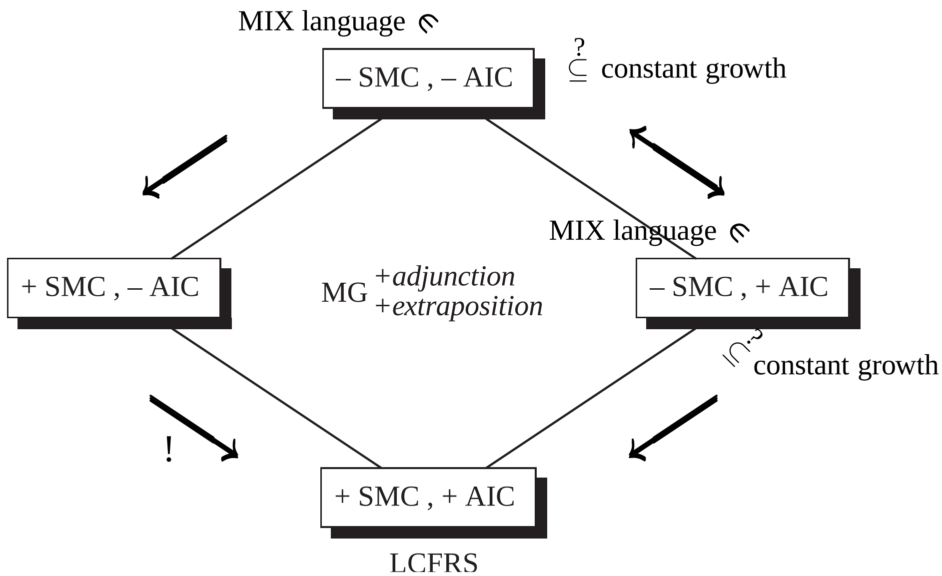


Figure 13. MG-diamond — Shortest Move (SMC) and Adjunct Islands (AIC)

The proof that MGs without late adjunction are mildly context-sensitive rests on the technical possibility of removing checked features from the structures.[9] 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

[9] See Stabler and Keenan (2003) for a reduced MG-format that cashes this out representationally. Chomsky (2005, p. 11) characterizes his "… 'no-tampering' condition of efficient computation" in almost the same way. Speaking of "operations forming complex expressions" Chomsky notes that it "sharply reduces computational load" if "what has once been constructed can be 'forgotten' in later computations." Without noting the tension created, Chomsky (2005, p. 12) introduces the "internal Merge" implementation of movement. This operation in fact requires an a priori not bounded amount of structure to remain available for copying and displacement. This undoes the effect of whatever structure may be 'forgotten' otherwise. Introducing the notion of "edge of a phase" (Chomsky 2001) as container for "still active" constituents does not essentially improve the situation, as long as there is no upper bound on the material inside such an edge. As far as we can see, this also negatively affects attempts by Chesi (2004) at providing any "measurable" complexity reductions in terms of phase-based locality. The point made by Berwick (1992) is closely related. Thus, "computational intractability" results if syntactic traces or "variables" are allowed to preserve an arbitrary amount of information (full copying being the extreme case).

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 m- or s-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. Fig. 13 summarizes our results for SMC/AIC-interaction. Again, addition of an LC does not automatically restrict the grammar, as the upper right side shows. We conjecture that the AIC is a formal restriction only where it complements the SMC.

4 Conclusion and Further Outlook

Let us take a step back and summarize what we have found out about LCs within Stablerian MGs. Taking restrictiveness to be defined as weak generative capacity, we have illustrated how imposition of an LC can have either:

(A) restrictive effects, or

(B) no restrictive effects, or

(C) anti-restrictive effects.

Thus, adding the SPIC to standard MGs raises them to type 0 grammars if the SMC is absent, while together with the SMC it induces a genuine restriction (Section 3.1). Adding the AIC to an MG extended with the operations of late adjunction and extraposition (via rightward scrambling) is without effects unless the SMC is co-present. In the latter case, the AIC guarantees mild context-sensitivity, which the extended MGs without it are likely to go beyond (Section 3.2). We think that these non-monotonic effects of LCs should be of interest to everyone caring about formal (and measurable) notions of restrictiveness. In a nutshell, the message is that “constraints do not always constrain.” Our result for MGs without SMC, but obeying the SPIC can be seen in the light of what Rogers (1998, p. 3f) concludes about a famous similar case:

> The significance of the [Peters & Ritchie-]results is [...] that, by itself, the hypothesis that natural languages are characterized by *Aspects*-style TGs [...] has no non-trivial consequences with respect to the class of natural languages.

Equally, by itself, the hypothesis that natural languages are characterized by the said MGs has no non-trivial consequences with respect to the class of natural languages.

As pointed out in Section 1, all of these issues have regained relevance due to the recent emergence of "cognitivist" studies of language evolution (Hauser et al. 2002, Piattelli-Palmarini and Uriagereka 2004) that reintroduce notions of classical formal complexity theory. Likewise, Chomskyan minimalism conceives of *computational efficiency* as contributing to the *design factors of language* (Chomsky 2005). This comes at a time where more and more grammar types have begun to converge on the mildly context-sensitive format (Joshi et al. 1991), Stablerian MGs among them.

There are some obvious ways to pursue the work begun here further. First of all, we have not looked at the interaction of SPIC and AIC. This is particularly relevant for MGs with late adjunction and extraposition for the following reasons. First, it is unclear whether the SPIC should constrain extraposition as much as it would in our current formalization. Secondly, the dynamics of late adjunction call for greater care to be taken in distinguishing static from dynamic formulations of LCs, i.e. LCs that put absolute bans on output structures vs. LCs that constrain individual derivational steps. On a more speculative note, it also remains to be seen whether a different division of labor between competence and performance aspects of grammars, as envisioned by Sternefeld (1998), could reorganize the (complexity) landscape of grammar formalisms in a fruitful fashion.

Appendix A

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:[10]

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

Base		a set of *(basic) categories*
M-Select	$= \{ \texttt{=x} \mid \texttt{x} \in \mathit{Base} \}$	a set of *m(erge)-selectors*
A-Select	$= \{ \approx\texttt{x} \mid \texttt{x} \in \mathit{Base} \}$	a set of *a(djoin)-selectors*
M-Licensors	$= \{ \texttt{+x} \mid \texttt{x} \in \mathit{Base} \}$	a set of *m(ove)-licensors*
M-Licensees	$= \{ \texttt{-x} \mid \texttt{x} \in \mathit{Base} \}$	a set of *m(ove)-licensees*
S-Licensees	$= \{ \sim\texttt{x} \mid \texttt{x} \in \mathit{Base} \}$	a set of *s(cramble)-licensees*

(F3) *Base* includes at least the category $\texttt{c}$.

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

Definition 4.1 An *expression (over Feat)*, also referred to as a *minimalist tree (over Feat)*, is a 6-tuple $\langle N_\tau, \triangleleft^*_\tau, \prec_\tau, <_\tau, \mathit{label}_\tau \rangle$ obeying (E1)–(E3).

(E1) $\langle N_\tau, \triangleleft^*_\tau, \prec_\tau \rangle$ is a finite, binary (ordered) tree defined in the usual sense: N_τ is the finite, non-empty set of *nodes*, and $\triangleleft^*_\tau$ and $\prec_\tau$ are the respective binary relations of *dominance* and *precedence* on N_τ.[11]

(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 $\langle N_\tau, \triangleleft^*_\tau, \prec_\tau \rangle$ either $x <_\tau \mathit{sibling}_\tau(x)$ or $\mathit{sibling}_\tau(x) <_\tau x$ holds.[12]

(E3) label_τ is the *leaf-labeling function* from the set of leaves of $\langle N_\tau, \triangleleft^*_\tau, \prec_\tau \rangle$ into $\mathit{Syn}^*\{\#\}\mathit{Syn}^*\mathit{Phon}^*\mathit{Sem}^*$.[13]

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

Let $\tau = \langle N_\tau, \triangleleft^*_\tau, \prec_\tau, <_\tau, \mathit{label}_\tau \rangle \in \mathit{Exp}(\mathit{Feat})$.[14]

---

[11] Thus, $\triangleleft^*_\tau$ is the reflexive-transitive closure of $\triangleleft_\tau \subseteq N_\tau \times N_\tau$, the relation of *immediate dominance* on N_τ.

[12] $\mathit{sibling}_\tau(x)$ denotes the (unique) sibling of any given $x \in N_\tau$ different from the root of $\langle N_\tau, \triangleleft^*_\tau, \prec_\tau \rangle$. If $x <_\tau y$ for some $x, y \in N_\tau$ then x is said to *(immediately) project over* y.

[13] 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*.

[14] Note that the leaf-labeling function label_τ can easily be extended to a total labeling function ℓ_τ from N_τ into $\mathit{Feat}^*\{\#\}\mathit{Feat}^* \cup \{<,>\}$, 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 $x \triangleleft_\tau y, z$, and $y \prec_\tau z$. In this sense a concrete $\tau \in \mathit{Exp}(\mathit{Feat})$ is depictable in the way indicated in Fig. 1.

For each $x \in N_\tau$, the *head of x (in* τ*)*, denoted by $head_\tau(x)$, is the (unique) leaf of τ with $x \triangleleft_\tau^* head_\tau(x)$ 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*).

An expression $\phi = \langle N_\phi, \triangleleft_\phi^*, \prec_\phi, <_\phi, label_\phi \rangle \in Exp(Feat)$ is a *subexpression of* τ in case $\langle N_\phi, \triangleleft_\phi^*, \prec_\phi \rangle$ is a subtree of $\langle N_\tau, \triangleleft_\tau^*, \prec_\tau \rangle$, $<_\phi = <_\tau \restriction_{N_\phi \times N_\phi}$, 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 $\{\phi \,|\, \tau \, comp_\tau^+ \, \phi\}$.

$spec_\tau \subseteq MaxProj(\tau) \times MaxProj(\tau)$ 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 $r_\phi \triangleleft_\tau^+ x \triangleleft_\tau^+ head_\tau(r_\phi)$, 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 $\{\phi \,|\, \tau \, spec_\tau \, \phi\}$.

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 $\beta, \beta' \in Feat^*$.[15]

τ is *complete* if its head-label is in $Syn^*\{\#\}\{\mathtt{c}\}Phon^*Sem^*$, 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 $\langle N_\tau, \triangleleft_\tau^*, \prec_\tau \rangle$, and replacing all instances of non-phonetic features with the empty string, afterwards.

For any $\phi, \chi \in Exp(Feat)$, $[_<\phi, \chi]$ (respectively, $[_>\phi, \chi]$) denotes the complex expression $\psi = \langle N_\psi, \triangleleft_\psi^*, \prec_\psi, <_\psi, label_\psi \rangle \in Exp(Feat)$ for which ϕ and

[15] Thus, e.g., the expression depicted in (3) has feature +x, while there is a maximal projection which has feature -x.

χ are those two subexpressions such that $r_\psi \triangleleft_\psi r_\phi$, $r_\psi \triangleleft_\psi r_\chi$ 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 $\phi, \chi, \psi \in \mathit{Exp}(\mathit{Feat})$ such that χ is a subexpression of ϕ, $\phi\{\chi/\psi\}$ is the expression which results from substituting ψ for χ in ϕ.

In the following we write *MG* as a shorthand for *minimalist grammar*.

Definition 4.2 An *MG without both SMC and SPIC ($MG^{/-,-/}$)* is a 5-tuple of the form $\langle \neg \mathit{Syn}, \mathit{Syn}, \mathit{Lex}, \Omega, \mathtt{c} \rangle$ where Ω is the operator set consisting of the structure building functions $\mathit{merge}^{/-/}$ and $\mathit{move}^{/-,-/}$ 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*, and each item $\tau \in \mathit{Lex}$ is of the form $\langle \{r_\tau\}, \triangleleft_\tau^*, \prec_\tau, <_\tau, \mathit{label}_\tau \rangle$ such that $\mathit{label}_\tau(r_\tau)$ is an element in $\{\#\}(\mathit{M\text{-}Select} \cup \mathit{M\text{-}Licensors})^* \mathit{Base}\, \mathit{M\text{-}Licensees}^* \mathit{Phon}^* \mathit{Sem}^*$.

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$^{\text{-SPIC}}$) $\mathit{merge}^{/-/}$ is a partial mapping from $\mathit{Exp}(\mathit{Feat}) \times \mathit{Exp}(\mathit{Feat})$ into $\mathit{Exp}(\mathit{Feat})$. For any $\phi, \chi \in \mathit{Exp}(\mathit{Feat})$, $\langle \phi, \chi \rangle$ is in $\mathit{Dom}(\mathit{merge}^{/-/})$ if for some category $\mathtt{x} \in \mathit{Base}$ and $\alpha, \alpha', \beta, \beta' \in \mathit{Feat}^*$, conditions (me.i) and (me.ii) are fulfilled:[16]

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

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

Then,

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

(me.2) $\mathit{merge}^{/-/}(\phi, \chi) = [_> \chi', \phi']$ 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. Fig. 2).

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

(mo$^{\text{-SMC,-SPIC}}$) $move^{/-,-/}$ is a partial mapping from $Exp(Feat)$ into the class $\mathcal{P}_{\text{fin}}(Exp(Feat))$.[17] A $\phi \in Exp(Feat)$ is in $Dom(move^{/-,-/})$ if for some $\text{-x} \in$ *M-Licensees* and $\alpha, \alpha' \in Feat^*$, (mo.i) and (mo.ii) are true:

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

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

Then,

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

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. Fig. 3).

Definition 4.3 An *MG without SMC, but with SPIC (MG$^{/-,+/}$)* is a five-tuple of the form $\langle \neg Syn, Syn, Lex, \Omega, \text{c} \rangle$ 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* defined as in Definition 4.2.

(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 $\text{x} \in Base$ 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 syntactic feature instance).

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

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

(mo$^{\text{-SMC,+SPIC}}$) $move^{/-,+/}$ is a partial mapping from $Exp(Feat)$ into the class $\mathcal{P}_{\text{fin}}(Exp(Feat))$. A $\phi \in Exp(Feat)$ is in $Dom(move^{/-,+/})$ if for some $\text{-x} \in$ *M-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 explicity 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 4.4 An *MG with SMC, but without SPIC (MG$^{/+,-/}$)* is a five-tuple of the form $\langle \neg Syn, Syn, Lex, \Omega, \text{c} \rangle$ where Ω is the operator set consisting of the structure building functions $merge^{/-/}$ and $move^{/+,-/}$ defined as in (me$^{\text{-SPIC}}$) above and (mo$^{\text{+SMC,-SPIC}}$) below, respectively, and where *Lex* is a lexicon over *Feat* defined as in Definition 4.2.

(mo$^{\text{+SMC,-SPIC}}$) $move^{/+,-/}$ is a partial mapping from $Exp(Feat)$ into the class $\mathcal{P}_{\text{fin}}(Exp(Feat))$. A $\phi \in Exp(Feat)$ belongs to $Dom(move^{/+,-/})$ if for some $\text{-x} \in$ *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 $\gamma\#\text{-x}\gamma'$ for some $\gamma, \gamma' \in Feat^*$ (i.e. exactly one $\chi \in MaxProj(\phi)$ has -x).[18]

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

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

Definition 4.5 An *MG with both SMC and SPIC* ($MG^{/+,+/}$) is a five-tuple of the form $\langle \neg Syn, Syn, Lex, \Omega, \mathrm{c}\rangle$ 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 4.2.

(mo$^{+\text{SMC},+\text{SPIC}}$) $move^{/+,+/}$ is a partial mapping from $Exp(Feat)$ into the class $\mathcal{P}_{\text{fin}}(Exp(Feat))$. A $\phi \in Exp(Feat)$ is in $Dom(move^{/+,+/})$ if for some $-\mathrm{x} \in$ *M-Licensees* and $\alpha, \alpha' \in Feat^*$, (mo.i), (mo.ii), (mo.spic) and (mo.smc) above are true. Then, $move^{/+,+/}(\phi) = move^{/-,-/}(\phi)$.[19]

Let $G = \langle \neg Syn, Syn, Lex, \Omega, \mathrm{c}\rangle$ 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 $k \in \mathbb{N}$,[20] $CL^{k+1}(G) \subseteq Exp(Feat)$ is recursively defined as the set

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

The set $\{\tau \,|\, \tau \in CL(G)$ and τ complete$\}$, denoted by $T(G)$, is the *minimalist tree language derivable by G*. The set $\{Y_{Phon}(\tau) \,|\, \tau \in T(G)\}$, denoted by $L(G)$, 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 4.6 An $MG_{adj,ext}$ *without both SMC and AIC* ($MG^{/-,-/}_{adj,ext}$) is a 5-tuple $G = \langle \neg Syn, Syn, Lex, \Omega, \mathrm{c}\rangle$ 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)*, i.e., a finite set of simple expressions over *Feat*, and each lexical item $\tau \in Lex$ is of the form $\langle \{r_\tau\}, \triangleleft^*_\tau, \prec_\tau, <_\tau, label_\tau\rangle$ such that $label_\tau(r_\tau)$ is an element belonging to $\{\#\}(\textit{M-Select} \cup \textit{M-Licensors})^*(\textit{Base} \cup \textit{A-Select})\textit{Licensees}^*\textit{Phon}^*\textit{Sem}^*$.

[19] Note that the 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 (SMC). Thus, these functions can easily be identified with one from $Exp(Feat)$ to $Exp(Feat)$.

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

(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 $\text{x} \in Base$ and $\alpha, \alpha' \in Feat^*$, conditions (ad.i) and (ad.ii) are fulfilled:

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

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

Then,

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

where ϕ' results from ϕ by interchanging the instances of # and $\approx\text{x}$, the latter directly following the former in the head-label of ϕ (cf. Fig. 7).

(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 $\text{x} \in Base$ and $\alpha, \alpha' \in Feat^*$, (sc.i) and (sc.ii) are true:

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

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

Then,

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

where $\phi' \in Exp(Feat)$ is identical to ϕ except for the fact that the subtree χ is replaced by a single node labeled ε. $\chi' \in Exp(Feat)$ arises from χ by interchanging the instance of # and the instance of $\sim\text{x}$ immediately to its right within the head-label of χ (cf. Fig. 8).

Definition 4.7 An $MG_{adj,ext}$ *without SMC, but with AIC* ($MG^{/-,+/}_{adj,ext}$) is a five-tuple of the form $\langle \neg Syn, Syn, Lex, \Omega, \mathrm{c}\rangle$ where Ω is the operator set consisting of the structure building functions $merge^{/-/}$, $move^{/-,-/}$, $adjoin^{/+/}$ and $scramble^{/-,+/}$ defined as in (me$^{\text{-SPIC}}$) 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 4.6.

(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 $\mathrm{x} \in Base$ 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 $\mathrm{x} \in Base$ 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 itself a maximal projection displaying an unchecked syntactic feature instance).

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

Definition 4.8 An $MG_{adj,ext}$ *with SMC, but without AIC* ($MG^{/+,-/}_{adj,ext}$) is a five-tuple of the form $\langle \neg Syn, Syn, Lex, \Omega, \mathrm{c}\rangle$ 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 4.6.

($\mathrm{sc}^{\text{+SMC,-AIC}}$) The function $\mathit{scramble}^{/+,-/}$ maps partially from $\mathit{Exp}(\mathit{Feat})$ into the class $\mathcal{P}_{\mathrm{fin}}(\mathit{Exp}(\mathit{Feat}))$. 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) above and (sc.smc) are true:

(sc.smc) exactly one $\chi \in \mathit{MaxProj}(\phi)$ exists with head-label $\gamma\#{\sim}\mathrm{x}\gamma'$ for some $\gamma, \gamma' \in \mathit{Feat}^*$ (i.e. exactly one $\chi \in \mathit{MaxProj}(\phi)$ has ${\sim}\mathrm{x}$).[21]

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

Definition 4.9 An $MG_{adj,ext}$ *with both SMC and AIC* ($MG^{/+,+/}_{adj,ext}$) 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 ($\mathrm{me}^{\text{-SPIC}}$), ($\mathrm{mo}^{\text{+SMC,-SPIC}}$) and ($\mathrm{ad}^{\text{+AIC}}$) above and ($\mathrm{sc}^{\text{+SMC,+AIC}}$) below, respectively, and where *Lex* is a lexicon over *Feat* defined as in Definition 4.6.

($\mathrm{sc}^{\text{+SMC,+AIC}}$) $\mathit{scramble}^{/+,+/}$ is a partial mapping from $\mathit{Exp}(\mathit{Feat})$ into the class $\mathcal{P}_{\mathrm{fin}}(\mathit{Exp}(\mathit{Feat}))$. 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), (sc.ii), (sc.aic) and (sc.smc) above are true. Then, $\mathit{scramble}^{/+,+/}(\phi) = \mathit{scramble}^{/-,-/}(\phi)$.

Consider an $\mathrm{MG}^{/-,-/}_{\mathrm{adj,ext}}$, $\mathrm{MG}^{/-,+/}_{\mathrm{adj,ext}}$, $\mathrm{MG}^{/+,-/}_{\mathrm{adj,ext}}$, respectively $\mathrm{MG}^{/+,+/}_{\mathrm{adj,ext}}$, G, of the form $\langle \neg \mathit{Syn}, \mathit{Syn}, \mathit{Lex}, \Omega, \mathrm{c} \rangle$. 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. The *closure of* G, $\mathit{CL}(G)$, is the set $\bigcup_{k \in \mathbb{N}} \mathit{CL}^k(G)$, where $\mathit{CL}^0(G) = \mathit{Lex}$, and for $k \in \mathbb{N}$, $\mathit{CL}^{k+1}(G) \subseteq \mathit{Exp}(\mathit{Feat})$ is recursively defined as the set

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

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

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

Appendix B

One phenomenon that appears to challenge the SMC adopted here is multiple wh-fronting in Slavic languages. Take (6) from Bulgarian (Richards 2001, p. 249).

(6) *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, (6) requires three m-licensee instances of type `-wh`, which are successively checked in the C-domain. The required pre-movement representation, (7), is ruled out by the strictest version of the SMC (see above).

(7) $[_{\text{IP}}$ `-wh`.koj e $[_{\text{VP}}$ pital `-wh`.kogo `-wh`.kakvo]]

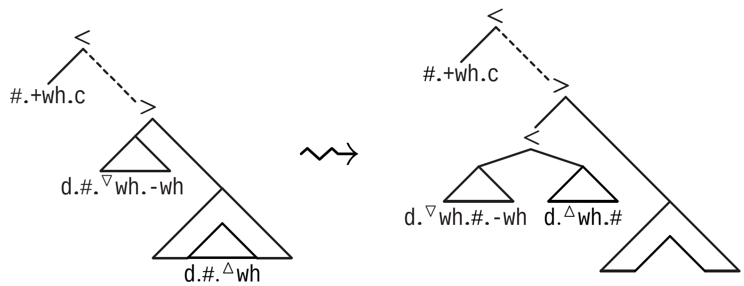
However, an SMC-violation can be circumvented if we adopt the wh-cluster hypothesis by Sabel (1998; 2001) and Grewendorf (2001). Under this perspective, wh-expressions undergo successive cluster-formation before the resulting cluster takes a single wh-movement step, in compliance with the SMC. For this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

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

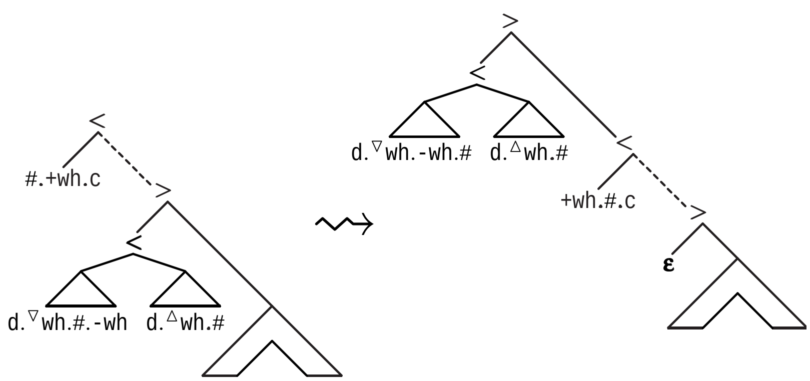
In Fig. 14 we show a derivation with two wh-phrases. For cases with three or more such phrases the intermediate ones have to be of type $\mathtt{d}.^{\triangledown}\mathtt{wh}.^{\triangle}\mathtt{wh}$. Note that additional word order variation can be found in Bulgarian, as shown in (8) (Richards 2001, p. 249).

(8) *Koj kakvo kogo e pital*

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 $\mathtt{d}.{\sim}\mathtt{v}.^{\triangledown}\mathtt{wh}$. See Sabel (1998) for more discussion of wh- and focus-driven movements in multiple wh-configurations. Semantically, wh-cluster-formation can be interpreted as quantifier composition, a.k.a. "absorption" (Higginbotham and May 1981).



Wh-clustering, n = 2, crucial step 1



Wh-clustering, n = 2, crucial step 2

Figure 14. Wh-clustering involving c-licensors and c-licensees.

Appendix C

A general picture of the MCSG landscape is given in the next figure, where, in particular, we have the following abbreviations: TAG = tree adjoining grammars, 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.

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.

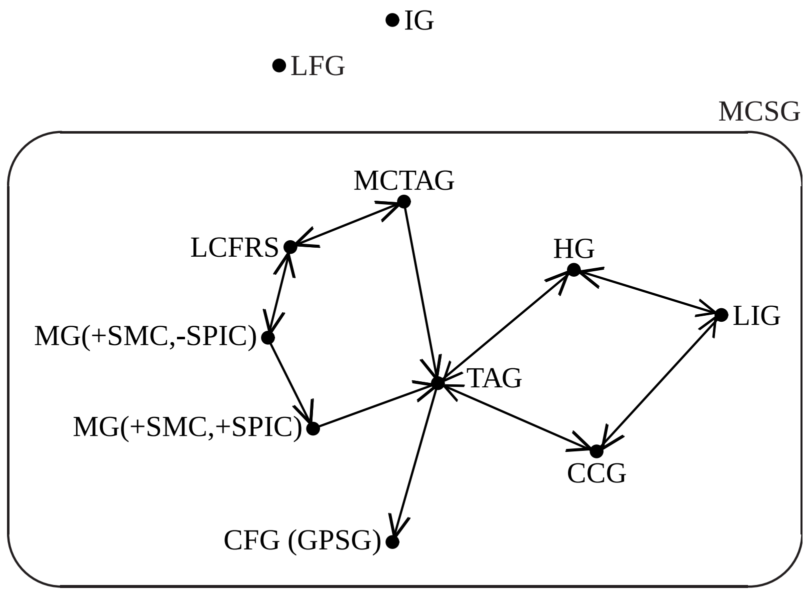


Figure 15. MCSG landscape

References

Berwick, Robert
1992 No variable is an island. Computational complexity and island constraints. In Goodluck and Rochemont (1992), pp. 35–59.

Berwick, Robert and Amy Weinberg
1982 Parsing efficiency, computational complexity, and the evaluation of grammatical theories. *Linguistic Inquiry*, 13: 165–191.

Chesi, Christiano
2004 *Complexity and Determinism in Linguistic Computation*. Manuscript, University of Siena, Siena.

Chomsky, Noam
1956 Three models for the description of language. In *IRE Transactions on Information Theory*, volume IT-2(3), pp. 113–124.
1959 On certain formal properties of grammars. *Information and Control*, 2: 137–167.
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.
1977 On wh-movement. In P. Culicover, T. Wasow, and A. Akmajian, (eds.), *Formal Syntax*, pp. 71–132. Academic Press, New York, NY.
1986 *Barriers*. MIT Press, Cambridge, MA.
1995 *The Minimalist Program*. MIT Press, Cambridge, MA.
2001 Derivation by phase. In Michael Kenstowicz, (ed.), *Ken Hale. A Life in Language*, pp. 1–52. MIT Press, Cambridge, MA.
2005 Three factors in language design. *Linguistic Inquiry*, 36: 1–22.

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

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

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.
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)*, Lecture Notes in Artificial Intelligence Vol. 3492, pp. 114–130. Springer, Berlin, Heidelberg.

Gibson, Edward
1991 Linguistic complexity. Locality of syntactic dependencies. *Cognition*, 68: 1–76.

Goodluck, Helen and Michael Rochemont, (eds.)
1992 *Island Constraints. Theory, Acquisition and Processing*. Kluwer, Dordrecht.

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.

Hauser, Marc, Noam Chomsky, and Tecumseh Fitch
2002 The faculty of language. What is it, who has it, and how did it evolve? *Science*, 298: 1569–1579.

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 P. Sells, S. M. Shieber, and T. Wasow, (eds.), *Foundational Issues in Natural Language Processing*, pp. 31–81. MIT Press, Cambridge, MA.

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.

Kolb, Hans-Peter
1997 *GB Blues. Two Essays on Procedures and Structures in Generative Syntax*. Bericht Nr. 110, Arbeitspapiere des SFB 340, Universität Tübingen.

Lebeaux, David
1991 Relative clauses, licensing, and the nature of the derivation. In Susan D. Rothstein, (ed.), *Perspectives on Phrase Structure. Heads and Licensing*, pp. 209–239. Academic Press, New York, NY.

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

Michaelis, Jens

1998 Derivational minimalism is mildly context-sensitive. In *Proceedings of the Conference on Logical Aspects of Computational Linguistics (LACL '98),* Grenoble, pp. 61–64.

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, pp. 179–198. Springer, Berlin, Heidelberg.

2001b *On Formal Properties of Minimalist Grammars*. Linguistics in Potsdam 13. Universitätsbibliothek, Publikationsstelle, Potsdam.

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

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.

Müller, Gereon and Wolfgang Sternefeld

1993 Improper movement and unambiguous binding. *Linguistic Inquiry*, 24: 461–507.

Piattelli-Palmarini, Massimo and Juan Uriagereka

2004 Immune syntax. The evolution of the language virus. In Lyle Jenkins, (ed.), *Variation and Universals in Biolinguistics*, pp. 341–377. Elsevier, Oxford.

Pritchett, Bradley

1992 Parsing with grammar. Islands, heads, and garden paths. In Goodluck and Rochemont (1992), pp. 321–349.

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.

Ross, John R.

1967 *Constraints on Variables in Syntax*. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA.

Sabel, Joachim

1998 *Principles and Parameters of Wh-Movement*. Habilitationsschrift, Universität Frankfurt.

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

Stabler, Edward P.

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

1998 Acquiring languages with movement. *Syntax*, 1: 72–97.
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.
2001 Recognizing head movement. In de Groote et al. (2001), pp. 245–260.

Stabler, Edward P. and Edward L. Keenan
2003 Structural similarity within and among languages. *Theoretical Computer Science*, 293: 345–363.

Sternefeld, Wolfgang
1998 *Grammatikalität und Sprachvermögen*. SfS-Report-02-98, Seminar für Sprachwissenschaft, Universität Tübingen.

Szabolcsi, Anna and Frans Zwarts
1997 Weak islands and an algebraic semantics for scope taking. In Anna Szabolcsi, (ed.), *Ways of Scope Taking*, pp. 217–262. Kluwer, Dordrecht.