Minimalism and Computational Linguistics

Thomas Graf

Computational linguistics is a large field that covers a wide range of topics and approaches, from the theory-minded work in mathematical linguistics and formal language theory to very applied issues such as automatic generation of descriptive captions for images. Computational work on Minimalism has largely been theory-focused, but there have been some major advances in the last few years that hold the promise of Minimalist ideas finding their way into more applied domains. This recent work builds on Stabler's *Minimalist grammars* (MGs) (Stabler 1997, 2011a; see also Kobele, this volume). While MGs are not the only computational formalization of Minimalist syntax (cf. Amblard et al. 2010, Fong and Ginsburg 2012), they constitute the largest and thematically most varied body of work to date. Hence this chapter will use MGs as the lens through which the intersection of Minimalism and computational linguistics will be explored.

After a brief introduction to MGs (Sec. 1), I will cover: their weak generative capacity and why it points towards copy-movement as possibly the most important innovation of Minimalism (Sec. 2); how investigating their strong generative capacity has yielded techniques for making MGs highly faithful to Minimalism without losing their core properties (Sec. 3); recent work on subregular complexity that is more sensitive to the empirical phenomena Minimalists care about, and also reveals surprising parallels to phonology (Sec. 4); and finally MGs as a useful for incorporating Minimalist ideas into computational modeling and language technology (Sec. 5). Throughout, emphasis will be put on how the computational insights might inform linguistic research, and in what ways linguists might contribute to our computational understanding of language. Either one may take a variety of forms: deeper insights into linguistic machinery, concrete questions about empirical phenomena, unexpected parallels between syntax and phonology or morphology, the join exploration of sentence processing, and the construction of Minimalist corpora. In order to keep this chapter accessible to as broad an audience as possible, I will focus on intuitions and the big picture — mathematically inclined readers are referred to the primary literature for the many technical details that are glossed over here. If there is one thing that I hope the reader will get out of this chapter, it is the realization that even if Minimalism and computational linguistics do not march in perfect lockstep, it is now easier than ever for each one to borrow insights from the other in order to unlock new research avenues.

1 Minimalist grammars in a nutshell

True understanding of any formalism involves two components: I) mastery of the technical machinery, and II) the ability to insightfully apply this machinery to concrete problems, e.g. raising, passive, or wh-movement. This chapter is primarily concerned with the former, i.e. the formal nuts and bolts of MGs. For an in-depth demonstration of MGs as a framework of linguistic analysis, the reader should consult Kobele (this volume).

MGs were originally defined in Stabler (1997) as a formalization of Minimalist syntax as presented in Chomsky (1995), although they have developed a lot since then. The goals of formalizing a syntactic theory are manifold. The mathematical rigor makes it possible to address questions such as: How powerful is the formalism, and given what notion of power? Is it too powerful for natural language, or not powerful enough? What is the source of its power, and how does it change if certain parameters are altered? What is the formal nature of specific syntactic proposals, and can they be stated in terms that are less theory-laden or dependent on specific notational conventions? Given these insights, which empirical or conceptual issues should linguists pay particular attention to? Are there unexpected predictions that can take syntactic research in completely new directions? The idea is that by gaining a better understanding of syntactic theory, we also get a better understanding of syntax (this will be the focus of Sec. 2, 3, and 4).

But formalization also extends the reach of syntactic theory. Mathematics is a powerful *lingua franca*, a tremendous bridge builder. By putting Minimalism on a mathematical foundation, one can link it to existing work on parsing and learnability (see Sec. 5). Not only does this strengthen the connection between theoretical syntax and psycholinguistics, it also opens up the gate to large-scale applications in modern language technology. If Minimalist ideas can be shown to be useful for practical applications, that is mutually beneficial for all involved fields.

The proof of the pudding is in the eating, though, so let us see what the MG pudding has to offer. First we need an actual definition of MGs. There are many different ones to choose from depending on whether MGs are viewed as generators of phrase structure trees (Stabler 1997), strings (Stabler and Keenan 2003), or a specification of well-formed syntactic derivations (Kobele et al. 2007; Graf and Kostyszyn 2021). These definitions are all formally equivalent; I will present a version here that draws primarily from the latter two.

Following the Chomsky-Borer hypothesis, MGs situate all language-specific variation in the lexicon. Hence every MG is just a finite set of lexical items. Each lexical item takes the form A :: α, where *A* is the item's phonetic exponent and α its string of features.

(1) **A small MG fragment for English**

car :: $\texttt{N}^-$	the :: $\texttt{N}^+\texttt{D}^-$	hit :: $\texttt{D}^+\texttt{D}^+\texttt{V}^-$
John :: $\texttt{D}^-$	the :: $\texttt{N}^+\texttt{D}^-\texttt{nom}^-$	ε :: $\texttt{V}^+\texttt{nom}^+\texttt{T}^-$
John :: $\texttt{D}^-\texttt{nom}^-$	which :: $\texttt{N}^+\texttt{D}^-\texttt{wh}^-$	ε :: $\texttt{T}^+\texttt{C}^-$
	which :: $\texttt{N}^+\texttt{D}^-\texttt{nom}^-\texttt{wh}^-$	did :: $\texttt{T}^+\texttt{wh}^+\texttt{C}^-$

Features must be checked in the order in which they appear on the lexical item. This is in line with versions of Minimalist syntax that use a very explicit feature calculus, e.g. Adger (2003) and Müller (2010). Feature checking may only occur between matching features of opposite polarity. For instance, $\texttt{N}^+$ and $\texttt{N}^-$ can check each other, but $\texttt{N}^+$ and $\texttt{D}^-$ cannot, nor can $\texttt{N}^-$ and $\texttt{N}^-$. Each feature checking step deletes the features involved and then triggers a specific operation depending on the type of the deleted features. Uppercase features like $\texttt{N}^+$ and $\texttt{D}^-$ trigger *Merge*, whereas lowercase features like $\texttt{wh}^+$ and $\texttt{nom}^-$ trigger *Move*. With two types (*Merge*, *Move*) and two polarities ($+$, $-$), there are four different kinds of MG features.

(2) **Names of MG feature types**

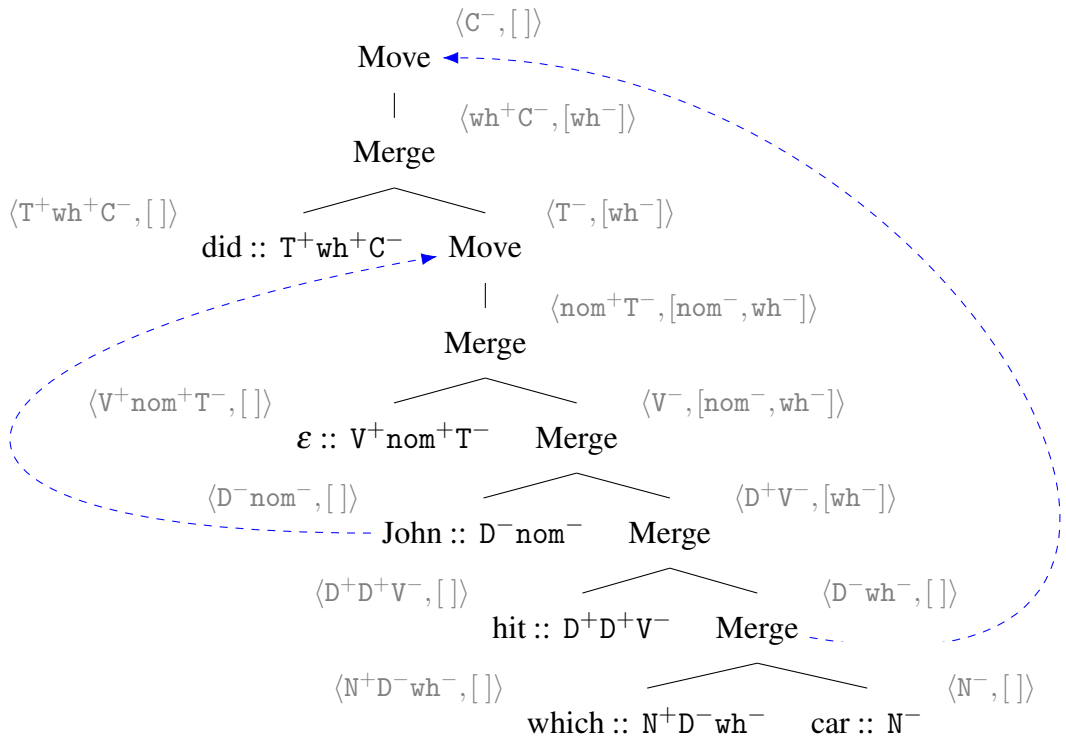
	Merge	*Move*
$+$	selector feature $\texttt{F}^+$	licensor feature $\texttt{f}^+$ (occurs at target site)
$-$	category feature $\texttt{F}^-$	licensee feature $\texttt{f}^-$ (occurs on mover)

MG polarities are quite different from the feature system of interpretable and uninterpretable features in Chomsky (1995), the subsequent switch to valued and unvalued features (Chomsky 2001), or any hybrid of the two (Pesetsky and Torrego 2007). Moreover, few Minimalists assume such a strict split between Merge features and Move features. However, these deviations from Minimalist syntax are less significant than they may seem. The MG feature calculus is not an ontological claim about how actual syntactic features operate, just as its definition of a lexicon as a finite set of feature-annotated lexical items is no theory of the human lexicon. Both are just convenient tools to regulate the application of Merge and Move, which is the actual core of MGs. They could have been defined differently to match syntactic ideas more closely; and as will be explained in Sec. 3, features could even be done away with completely to allow for free Merge and Move in the vein of Chomsky (2013). But the current system is mathematically convenient, and it is good practice to start with a simple system first and then adapt and expand it as one develops a deeper understanding of its formal plumbing.

Given some fixed MG *G*, i.e. some finite set of feature-annotated lexical items, one can then look at how the lexical items of *G* may be combined as part of a syntactic derivation. Example (3) shows a specific derivation that yields *which car did John hit* (movement arrows have been added as a visual aid). In order to make fully explicit the MG feature calculus that produces this derivation, I annotate each node with a pair where the first component is the remaining feature string of the current head whereas the second component is a list consisting of all other incomplete

feature strings that still occur in the whole subtree. I will call such sets **f[eature]-expressions**.

(3)



Note how Merge and Move differ not only in the types of features involved, but also where these features come from. Merge always combines two separate f-expressions into a new f-expression. Move, on the other hand, applies to a single f-expression that starts with a licensor feature $\texttt{f}^+$ and then reaches inside the list of that f-expression to find a feature string that starts with a matching licensee feature $\texttt{f}^-$ — in MGs, Move is literally internal Merge.

This annotation system also makes it easy to state when an MG derivation is well-formed. First, the root node must be annotated with $\langle \texttt{C}^-, [\,] \rangle$, which ensures that the derivation yields a CP with all features checked except the category feature on the head. Second, at no point in the derivation may there be an f-expression whose list contains two feature strings that start with the same feature. This is known as the **Shortest Move Constraint** (SMC) but should not be confused with the eponymous constraint in Chomsky (1995).

The SMC is at odds with many Minimalist analyses. For example, the natural implementation of the standard analysis for sentences like *who likes what* would be that both wh-phrases carry the licensee feature $\texttt{wh}^-$ but only the subject gets to move because it is closer to the target. But then the derivation would include the pair $\langle \texttt{V}^-, [\texttt{wh}^-, \texttt{wh}^-] \rangle$, which violates the SMC. Why, then, do standard MGs contain such a strong constraint? The SMC serves two purposes. First, it keeps the expressive power of MGs in check (see Sec. 2.2). Second, it ensures that Move is a deterministic operation — whenever some licensor feature $\texttt{f}^+$ gets checked, there is no ambiguity as to which $\texttt{f}^-$ feature is involved in this checking operation (this is also why the movement arrows in (3)

are just an optional visual aid). As we will see in Sec. 3, there are now more elegant means of guaranteeing those two properties, so that the SMC is no longer required. However, MGs with the SMC are still easier to study mathematically, and this is why they are the default.

The simple derivational calculus outlined so far is the core of MGs. For each MG G, the feature calculus determines all the syntactic derivations that are well-formed according to G. The derivations can then be translated into the syntactic representation they compute, e.g. bare phrase structure trees. This involves little more than relabeling interior nodes and placing each mover in its final landing site as the derivation already closely mirrors the structure of the phrase structure tree (the reader is invited to verify this by attempting such a conversion for (3)). Once the phrase structure tree has been constructed, its string yield can be obtained in the usual manner (alternatively, there are also ways to directly compute the string yield from the derivation). In contrast to the MG feature calculus and the SMC, these steps contain no surprises for anybody accustomed to Minimalist syntax. Again some technical notions such as Transfer (Chomsky 2008) are not explicitly incorporated, but this is in line with standard mathematical inquiry: we want to understand the simple before we make it complex.

2 Linguistic insights from weak generative capacity

Defining a computational formalism is of little interest on its own, it is the study of its properties that breathes life into the formalism. Keeping in the spirit of starting with simple concepts that more complex ones can be built on, one should first consider **weak generative capacity**: I) how complex are the string sets that the formalism can generate, II) how complex are natural languages if we view them as sets of well-formed strings, III) how close do specific formalism get towards characterizing the class of natural languages in this case, and IV) what does this tell us about language and linguistic theories?

Those are very reductionist questions. Section 1 explicitly introduces MGs as generators of syntactic derivations, not strings. Even more obviously, no natural language is adequately described as a collections of well-formed word sequences. Linguists have pointed this out many times, and it is one of the reasons why weak generative capacity no longer receives the attention among generativists that it enjoyed in the early days of Transformational Grammar (Chomsky 1956, 1959; see also Pullum and Gazdar 1982 and references therein). Some might also argue that given the split between I-language and E-language (Chomsky 1986), weak generative capacity cannot enjoy any kind of ontological or cognitive reality and is hence anathema to linguistic concerns. While that may well be so, and while weak generative capacity is the proverbial sledgehammer in the toolbox of formal language theory, it has proven very useful. This is perhaps the most interesting aspect of weak generative capacity: although it is possible to construct artificial scenarios

where weak generative capacity tells us very little about language and grammar formalisms, this does not seem to be the case in practice.

In the following, I first address some general issues surrounding weak generative capacity (Sec. 2.1) before discussing specific results about MGs that are of interest to Minimalist syntax (Sec. 2.2), in particular the importance of remnant movement and copy movement, certain insights from Chomsky (1957) and Chomsky (1965) that are corroborated by MGs nowadays, and how the expressivity of MGs is (not) affected by various extensions such as adjunction, sidewards movement, and Late Merge.

2.1 General remarks

Since weak generative capacity is a contentious concept among many linguists, its inclusion here requires motivation. What, specifically, does weak generative capacity have to offer to computational and/or theoretical linguistics? Among other things, it provides a fairly theory-neutral classification of natural language phenomena, which in turn provides a lower bound on what grammar formalism have to be capable of.

The best-known example of that is the demise of GPSG (Gazdar et al. 1985). GPSG was built on the assumption that a context-free formalism is sufficient to handle all syntactic phenomena. But Huybregts (1984) and Shieber (1985) provided a powerful argument against that in the form of crossing dependencies in Dutch and Swiss German, respectively. This argument consists of three steps: First, embedding constructions in Dutch and Swiss German can yield strings of the form $\mathrm{NP}_1\mathrm{NP}_2\cdots\mathrm{NP}_n\mathrm{V}_1\mathrm{V}_2\cdots\mathrm{V}_n$, where the i-th NP is an argument of the i-th verb. Next, the competence-performance distinction entails that there should be no finite cutoff point k in the grammar such that this construction does not work once n exceeds k. Finally, a series of mathematical steps is used to convert the natural language data sample into a formal language that is not context-free, and based on that it is shown that no context-free language allows for such unbounded crossing dependencies. As a result, language is at least **mildly context-sensitive** (Joshi 1985), and hence the context-free GPSG formalism necessarily undergenerates, which renders it inadequate as a model of syntax.

This finding left GPSG practitioners with three choices. They could reject the competence-performance distinction and assume some finite cutoff point for crossing dependencies. But this would have been unsatisfying as GPSG was built on the assumption that other embedding constructions are unbounded. Alternatively, they could have doubted the validity of the data presented by Huybregts and Shieber, or at least how the finite data sample was generalized to an infinite language — perhaps a different formal language would have been context-free while accommodating the data. This was indeed attempted by Pullum and Gazdar (1982) in response to the Dutch data,

but their solution failed to extend to Swiss German. The only remaining option is the one that was actually taken in the end: the formalism was modified to handle the new data, which led to the creation of HPSG (Pollard and Sag 1994). This shows that weak generative capacity can be a powerful factor in deciding the linguistic adequacy of a formalism and can drive new developments in syntax.

Crucially, many of the standard responses in syntactic debates were not available to GPSG practitioners. They could not quibble with the theoretical priors, the assumed structures, or the syntactic dependencies because the claim is descriptive, there is no structure except for the string, and there are no dependencies beyond argument-hood, which is obvious through the semantics and is also indicated via case marking in Swiss German. This reliance on very few, largely innocuous assumptions is the strength of weak generative capacity arguments. Their lack of syntactic sophistication is also what gives them tremendous robustness — it is a feature, not a bug.

2.2 Minimalism and weak generative capacity

Weak generative capacity also provides some key insights into Minimalism via the proxy of MGs. It was quickly realized that MGs can handle unbounded crossing dependencies and thus are not undermined by the Dutch and Swiss German data. More precisely, MGs generate **multiple context-free languages** (MCFLs; see Fig. 1).

While the mathematical proof is complex, the underlying intuition is simple enough. Consider once more the annotated derivation in (3). We thought of this in a bottom-up fashion, where one or two f-expressions are combined into a new f-expression via feature checking. For example, $\langle \mathtt{T}^{+}\mathtt{wh}^{+}\mathtt{C}^{-}, [\,] \rangle$ and $\langle \mathtt{T}^{-}, [\mathtt{wh}^{-}] \rangle$ combine into the new pair $\langle \mathtt{wh}^{+}\mathtt{C}^{-}, [\mathtt{wh}^{-}] \rangle$. But we could also look at it in a top-down fashion: $\langle \mathtt{wh}^{+}\mathtt{C}^{-}, [\mathtt{wh}^{-}] \rangle$ is split or *unmerged* into $\langle \mathtt{T}^{+}\mathtt{wh}^{+}\mathtt{C}^{-}, [\,] \rangle$ and $\langle \mathtt{T}^{-}, [\mathtt{wh}^{-}] \rangle$. Similarly, $\langle \mathtt{T}^{-}, [\mathtt{wh}^{-}] \rangle$ can be *unmoved* to yield $\langle \mathtt{nom}^{+}\mathtt{T}^{-}, [\mathtt{nom}^{-}, \mathtt{wh}^{-}] \rangle$. Both examples are easily represented as context-free rewrite rules:

(4) **Unmerge and Unmove as rewrite rules**

$$\begin{aligned} \langle \mathtt{wh}^{+}\mathtt{C}^{-}, [\mathtt{wh}^{-}] \rangle &\rightarrow \langle \mathtt{T}^{+}\mathtt{wh}^{+}\mathtt{C}^{-}, [\,] \rangle \; \langle \mathtt{T}^{-}, [\mathtt{wh}^{-}] \rangle \\ \langle \mathtt{T}^{-}, [\mathtt{wh}^{-}] \rangle &\rightarrow \langle \mathtt{nom}^{+}\mathtt{T}^{-}, [\mathtt{nom}^{-}, \mathtt{wh}^{-}] \rangle \end{aligned}$$

Given a fixed MG G, one can compute all possible ways of unmerging and unmoving, and convert those to a finite collection of rewrite rules. Moreover, for each lexical item $A :: \alpha$ of G one adds a rewrite rule $\langle \alpha, [\,] \rangle \rightarrow A$. So the lexical item John :: $\mathtt{D}^{-}\mathtt{nom}^{-}$ becomes the rewrite rule $\langle \mathtt{D}^{-}\mathtt{nom}^{-}, [\,] \rangle \rightarrow$ John. The result is a context-free grammar (CFG) that generates all the well-formed derivations of the MG G (and only those).

But generating syntactic derivations is not enough for weak generative capacity, we need the actual string yields. This is where the CFG reaches its limits because it linearizes the movers in

			mildly context-sensitive				
Class	REG	< CFL	< TAL	< MCFL	< PMCFL	⋯	RE
	regular	context-free	tree-adjoining	multiple context-free	parallel MCFL		recursively enumerable
Formalism	SPE	GPSG	CCG TAG	MGs with standard movement	MGs with copy movement		Aspects model, HPSG
Example string set	set of all even-length strings	set of all palindromes	ww (w = any string)	$a^n b^n c^n d^n e^n$	a^{2^n}		all theorems of first-order logic
Natural phenomena	most of phonology and morphology	center embedding	crossing dependencies, reduplication		relativized predicates in Yoruba, case stacking in Old Georgian		

Figure 1: Classes of string languages that play a key role in computational linguistics, sorted by complexity; each class is given with at least one formalism that generates all and only those string languages that belong to the class, as well as example string sets and related natural language phenomena (compiled from Culy 1985, Joshi 1985, Kaplan and Kay 1994, Vijay-Shanker and Weir 1994, Michaelis and Kracht 1997, Harkema 2001a, Michaelis 2001, Kobele 2006)

the position where they enter the derivation, rather than their final landing site. The step from context-free to multiple context-free fixes this problem by providing the means to keep parts of the string non-linearized so that they may be inserted in a different position from where they are generated. This is exactly what movement in MGs does: the overall shape of the mover is already determined in its base position, but it is linearized at the final landing site instead. Harkema (2001a) and Michaelis (2001) extend this sketch into a proof that MGs are weakly equivalent to multiple context-free grammars (MCFGs).

Several insights build on the weak equivalence of MGs and MCFGs. First of all, it shows that MGs are essentially context-free grammars with a more elaborate linearization procedure. This is similar to the original Aspects model (Chomsky 1965), where a context-free D-structure was rearranged by transformations — the transformational roots of Minimalism are still apparent in MGs. Second, Kobele (2010) proved that not all types of movement are the same in MGs. Head movement and remnant movement require the step from context-free to multiple context-free, whereas ECP-obeying phrasal movement only requires some smart coding tricks in order to be handled with the CFG sketched above. Recall that Huybregts (1984) and Shieber (1985) showed that syntax is at least mildly context-sensitive, so an empirically viable version of Minimalism cannot rely on ECP-obeying phrasal movement alone. Finally, Stabler (2013) points out that even though MGs and MCFGs are weakly equivalent, MGs are exponentially more succinct: an MG with n lexical

items corresponds to an MCFG with up to 2^n rewrite rules. This again harkens back to the early days of Transformational Grammar, as Chomsky (1957) argued against CFGs not on the basis of their generative capacity but rather their inability to succinctly express generalizations. The same still holds over 60 years later: Minimalist syntax provides a much more compact description than (multiple) context-free rewrite rules.

The weak equivalence of MGs and MCFGs also allows us to roughly compare Minimalism to other frameworks, to assess how well the typology it predicts actually matches the class of natural languages, and how modifications to the formalism may affect its power. As can be seen in Fig. 1, MGs are more expressive than Tree Adjoining Grammar (TAG) (Joshi 1985; Joshi and Schabes 1997) and Combinatory Categorial Grammar (CCG) (Steedman and Baldridge 2003). At the same time, they are not completely unrestricted formalisms like HPSG or early versions of Transformational Grammar (Peters and Ritchie 1971, 1973). Hence no additional assumptions (e.g. substantive universals or learnability requirements) are needed to rule out truly implausible languages such as the set of all strings whose length is prime. MGs, and by extension Minimalism, are in roughly the right ballpark.

It is curious that there seems to be no syntactic phenomenon that falls within the class of MCFLs but not the class of TALs. In other words, there seems to be no pressing need for the additional expressivity of standard MGs relative to TAG or CCG. However, in contrast to those formalisms, MGs allow for a straightforward extension that is in fact required for natural language: when movement may leave behind pronounced copies, MGs cross the threshold from MCFLs to parallel MCFLs (PMCFLs), and there are several language phenomena that apparently require this increased power. This includes Chinese number names (Radzinski 1991), case stacking in Old Georgian (Michaelis and Kracht 1997), and relativized predicates in Yoruba (Kobele 2006). MGs are the only formalism on the market where the empirically motivated step from MCFLs to PMCFLs is almost trivial on a technical level and is also linguistically motivated through the copy-theory of movement.

It is not surprising, then, that the status of these PMCFL phenomena is contentious, prompting debates that mirror GPSG's reaction to the crossing dependencies data. For instance, Bhatt and Joshi (2004) argue that Old Georgian case stacking could be reanalyzed as a simpler pattern that falls within the purview of TAGs. Along the same lines, the naturalness of the Chinese number names data has been doubted because the relevant patterns arise with very large numbers that are rarely encountered outside mathematical contexts. To the best of my knowledge, no counterargument has been presented to the Yoruba data in Kobele (2006). Nonetheless there is no shared consensus at this point that a formalism has to be able to generate PMCFLs in order to handle all aspects of syntax. This is an area where syntacticians could use their expertise to greatly strengthen the cases presented so far and even add new ones to the list. Because weak generative capacity

arguments depend on such a small number of assumptions, they are a powerful addition to the Minimalist case for copy movement.

While copy movement has a significant impact on weak generative capacity, the same does not hold for many other proposals in the Minimalist literature. As MGs are such a bare-bones interpretation of Minimalism, they have been extended in numerous ways over the years: rightward movement (Graf 2012a), sideward movement (Stabler 2006), across-the-board movement (Kobele 2008; Torr and Stabler 2016), lowering movement (Graf 2012a), clustering movement (Gärtner and Michaelis 2010), adjunction (Frey and Gärtner 2002; Fowlie 2013; Graf 2014b; Hunter 2015), TAG-style tree adjunction (Graf 2012b), Late Merge, (Graf 2014a), phases (Stabler 2011a), and transderivational constraints (Graf 2013), among others. None of them increase the weak generative capacity of the formalism — at the level of strings, all of them can be emulated by the basic MG version in Sec. 1.

This emphatically does not imply that all these proposals are linguistically equivalent, but rather that any differences between them cannot be related to the well-formedness of surface strings. There may still be differences in the structural descriptions, the mapping to semantics or prosody, or the succinctness of the grammar. Weak equivalence does not mean that there is no point in comparing these different proposals, but that any data that is taken to differentiate between them only does so when embedded in a network of additional syntactic assumptions. If these assumptions change, so might the relation between the proposals.

It is also worth keeping in mind that these results have only been established for MGs with the SMC. They also hold for the MG variants with relaxed SMCs that I will discuss in the next section, but if those variants should still prove too restrictive for natural language, there is no telling how the additions and modifications studied so far affect weak generative capacity. While MGs without any SMC-like constraints are known to have greater weak generative capacity (Salvati 2011), the real problem is that such MGs behave in unexpected ways. For example, adding the Specifier Island constraint to MGs with the SMC reduces their expressivity to so-called monadic branching MCFLs (Michaelis 2009; Kobele and Michaelis 2011). But adding the very same constraint to MGs without the SMC turns them into a completely unrestricted formalism that can generate any recursively enumerable stringset (Kobele and Michaelis 2005; Gärtner and Michaelis 2007). While overgeneration is not as much of an issue as undergeneration, and while some of the slack could be picked up by substantive universals as well as processing and learnability considerations, generating all recursively enumerable languages reduces MGs to the status of a pure description language that puts no meaningful restrictions at all on what is a possible natural language. This is yet another case where weak generative capacity, whatever its linguistic shortcomings may be, helps us pinpoint the parts of the formalism that keep it closely aligned with natural language syntax and should be preserved if at all possible: the SMC (or some variant thereof) is an indispensable cap

on the power of any system with Merge and Move.

3 Strong generative capacity and the malleability of MGs

From the perspective of weak generative capacity, MGs do a lot right, but their behavior depends on the SMC, which seems at odds with Minimalist practice. MGs may also appear too far removed from modern Minimalist syntax, the various extensions mentioned above notwithstanding. Once the study of MGs shifted from weak generative capacity to **strong generative capacity**, these issues could suddenly be addressed in an elegant manner because MGs turned out to be much more malleable than their original definition suggests.

Strong generative capacity measures the complexity of the structural descriptions a formalism can provide. But there are many types of structural descriptions: phrase structure trees, dependency trees, derivations, syntax-LF mappings, syntax-PF mappings, and possibly more. It has become clear, though, that derivations provide a baseline for MGs from which these other types of structures are produced. Grammar formalisms that generate (P)MCFLs or TALs can be decomposed into two components: I) a **monadic second-order (MSO)** definable tree language, and II) an MSO-definable mapping from those trees to the desired structural description (Morawietz 2003; Mönnich 2006). In the case of MGs, these correspond, respectively, to the set of well-formed derivations and the procedure that translates derivations into the desired output structures. But let us briefly discuss MSO before continuing this big picture exploration of MGs.

MSO is an extension of first-order logic with quantification over sets. Even though linguists tend to think of logic as a semantic tool, it is also very well-suited as a description language for structures and mappings. While MSO may be used with strings, trees, or graphs, I will only discuss the string case here to keep the math as simple as possible. Suppose that we wish to define the set of all strings that start with a, end with b, and are of even length. This is very easy in MSO if we have the successor relation S for string precedence: $x\ S\ y$ iff x appears immediately to the left of y. First, we will define two ancillary predicates.

(5) **Defining two ancillary predicates with MSO**

a. $\text{first}(x) \Leftrightarrow \neg\exists y[y\ S\ x]$
"x is the first node iff there is no y to its left."

b. $\text{last}(x) \Leftrightarrow \neg\exists y[x\ S\ y]$
"x is the last node iff there is no y to its right."

Next, we define a constraint on what the first and last node have to look like.

(6) **Constraining the first and last node in a string**
$\forall x[(\text{first}(x) \rightarrow a(x)) \wedge (\text{last}(x) \rightarrow b(x))]$

"If x is the first node, it must be labeled a, and if x is the last node, it must be labeled b."

Any string that satisfies the formula above necessarily starts with a and ends with b. But we still have to ensure that every string is of even length. This is where MSO's set quantification enters the picture. We will define a set X that contains all nodes in odd positions. Requiring the string to be of even length is the same as requiring that the last node does not belong to our set X.

(7) **Constraining the string length to be even**
$\exists X \forall x \forall y[(\text{first}(x) \rightarrow X(x)) \wedge (\text{last}(x) \rightarrow \neg X(x)) \wedge (x\ S\ y \rightarrow (X(x) \leftrightarrow \neg X(y)))]$
"There must be a set X such that X contains the first node, does not contain the last node, and if y is the successor of x, then X contains x iff X does not contain y."

This is an example of MSO defining a set of well-formed strings. But MSO can also define mappings that translate one structure (e.g. derivations) into another (e.g. phrase structure trees). Suppose, for instance, that we want to map each string to its mirror image. All we have to do is to define a new predicate S_o that encodes the successor relation for the output in terms of the input.

(8) **Reversing a string's order with MSO**
$x\ S_o\ y \Leftrightarrow y\ S\ o$

These are very simple examples, of course, but MSO is capable of expressing highly complex constraints and mapping with shocking succinctness. Rogers (1998) gives a full MSO-implementation of GB as presented in Haegeman (1994), deftly demonstrating that MSO is more than capable of handling linguistic ideas. Syntactic constraints are easily converted to MSO-constraints, and syntactic operations are easily converted to MSO-mappings.

With this foundation, we can now make sense of the claim in Sec. 2.2 that most MG extensions do not affect weak generative capacity. It is known that an MSO-definable tree language with an MSO-definable mapping yields at most MCFLs (or PMCFLs if the MSO-mapping is combined with a copying mechanism). MGs already generate (P)MCFLs, putting them at the expressive edge of what is possible in this MSO-based setup. Consequently, any extension that can be expressed as an MSO-constraint or an MSO-mapping — which is virtually all Minimalist proposals in the literature, for the same reason that Rogers (1998) could fit even arcane details of GB into MSO — cannot push MGs beyond (P)MCFLs.

Of course one should not infer form this that any combination of an MSO-tree language with an MSO-mapping is a possible MG. To wit, TAGs are also the combination of two such MSO-components, yet they differ from MGs in many important ways. Not only is TAGs' weak generative capacity less than that of MGs, they also generate a very different class of phrase structure trees,[1] and their parsing algorithms (Kallmeyer 2010) differ noticeably from those for MGs (Harkema

[1] Standard MGs generate multiple regular tree languages (Mönnich 2006, 2007) whereas TAGs generate monadic linear context-free tree languages (Kepser and Rogers 2011). At this point it is unclear which one of the two classes

2001b; Stabler 2013; Stanojević and Stabler 2018). MSO is a very large class, so the fact that MGs occupy a section of that class does not mean that everything in the class is an MG.

But many MG extensions are not only MSO-definable, they can be fully reduced to standard MGs. This entails that these extensions do not affect strong generative capacity either; in fact, they are just a convenient shorthand for defining standard MGs. That is because MSO-constraints have a principled connection to the MG feature calculus: every MSO-constraint can be compiled into the feature calculus (Kobele 2011; Graf 2011), and the whole MG feature calculus can be abstracted into MSO (Graf 2017). As far as weak and strong generative capacity are concerned, there is no difference between a purely feature-driven MG as in Sec. 1 and a purely constraint-based one where operations apply freely in a generate-and-filter manner (and this is also why the differences between MG features and Minimalist features are largely innocuous).

Note that this surprising equivalence only holds for MSO-constraints — changes to the MSO mapping might still affect what kind of output structures can be produced. But if one takes seriously the notion that the core of syntax is about the application of Merge and Move, then the central data structure for syntax should be derivations, not the output of those derivations. And once our focus is squarely on derivations, all Minimalist proposals correspond to MSO-constraints on MG derivations. Even Agree, which is usually thought of as an operation, can be reconceptualized as a derivational constraint along those lines, and the same goes for new movement types such as sideward movement (Graf 2012a). MSO effectively reinterprets all of Minimalism as a constraint-based theory of derivations, and in doing so it allows us to quickly and safely extend MGs in numerous ways.

The number of papers studying MG extensions has decreased significantly in recent years exactly because the MSO perspective makes it so easy to extend MGs while preserving both weak and strong generative capacity (when the latter is construed in terms of derivations). At this point, MGs are so malleable that it is hard to imagine a Minimalist proposal that cannot be made to fit. The only plausible candidate are constraints at the syntax-semantics interface that invoke identity of meaning (e.g. Rule I; Grodzinsky and Reinhart 1993; Heim 1998), and that is because identity of meaning is an undecidable property in any semantic theory that is at least as expressive as first-order logic. But even in those cases the notion of identity of meaning might just be a convenient shorthand for an intuitively more complicated but computationally less complex concept, a concept that might be expressible in MSO. Overall, then, MGs turn out to be a much more flexible toolbox than the sparse definition might suggest: All MSO-definable constraints are reducible to standard MGs that only know Merge and Move, and in a world where syntax is about derivations, almost

is a better fit for natural languages. Graf (2012b) shows that TAGs can be translated to MGs where standard phrasal movement is replaced a specific kind of lowering movement. If this lowering movement were to be shown to be inadequate for syntax, that would constitute structural evidence against TAGs.

all Minimalist proposals are MSO-definable constraints.

MSO also provides the means of relaxing the SMC, which is the biggest hurdle towards implementing Minimalist analyses in a maximally faithful manner. Consider once more the example of *who likes what* from the end of Sec. 1. The problem was that a faithful analysis should posit two wh-features, one on *who*, the other on *what*, and both features should be active at the same time, which violates the SMC. With MSO, one can quickly devise alternative systems. For instance, one could say that licensee features are checked if at all possible but may remain unchecked if that violates other locality constraints. Or one could say that *who likes what* contains only a single wh-feature, but there are locality constraints that forbid this wh-feature from occurring on *what* instead of *who*. In both cases, the resulting grammar could still be automatically converted to a standard MG, making these two strategies just different ways of defining standard MGs in a manner that is closer to linguistic conventions.

If one is willing to trade in reducibility to standard MGs for even more linguistic faithfulness, the SMC can be relaxed even further without suffering the overgeneration issues of MGs without the SMC, which were mentioned at the end of Sec. 2.2. In the MSO-framework, the primary role of the SMC is to guarantee two properties of MGs: that the set of well-formed derivations is MSO-definable, and that there be no ambiguity as to what moves where. In MGs without the SMC or some comparable replacement, neither property holds. But that is not to say that the SMC is indispensable. The SMC can be removed without losing those two properties if one also adds a collection of MSO-definable locality constraints on Move. These MSO-constraints do not need to be as rigid as the SMC and can instead be closely modeled after linguistic constraints such as Relativized Minimality. Once such an SMC-less system is in place, we can also allow licensor and/or licensee features to be checked more than once, so that a single mover can undergo an unbounded number of feature-driven movement steps, e.g. for raising, or a single landing site can be targeted by an unbounded number of movers, e.g. for multiple wh-movement (Stabler 2011a; Laszakovits 2018; Graf and Kostyszyn 2021). Because the resulting system is still MSO-definable, these extended MGs have the same weak generative capacity as standard MGs, but the removal of the SMC means that derivations may be more expressive.

This leaves us with two key insights from the MSO-perspective of MGs. First, standard MGs are a lot closer to Minimalism than their barebones definition would suggest. Almost all Minimalist proposals correspond to MSO-constraints on MG derivations, which are the central data structure of MGs. Hence these proposals can already be expressed by the feature calculus of standard MGs, although the result is much less elegant than the MSO-formulas. Only a handful of analytic options require more drastic changes because they are at odds with the SMC. But MSO can safely replace the SMC with a more flexible system without losing many of the core properties of MGs. The constraint-based view of MGs and Minimalism thus has greatly expanded our understanding of

what the formalism is capable of.

4 Subregular complexity: syntax ≈ phonology?

The MSO perspective has made MGs a tremendously malleable toolbox that easily accommodates most Minimalist machinery. This analytical flexibility is desirable for computational projects that rely on MGs as computational wrapper around Minimalism to incorporate syntactic ideas (some of those will be discussed in the next section). But it is also a worrisome state of affairs that has prompted a more fine-grained approach to MGs in recent years.

The extreme flexibility of MGs reveals them, and by extension Minimalism, to be a much looser characterization of natural language than one would assume. Remember, the duality of features and MSO-constraints means that every MSO-definable constraint can be enforced by the basic MGs we saw in Sec. 1. This includes many natural constraints, but even more unnatural ones. We already saw in the previous section that MSO can distinguish between odd and even, which syntactic constraints do not. Graf (2017) discusses many more examples of MSO-based overgeneration, each one more unnatural than the previous one. Some of the MSO-constraints also punch big loopholes into syntactic constraints, e.g. island constraints, depriving them of all empirical content. As a short demonstration of how bad things are, the reader is invited to write down an arbitrary MSO formula, with quantifiers and logical connectives combined in a random fashion. As long as the result obeys the syntax of MSO and contains no free variables, it is a constraint that can be enforced by standard MGs. But odds are this random MSO-formula has nothing in common with any constraint known to syntacticians — MSO is a giant class, with natural constraints but tiny islands in a vast ocean of the unnatural.

Due to the expressive equivalence of features and MSO-constraints, fixing this problem requires a two-pronged approach that limits both the power of features and the power of constraints. Let us consider the former first. The problem with MG features is that they have no computational cost attached to them. Recall from Sec. 1 that the lexicon already contains fully feature-annotated items, with no attention being paid to where those feature annotations come from. Consider once more the lexicon in (1) and contrast it against the one in (9) where D-features have been split into the subtypes $\mathtt{D_o}$ and $\mathtt{D_e}$ to keep track of whether the DP contains an odd or an even number of lexical items.

(9)

car :: $\mathtt{N^-}$	the :: $\mathtt{N^+D_e^-}$	hit :: $\mathtt{D_e^+D_o^+V^-}$
John :: $\mathtt{D_o^-}$	the :: $\mathtt{N^+D_e^-nom^-}$	ε :: $\mathtt{V^+nom^+T^-}$
John :: $\mathtt{D_o^-nom^-}$	which :: $\mathtt{N^+D_e^-wh^-}$	ε :: $\mathtt{T^+C^-}$
	which :: $\mathtt{N^+D_e^-nom^-wh^-}$	did :: $\mathtt{T^+wh^+C^-}$

As far as MGs are concerned, (9) is just as good a lexicon as (1), even though it enforces the

unnatural constraint that objects must have an even number of nodes and subjects an odd number. One could object against the splitting of D into $\texttt{D}_\texttt{o}$ and $\texttt{D}_\texttt{e}$ on linguistic grounds, but as Graf (2017) explains in detail, it is very difficult to convert such substantive universals into tight formal restrictions. It would be preferable to have a formal, substance-neutral criterion for distinguishing the natural (1) from the unnatural (9).

To this end, Graf (2020) proposes that Merge features, which are the primary source of the feature-constraint equivalence, have to be inferrable in a local manner. Suppose, for instance, that the derivation in (3) lacks all feature annotations. Would it be possible to at least infer the correct category and selector features for each lexical item? Graf (2020) argues that this is indeed the case, and that very little needs to be considered to determine a lexical item's category and selector features: the lexical item itself, the heads of the arguments it takes, and the head that takes it as an argument.

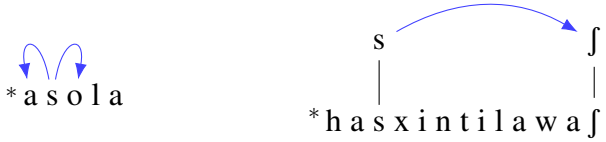
The underlying intuition is easily conveyed by the example of English *water*. In isolation, this could be a noun or a verb. But the verb takes a DP argument and is selected by a functional head, whereas the noun lacks an argument and is selected by a (possibly empty) D-head. This information is readily available from the local context of *water* in the derivation. Now contrast that to distinguishing $\texttt{D}_\texttt{o}$ from $\texttt{D}_\texttt{e}$. For the small grammar in (9), that is indeed doable by looking only at the D-head itself and whether it takes an argument. But with a more realistic grammar of English, at least some DPs will be much larger, and the only way to correctly tell $\texttt{D}_\texttt{o}$ from $\texttt{D}_\texttt{e}$ is to inspect the entire DP, which goes far beyond the locality domain conjectured by Graf (2020). A lot of empirical work remains to be done before we can safely conclude that Graf's locality domain is empirically adequate across all languages. In addition, the precise size of the domain will vary depending on one's theoretical assumptions, e.g. whether syntax contains fully inflected forms as in early Minimalism or DM-style roots. The central claim, however, is not so much how large the locality domain is, but that there is some locality domain of fixed size. This is probably true, in which case the local inference requirement greatly limits what kind of constraints can be expressed via the MG feature calculus, limiting one source of unnaturalness in MGs.

Graf (2020) formalizes the local inference requirement in terms that were first developed in the field of **subregular phonology** (see Heinz 2009, 2018, Chandlee (2014), Jardine (2016), and references therein). Subregular phonology was borne out of the desire for a tighter computational fit for phonology. While phonology is commonly modeled in terms of regular languages (see Fig. 1), this poses exactly the same overgeneration issues as the MSO-view of syntax. This prompted the development of weaker subclasses of mappings and constraints, and one of the most surprising findings in recent years is that the classes that play a major role in phonology are also tremendously useful for syntax.

Within the class of constraints, it quickly became clear that most phonotactic phenomena are

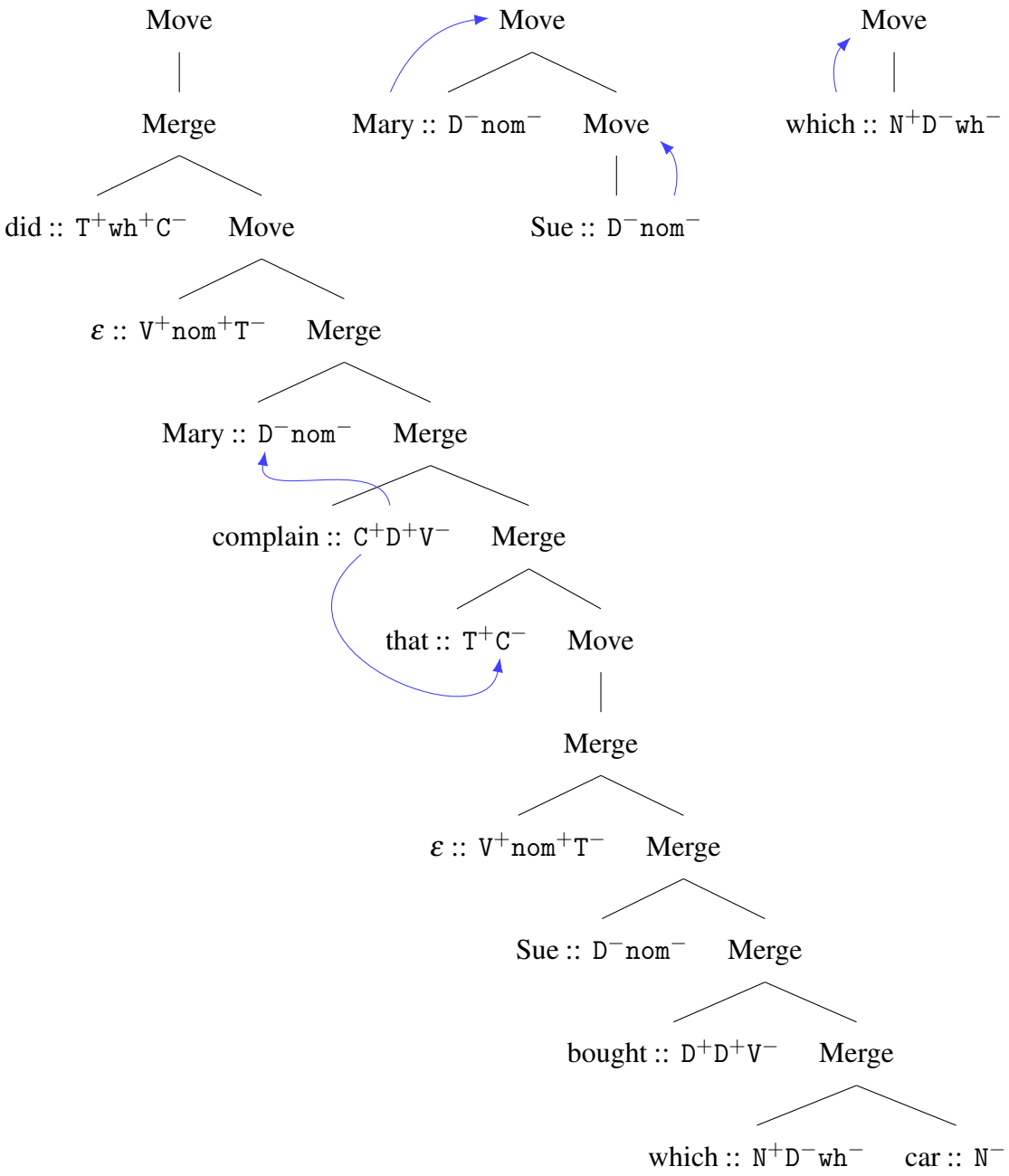
either **strictly local** or **tier-based strictly local** (Heinz et al. 2011; De Santo and Graf 2019), and that these classes are also connected to Merge and Move. A phonotactic constraint is strictly local iff it involves only a finitely bounded number of adjacent nodes. Intervocalic voicing, for example, is strictly local because it only involves three adjacent segments — to be more precise, one segment flanked by two vowels. A phonotactic constraint is tier-based strictly local iff it is local over a tier (inspired by Goldsmith 1976). Take Samala sibilant harmony, where a word may not contain any two sibilants that disagree in anteriority, no matter how far apart those sibilants are (Applegate 1972). This is not strictly local, but it is strictly local over a sibilant tier. This tier contains only the sibilants of the word, and on the tier no two adjacent segments may disagree in anteriority.

(10) **Intervocalic voicing (strictly local) and sibilant harmony (tier-based strictly local)**



Curiously, the most fundamental aspects of MGs show the very same split between strictly locality and tier-based strictly locality (Graf 2018). Merge is strictly local because the distance between the selector feature of a head and the category feature of its argument is finitely bounded, as is shown in (11). Movement dependencies, on the other hand, can cross arbitrarily long distances. However, for each movement feature `f` (`wh`, `nom`, `top`, and so on), one can construct a movement tier that only contains the lexical items carrying `f` and the Move nodes involved in checking these features. On this tier, the long-distance dependency between $\texttt{f}^{+}$ and $\texttt{f}^{-}$ is again local.

(11) **Derivation** **nom-Tier** **wh-Tier**



It seems, then, that phonology is strictly local and tier-based strictly local over strings, whereas syntax is strictly local (Merge) and tier-based strictly local (Move) over trees. This is a surprising parallel, and many more emerge once one lifts subregular concepts from phonology to syntax. Take the case of Irish, where complementizers surface with special morphology if they occur along a wh-movement path (McCloskey 2001). From a subregular perspective, this looks very much like unbounded tone plateauing, where no low tone may appear within an interval spanned

by two high tones. The high tones correspond to $\texttt{wh}^+$ and $\texttt{wh}^-$, and the illicit low tone would be a complementizer that does not show the required morphology. All of this points towards syntax and phonology (and possibly morphology, see Dolatian 2020 and references therein) having a lot more in common than is usually believed. If so, this would have far-reaching repercussions for how we think about these language modules and how we do research, as phonological evidence could suddenly be brought to bear on syntax, and the other way round.

This is still a young enterprise, and a lot of work remains to be done. It is unclear if all aspects of syntax are at most tier-based strictly local (Graf and De Santo 2019), but there is plenty of evidence that a much smaller class than MSO is sufficient for syntactic constraints. Because subregular complexity is a much more fine-grained than previous notions of complexity, it is also much more sensitive to subtle details of the linguistic data. This means that syntacticians can play a much more prominent rule in determining the subregular bounds of syntax, and how those line up with other aspects of language.

5 MGs as a research tool

The previous sections revolved around research where MGs take center stage, but there is also a rich body of work where MGs play a less prominent yet equally important role: whenever a computationally minded project wants to incorporate insights from Minimalism, MGs are an easy way to do so.

For example, Yun et al. (2014) present a computational model of relative clause processing in East Asian languages. The formal core of the paper draws from probability theory and information theory, but the analysis of relative clauses is taken from Minimalism. In order to combine these two approaches, one needs I) a rigorously specified grammar that can express Minimalist analyses, and II) a reliable method for calculating probabilities with said grammar. This is exactly what MGs excel at. They provide a very direct implementation of Minimalist ideas, do not leave any parts of the formalism underspecified, and can be made probabilistic by either converting them to MCFGs or by defining probability distributions directly over the syntactic derivations (Hunter and Dyer 2013). Without MGs, this work would have been a lot harder, if not impossible.

Another model of sentence processing has been built on the basis of existing MG parsing techniques. Stabler (2011b, 2013), extending earlier work by Mainguy (2010), presented a new top-down parser for MGs. Since we usually think of Minimalist structure building in a bottom-up fashion, it might be surprising that parsing can be done in a top-down fashion. But recall from Sec. 2.2 that MGs are, in a loose sense, CFGs with a more complex linearization procedure, and CFGs are top-down rewriting devices. Hence one can take standard top-down parsing techniques for CFGs and adapt them to MGs by keeping track of how movement affects linearization, which

is exactly what Stabler did. Kobele et al. (2013) then took this MG parser and combined it with a specific way of measuring its memory usage during the structure building process. Curiously, memory usage lines up with processing difficulty in a number of constructions. For instance, it is well-known that crossing dependencies are easier to parse than nested dependencies (Bach et al. 1986), even though the latter are simpler in terms of formal language theory. But nested dependencies increase the MG parser's memory usage more than crossing dependencies do. Under the plausible assumption that higher memory usage makes sentence processing harder for humans, this explains why crossing dependencies are easier than nested dependencies.

This approach has since then be applied to many other phenomena, including relative clause processing, attachment ambiguities, heavy NP shift, quantifier scope, gradability, and structural priming (see Gerth 2015, Graf et al. 2017, De Santo 2020, Pasternak and Graf 2021, and references therein). The approach is very accessible and does not require deep familiarity with MGs. The formalism's role is once again to provide an easy way to take Minimalist analyses and combine them with a computational model, in this case top-down parsing. It is only because of all the mathematical groundwork discussed in Sec. 2 and 3 that we now understand MGs enough to effectively employ them in this approachable, "math-light" manner.

In the future, MGs may allow Minimalist ideas to spread even deeper into neighboring disciplines. Torr et al. (2019) developed the first wide-coverage parser for MGs that is suitable for use in practical applications that require rich structural descriptions, e.g. machine translation. The parser builds on a technique called **supertagging**, which has already been used to great success with TAG and CCG. In a highly lexicalized formalism like MGs, parsing is much easier if each word is already annotated with its feature string — once the feature strings are known, it is fairly obvious how the words should be combined via Merge and Move. Supertagging decomposes parsing into a two-step procedure: feature assignment, and structure building. As the first step, each word in the input is automatically assigned a feature string, usually by a neural network or some other probabilistic device. Crucially, no structural information is used at this point, all decisions are made based only on what the input string looks like. Once all words have been annotated with features, a very fast, probabilistic parsing technique (A^* parsing) quickly computes the most likely tree structure. By design, supertagging does not always find the right parse because the initial feature annotation step only uses string-based heuristics, which can easily result in errors. This is the price one has to pay for the greatly increased speed relative to standard MG parsing techniques (Fowlie and Koller 2017) The accuracy of this supertagging parser is good enough for practical applications, many of which have a bigger need for fast parsing than for perfectly accurate parsing.

Not only it this is a unique opportunity for Minimalist syntax to raise its profile in other fields, it is also a valuable direction to take from a theoretical perspective: In general, science has benefited greatly from the interplay of theory and application, and language technology may reveal areas

where Minimalism is currently falling short. And more specifically, supertagging is opening up a turf for Minimalism that so far has been dominated by CCG — if one of the two turns out to simply be better-suited to some real-world problems than the other, that is a valuable insight.

Minimalism's main asset in comparison to other approaches may be its wide empirical coverage, with numerous analyses for many understudied languages. There is growing awareness that the majority of languages do not have the resources (word lists, corpora, online message boards, and so on) that language technology now relies on in the age of Big Data. One solution to this is a more universal, less language specific framework, as has been advocated by Bender et al. (2002). Minimalist syntax, much more so than other syntactic theories, is built around a large common core shared by all languages, making it the perfect champion of such an approach to language technology.

This is also a project that linguists can contribute to in a very concrete manner. The supertagging parser for MGs would not have been possible without the work of Torr (2017), who constructed an English MG corpus from a significant chunk of the Penn treebank. This corpus is the basis for the probabilities used by the supertagger and the A*-parser of Torr et al. (2019). But this MG corpus is still just a starting point; ideally, there would be a plethora of large MG corpora for a variety of languages. This may not be feasible given the required (wo)man power, but the Minimalist community should explore ways to convert its rich research body on numerous languages into resources that support language technology. Even very small corpora have a measurable effect on a system's performance when they are combined with linguistic insights (Srivastava et al. 2020). While some Minimalists doubt the scientific value of corpus work, this is a case where comparatively little effort could have a tremendous payoff for the whole community, and society at large.

6 Conclusion

In the over 20 years since their inception, MGs have been the driving force in combining Minimalism and computation. In doing so, they have assumed two very different, yet closely related roles. One is as a mathematical window into Minimalism, granting us deeper understanding of its machinery and thus the claims it makes about language. Examples of this are weak generative capacity arguments for copy movement, the central role of derivations, the expressive equivalence of features and constraints, and the computational parallels between phonology and syntax. The other role is that of a computational wrapper around Minimalism, which makes syntactic proposals easier to incorporate into computational research. This has allowed for several new processing models informed by Minimalist analyses, and with the arrival of a robust wide-coverage parser for MGs, many new applications are suddenly within reach.

Both types of MG work have traditionally required a high degree of mathematical expertise and thus have seen very little direct involvement from Minimalists. But it seems that a tipping point has been reached in the last few years, with MGs now more approachable and linguistically faithful than ever before, but also in greater need of direct support from linguists. Be it the use of MGs in sentence processing, the design of Minimalist corpora, or the exploration of phenomena that push the limits of subregular complexity, there are now many areas where computational and theoretical linguists can fruitfully collaborate without any major mathematical stumbling blocks along the way.

References

Adger, David. 2003. *Core syntax: A Minimalist approach*. Oxford: Oxford University Press.

Amblard, Maxime, Alain Lecomte, and Christian Retoré. 2010. Categorial minimalist grammars: From generative syntax to logical forms. *Linguistic Analysis* 36:273–308.

Applegate, Richard B. 1972. *Ineseño Chumash grammar*. Doctoral Dissertation, University of California, Berkeley.

Bach, Emmon, Colin Brown, and William Marslen-Wilson. 1986. Crossed and nested dependencies in German and Dutch: A psycholinguistic study. *Language and Cognitive Processes* 1:249–262.

Bender, Emily M, Dan Flickinger, and Stephan Oepen. 2002. The grammar matrix: An open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In *Proceedings of the 2002 Workshop on Grammar Engineering and Evaluation - Volume 15*, COLING-GEE '02, 1–7. Association for Computational Linguistics.

Bhatt, Rajesh, and Aravind Joshi. 2004. Semilinearity is a syntactic invariant. A reply to Kracht and Michaelis 1997. *Linguistic Inquiry* 35:683–692.

Chandlee, Jane. 2014. *Strictly local phonological processes*. Doctoral Dissertation, University of Delaware. URL `http://udspace.udel.edu/handle/19716/13374`.

Chomsky, Noam. 1956. Three models for the description of language. *IRE Transactions on Information Theory* 2:113–124.

Chomsky, Noam. 1957. *Syntactic structures*. The Hague: Mouton.

Chomsky, Noam. 1959. On certain formal properties of grammars. *Information and Control* 2:137–167. URL https://doi.org/10.1016/S0019-9958(59)90362-6.

Chomsky, Noam. 1965. *Aspects of the theory of syntax*. Cambridge, MA: MIT Press.

Chomsky, Noam. 1986. *Knowledge of language: Its nature, origin, and use*. New York: Praeger.

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

Chomsky, Noam. 2001. Derivation by phase. In *Ken Hale: A life in language*, ed. Michael J. Kenstowicz, 1–52. Cambridge, MA: MIT Press.

Chomsky, Noam. 2008. On phases. In *Foundational issues in linguistic theory: Essay in honor of Jean-Roger Vergnaud*, ed. Robert Freidin, Carlos P. Otero, and Maria Luisa Zubizarreta, 133–166. Cambridge, MA: MIT Press.

Chomsky, Noam. 2013. Problems of projection. *Lingua* 130:33–49.

Culy, Christopher. 1985. The complexity of the vocabulary of Bambara. *Linguistics and Philosophy* 8:345–351. URL https://doi.org/10.1007/978-94-009-3401-6_14.

De Santo, Aniello. 2020. *Structure and memory: A computational model of storage, gradience, and priming*. Doctoral Dissertation, Stony Brook University.

De Santo, Aniello, and Thomas Graf. 2019. Structure sensitive tier projection: Applications and formal properties. In *Formal Grammar*, ed. Raffaella Bernardi, Gregory Kobele, and Sylvain Pogodalla, 35–50. Berlin, Heidelberg: Springer.

Dolatian, Hossep. 2020. *Computational locality of cyclic phonology in Armenian*. Doctoral Dissertation, Stony Brook University.

Fong, Sandiway, and Jason Ginsburg. 2012. Computation with doubling constituents: Pronouns and antecedents in phase theory. In *Towards a biolinguistic understanding of grammar*, ed. Anna Maria Di Sciullo, 303–338. Amsterdam: John Benjamins.

Fowlie, Meaghan. 2013. Order and optionality: Minimalist grammars with adjunction. In *Proceedings of the 13th Meeting on the Mathematics of Language (MoL 13)*, ed. András Kornai and Marco Kuhlmann, 12–20.

Fowlie, Meaghan, and Alexander Koller. 2017. Parsing minimalist languages with interpreted regular tree grammars. In *Proceedings of the 13th International Workshop on Tree Adjoining Grammars and Related Formalisms*, 11–20.

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*, ed. Gerhard Jäger, Paola Monachesi, Gerald Penn, and Shuly Wintner, 41–52.

Gärtner, Hans-Martin, and Jens Michaelis. 2007. Some remarks on locality conditions and Minimalist Grammars. In *Interfaces + recursion = language? Chomsky's minimalism and the view from syntax-semantics*, ed. Uli Sauerland and Hans-Martin Gärtner, 161–196. Berlin: Mouton de Gruyter.

Gärtner, Hans-Martin, and Jens Michaelis. 2010. On the treatment of multiple-wh-interrogatives in Minimalist grammars. In *Language and logos*, ed. Thomas Hanneforth and Gisbert Fanselow, 339–366. Berlin: Akademie Verlag.

Gazdar, Gerald, Ewan Klein, Geoffrey K. Pullum, and Ivan A. Sag. 1985. *Generalized phrase structure grammar*. Oxford: Blackwell.

Gerth, Sabrina. 2015. *Memory limitations in sentence comprehension. a structure-based complexity metric of processing difficulty*. Doctoral Dissertation, University of Potdsam.

Goldsmith, John. 1976. *Autosegmental phonology*. Doctoral Dissertation, MIT.

Graf, Thomas. 2011. Closure properties of Minimalist derivation tree languages. In *LACL 2011*, ed. Sylvain Pogodalla and Jean-Philippe Prost, volume 6736 of *Lecture Notes in Artificial Intelligence*, 96–111. Heidelberg: Springer. URL `https://dx.doi.org/10.1007/978-3-642-22221-4_7`.

Graf, Thomas. 2012a. Movement-generalized Minimalist grammars. In *LACL 2012*, ed. Denis Béchet and Alexander J. Dikovsky, volume 7351 of *Lecture Notes in Computer Science*, 58–73. URL `http://dx.doi.org/10.1007/978-3-642-31262-5_4`.

Graf, Thomas. 2012b. Tree adjunction as Minimalist lowering. In *Proceedings of the 11th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+11)*, 19–27. URL `http://www.aclweb.org/old_anthology/W/W12/W12-4603.pdf`.

Graf, Thomas. 2013. *Local and transderivational constraints in syntax and semantics*. Doctoral Dissertation, UCLA. URL `http://thomasgraf.net/doc/papers/Graf13Thesis.pdf`.

Graf, Thomas. 2014a. Late merge as lowering movement in Minimalist grammars. In *LACL 2014*, ed. Nicholas Asher and Sergei Soloviev, volume 8535 of *Lecture Notes in Computer Science*, 107–121. Heidelberg: Springer. URL `https://doi.org/10.1007/978-3-662-43742-1_9`.

Graf, Thomas. 2014b. Models of adjunction in Minimalist grammars. In *Formal Grammar 2014*, ed. Glynn Morrill, Reinhard Muskens, Rainer Osswald, and Frank Richter, volume 8612 of *Lecture Notes in Computer Science*, 52–68. Heidelberg: Springer. URL https://doi.org/10.1007/978-3-662-44121-3_4.

Graf, Thomas. 2017. A computational guide to the dichotomy of features and constraints. *Glossa* 2:1–36. URL https://dx.doi.org/10.5334/gjgl.212.

Graf, Thomas. 2018. Why movement comes for free once you have adjunction. In *Proceedings of CLS 53*, ed. Daniel Edmiston, Marina Ermolaeva, Emre Hakgüder, Jackie Lai, Kathryn Montemurro, Brandon Rhodes, Amara Sankhagowit, and Miachel Tabatowski, 117–136.

Graf, Thomas. 2020. Curbing feature coding: Strictly local feature assignment. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2020*, 362–371.

Graf, Thomas, and Aniello De Santo. 2019. Sensing tree automata as a model of syntactic dependencies. In *Proceedings of the 16th Meeting on the Mathematics of Language*, 12–26. Toronto, Canada: Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W19-5702.

Graf, Thomas, and Kalina Kostyszyn. 2021. Multiple wh-movement is not special: The subregular complexity of persistent features in Minimalist grammars. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2021*. To appear.

Graf, Thomas, James Monette, and Chong Zhang. 2017. Relative clauses as a benchmark for Minimalist parsing. *Journal of Language Modelling* 5:57–106. URL http://dx.doi.org/10.15398/jlm.v5i1.157.

Grodzinsky, Yosef, and Tanja Reinhart. 1993. The innateness of binding and coreference. *Linguistic Inquiry* 24:69–102.

Haegeman, Liliane. 1994. *Introduction to government and binding theory*. Oxford: Blackwell, 2 edition.

Harkema, Henk. 2001a. A characterization of Minimalist languages. In *Logical aspects of computational linguistics (LACL'01)*, ed. Philippe de Groote, Glyn Morrill, and Christian Retoré, volume 2099 of *Lecture Notes in Artificial Intelligence*, 193–211. Berlin: Springer.

Harkema, Henk. 2001b. *Parsing Minimalist languages*. Doctoral Dissertation, University of California.

Heim, Irene. 1998. Anaphora and semantic interpretation: A reinterpretation of Reinhart's approach. In *The interpretive tract*, ed. Uli Sauerland and O. Percus, volume 25 of *MIT Working Papers in Linguistics*, 205–246. Cambridge, MA: MIT Press.

Heinz, Jeffrey. 2009. On the role of locality in learning stress patterns. *Phonology* 26:303–351. URL `https://doi.org/10.1017/S0952675709990145`.

Heinz, Jeffrey. 2018. The computational nature of phonological generalizations. In *Phonological typology*, ed. Larry Hyman and Frank Plank, Phonetics and Phonology, chapter 5, 126–195. Mouton De Gruyter.

Heinz, Jeffrey, Chetan Rawal, and Herbert G. Tanner. 2011. Tier-based strictly local constraints in phonology. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, 58–64. URL `http://www.aclweb.org/anthology/P11-2011`.

Hunter, Tim. 2015. Deconstructing merge and move to make room for adjunction. *Syntax* 18:266–319.

Hunter, Tim, and Chris Dyer. 2013. Distrubtions on Minimalist grammar derivations. In *Proceedings of the 13th meeting on the mathematics of language (mol 13)*, 1–11.

Huybregts, M. A. C. 1984. The weak adequacy of context-free phrase structure grammar. In *Van periferie naar kern*, ed. Ger J. de Haan, Mieke Trommelen, and Wim Zonneveld, 81–99. Dordrecht: Foris.

Jardine, Adam. 2016. Computationally, tone is different. *Phonology* 33:247–283. URL `https://doi.org/10.1017/S0952675716000129`.

Joshi, Aravind. 1985. Tree-adjoining grammars: How much context sensitivity is required to provide reasonable structural descriptions? In *Natural language parsing*, ed. David Dowty, Lauri Karttunen, and Arnold Zwicky, 206–250. Cambridge: Cambridge University Press.

Joshi, Aravind, and Y. Schabes. 1997. Tree-adjoining grammars. In *Handbook of formal languages*, ed. Grzegorz Rosenberg and Arto Salomaa, 69–123. Berlin: Springer.

Kallmeyer, Laura. 2010. *Parsing beyond context-free grammars*. Berlin: Springer.

Kaplan, Ronald M., and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics* 20:331–378. URL `http://www.aclweb.org/anthology/J94-3001.pdf`.

Kepser, Stephan, and James Rogers. 2011. The equivalence of tree adjoining grammars and monadic linear context-free tree grammars. *Journal of Logic, Language and Information* 20:361–384.

Kobele, Gregory M. 2006. *Generating copies: An investigation into structural identity in language and grammar*. Doctoral Dissertation, UCLA. URL `http://home.uchicago.edu/~gkobele/files/Kobele06GeneratingCopies.pdf`.

Kobele, Gregory M. 2008. Across-the-board extraction and Minimalist grammars. In *Proceedings of the Ninth International Workshop on Tree Adjoining Grammars and Related Frameworks*.

Kobele, Gregory M. 2010. Without remnant movement, MGs are context-free. In *MOL 10/11*, ed. Christian Ebert, Gerhard Jäger, and Jens Michaelis, volume 6149 of *Lecture Notes in Computer Science*, 160–173.

Kobele, Gregory M. 2011. Minimalist tree languages are closed under intersection with recognizable tree languages. In *LACL 2011*, ed. Sylvain Pogodalla and Jean-Philippe Prost, volume 6736 of *Lecture Notes in Artificial Intelligence*, 129–144. URL `https://doi.org/10.1007/978-3-642-22221-4_9`.

Kobele, Gregory M., Sabrina Gerth, and John T. Hale. 2013. Memory resource allocation in top-down Minimalist parsing. In *Formal Grammar: 17th and 18th International Conferences, FG 2012, Opole, Poland, August 2012, Revised Selected Papers, FG 2013, Düsseldorf, Germany, August 2013*, ed. Glyn Morrill and Mark-Jan Nederhof, 32–51. Berlin, Heidelberg: Springer. URL `https://doi.org/10.1007/978-3-642-39998-5_3`.

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*, 81–93. Edinburgh.

Kobele, Gregory M., and Jens Michaelis. 2011. Disentangling notions of specifier impenetrability. In *The Mathematics of Language*, ed. Makoto Kanazawa, András Kornia, Marcus Kracht, and Hiroyuki Seki, volume 6878 of *Lecture Notes in Artificial Intelligence*, 126–142. Berlin, Heidelberg: Springer. URL `https://doi.org/10.1007/978-3-642-23211-4_8`.

Kobele, Gregory M., Christian Retoré, and Sylvain Salvati. 2007. An automata-theoretic approach to Minimalism. In *Model Theoretic Syntax at 10*, ed. James Rogers and Stephan Kepser, 71–80.

Laszakovits, Sabine. 2018. Case theory in Minimalist grammars. In *Proceedings of Formal Grammar 2018*, ed. Annie Foret, Greg Kobele, and Sylvain Pogodalla, 37–61. Berlin: Springer.

Mainguy, Thomas. 2010. A probabilistic top-down parser for Minimalist grammars. ArXiv:1010.1826v1.

McCloskey, James. 2001. The morphosyntax of wh-extraction in Irish. *Journal of Linguistics* 37:67–100.

Michaelis, Jens. 2001. Transforming linear context-free rewriting systems into Minimalist grammars. *Lecture Notes in Artificial Intelligence* 2099:228–244.

Michaelis, Jens. 2009. An additional observation on strict derivational minimalism. In *FG-MOL 2005*, ed. James Rogers, 101–111.

Michaelis, Jens, and Marcus Kracht. 1997. Semilinearity as a syntactic invariant. In *Logical Aspects of Computational Linguistics*, ed. Christian Retoré, volume 1328 of *Lecture Notes in Artifical Intelligence*, 329–345. Springer. URL `http://dx.doi.org/10.1007/BFb0052165`.

Mönnich, Uwe. 2006. Grammar morphisms. Ms. University of Tübingen.

Mönnich, Uwe. 2007. Minimalist syntax, multiple regular tree grammars and direction preserving tree transductions. In *Model Theoretic Syntax at 10*, ed. James Rogers and Stephan Kepser, 83–87.

Morawietz, Frank. 2003. *Two-step approaches to natural language formalisms*. Berlin: Walter de Gruyter.

Müller, Gereon. 2010. On deriving CED effects from the PIC. *Linguistic Inquiry* 41:35–82.

Pasternak, Robert, and Thomas Graf. 2021. Cyclic scope and processing difficulty in a Minimalist parser. *Glossa* 6:1–34.

Pesetsky, David, and Esther Torrego. 2007. The syntax of valuation and the uninterpretability of features. In *Phrasal and clausal architectur*, ed. Simin Karimi, Samiian Samiian, and Wendy K. Wilkins, 262–294. Amsterdam: Benjamins.

Peters, Stanley, and Robert W. Ritchie. 1971. On restricting the base component of transformational grammars. *Information and Control* 18:483–501.

Peters, Stanley, and Robert W. Ritchie. 1973. On the generative power of transformational grammars. *Information Sciences* 6:49–83.

Pollard, Carl, and Ivan Sag. 1994. *Head-driven phrase structure grammar*. Stanford and Chicago: CSLI and The University of Chicago Press.

Pullum, Geoffrey K., and Gerald Gazdar. 1982. Natural languages and context free languages. *Linguistics and Philosophy* 4:471–504.

Radzinski, Daniel. 1991. Chinese number names, tree adjoining languages, and mild context sensitivity. *Computational Linguistics* 17:277–300. URL `http://ucrel.lancs.ac.uk/acl/J/J91/J91-3002.pdf`.

Rogers, James. 1998. *A descriptive approach to language-theoretic complexity*. Stanford: CSLI.

Salvati, Sylvain. 2011. Minimalist grammars in the light of logic. In *Logic and grammar — essays dedicated to Alain Lecomte on the occasion of his 60th birthday*, ed. Sylvain Pogodalla, Myriam Quatrini, and Christian Retoré, number 6700 in Lecture Notes in Computer Science, 81–117. Berlin: Springer.

Shieber, Stuart M. 1985. Evidence against the context-freeness of natural language. *Linguistics and Philosophy* 8:333–345. URL `http://dx.doi.org/10.1007/BF00630917`.

Srivastava, Aarohi, Robert Frank, Sarah Widder, and David Chartash. 2020. The role of linguistic features in domain adaptation: TAG parsing of questions. In *Proceedings of the Society for Computation in Linguistics*, volume 3. URL `https://scholarworks.umass.edu/scil/vol3/iss1/41`.

Stabler, Edward P. 1997. Derivational Minimalism. In *Logical aspects of computational linguistics*, ed. Christian Retoré, volume 1328 of *Lecture Notes in Computer Science*, 68–95. Berlin: Springer. URL `https://doi.org/10.1007/BFb0052152`.

Stabler, Edward P. 2006. Sidewards without copying. In *Formal Grammar '06, Proceedings of the Conference*, ed. Gerald Penn, Giorgio Satta, and Shuly Wintner, 133–146. Stanford: CSLI.

Stabler, Edward P. 2011a. Computational perspectives on Minimalism. In *Oxford handbook of linguistic Minimalism*, ed. Cedric Boeckx, 617–643. Oxford: Oxford University Press.

Stabler, Edward P. 2011b. Top-down recognizers for MCFGs and MGs. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*, 39–48.

Stabler, Edward P. 2013. Two models of minimalist, incremental syntactic analysis. *Topics in Cognitive Science* 5:611–633. URL `https://dx.doi.org/10.1111/tops.12031`.

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

Stanojević, Miloš, and Edward Stabler. 2018. A sound and complete left-corner parser for Minimalist grammars. In *Proceedings of the 8th Workshop on Cognitive Aspects of Computational Language Learning and Processing*, 65–74.

Steedman, Mark, and Jason Baldridge. 2003. Combinatory categorial grammar. Unpublished tutorial paper.

Torr, John. 2017. Autobank: a semi-automatic annotation tool for developing deep Minimalist grammar treebanks. In *Proceedings of the Demonstrations at the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 81–86.

Torr, John, and Edward P. Stabler. 2016. Coordination in Minimalist grammars: Excorporation and across the board (head) movement. In *Proceedings of the 12th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+12)*, 1–17. Düsseldorf, Germany. URL `https://www.aclweb.org/anthology/W16-3301`.

Torr, John, Miloš Stanojević, Mark Steedman, and Shay Cohen. 2019. Wide-coverage neural A* parsing for Minimalist grammars. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Florence, Italy: Association for Computational Linguistics.

Vijay-Shanker, K., and David J. Weir. 1994. The equivalence of four extensions of context-free grammars. *Mathematical Systems Theory* 27:511–545.

Yun, Jiwon, Zhong Chen, Tim Hunter, John Whitman, and John Hale. 2014. Uncertainty in processing relative clauses across East Asian languages. *Journal of East Asian Linguistics* 1–36.