

Two models of minimalist, incremental syntactic analysis

Edward P. Stabler

Minimalist grammars (MGs) and multiple context free grammars (MCFGs) are weakly equivalent in the sense that they define the same languages, a large mildly context sensitive class that properly includes context free languages. But in addition, for each MG, there is an MCFG which is strongly equivalent in the sense that it defines the same language with isomorphic derivations. However, the structure building rules of MGs but not MCFGs are defined in a way that generalizes across categories. Consequently, MGs can be exponentially more succinct than their MCFG equivalents, and this difference shows in parsing models too. An incremental, top-down beam parser for MGs is defined here, sound and complete for all MGs, and hence also capable of parsing all MCFG languages. But since the parser represents its grammar transparently, the relative succinctness of MGs is again evident. And although the determinants of MG structure are narrowly and discretely defined, probabilistic influences from a much broader domain can influence even the earliest analytic steps, allowing frequency and context effects to come early and from almost anywhere, as expected in incremental models.

Keywords minimalist grammar, parsing, multiple context free grammar

A psychological model is not adequate if a response, any response really due to the mechanism being modeled, is simply not in the range of the model. But suppose we compare two models that agree on the range of behaviors to be modeled; in fact, suppose their input/output behaviors are provably identical. Then can there be a reason to prefer one over the other? Yes. It is a familiar fact that very different algorithms, with very different data structures, can compute exactly the same function. And in such cases, it can matter which one is implemented. Since recent mathematical work on grammars has established a wide range of equivalence results, comparisons of models that are in some relevant sense equivalent are likely to arise often. This paper considers exactly such a case. For every minimalist grammar (MG) there is a multiple context free grammar (MCFG) which is weakly equivalent (defining the same languages) and strongly equivalent (isomorphic derivation trees for every string). But while MGs will look at least vaguely familiar to linguists, MCFGs are surprising. Like a context free grammar, each MCFG has only a finite set of production rules, and does not distinguish movement dependencies from complement selection, or either of those from agreement. All dependencies are enforced uniformly through the category system. As a result, MCFGs are often much larger, even exponentially larger, than MGs they are strongly equivalent to. And while the extension of MCFGs to probabilis-

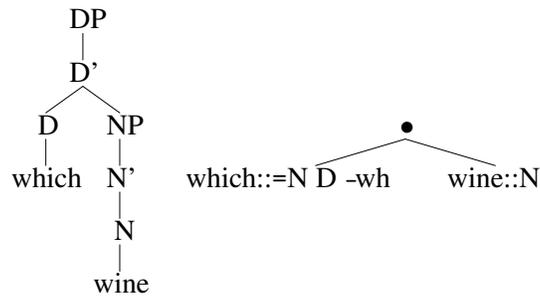
tic models is straightforward, like the extension of CFGs to probabilistic CFGs (PCFGs), probabilistic MCFGs share with PCFGs unrealistic independence assumptions, motivating a model in which probabilities are not determined by the very restricted mechanisms needed for definition of structure.

The relevance of such considerations to performance models is illustrated in §2, where a top-down beam MCFG parser (Stabler 2011b) is adapted to the MGs of §1. Since this implementation of MGs is simple and transparent, the structure and relative succinctness of the grammars is evident. A successful MG parse is a kind of structurally conditioned check of lexical requirements, implemented as the traversal of a graph representing the lexicon. No such perspective is available in the similar MCFG parsing. §3 briefly observes how extragrammatical and interface conditions can have gradient influences. The model presented here is simplified in certain respects (and perhaps it is more complex than necessary in other respects), but it can be extended to a wide range of live proposals, some of which are mentioned in §4.

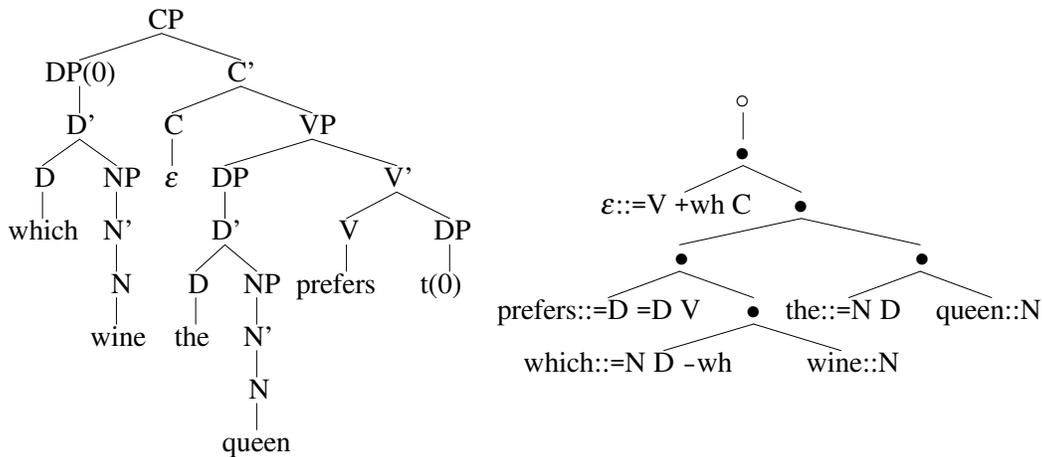
§5 reviews some of the evidence that could support this kind of MG model over equivalent MCFG-based models (of which CFG-based models are a special case), noting that the issues at stake here are old ones. The reasons offered in favor of MGs over MCFGs echo the informal arguments given against phrase structure grammars by Chomsky (1956), but now the relevant formal comparison can be made more rigorous and applied to grammars for mildly context sensitive languages. The important question then, as now, is not expressive power, but what the linguistic structures are and how they could be calculated in language use. This remark may seem puzzling at first, but I will explain it more carefully in proposals H4 and H5 of the concluding section, §6. In the simple setting of our MG/MCFG comparison, where expressive power does not distinguish the alternatives at all, at least some relevant aspects of these questions can be made completely clear. Many issues are clearer now than they were in 1956, and the evidence still overwhelmingly and uncontroversially disfavors (M)CFG-based models.

1 MGs and MCFGs

Minimalist grammars (MGs) aim to formalize some of the fundamental ideas of the the minimalist program (Chomsky 1995:and much following work). MGs and strongly equivalent MCFGs are gently introduced in Stabler (2011a), with reference to the technical literature that explores these grammars in great detail. But the basic situation is easily sketched here. When a determiner combines with a noun, it is common to depict the result with a tree like the one on the left; but we can also draw a simpler tree like the one on the right that indicates the derivational step by itself:



The • in the tree on the right signifies the merge operation that builds the tree on the left. Also notice that in the tree on the right, each lexical item has been associated with a sequence of features. For example, *which* has a sequence of 3 features: =ND -wh. The = means ‘selects’, and the - marks licensing requirements, so this sequence of features indicates that *which* selects a noun phrase in order to form a determiner phrase, one which needs to move to a position where its -wh feature is licensed. The merge operation • can apply because of the match between =N and the category N. We assume for the moment that features are matched from left to right, and that each matching operation checks and deletes both features.¹ With the additional steps shown in the tree below on the right, culminating in the final movement of the wh-phrase, indicated by ◦ in the derivation tree, we build the derived tree on the left:



In this simple structure for the clause *which wine the queen prefers*, as it might appear in the sentence *I wonder [which wine the queen prefers]*, notice that the the second, higher DP selected by V, *the queen*, attaches to the left of V in the derived tree, as a ‘specifier’,

¹The idea that grammatical features are ordered in a sequence is not mainstream, at least not explicitly. Feature ordering is sometimes explicit (Abels 2012, 2007; Müller 2010; Brody 2000; Chomsky 1995:§3), but roughly equivalent ideas are implicit in many theories that assume ordered projections each of which has just one non-complex categorial feature (Sportiche 1998; Caha 2009; Adger 2010; Bobaljik 2011:and many others). The assumption that matched features are both checked and deleted is also not mainstream, but again it is easy to allow certain kinds of persistent features and asymmetric checking without changing the computational properties of interest here (Stabler 2011a). And the Kayneian SVO hypothesis has been challenged (Abels and Neeleman 2012), but is easily relaxed (Stabler 2011a) without affecting the computational properties discussed here.

following Kayne’s (1994) proposal that specifiers are on the left. And notice that the top step \circ shown in the derivation tree is unary, since rather than attaching two separate constituents, it moves the *-wh* phrase from inside the tree and attaches it at the root, checking and deleting both the *+wh* licensing feature of the (silent) complementizer *C* and the *-wh*. In this simple system, counting the features of all the lexical items at the leaves of the derivation, there are 13. Each derivation step matches, checks and deletes the leftmost features in the pair of constituents it applies to, so after the 6 indicated steps (the 6 internal nodes of the tree on the right), there is one feature left, namely, the ‘start category’ *C*.

This example is not meant to be a serious structural proposal, but only to indicate roughly how MG structure building mechanisms work. With simple definitions of the binary merge step \bullet and the unary move step \circ , it is possible to assign features to lexical items in a way that allows us to formalize many proposals in the syntactic literature. What is more surprising is that the class of languages that can be defined by applying \bullet and \circ to a finite lexicon is one that was already well known before this kind of grammar was proposed. The MG-definable languages are exactly the languages defined by multiple context free grammars, by set-local tree adjoining grammars, and many other formalisms (Michaelis 2001; Harkema 2001a). Furthermore, like these other grammars, it is natural to regard MGs as folding together two different sorts of principles: those that define the derivation tree, and those that map the derivation to its pronounced and interpreted form. One insight that comes from this perspective is that both of those steps, the definition of derivations (like the tree on the right, above), and the mappings to derived structures (like the tree on the left), are very simple, finite state computations. (Appendix B explains more carefully how the derivation tree depicted above, on the right, relates to the the derived tree on the left. Appendix C precisely defines the operations \bullet , \circ , and the parsing algorithm.)

MGs as MCFGs. MCFGs are a restricted version of Pollard’s (1984) generalized context free grammars, studied by Seki, Matsumura, Fujii, and Kasami (1991). MCFGs do not have movement, but they have something similar, namely, categories with two or more string parts. For example, the MG step above that combines the determiner and noun could be represented with a CFG rule like this,

$$D\text{-wh}(x_1x_2) \rightarrow =N=D\text{-wh}(x_1) \quad N(x_2).$$

This rule says that a string x_1 with category $=N=D\text{-wh}$ can combine with a string x_2 with category *N* to yield the concatenated string x_1x_2 with category $=D\text{-wh}$. That rule is context free, but the expressive power of a MCFG comes from allowing categories that classify tuples of strings. For example, the MG step shown above which combines the verb and its object corresponds to the use of a MCFG rule like this:

$$=DV, \text{-wh}(x_1, x_2) \rightarrow =D=DV(x_1) \quad D\text{-wh}(x_2).$$

This rule looks odd at first, because the category name on the left side of the rule is given two MG feature sequences separated by a comma. Intuitively, the sequence $=DV$ is the category of the VP, and *-wh* is the category of the moving DP, combined into a single category for the pair of strings. In the MCFG, though, this is a single, atomic category label. We write it with a comma in it to aid in the comparison with MGs, but MCFG categories (like CFG categories) are atoms as far as the grammar is concerned. So the rule above simply says that a string x_1 with category $=D=DV$ can combine with x_2 of category *D -wh* to yield a

pair of strings x_1, x_2 with category =D V, -wh. So rather than deriving a tree from which a -wh phrase can move, as in Chomskian syntax, we have a VP category with two parts, two strings, one of which is the ‘mover’, the -wh phrase, Pollard-style. The moving element x_2 ‘lands’ in its surface position with the last rule of the derivation:

$$C(x_2x_1) \rightarrow +whC, -wh(x_1, x_2).$$

With essentially this idea, Michaelis (1998) proved that, for any MG, we can formulate a MCFG that derives exactly the same strings as the MG with derivations that have exactly the same shape. That is, we can design an MCFG that derives *which wine the queen prefers* with a derivation that is isomorphic to the MG derivation shown above. For every MG, there is an MCFG that derives exactly the same strings with the same derivation trees. This is our strong equivalence result.

The relative succinctness of MGs. Every MG has a strongly equivalent MCFG, but it is easy to see that the strongly equivalent MCFG must sometimes be very much larger than the MG. One problem is movement. Intuitively, if we have i movers that might or might not be extractable from a VP, we will need VP categories for every possible subset of the i movers, all 2^i of them. (Appendix A presents the argument in more detail.) So although we could, in principle, compute MG derivations by parsing with the strongly equivalent MCFG, it is at the cost of missing a generalization and consequently having a larger grammar with separate rules for each instance of the general patterns. The hypothesis that the human parser neglects these fundamental generalizations is not plausible, if there is an incremental parser that can use the more general operations of MGs directly.

2 An incremental, probabilistic, top-down MG parser

Top-down parsers are simple and have some nice properties (Roark and Johnson 1999; Roark 2001). Perhaps most significantly, they are incremental in the sense that each word is attached to a fully connected derivation, allowing the full, left grammatical context to be assessed, for example by processes assessing the relation between compositional meaning and discourse context. Left-corner and bottom-up parsers do not have this property. But MG derivations are usually defined bottom-up to avoid indeterminacies, which are equally present for MG derivations and strongly equivalent MCFG derivations. In the MCFG bottom-up derivation of *which wine the queen prefers* above, for example, the last rule to apply is:

$$C(x_2x_1) \rightarrow +whC, -wh(x_1, x_2).$$

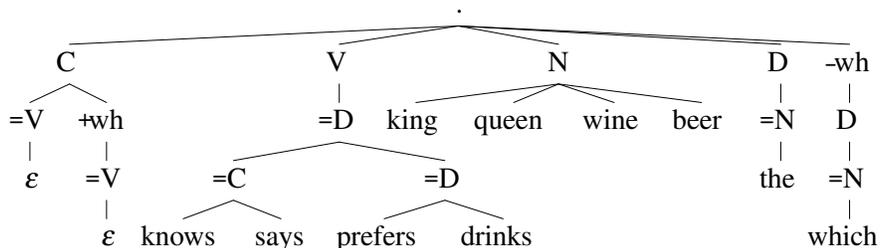
Using this rule top-down, a troubling indeterminacy comes from the fact it requires correctly splitting the complete string to be derived, x_2x_1 , into two parts. At the beginning of a top-down parse, we do not have the complete string, let alone a criterion for splitting it appropriately. Following earlier work by Mainguy (2010) and others, Stabler (2011b) shows how a top-down MCFG parser can avoid this problem by ‘threading’ the input; that is, intuitively, instead of passing the input down from the root of the derivation, we pass it from leaf to leaf. And instead of using a stack of predictions as the standard top-down CFG parser does, predictions are sorted into linear order at every step (so this is a ‘priority queue’, not a stack), expanding the leftmost prediction at each point. Here we show how to adapt those ideas to the particular properties of MGs.

Top-down MG parsing faces another indeterminacy that depends on a particular grammatical assumption: In a binary expansion of a constituent with n movers, there are 2^n different ways to partition the movers between the two children. Does the grammar really allow every moved element to come from any possible c-commanded position? One constraint which would make a big difference here is a version of a specifier island constraint. One version of this constraint, SpIC_{mrg} , says there can be no extraction of any proper subconstituent of a merged specifier. With SpIC_{mrg} , in a binary expansion of a constituent with n movers, at most one of those movers can come from a merged specifier, and only if the movement feature is lexically associated with the head of that specifier. Let's adopt this constraint for the moment (partly because it simplifies the presentation) and then reassess it in §4 below.

A transparent, top-down parsing strategy for $\text{MG}[\text{+SpIC}_{\text{mrg}}]$ grammars can be illustrated with an example. This lexicon derives the example shown in §1:

| | | | | | | | | | |
|-------------------|-------|--------------------|----------------------|------|-------------------|--------------------|--------------------|-----|-------|
| $\varepsilon::=V$ | C | $\text{knows}::=C$ | $=D$ | V | $\text{king}::=N$ | $\text{the}::=N$ | D | | |
| $\varepsilon::=V$ | $+wh$ | C | $\text{says}::=C$ | $=D$ | V | $\text{queen}::=N$ | $\text{which}::=N$ | D | $-wh$ |
| | | | $\text{prefers}::=D$ | $=D$ | V | $\text{wine}::=N$ | | | |
| | | | $\text{drinks}::=D$ | $=D$ | V | $\text{beer}::=N$ | | | |

Notice that many lexical items have the same syntactic features or common suffixes in their feature sequences. Since the top-down parser will check lexical features right-to-left, it is convenient to reduce that redundancy by representing lexicons with a tree:²

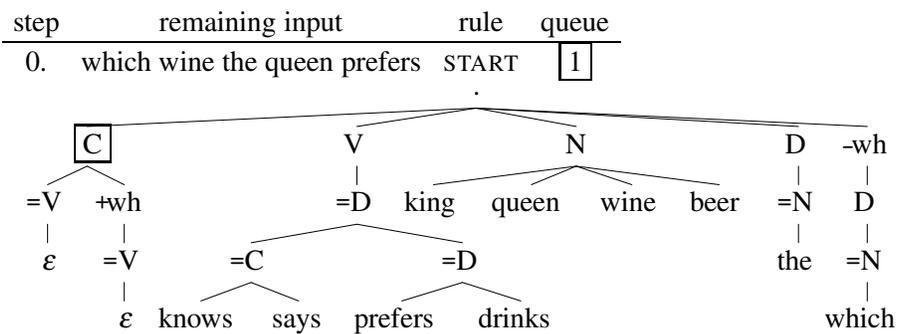


It is easy to see that this graph exactly represents the lexicon listed above: reading the labels of each path from a leaf to the root yields a lexical item in that list. It would be natural to add weights indicating how likely the associations among the lexical properties are taken to be. And notice that here, as in any language model where syntax and semantics correspond appropriately, the tree representation of the lexicon reflects semantic distinctions, Curry-Howard-like. Most importantly for present purposes, with the operation of the parser defined directly on this lexical representation, each next possible step can be determined directly without any search or calculation at all. In effect, if the categories used by the parser are tuples of (pointers to) subtrees of the lexicon, then each parsing step is simply a step down to a descendant in the head category. That is, the queue is the memory structure that lists the predicted but as yet unanalyzed categories, which are identified as nodes in the tree

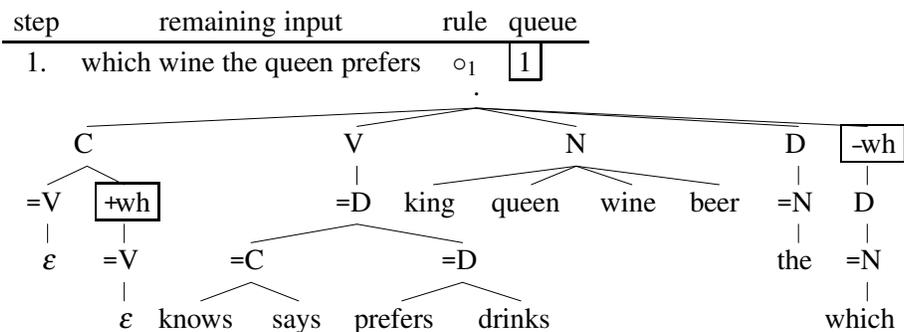
²Though lexicons are here represented as trees, clearly nodes dominating identical subtrees in this graph could be equated, yielding multidominance structures. And a possible way to represent persistent features – if they vary lexically – is with trivial cycles, that is, arcs from a node to itself. We stick to a tree representation here for ease of exposition.

representation of the lexicon. An example will show how this works. Each step expands one node of the derivation tree, so the 12 node derivation tree shown above is parsed in the following 12 steps. The rules indicated at each step (START, SCAN, internal merge subcases \bullet_i and external merge subcases \circ_i) are precisely defined in Appendix C.

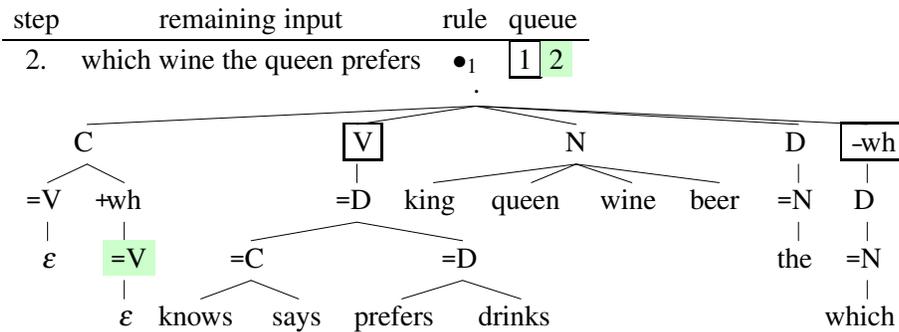
A successful top-down recognition of *which wine the queen prefers* begins with the start category C, which requires a lexical item that has C as its last feature. So, beginning at the root of the lexical tree, we step down to the unique node labeled C, representing the prediction of a completed complementizer phrase. At initialization, this category is the only element in the queue of predictions:



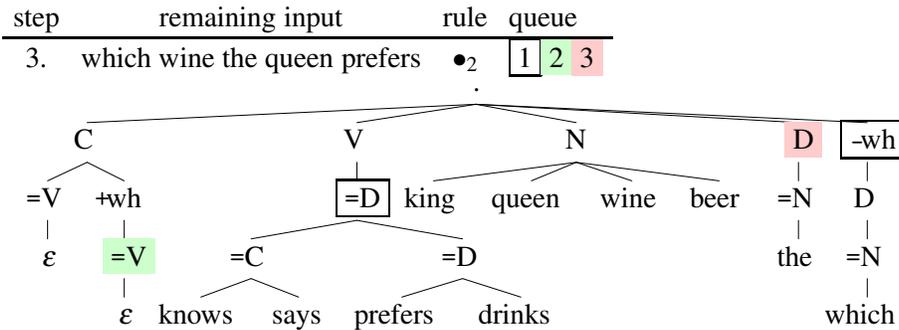
Just below the root of the start category, there are two options, two descendants. The parser constructs both analyses (see the discussion of the beam below), but here, to compute the analysis corresponding to the parse depicted earlier, we descend to +wh. This licenser then triggers the step down from the root of the lexicon to the corresponding licensee -wh subtree. Since this is a mover, these two elements form one constituent; the head +wh has a “mover” -wh:



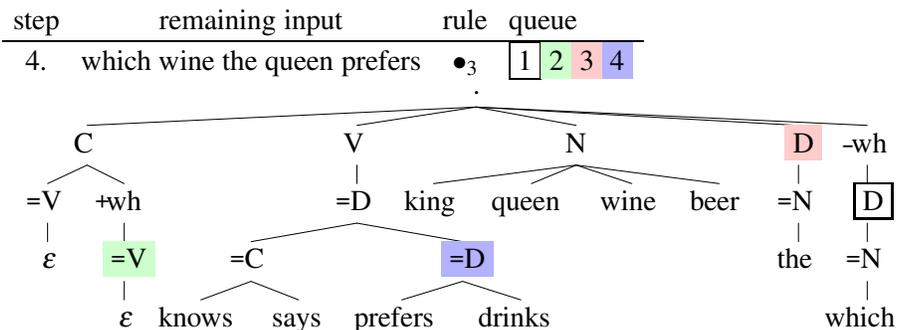
Continuing down towards the lexical leaves, we step down from +wh to find =V, and this selection feature triggers a new path from the lexical root to the V. This new predicted V must be to the left of the selector (since it contains the moved element, which is in spec position), so it goes to the front of the linearly ordered predictions queue, with the selector in second position (indicated here by green shading):



Stepping down from V we find =D which triggers a step from the root to the D subtree, now third in linear order (indicated by red shading):

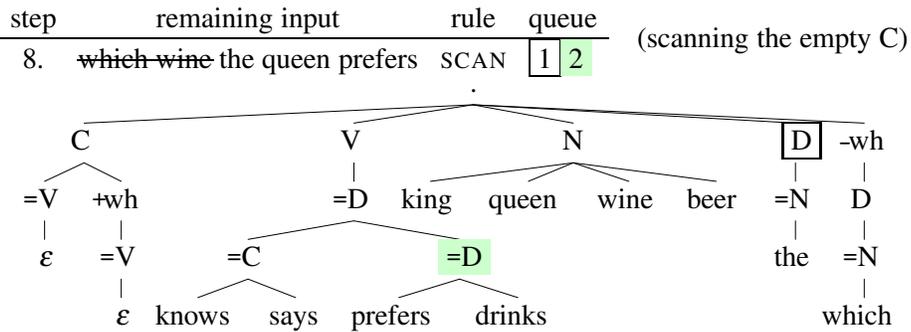
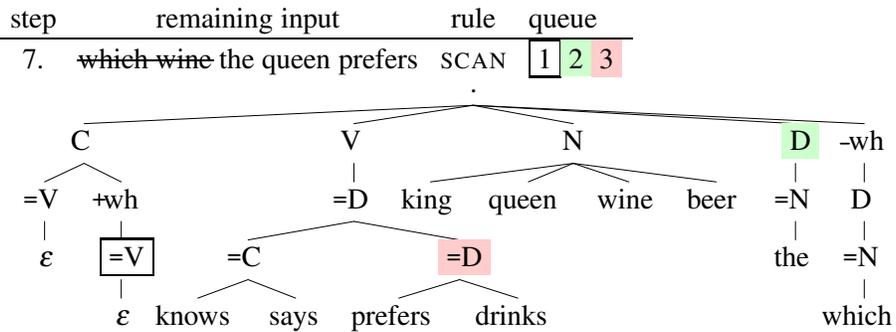
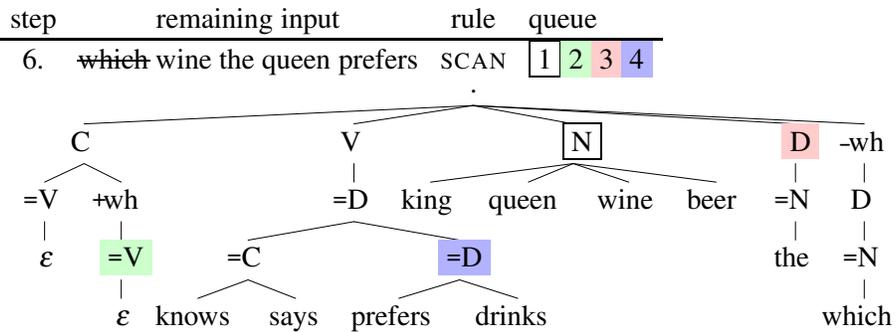
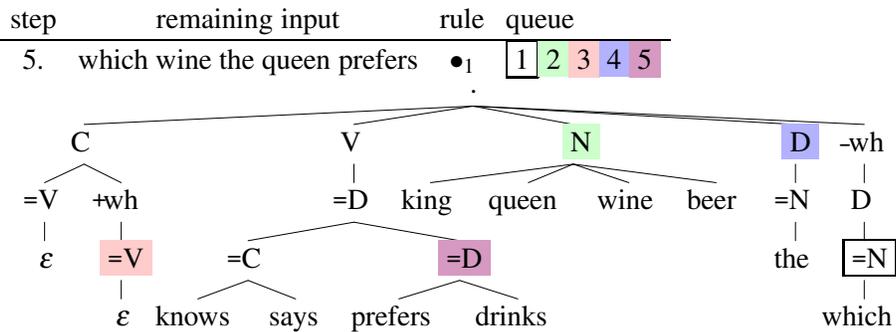


Step 4 is the most interesting step in the derivation. From the previous step, we step down from the ‘external’ =D to find the ‘internal’ argument selector =D. But this time instead of introducing a new D, we find it among the movers, stepping down from -wh and bringing that mover to the front of the queue of predictions:

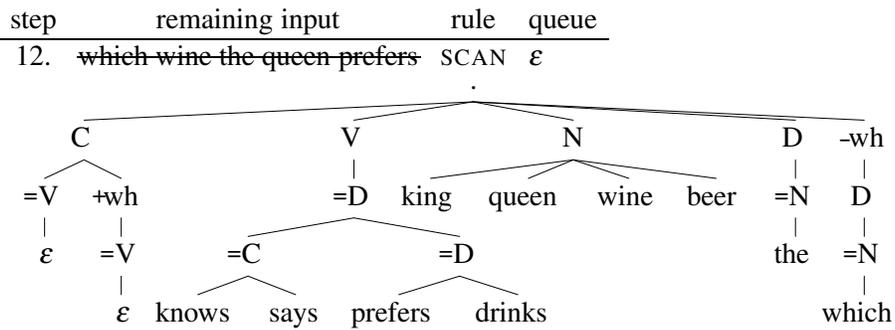
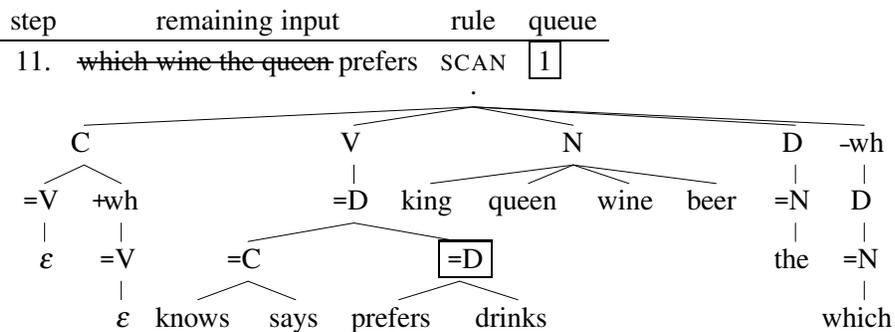
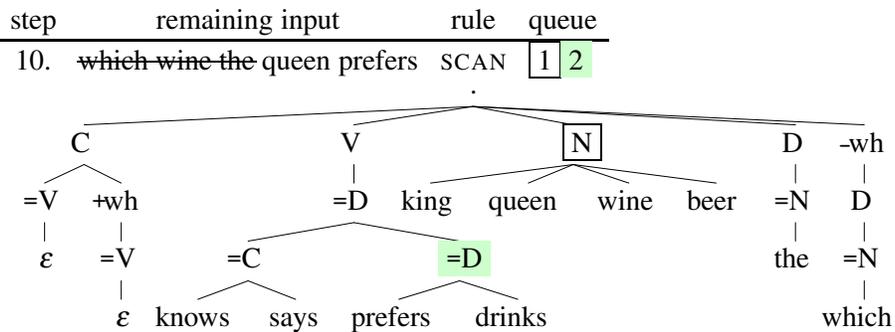
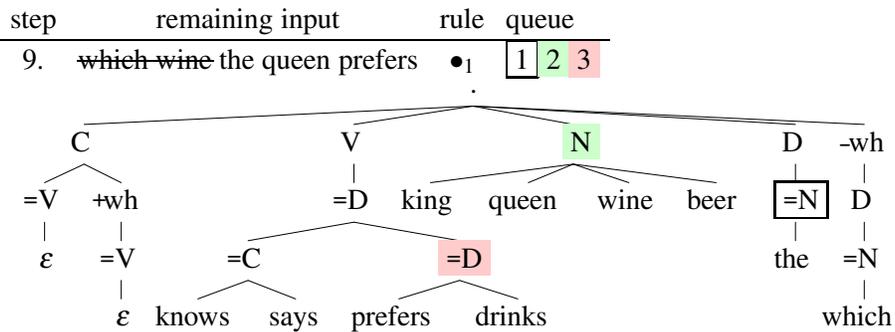


The remaining steps are easy. At each step we simply step down from the leftmost element in our queue, which is the position in the lexical graph indicated by the white box, until all predictions reach the bottom where a (sometimes empty) head is checked against the input with a SCAN rule. Follow the white box (first in queue); at each step, the action is there.

Predicting the NP complement of the wh-DP, we can scan 3 lexical items:

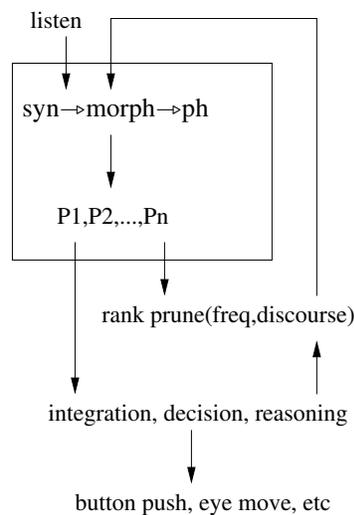


Predicting the NP complement of the DP subject of *prefers*, we can scan 3 more lexical items:



With step 12 the analysis has reached a successful conclusion, since the input is consumed and no predictions remain outstanding in the queue.

A probabilistically ranked beam of alternatives. When the parser faces alternatives, all possible next steps are taken and the results are put into a ranked set of partial derivations, the ‘beam’. Highly improbable analyses may be discarded, and at each point, the most probable analysis in the beam is developed one more step, in all possible ways. There are many ways to specify a probability distribution over partial derivations, each with a given history and context. A careful assessment is beyond the scope of this study. Given the obvious importance of contextual factors though, the architecture we need must be roughly like the following, where the parser’s expansion of the beam of candidate analyses P_1, P_2, \dots, P_n , at each step (sensitive only to lexical features of the heads being combined), is distinguished from probabilistic ranking and pruning (sensitive to frequency effects, discourse, etc.):



In this model, the extra-grammatical ranking and pruning of the queue can consider factors and draw distinctions that are irrelevant to the parsing rules. Obviously, what we understand or say, at each point, depends not only on what the parser can see, but on many factors irrelevant to the definition of possible structures. Compare Roark’s (2001) similar observations about his parser. §3 emphasizes some important properties of these models.

3 Gradient in models of syntactic analysis

Chomskian syntax is criticized by linguists and psychologists for defining discrete structures that provide no account of frequency and gradient effects that are evident in human linguistic judgments and abilities.³ These worries would make it worth considering

³Ferreira (2005:pp.365, 370) says, for example,

... many psycholinguists are disenchanted with generative grammar. One reason is that the Minimalist Program is difficult to adapt to processing models. Another is that generative theories appear to rest on a weak empirical foundation. ... [I]t is now clear that no one interested in human performance can ignore the possible effects of things such as frequency and exposure on ease of processing.

And Crocker and Keller (2006) say:

While the study of gradient grammaticality has a long history in the generative tradition (Chomsky, 1955, 1964), recent approaches such as the Minimalist Program (Chomsky, 1995) do not

whether there is something wrong with our MGs because their definitions of structures are discrete, and whether we have an appropriate notion of gradience in the performance model of the previous section. But these issues also have a more direct relevance to the main point of this paper, namely, that when different phenomena have different sorts of causes – like difference between selection and movement – this is something a psychological model must explain, typically by providing different mechanisms with appropriately different sensitivities.

Grammar is sometimes regarded as a device for partitioning the set of possible strings into grammatical and ungrammatical. But this conception does not fit with mainstream linguistic practice. Linguists often give particular attention to explaining differences in the acceptability of and interpretations of structures that are all marginal. Consider, for example, recent linguistic discussions about island effects (Szabolcsi 2006:and references cited here, for example). This work does not provide clear classifications into grammatical and ungrammatical; rather, it typically explores contrasts in the acceptability of marginal cases, not with the goal of determining the ‘right’ classification of strings into the categories of grammatical/ungrammatical, but to discern which properties are relevant to the contrasts. That is, linguists are interested in the structures of linguistic expressions, regardless of whether those expressions are fully grammatical or not. A structure is a set of relations, and once we have a set of various relations, those can provide much more than a binary classification of strings. The model of §2 generates structures, trees with various properties that could be related to linguistic abilities in many different ways.

From this point of view, we are not free to stipulate, for example, that corpus frequency data are “beyond the scope of generative grammar” (Fanselow, Féry, Schlesewsky, and Vogel 2006:p.4). The questions are: what mechanisms are responsible for corpus frequency effects, and are those mechanisms anything that the model of §2 says anything about? If part of the explanation of corpus frequencies comes from the structure of individual grammars (i.e. from the relation between the lexicon and the structures that lexical items determine, the boxed part of our diagram), then the model must be compatible with that. And if part of the explanation of corpus frequencies comes from the way possible structures are selected (by ranking/pruning), the model must be compatible with that as well. These are not matters for stipulation. There are many ideas about this (Hawkins 2006:for example) but they are rather indirect because they must be mediated by assumptions about how individual grammars are related to the content of the corpus under study. Here let’s seek out evidence that is more directly relevant.

Consider the popular question “to what extent do the mechanisms underlying language comprehension rely on previous linguistic encounters to guide them in resolving the ambiguity they currently face?” (Crocker and Keller 2006:p.229). The model of §2 says something about how the grammar works – that the grammar is mildly context sensitive, etc. – and it has been suggested that the ranking/pruning can be modeled probabilistically. But it has not restricted the source of probabilistic influences on ranking/pruning, and so it has nothing to say about the question of the extent to which those sources rely on previous linguistic encounters. Closer to the relevant issues, though, in the model of §2, previous linguistic encounters influence the possible syntactic structures (the grammar, in the boxed

explicitly allow for gradience as part of the grammar.

part of the diagram above) only to the extent that they determine the syntactic features of lexical items. Are there any challenges to that hypothesis? There is some controversy about this in the literature, but it is very hard to see any reason for thinking that the relation between lexical properties and structure – which is what the grammar defines – calls for any sort of gradience at all. We have other mechanisms in parsing/production models to explain gradient effects. In their survey of gradience, Sorace and Keller (2005:p.1521) conclude “soft constraints are at the syntax-pragmatics interface, while hard ones belong to the core of computational syntax.” The present paper is committed to that view.

A second question directly relevant to the proposal in §2 is whether there is any reason to assume that the ranking/pruning is separate from the structure building operations, as indicated here. Two related, fundamental kinds of evidence support this. First, while the structure building operations are completely determined by lexical features, it is obvious that the factors influencing choice among alternative structures goes beyond those; discourse context and background knowledge about the topics of discussion are clearly relevant. In the second place, note that the syntactic properties of a complex are “compositional”, determined by the properties of the parts. Probabilistic CFGs are compositional in this sense; the probability of a complex formed by a rule is the product of the probabilities of the rule and of the constituents combined. But PCFG estimates of probability are incorrect unless each choice of rule is made independently from all other choices in the derivation, and this is obviously not what happens in human language. In human language, the acceptability of a complex is not a function of the acceptability of its parts, just as in (both subjective and objective) probability theory, the probability $P(p \wedge q)$ is not a function of $P(p)$ and $P(q)$. This point is old and uncontroversial. It is a clear concern even in Grenander’s (1967) pioneering work on probabilistic language models. Charniak (1993:p.83) famously remarks that PCFG models are so poor that even trigram models do better. The factors conditioning what we say are diverse and poorly understood, but it is completely clear that the independent acceptabilities (or relative frequencies) of the parts of a complex do not determine the acceptability (or relative frequency) of the complex. The broad influences on acceptability and plausibility completely cross-cut the much narrower, compositional determinants of syntactic properties. It is right for the model in §2 to separate them, and one expects development of this kind of model to require many similar steps to a much finer articulation of what influences what.⁴

4 Comparisons and extensions

SpIC_{mrg.} Even though the proposed parsing strategy can be applied to an infinite class of grammars, it is easy to see that particular assumptions about linguistic structure, particular choices of grammar will have an impact on parser performance. In this paper, we see this happen with the Specifier Island Constraint (SpIC). As usually formulated, SpIC makes specifiers into islands, prohibiting the extraction of any proper part of a specifier. Since movement is always to specifier position, this provides a kind of “freezing” constraint. That

⁴One standard way to improve a PCFG (or PMCFG) is to break each rule into many ‘lexicalized’ instances (Eisner and Satta 1999; Eisner 2000; Nederhof and Satta 2000) that can have different probabilities. Obviously, this step serves exactly to introduce many distinctions into categories that are irrelevant to the possible structures, for the purpose of defining a choice among those possibilities that depends on other things.

is, SpIC blocks extractions from a constituent after it has moved, because the constituent is then in specifier position. But very many (perhaps even most) linguists think freezing is too restrictive (Sauerland 1999; Koopman and Szabolcsi 2000; Collins 2005; Abels 2007; and others), so it is important that we have not adopted a version of SpIC that enforces freezing.

MG[+SpIC] and variants have been studied in a series of papers.⁵ Recently, Kobele and Michaelis (2011) shows that blocking extraction from specifiers formed by merge reduces expressive power, while prohibiting only extractions from inside specifiers formed by move does not. So with the addition of SpIC_{mrg}, MGs are no longer weakly equivalent to MCFGs, but this addition significantly simplifies the top-down recognizer because, roughly speaking, in top-down MG[+SpIC_{mrg}] parsing, movers that do not simply land in specifier position are deterministically passed to the complement.

SpIC_{mrg} is obviously unlike “left branch” conditions which block or restrict extractions of full specifiers, and so it of course allows Ross’s (1967, §4.3.2.5) examples of left branch extraction from verbal complements in Russian:

Čuju ty čitaješ knigu?
whose you are-reading book
‘whose book are you reading?’

SpIC_{mrg} blocks only subject ‘subextraction’, extraction of a proper subconstituent of a phrase that is second-merged. Subject subextraction seems to be blocked, or at least restricted, across languages. den Besten (1985:§3.2) observes that the so-called Germanic *was-für* split is fine for objects but degraded for subjects (cf. also Corver 1990, 2006):

Was hast du in Italien für Museen besucht?
What have you in Italy for museums visited?
‘What sort of museums did you visit in Italy?’

*Was haben für Leute deine Mutter besucht?
what have for people your mother visited?
‘What sort of people have visited your mother?’

And in a recent study, Chomsky (2008) observes that one English instance of apparent subextraction, fine for objects but not for subjects, is acceptable for the subjects of passives and unaccusatives, as predicted by SpIC_{mrg} if those subjects are underlyingly complements:

it was the CAR (not the TRUCK) of which [they found the (driver, picture)]
it was the CAR (not the TRUCK) of which [the (driver, picture) was found]
*it was the CAR (not the TRUCK) of which [the (driver, picture) caused a scandal]

Accounts of such restrictions include the Subject Condition of Chomsky (1973), the CED of Huang (1982), and more recent PIC-based accounts (Chomsky 2008; Müller 2010). But there are some other puzzling cases of apparent subject subextraction. Wexler and Culicover (1980:p.183) note that while subject subextraction is usually impossible in English, we have examples like this:

[Some people t_i] greeted me [from Philadelphia]_i

⁵Stabler 1999; Michaelis 2004, 2005; Kobele and Michaelis 2005; Gärtner and Michaelis 2007; Kanazawa, Michaelis, Salvati, and Yoshinaka 2011.

Szabolcsi (1983) observes subextraction of possessors from postverbal subjects in Hungarian:

Ki- nek_i alsz-ik [a t_i vendég-e- $\emptyset - \emptyset$]?
who-DAT sleep-3sg the guest-POSS-3SG-NOM
'Whose guest sleeps?'

And Bašić (2004:p.30) observes constructions like this in Serbo-Croatian:

Ovaj nam je predsednik obećao veće plate
this us-cl aux-cl president promised higher salaries
'This president promised us higher salaries'

There are many proposals for relaxing or replacing $SpIC_{mrg}$. It is easy to remove this assumption from the parsing rules, but this brief discussion of the issues is included here to indicate how grammatical assumptions bear directly on aspects of parsing that matter for basic complexity assessments. In parsing models like the one in §2, allowing all extractions from specifiers imposes additional demands on memory, and so it is intriguing that Jurka, Nakao, and Omaki (2011) have shown that subject *was-für* split generally reduces acceptability, even in speakers most inclined to accept it. This could be a consequence of the memory burden imposed on the parser, but careful assessment of these matters must be left for future work.

Complexity filters. Koopman observes that heads often restrict the complexity of their specifiers, both in the morphology and in the syntax (Koopman and Szabolcsi 2000; Koopman 2002, 2012). And Kobele (2011) notes that these restrictions could be enforced directly in MGs by coding them into the category system. But these constraints could also be factored away from the category system, possibly with a significant gain in succinctness from, intuitively, capturing the generalizations succinctly. In performance too, the constraints on specifiers could be treated separately, imposed by a 'transducer' running concurrently with the parser, checking each step of each analysis.

Head movement, agreement, distributed morphology. Similarly, Kobele (2011) observes that operations assumed in distributed morphology (Marantz 1997; Halle and Marantz 1993) could be applied to MGs without affecting the class of definable languages (cf. also Graf 2011). Approaches to head movement and agreement could be compiled into the MG category system too. But again a factored account may be more succinct and better evidenced. It is not yet clear which accounts will fare best.

5 Evidence

MG-based parsers have i/o identical MCFG-based models that compute isomorphic derivations. Let's briefly survey evidence that could distinguish these rather similar models.

Succinctness. Chomsky (1956:p.119) says "I do not know whether English is. . . literally beyond the bounds of phrase structure description. . . When we turn to the question of the complexity of description. . . , however, we find that there are ample grounds for the conclusion that this theory of linguistic structure is fundamentally inadequate." This paper makes the same kind of comparison. MGs have strongly equivalent MCFGs, but those MCFGs must simply list all instances of generalizations that the MG formalism captures in its lexicon. This is explicit in frameworks powerful enough to formulate both perspectives

precisely, as for example in the work of Salvati (2011:p.98), where the MG quantification over categories with common features is explicit.

With two equivalent models that differ in size but are similar in runtime complexity, it is natural to prefer the smaller, simpler model. In the present case, the larger, strongly equivalent MCFGs could even be too large for feasible representation. But the relevance of this consideration for psychological and neurophysiological models depends on unclear assumptions about the representational capacity of the human sentence recognizer, how many types of movements human grammars allow, and other factors. So we cannot be satisfied with succinctness arguments alone. We expect corroborating evidence from other sources.

Other evidence. In MCFGs, every rule is a phrase structure expansion, and there is no mechanism for grouping categories according to their common properties. This is evident in transparent implementations MCFGs (e.g. those described in Stabler 2012, upon which §2 is based). But in transparent representations of MGs, like the one in §2, heads of phrases that move are distinguished in the lexicon; heads of phrases that select a common element (or a common sequence of elements) are treated alike, and so on. If something like the parser of §2 were neurophysiologically implemented, we might even find neurally distributed lexical representations in which the activation of movers is spatially distinct in memory. More generally, evidence for MG-like grouping and processing of constituents could take the form of differential influences on the recognition/production of movement relations and differential neurophysiological/psychological correlates of movement recognition/production.⁶ A review of the literature turns up an abundance of results of both sorts, supporting MG-like models.

Are there any supported psycholinguistic generalizations that quantify over movement constructions, or over local selection constructions? That there is something special about wh-movement and other A-bar dependencies has been long recognized. Many measures show that object relatives are more difficult than subject relatives, and more generally that online complexity increases with the distance spanned by the dependencies recognized. Many of these results are summarized in, for example, Gibson (1998), Yoshida (2006), and Grillo (2008).⁷ Hofmeister and Sag (2010) argue that island effects may be due to a coalition of processing pressures, but that is an issue of detail compared to the very basic point considered here, and so it is no surprise that their alternative account, even without islands, quantifies across different categories that are similar with respect to long various distance dependencies. In acquisition of movement relations, it is also common to find patterns specific to types of movement, cutting across the particular categories of the elements involved (Friedmann and Lavi 2006; van Kampen 1997:and refs. there).

⁶Note that the questions of interest here are not the difficult questions of theoretical detail usually discussed in the literature: we are not concerned here with the question of whether movement leaves ‘traces’ in particular surface positions, whether movement is best defined with feature passing mechanisms, or any such thing. The question here is just whether the recognition of discontinuous licensing conditions is distinguished from other sorts of analysis, as a transparent MG implementation would predict.

⁷Bartek, Lewis, Vasishth, and Smith (2011) argue that locality effects have been overestimated, but acknowledge that they are robustly evidenced in at least some experimental paradigms.

6 A methodology for language modeling

The incremental model proposed here is motivated by a number of general and supported hypotheses about appropriate ways of factoring explanations of linguistic phenomena. Hypotheses H1 and H2 directly inspire the model of §2, but the whole project is founded on the more general setting of H3-H5:

- H1. As discussed in §2, there are regularities in what lexical heads select, what they are selected by, and how they are licensed by discontinuous dependencies. This is recognized by all mainstream grammatical traditions, and MGs provide just one possible formalization of this idea. But it is for this reason that MGs can be much simpler and more plausible than strongly equivalent MCFGs.
- H2. As mentioned in §3, the acceptability of complexes is not a function of the acceptability of their parts. Nor are the conditioning factors restricted to those properties that determine the applicability of grammatical rules. The probabilistic ranking and pruning introduce extra-grammatical influences.
- H3. As noted in point (iii) of §3, it is assumed that the grammar defines all the syntactic structures that the parser ever constructs: full sentences in some cases, fragments in other cases, built by grammatical mechanisms.
- H4. We have adopted standard assumptions about the relevance and irrelevance of the expressive power of grammars. Expressive power is relevant to the basic requirement of descriptive adequacy that was mentioned in the first sentence of this paper. However, that requirement is very far from sharp for two different kinds of reasons. First, we do not have any direct access to the mechanisms we want to model. Instead, we are trying to find evidence about mechanisms in the noise of everything else that is also happening. Second, our evidence is always drastically incomplete, so even very different models can be compatible with all the evidence seen so far. In these cases, there can be strong reasons for preferring one theory even over weakly and strongly equivalent alternatives. Considerations of this latter sort are arguably the ones that are most substantial in getting us to models that explain the phenomena at hand, as in this paper.
- H5. Could a neurophysiological or artificial implementation of a MG[+SpIC_{mrg}] parser also be an implementation of the corresponding strongly equivalent MCFG model? It does not really matter how that question gets answered, since the question of interest is not what counts as an *implementation*. The real question is: (Q) does the human parser analyze categories in something like the way MGs do? MCFGs do not analyze categories, and so they do not have the means to formulate rules that target whole groups of similar categories. Obviously, it makes sense to detect and take advantage of the structure that is only implicit in MCFGs – and *explaining* the regularities demands it. And then it is not natural to say this is an implementation of the MCFG model any more, with any normal sense of ‘implementation’. But what you call it does not matter; the question we care about is (Q), and the answer to that question is: yes.

In sum, the MG idea defended here (and shared by many grammatical traditions) is that cross-categorial regularities in human languages (roughly speaking: selection, movement, spellout) reflect the operation of corresponding mechanisms in performance. The rules apply according to first features, regardless of the rest of the ‘categorial’ specification given by any following features. And while the structure building rules may be category functional, global influences on preference are not. A wide range of psychological evidence supports these two fundamental properties. These fundamentals represent a consensus position, and it is a non-trivial one. The particular incremental MG parsing model proposed in §2 has these properties, and is otherwise distinguished by its remarkable simplicity, its adequacy for a large class of mildly context sensitive languages, and its close, rather well-understood ties with a number of prominent, critically assessed proposals in theoretical and computational syntax.

Acknowledgments

Thanks to Thomas Graf, Ed Keenan, Hilda Koopman, Colin Phillips, TopiCS editors and reviewers for helpful suggestions.

References

- Abels, Klaus. 2007. Towards a restrictive theory of (remnant) movement: Improper movement, remnant movement, and a linear asymmetry. *Linguistic Variation Yearbook 2007* 7:53–120.
- Abels, Klaus. 2012. The Italian left periphery: A view from locality. *Linguistic Inquiry* Forthcoming.
- Abels, Klaus, and Ad Neeleman. 2012. Linear asymmetries and the LCA. *Syntax* 12:25–74.
- Adger, David. 2010. A minimalist theory of feature structure. In *Features: Perspectives on a key notion in linguistics*, ed. A. Kibort and G. Corbett, 185–218. Oxford: Oxford University Press.
- Bartek, Brian, Richard L. Lewis, Shravan Vasishth, and Mason R. Smith. 2011. In search of on-line locality effects in sentence comprehension. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 37:1178–1198.
- Bašić, Monnika. 2004. Nominal subextraction and the structure of NPs in Serbian and English. Doctoral Dissertation, Universitetet i Tromsø.
- den Besten, Hans. 1985. The ergative hypothesis and free word order in Dutch and German. In *Studies in German grammar*, ed. Jindřich Toman, 23–64. Dordrecht: Foris.
- Bobaljik, Jonathan D. 2011. *Universals in comparative morphology: Suppletion, superlatives and the structure of words*. Cambridge, Massachusetts: MIT Press. Forthcoming.
- Brody, Michael. 2000. Mirror theory: syntactic representation in perfect syntax. *Linguistic Inquiry* 31:29–56.

-
- Caha, Pavel. 2009. The nanosyntax of case. Doctoral Dissertation, University of Tromsø.
- Charniak, Eugene. 1993. *Statistical language learning*. Cambridge, Massachusetts: MIT Press.
- Chomsky, Noam. 1955. *The logical structure of linguistic theory*. NY: Plenum. 1955 typescript published, in part, in 1975.
- Chomsky, Noam. 1956. Three models for the description of language. *IRE Transactions on Information Theory* IT-2:113–124.
- Chomsky, Noam. 1964. Degrees of grammaticalness. In *The structure of language: Readings in the philosophy of language*, ed. J.A. Fodor and J.J. Katz, 384–389. Englewood Cliffs, New Jersey: Prentice-Hall.
- Chomsky, Noam. 1973. Conditions on transformations. In *A Festschrift for Morris Halle*, ed. Stephen R. Anderson and Paul Kiparsky. NY: Holt, Rinehard & Winston.
- Chomsky, Noam. 1995. *The minimalist program*. Cambridge, Massachusetts: MIT Press.
- Chomsky, Noam. 2008. On phases. In *Foundational issues in linguistic theory: Essays in honor of Jean-Roger Vergnaud*, ed. Robert Freidin, Carlos P. Otero, and Maria Luisa Zubizarreta. Cambridge, Massachusetts: MIT Press.
- Chomsky, Noam. 2012. Problems of projection. *Lingua* forthcoming.
- Collins, Chris. 2005. A smuggling approach to passive in English. *Syntax* 8:81–120.
- Corver, Norbert. 1990. *The syntax of left branch extraction*. Doctoral Dissertation, Katholieke Universiteit Brabant.
- Corver, Norbert. 2006. Subextraction. In *The blackwell companion to syntax, volume iv*, ed. Martin Everaert and Henk van Riemsdijk, 566–600. Oxford: Blackwell Publishing.
- Crocker, Matthew W., and Frank Keller. 2006. Probabilistic grammars as models of gradience in language processing. In *Gradience in grammar: Generative perspectives*, ed. Gisbert Fanselow, Caroline Féry, Matthias Schlesewsky, and Ralf Vogel, 227–245. Oxford: Oxford University Press.
- Eisner, Jason. 2000. Bilexical grammars and their cubic-time parsing algorithms. In *Advances in probabilistic and other parsing technologies*, ed. Harry Bunt and Anton Nijholt, 29–62. Kluwer.
- Eisner, Jason, and Giorgio Satta. 1999. Efficient parsing for bilexical context-free grammars and head automaton grammars. In *Proceedings of the 37th Annual Meeting, ACL'99*, 457–464. Association for Computational Linguistics.
- Fanselow, Gisbert, Caroline Féry, Matthias Schlesewsky, and Ralf Vogel. 2006. Gradience in grammar. In *Gradience in grammar: Generative perspectives*, ed. Gisbert Fanselow, Caroline Féry, Matthias Schlesewsky, and Ralf Vogel, 1–21. Oxford: Oxford University Press.

-
- Ferreira, Fernanda. 2005. Psycholinguistics, formal grammars, and cognitive science. *Linguistic Review* 22:365–380.
- Friedmann, Naama, and Hedva Lavi. 2006. On the order of acquisition of A-movement, wh-movement and V-C movement. In *Language acquisition and development*, ed. A. Belletti, E. Bennati, C. Chesì, E. Di Domenico, and I. Ferrari. Cambridge: Cambridge Scholars Press.
- 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. NY: Mouton de Gruyter.
- Gibson, Edward. 1998. Linguistic complexity: Locality of syntactic dependencies. *Cognition* 68:1–76.
- Graf, Thomas. 2011. Closure properties of minimalist derivation tree languages. In *Logical Aspects of Computational Linguistics, LACL'11*.
- Grenander, Ulf. 1967. Syntax-controlled probabilities. Technical report, Brown University, Providence, Rhode Island.
- Grillo, Antonino. 2008. Generalized minimality: Syntactic underspecification in Broca's aphasia. Doctoral Dissertation, Universiteit Utrecht, The Netherlands.
- Groenink, Annius. 1995. Literal movement grammars. In *Proceedings of the 7th Meeting of the European Association for Computational Linguistics*, 90–97.
- Halle, Morris, and Alec Marantz. 1993. Distributed morphology and the pieces of inflection. In *The View from Building 20*, ed. Kenneth Hale and Samuel Jay Keyser, 111–176. MIT Press.
- Harkema, Henk. 2001a. A characterization of minimalist languages. In *Logical Aspects of Computational Linguistics*, ed. P. de Groote, G. Morrill, and C. Retoré, LNCS 2099, 193–211. NY: Springer.
- Harkema, Henk. 2001b. Parsing minimalist languages. Doctoral Dissertation, University of California, Los Angeles.
- Hawkins, John A. 2006. Gradedness as relative efficiency in the processing of syntax and semantics. In *Gradience in grammar: Generative perspectives*, ed. Gisbert Fanselow, Caroline Féry, Matthias Schlesewsky, and Ralf Vogel, 207–226. Oxford: Oxford University Press.
- Hofmeister, Philip, and Ivan A. Sag. 2010. Cognitive constraints and island effects. *Language* 86:366–415.
- Huang, Cheng-Teh James. 1982. Logical relations in Chinese and the theory of grammar. Doctoral Dissertation, Massachusetts Institute of Technology, Cambridge, Massachusetts.

-
- Jurka, Johannes, Chizuro Nakao, and Akira Omaki. 2011. It's not the end of the CED as we know it. In *Proceedings of the 28th West Coast Conference on Formal Linguistics*, ed. M. B. Wasburn, K. McKinney-Bock, E. Varis, A. Sawyer, and B. Tomaszewicz. Somerville, Massachusetts: Cascadilla.
- Kanazawa, Makoto, Jens Michaelis, Sylvain Salvati, and Ryo Yoshinaka. 2011. Well-nestedness properly subsumes strict derivational minimalism. In *Logical Aspects of Computational Linguistics, LACL'11*, ed. S. Pogodalla and J.-P. Prost, LNCS/LNAI Volume 6736. Berlin: Springer.
- Kayne, Richard S. 1994. *The antisymmetry of syntax*. Cambridge, Massachusetts: MIT Press.
- Kobele, Gregory M. 2011. Minimalist tree languages are closed under intersection with recognizable tree languages. In *Logical Aspects of Computational Linguistics, LACL'11*.
- Kobele, Gregory M., and Jens Michaelis. 2005. Two type 0 variants of minimalist grammars. In *Proceedings of the 10th conference on Formal Grammar and the 9th Meeting on Mathematics of Language, FGMOL05*.
- Kobele, Gregory M., and Jens Michaelis. 2011. Disentangling notions of specifier impenetrability. In *The mathematics of language*, ed. M. Kanazawa, A. Kornai, M. Kracht, and H. Seki, LNCS/LNAI Vol. 6878, 126–142. Berlin: Springer.
- Kobele, Gregory M., Christian Retoré, and Sylvain Salvati. 2007. An automata-theoretic approach to minimalism. In *Model Theoretic Syntax at 10. ESSLLI'07 Workshop Proceedings*, ed. James Rogers and Stephan Kepser.
- Koopman, Hilda. 2002. Derivations and complexity filters. In *Dimensions of movement: From features to remnants*, ed. Artemis Alexiadou, Elena Anagnostopoulou, Sjef Barbiers, and Hans-Martin Gärtner. Philadelphia: John Benjamins.
- Koopman, Hilda. 2012. Recursion restrictions: Where grammars count. UCLA. Forthcoming.
- Koopman, Hilda, and Anna Szabolcsi. 2000. *Verbal Complexes*. Cambridge, Massachusetts: MIT Press.
- Mainguy, Thomas. 2010. A probabilistic top-down parser for minimalist grammars. [Http://arxiv.org/abs/1010.1826v1](http://arxiv.org/abs/1010.1826v1).
- Marantz, Alec. 1997. No escape from syntax: Don't try morphological analysis in the privacy of your own lexicon. In *Proceedings of the 21st Annual Penn Linguistics Colloquium*, 201–225. University of Pennsylvania.
- Michaelis, Jens. 1998. Derivational minimalism is mildly context-sensitive. In *Proceedings, Logical Aspects of Computational Linguistics, LACL'98*, 179–198. NY: Springer.
- Michaelis, Jens. 2001. Transforming linear context free rewriting systems into minimalist grammars. In *Logical Aspects of Computational Linguistics*, ed. Philippe de Groote,

-
- Glyn Morrill, and Christian Retoré, *Lecture Notes in Artificial Intelligence*, No. 2099, 228–244. NY: Springer.
- Michaelis, Jens. 2004. Observations on strict derivational minimalism. *Electronic Notes in Theoretical Computer Science* 53:192–209. Proceedings of the joint meeting of the 6th Conference on Formal Grammar and the 7th Conference on Mathematics of Language (FGMOL '01).
- Michaelis, Jens. 2005. An additional observation on strict derivational minimalism. In *Proceedings of the 10th conference on Formal Grammar and the 9th Meeting on Mathematics of Language*, FGMOL05, ed. J. Rogers. Stanford: CSLI.
- Michaelis, Jens, Uwe Mönnich, and Frank Morawietz. 2000. Derivational minimalism in two regular and logical steps. In *Proceedings of the 5th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+5)*, 163–170.
- Mönnich, Uwe. 1998. TAGs M-constructed. In *TAG+ Workshop*, Institute for Research in Cognitive Science, University of Pennsylvania.
- Mönnich, Uwe. 2010. Well-nested tree languages and attributed tree transducers. In *The 10th International Conference on Tree Adjoining Grammars and Related Formalisms TAG+10*.
- Morawietz, Frank. 2003. *Two step approaches to natural language formalisms*. Berlin: de Gruyter.
- Müller, Geroen. 2010. On deriving CED effects from the PIC. *Linguistic Inquiry* 41:35–82.
- Nederhof, Mark-Jan, and Giorgio Satta. 2000. Left-to-right parsing and bilexical context-free grammars. In *Proceedings of ANLP-NAACL 2000*.
- Pereira, Fernando C. N. 1981. Extraposition grammars. *American Journal of Computational Linguistics* 7:243–256.
- Pollard, Carl. 1984. *Generalized phrase structure grammars, head grammars and natural language*. Doctoral Dissertation, Stanford University.
- Roark, Brian. 2001. Probabilistic top-down parsing and language modeling. *Computational Linguistics* 27:249–276.
- Roark, Brian, and Mark Johnson. 1999. Efficient probabilistic top-down and left-corner parsing. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, 421–428.
- Rogers, James. 2003. wMSO theories as grammar formalisms. *Theoretical Computer Science* 293:291–320.
- Ross, John R. 1967. *Constraints on variables in syntax*. Doctoral Dissertation, Massachusetts Institute of Technology.

-
- Salvati, Sylvain. 2011. Minimalist grammars in the light of logic. In *Logic and grammar*, ed. Sylvain Pogodalla, Myriam Quatrini, and Christian Retoré, number 6700 in *Lecture Notes in Computer Science*. Berlin: Springer.
- Sauerland, Uli. 1999. Erasability and scrambling. *Syntax* 2:161–188.
- Seki, Hiroyuki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science* 88:191–229.
- Sorace, A., and F. Keller. 2005. Gradience in linguistic data. *Lingua* 115:1497–1524.
- Sportiche, Dominique. 1998. *Partitions and atoms of clause structure : Subjects, agreement, case and clitics*. NY: Routledge.
- Stabler, Edward P. 1999. Remnant movement and complexity. In *Constraints and resources in natural language syntax and semantics*, ed. Gosse Bouma, Erhard Hinrichs, Geert-Jan Kruijff, and Dick Oehrle, 299–326. Stanford, California: CSLI Publications.
- Stabler, Edward P. 2011a. Computational perspectives on minimalism. In *Oxford handbook of linguistic minimalism*, ed. Cedric Boeckx, 617–641. Oxford: Oxford University Press.
- Stabler, Edward P. 2011b. Top-down recognizers for MCFGs and MGs. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics (CMCL), 49th Annual Meeting of the Association for Computational Linguistics*, ed. Frank Keller and David Reitter.
- Stabler, Edward P., and Edward L. Keenan. 2003. Structural similarity. *Theoretical Computer Science* 293:345–363.
- Szabolcsi, Anna. 1983. The possessor that ran away from home. *The Linguistic Review* 3:89–102.
- Szabolcsi, Anna. 2006. Strong vs. weak islands. In *The Blackwell companion to syntax*, volume 4, ed. Martin Everaert and Henk van Riemsdijk, 479–531. Oxford: Blackwell.
- van Kampen, Jacqueline. 1997. First steps in wh-movement. Doctoral Dissertation, University of Utrecht.
- Villemonte de la Clergerie, Éric. 2002. Parsing MCS languages with thread automata. In *Proceedings, 6th International Workshop on Tree Adjoining Grammars and Related Frameworks, TAG+6*.
- Wexler, Kenneth, and Peter W. Culicover. 1980. *Formal principles of language acquisition*. Cambridge, Massachusetts: MIT Press.
- Yoshida, Masaya. 2006. Constraints and mechanisms in long-distance dependency formation. Doctoral Dissertation, University of Maryland.

Appendix A. The relative succinctness of MGs

MGs can be exponentially smaller than their strongly equivalent MCFGs because MCFGs explicitly code each movement possibility into the category system, while MGs can, in effect, quantify over all categories with a given feature. One response that has been proposed to me is that this MCFG size problem may arise just because our MGs are too liberal. When all appropriate restrictions on movement (and selection, etc.) are imposed, for example, perhaps movements will be so restricted that the MCFG explosion will never happen? That this is not a reasonable idea can be shown by establishing that MGs are relatively succinct even when they respect overly strict constraints on movement. In particular, this appendix shows the relative succinctness of MGs that respect the specifier island condition for merged specifiers (SpIC_{mrg}) discussed above. Reflecting on how this result is established will make clear that no plausible constraints on movement will undermine this relative succinctness claim, at least given anything like current conceptions of grammar.

Consider an infinite series of grammars $\text{MG}_0, \text{MG}_1, \dots$ defined as follows. For each $i \in \mathbb{N}$, the lexicon of MG_i contains the lexical items specified in (i-iv) and nothing else, with A the ‘start’ category:

- i. the following lexical item, with $i + 2$ syntactic features:

$$a :: =B +1 +2 \dots +i A$$

- ii. the following lexical item, with 3 syntactic features:

$$b :: =B =CB$$

- iii. the following lexical item, with 1 syntactic feature:

$$d :: B$$

- iv. And for each $1 \leq j \leq i$, this lexical item with 2 syntactic features,

$$c :: C -j.$$

In each MG_i , clause (iv) introduces i lexical items with 2 features each, so each MG_i lexicon has $3i + 6$ features altogether, and for all n , provides $n!$ derivations of $c^n ab^n d$. So the number of features grows linearly with i . MG_2 , for example, has 12 features and allows $2!$ derivations of $ccabbd$, including this one:

allow rules of the form $A \rightarrow X$ where X is any sequence of 0 or more categories and terminal symbols. But a more insightful perspective on the context free languages comes from a logical, model-theoretic perspective, which can be roughly described as follows – cf. Rogers (2003) and references cited there for more rigorous presentations. Regular, finite state grammars define a kind of successor relation, telling us which symbols can occur in a sequence. Context free grammars define regular trees, where a tree is a structure similar to a string but allowing elements to have multiple successors. Regular trees can be recognized by finite state tree acceptors, and a language is context free iff it is the set of strings that are the leaves of a regular tree language. So then, what are the mildly context sensitive languages defined by minimalist grammars? These languages are characterized by two regular steps: we take a regular tree language, and then use a certain kind of regular transduction to rearrange the constituents of the trees. There are various ‘two step’ approaches to mildly context sensitive languages in the literature (Mönnich 1998, 2010; Michaelis, Mönnich, and Morawietz 2000; Morawietz 2003), but the most intuitive one for MGs is given by Kobele, Retoré, and Salvati (2007). On this perspective, we (1) define what counts as a good derivation, and then (2) map it to the desired PF (or LF) structure, which has the elements in their pronounced (or interpreted) positions. It is easy to sketch how these 2 steps work. Both steps can be done by simple, deterministic, finite state devices.

Suppose we are given any MG lexicon – a finite set of associations between phonological and syntactic features. In one standard notation, the lexical item

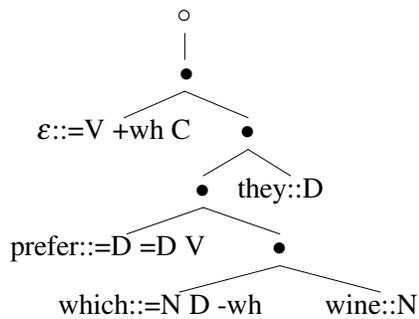
which ::= N D -wh

indicates that an element pronounced *which* selects a Noun phrase complement to form a Determiner phrase which must move to a position that licenses wh elements. Beginning with the lexical items, suppose we just merge things together arbitrarily, merging either two separate things (indicated by the binary symbol \bullet on the left below) or merging a structure with some constituent that the structure contains (indicated by \circ , on the right).

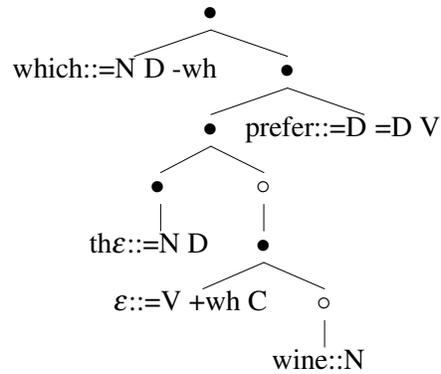


(Compare the suggestion in Chomsky (2012) that merge simply takes any two elements X, Y to form $\{X, Y\}$, with interface constraints determining whether the resulting derivations are good.) Some trees built in this way are good derivations like the one on the left below. Others are not good, like the one on the right below.

good derivation

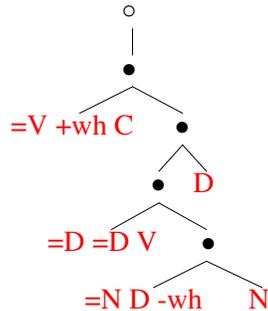


bad derivation

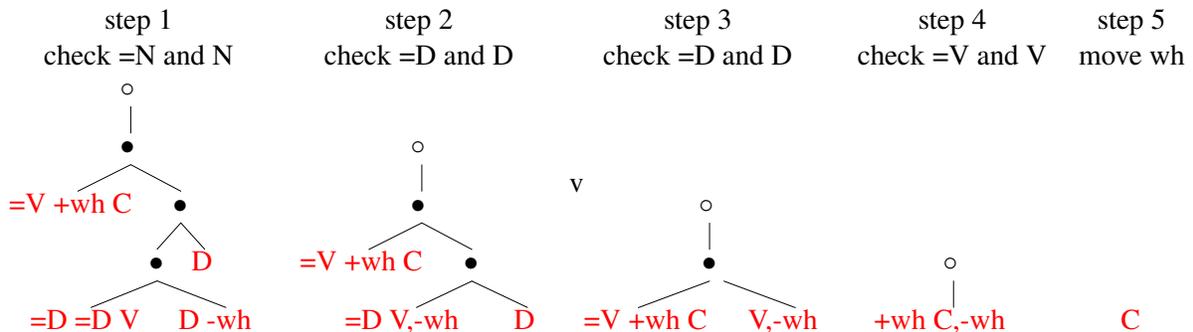


It turns out that checking to see which tree forms a good derivation can be done extremely easily by a finite state device (here, we use a deterministic finite state bottom-up tree acceptor), and the mapping to output representations ('derived trees') is also finite state.

To check whether an MG derivation is good, let's map each leaf, each lexical item to a state which is named by its sequence of features. We indicate acceptor states (the features) in red:

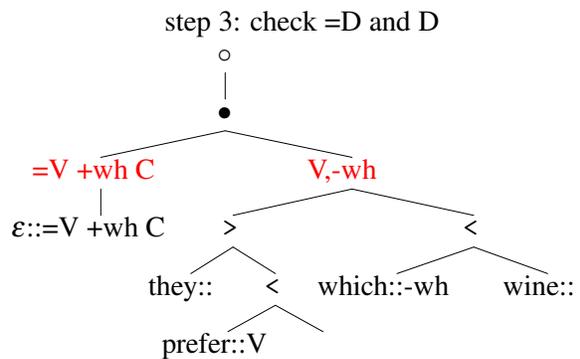
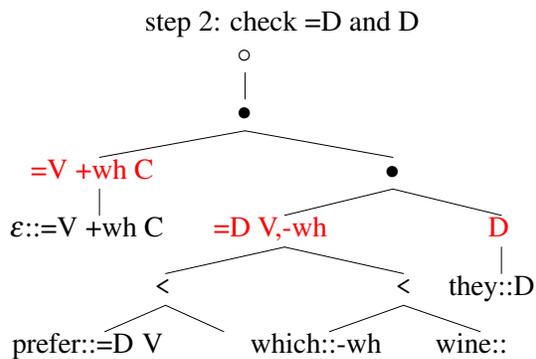
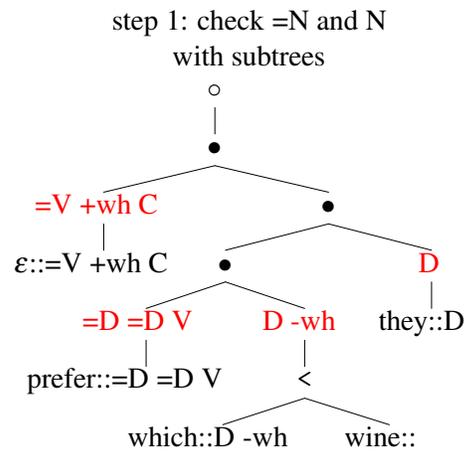
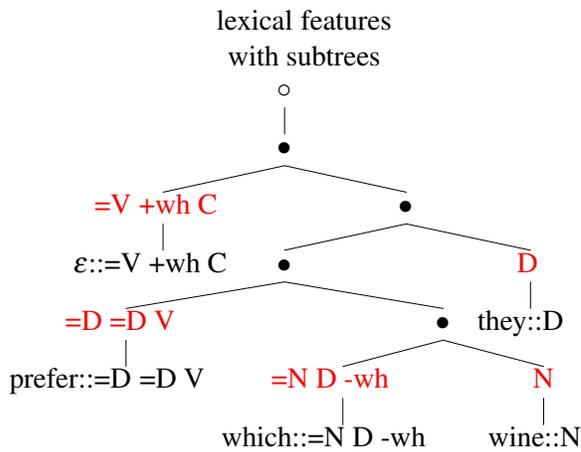


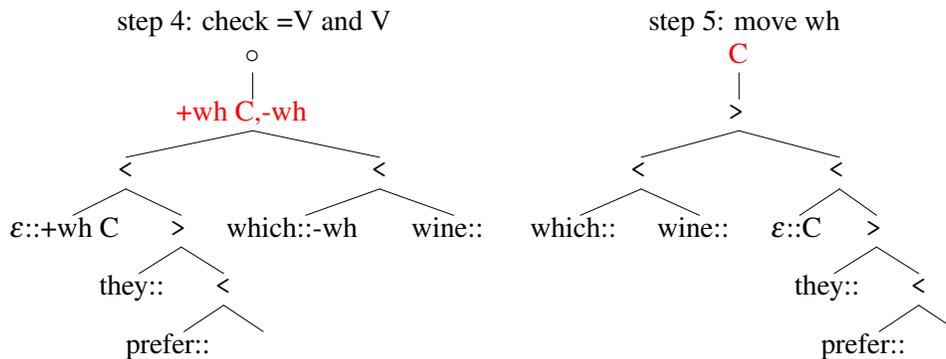
Now instead of lexical items at the leaves, we have just feature sequences, which will be the 'states' of our finite state tree acceptor. The acceptor calculates the states for internal nodes by checking features of its daughters in the standard way for MGs (Stabler 2011a), and the tree is a good derivation if at the end, at the root, we have just the single category feature C (or whatever category one assumes is the 'start' category). This is done in 5 trivial, deterministic steps:



That these steps can be done by with a regular, deterministic bottom-up tree acceptor is pointed out in Kobele et al. (2007), but the essential insight about this structure was already implicit in Michaelis (1998).

Calculating a derived tree, with the moved constituents in their proper positions, adds very little extra effort. Kobele et al. (2007) point out that it can be done by a deterministic, multi bottom-up tree transduction. Traversing the tree bottom-up as before, now we let each state have subtrees, and each step can apply a trivial assembly step to the subtrees of the states it is applying to. So now the first step replaces each leaf not just with a state (= the features), but with a state that keeps the original leaf as a subtree. And then, again, we take 5 steps, checking features as before but this time computing subtrees for each state. And again, we put the states in red:





Step 5 is the derived bare phrase structure representation of *which wine they prefer*. Getting X-bar representations instead is only slightly more difficult. Full specifications and computer implementations of these translations are available at the sites

<http://www.linguistics.ucla.edu/people/stabler/coding.html>
<https://github.com/epstabler/mgtadb/wiki>.

For a more thorough introduction to these grammars and a variety of examples, see Stabler (2011a) and references cited there.

Appendix C. A top-down MG[+SpIC_{mrg}] recognizer

The parser described informally in §2 is here specified precisely. It is a fairly straightforward adaptation of the MCFG recognizer of Stabler (2011a), and so shares many of that parser’s properties. The key innovation is, intuitively, to recognize the derivation tree, not the derived tree with strings in their linear order. The reason this is important is: the language is defined by its derivations. And recognizing the derivation, even when its elements may not be in their surface linear order, is extraordinarily simple when linear order is determined by the operations of the grammar, as it is in both MCFGs and MGs. Parsing MGs or MCFGs top-down, when a constituent is expanded, we know, at that point, the relative linear surface order of its constituents, in a sense that is easy to make precise. In MCFGs, relative linear order is stipulated arbitrarily rule-by-rule, but in standard MGs the order is specifier/head/complement (that is, second-merged/head/first-merged). MGs with alternative spell-out rules are briefly mentioned in §3 (MGs with transductions for morphology, etc.), but here we stick to the Kayne-ian SVO assumption. With linear order specified for each rule, it is easy to index the predictions so that they can be sorted into linear order, putting the leftmost on ‘top’ of the queue – see point (ix) just below. So instead of the stack of predictions used in CF parsing, we have a priority queue of predictions, with the least, leftmost element always first, on top. The analyses in the beam are also sorted, in order of decreasing probability, so that the most probable analysis is first, on top at each step. In the MG parsing defined here, though, instead of stipulated MCFG rules, the lexicon determines all the structural options. The algorithm can be represented in pseudocode as follows, and a number of different implementations of the parser are available from the websites listed just above, at the end of Appendix B.

Given *input*, *lexicon*, and a minimum probability bound *min*, and a function *P* that tells us the probability of each expansion, the recognition algorithm works with a priority queue *dq* of derivations, where each derivation has 3 parts (*i,p,q*), where *i* is the remaining input, *p* is the probability of the parse, and *q* is a priority queue of predicted indexed categories.

```

initialization
  ic := (startCat,  $\emptyset$ ) $\epsilon$            predict start from lex tree, with index  $\epsilon$ 
  p := 1.                               initial probability
  q := insert ic into empty queue        initial queue of predicted categories
  dq := push (i, q, p) into empty queue  initial queue of partial derivations
procedure derive(P, lexicon, min, dq):
  while dq  $\neq$  []  $\wedge$  max dq  $\neq$  ([], [], p)
    (i, q, p) := pop(dq)                pop max probability derivation
    exps := all expansions of (i, q)    apply rules (below), using lexicon
    for d in exps
      if P(d) > min, push d onto dq     only keep steps with probability > min
  if dq = [] then false else true

```

The rules for computing the ‘expansions’ of each predicted category are, in effect, the inverses of the standard bottom-up MG rules Stabler and Keenan (2003), slightly modified for SpIC_{mrq}. To define these rules precisely, the following conventions will be useful. Each rule is an operation on a pair *i, q* where *i* is the input and *q* the queue of predicted categories:

1. Given any sequence of trees *x*, let Σx be the elements of that sequence, in order, whose roots are labeled with vocabulary elements (i.e. leaves of the lexical tree). And let $\bar{\Sigma} x$ be the elements of that sequence, in order, whose roots are labeled with syntactic features (i.e. those roots are internal nodes of the lexical tree).
2. Let *t(x)* be a tree in which the sequence of subtrees immediately dominated by the root is *x*. And let *t[u]* be a tree whose the root immediately dominates subtree *u*. That is, *t[u]* is a tree *t(x)* where *u* occurs in *x*.
3. The notation *t_i* indicates that *t* is associated with index $i \in \mathbb{N}^*$. The result of extending *i* with 0 is *i0*; extending *i* with 1 yields *i1*; and so on. As in Stabler (2011a), two indices are ordered by the standard lexicographic convention for lists, built into OCaml and other languages; two predictions are ordered by their least indices.
4. *t * q* in a premise indicates that the result of removing the least prediction *t* from the current queue is *q*. In a conclusion, *t * q* represents the result of inserting *t* into *q*. This operator associates to the right, so $t * u * q = t * (u * q)$.
5. Similarly for the movers, $-f(x)_i \uplus \mu$ in a premise indicates that the result of removing $-f(x)_i$ from the multiset of current movers is μ . $-f(x)_i \uplus \mu$ in a conclusion represents the value of adding $-f(x)_i$ to the mover multiset μ , only possible if the SMC is respected.
6. Let ℓ be the tree representation of the lexicon. It follows from the structure of the lexical tree that, in any $\ell[t]$, and in any $t[u]$ where the root of *t* is a licensee, the root of *u* is labeled with a category or a licensee, never with a selector or licenser.

With these conventions, the structure building rules can be restated as operations on $input, q$, the remaining input and the priority queue of predictions:

$$\frac{}{input, (C(x), \emptyset)_\varepsilon} \text{ (START) } \ell[C(x)], \text{ for start category } C$$

$$\frac{w * input, (t[w], \emptyset)_i * q}{input, q} \text{ (SCAN)}$$

$$\frac{input, (t[=f(x)], \mu)_i * q}{input, (=f(\Sigma x), \emptyset)_{i0} * (f(y), \mu)_{i1} * q} \text{ (}\bullet_1\text{) } \ell[f(y)] \wedge \Sigma x \neq \varepsilon$$

$$\frac{input, (t[=f(x)], \mu)_i * q}{input, (=f(\bar{\Sigma}x), \mu)_{i1} * (f(y), \emptyset)_{i0} * q} \text{ (}\bullet_2\text{) } \ell[f(y)] \wedge \bar{\Sigma}x \neq \varepsilon$$

$$\frac{input, (t[=f(x)], u[f(y)]_j \uplus \mu)_i * q}{input, (=f(\Sigma x), \emptyset)_i * (f(y), \mu)_j * q} \text{ (}\bullet_3\text{) } \Sigma x \neq \varepsilon$$

$$\frac{input, (t[=f(x)], u[f(y)]_j \uplus \mu)_i * q}{input, (=f(\bar{\Sigma}x), \mu)_i * (f(y), \emptyset)_j * q} \text{ (}\bullet_4\text{) } \bar{\Sigma}x \neq \varepsilon$$

$$\frac{input, (t[+f(x)], \mu)_i * q}{input, (+f(x), -f(y)_{i0} \uplus \mu)_{i1} * q} \text{ (}\circ_1\text{) } \ell[-f(y)]$$

$$\frac{input, (t[+f(x)], u[-f(y)]_j * \mu)_i * q}{input, (+f(x), -f(y)_j \uplus \mu)_i * q} \text{ (}\circ_2\text{)}$$

Some points to note. Each rule removes exactly one element from the queue of predictions, the minimum (i.e. leftmost). The merge rules each insert two new elements into the queue; the move rules insert one element. Four rules fetch a subtree from the lexicon ℓ . The indexing of chains is trivially determined in \bullet_1, \bullet_2 and \circ_1 by the standard spec-head-comp rule that standard MGs implement. The queue is sorted by index, as in Stabler (2011a) and point 3 above (cf. Mainguy, 2010; Villemont de la Clergerie, 2002; Harkema, 2001). To enforce the SpIC_{mrg} , the third merge rule of Stabler and Keenan (2003) is split into two, \bullet_3 and \bullet_4 , for first- and second-merge respectively, so that the second-merged elements, the specifiers, can be treated appropriately. Rules \bullet_2, \bullet_4 do not pass any movers into specifiers, as required by SpIC_{mrg} . Note also that with SpIC_{mrg} , since we do not have the freezing that comes with SpIC, when a complement with movers is merged, no record need be kept of which complement came with which movers, significantly simplifying the inference rules.

Example 1. The 12 node derivation for *which wine the queen prefers*, depicted in §1, is found by the following 12 step recognition sequence. For readability, we put ... in place of subtrees that are not lexical. So to read the following derivation, it is important to refer to the depiction of the lexicon above to see which subtrees are being referred to in each step.

| step | queue | rule |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| 0. | $(C(\dots), \emptyset)_\varepsilon$ | START |
| 1. | $(+wh(\dots), -wh(\dots))_0$ | \circ_1 |
| 2. | $(V(\dots), -wh(\dots))_{011}, (=V(\varepsilon), \emptyset)_{10}$ | \bullet_1 |
| 3. | $(=D(\dots), -wh(\dots))_{0111}, (=V(\varepsilon), \emptyset)_{10}, (D(\dots), \emptyset)_{110}$ | \bullet_2 |
| 4. | $(D(\dots), \emptyset)_0, (=V(\varepsilon), \emptyset)_{10}, (D(\dots), \emptyset)_{110}, (=D[\text{prefers}], \emptyset)_{111}$ | \bullet_3 |
| 5. | $(=N(\text{which}), \emptyset)_{00}, (N[\text{wine}], \emptyset)_{01}, (=V(\varepsilon), \emptyset)_{10}, (D(\dots), \emptyset)_{110}, (=D[\text{prefers}], \emptyset)_{111}$ | \bullet_1 |
| 6. | $(N[\text{wine}], \emptyset)_{01}, (=V(\varepsilon), \emptyset)_{10}, (D(\dots), \emptyset)_{110}, (=D[\text{prefers}], \emptyset)_{111}$ | SCAN |
| 7. | $(=V(\varepsilon), \emptyset)_{10}, (D(\dots), \emptyset)_{110}, (=D[\text{prefers}], \emptyset)_{111}$ | SCAN |
| 8. | $(D(\dots), \emptyset)_{110}, (=D[\text{prefers}], \emptyset)_{111}$ | SCAN |
| 9. | $(=N(\text{the}), \emptyset)_{1100}, (N[\text{queen}], \emptyset)_{1101}, (=D[\text{prefers}], \emptyset)_{111}$ | \bullet_1 |
| 10. | $(N[\text{queen}], \emptyset)_{1101}, (=D[\text{prefers}], \emptyset)_{111}$ | SCAN |
| 11. | $(=D[\text{prefers}], \emptyset)_{111}$ | SCAN |
| 12. | ε | SCAN |

To simplify the presentation, I am not trimming the indices here in the way I did in Stabler (2011a), but obviously that could be done.

Example 2. To see moving specifiers (as allowed by SpIC_{mrg}), here is the recognition of the 13 node derivation of *ccabbd* from MG_2 depicted in Appendix 6. (I omit the lexical tree for reasons of space but recommend, dear reader, that you draw it for yourself.)

| step | queue | rule |
|------|------------------------------------------------------------------------------------------------------------|-------------|
| 0. | $(A(\dots), \emptyset)_\varepsilon$ | START |
| 1. | $(+2(\dots), -2(\dots))_0$ | \circ_1 |
| 2. | $(+1(\dots), -1(\dots))_{010} - 2(\dots)_{011}$ | \circ_1 |
| 3. | $(B(\dots), -1(\dots))_{010} - 2(\dots)_{0111} (=B(a), \emptyset)_{110}$ | \bullet_1 |
| 4. | $(C(c), \emptyset)_0, (=C(\dots), -1(\dots))_{010} (=B(a), \emptyset)_{110}$ | \bullet_4 |
| 5. | $(=C(\dots), -1(\dots))_{010} (=B(a), \emptyset)_{110}$ | SCAN |
| 6. | $(B(\dots), -1(\dots))_{010} (=B(a), \emptyset)_{110} (=B(b), \emptyset)_{1110}$ | \bullet_1 |
| 7. | $(C(c), \emptyset)_{10}, (=B(a), \emptyset)_{110} (=B(b), \emptyset)_{1110} (=C(\dots), \emptyset)_{1111}$ | \bullet_4 |
| 8. | $(=B(a), \emptyset)_{110} (=B(b), \emptyset)_{1110} (=C(\dots), \emptyset)_{1111}$ | SCAN |
| 9. | $(=B(b), \emptyset)_{1110} (=C(\dots), \emptyset)_{1111}$ | SCAN |
| 10. | $(=C(\dots), \emptyset)_{1111}$ | SCAN |
| 11. | $(=B(b), \emptyset)_{11110} (B(d), \emptyset)_{11111}$ | \bullet_1 |
| 12. | $(B(d), \emptyset)_{11111}$ | SCAN |
| 13. | ε | SCAN |

Example 3. To see remnant movement, it is useful to consider artificial examples like this 7 item grammar for the copy language $\{xx|x \in \{a,b\}^*\}$, with start category T, before tackling the less explicit and more complex proposals in the syntax literature:

$$\begin{array}{ll}
 \varepsilon ::= T +r +l T & \varepsilon ::= T -r -l \\
 a ::= A +l T -l & b ::= B +l T -l \\
 a ::= T +r A -r & b ::= T +r B -r
 \end{array}$$

It is immediately obvious from this lexicon that all MG derivations respect SpIC_{mrg} , since no specifier is ever merged; all specifiers are created by movement. The 11 step recognizer sequence that accepts *abab* is easily computed, a goo exercise for the reader. (Or see the websites given in the first paragraph of this appendix for implemented versions of this algorithm, which provide this grammar as one of the examples.)