Comparing methods of tree-construction across mildly context-sensitive formalisms

Tim Hunter
Department of Linguistics
University of California, Los Angeles
timhunter@ucla.edu

Robert Frank
Department of Linguistics
Yale University
bob.frank@yale.edu

A central constraint on minimalist derivations is cyclicity, as instantiated in the extension condition (**Ext**), which permits trees to grow only at (or near) the root. A second key idea concerns the notion of *derivational state*: how much information about the derivational past can the applicability of a certain operation be contingent on? Conditions along the lines of Phase Impenetrability and Shortest Move, for example, can sometimes limit the amount of information that conditions subsequent derivational steps; some versions of such conditions result in a finite bound (**Fin**).

Interestingly, **Ext** and **Fin** define points of variation across mildly context-senstive grammar formalisms, systems that have been proposed to characterize the computational structure of syntactic derivations. On the one hand, Minimalist Grammars (MGs; Stabler, 2011) abide by **Ext**, making use of the standard minimalist operations of Merge and Move, as do Combinatory Categorial Grammars (CCGs; Steedman, 1996), whose derivations involve bottom-up concatenation of lexical items, and the closely related Linear Indexed Grammars (LIGs; Gazdar, 1988). In contrast, the adjoining operation of Tree Adjoining Grammars (TAGs; Joshi and Schabes, 1997) allows trees to grow "in the middle", contra **Ext**. Turning to **Fin**, both MG and TAG operate with bounded derivational state, whereas CCG, with its unboundedly large categories, and LIG, with its stack-valued non-terminals, permit unbounded state. These relationships are summarized in the table in (1).

(1)

	Ext	**¬Ext**
Fin	MG	TAG
¬Fin	LIG, CCG	

Our goal here is to use a pattern of extractions in languages like Bulgarian, which lack the wh-island constraint, to argue against the conjunction of **Ext** and **Fin**, and therefore against the viability of the standard version of the MG formalism. Adequately capturing this pattern requires abandoning either **Ext** (as in TAG) or **Fin** (as in CCG and LIG). The ability of TAG/LIG/CCG to capture this structural pattern is striking, given that they are strictly less powerful than MGs in weak generative capacity.

1 The relevant empirical pattern

Our empirical starting point is Bulgarian sentences like (2) and (3) (Rudin, 1988; Richards, 1997). The significant point here is that these sentences exemplify the abstract structure shown in (4).

(2) Koja kniga$_1$ te popita učitelja [kogo$_2$ [
which book you asked teacher who
ubedi Ivan t_2 [da publikiva t_1]]]
convinced Ivan to publish
"Which book did the teacher ask you who Ivan convinced to publish?"

(3) Koj kontinent$_1$ te popita učitelja [koj$_2$ [
which continent you asked teacher who
t_2 e otkril t_1]]?
has discovered
"Which continent did the teacher ask you who discovered?"

(4) [wh . . . [wh [. . . t . . . t . . .]]]

We assume that the pattern can be extended without bound as indicated in Figure 1, modulo performance complications (Miller and Chomsky 1963, but cf. Joshi et al. 2000). To be clear, the point here is independent of the linear order of the words in the sentence, and relates only to the structural configuration we wish to assign to (2) and (3): (4) describes a tree in which some subtree (corresponding to the innermost square brackets) contains two traces but neither of the corresponding wh-phrases. The key point is also independent of whether the wh-phrases and traces are organized in nested or crossing configurations (Pesetsky, 1982); note that no subscripts are shown in Figure 1.

Proceedings of the Society for Computation in Linguistics (SCiL) 2021, pages 355-358.
Held on-line February 14-19, 2021

2 wh-phrases: [wh . . . [wh [. . . t . . . t . . .]]]

3 wh-phrases: [wh . . . [wh . . . [wh [. . . t . . . t . . . t . . .]]]]

4 wh-phrases: [wh . . . [wh . . . [wh . . . [wh [. . . t . . . t . . . t . . . t . . .]]]]]

⋮

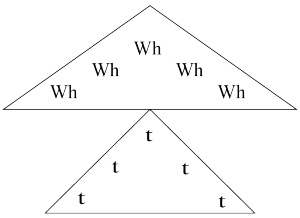


Figure 1

(a) discovered which-continent — unchecked: 1

(b) who discovered which-continent — unchecked: 2

(c) [$_{CP}$ who [$_{TP}$ t discovered which-continent]] — unchecked: 1

(d) teacher ask you [$_{CP}$ who [$_{TP}$ t discovered which-continent] — unchecked: 1

(e) [$_{CP}$ which-continent teacher ask you [$_{CP}$ who [$_{TP}$ t discovered t]]] — unchecked: 0

Figure 2

A natural minimalist-style bottom-up (**Ext**-satisfying) strategy for deriving (3) is indicated informally in Figure 2, where highlighting indicates phrases with unchecked featural requirements. The number of highlighted elements is therefore one thing that contributes to the derivational state that conditions subsequent grammatical operations. The important point is that in step (b), all of the derivation's wh-phrases (here, two) have unchecked featural requirements. Assuming that the relevant pattern extends without bound as indicated in Figure 1, there will be no bound on the number of highlighted elements that a derivation might need to track.

Notice that imposing the Phase Impenetrability Condition (PIC) on a derivation like Figure 2 and requiring successive cyclic movement (with multiple specifiers allowed) does not change the fact that there will be a point at which all of the derivation's wh-phrases will need to be "highlighted". The PIC constrains *where* the highlighted elements are allowed to be (roughly, it says they are not allowed to be too far away from the root of the tree), but does not impose any upper limit on *how many* there are allowed to be at a given time.

Consequently, as the crucial pattern is extended, this familiar **Ext**-satisfying strategy (with or without the PIC) will require unbounded derivational state (i.e. ¬**Fin**).

2 Two sufficient derivational strategies

We will consider two derivational strategies for generating the structural pattern in Figure 1, corresponding to two strategies for generating the familiar $a^n b^n$ string language. First, the strategy implicit in a CFG: a rule like 'S $\rightarrow$ a S b' introduces an a along with its corresponding b, and this a-b pair can be "forgotten" for the purposes of subsequent rewrites because those rewrites occur in the middle of the string. The second strategy is exemplified by a pushdown automaton (PDA): being restricted to reading through a string from one end to the other, a PDA must use its unbounded stack to maintain a count of unmatched occurrences of a, and this count can grow without bound.

TAG, being relevantly similar to a certain kind of context-free tree grammar (Kepser and Rogers, 2011), can generate the pattern in Figure 1 using a form of the first strategy. Elementary trees introduce matched pairs of a wh-phrase and trace, as shown in Figure 3 for (2), like the a-b pairs introduced by the CFG rule 'S $\rightarrow$ a S b'. The fact that the trees are not constrained to grow only at one end (¬**Ext**) allows the tree-building system to operate with a finite amount of memory (**Fin**).

LIG, like its close relative CCG, generates the pattern in Figure 1 using a version of the second strategy. What LIG adds to CFGs is the ability to store information in an unbounded stack at each node of a tree, and this information can be propagated upwards/downwards in order to control non-local dependencies, such as the dependency between a wh-phrase and its trace. While LIG derivations construct trees in the same end-to-end (**Ext**) (whether bottom-up or top-down) manner as CFG derivations, LIG has an unbounded (¬**Fin**) amount

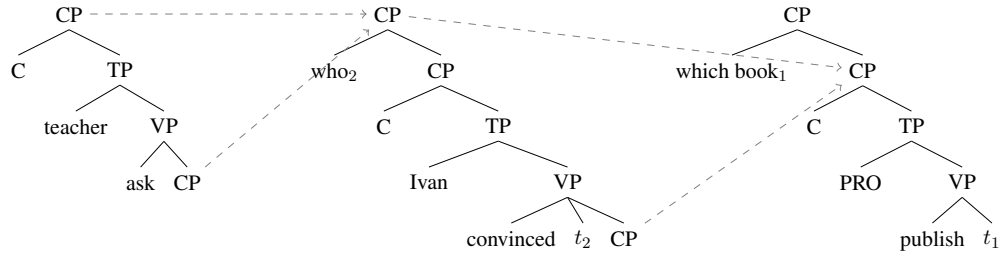


Figure 3

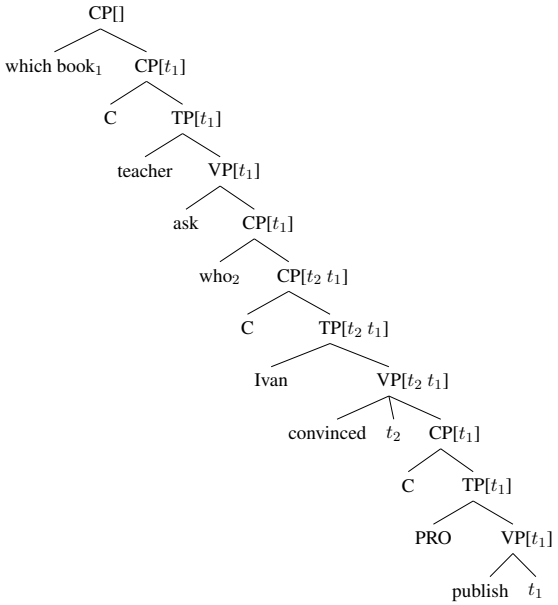


Figure 4

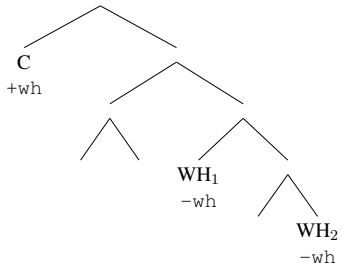
of memory available to ensure that wh-phrases and traces are appropriately paired up — just as a PDA uses its stack to pair as with bs when generating $a^n b^n$. An LIG derivation for (2) is shown in Figure 4, where each node's stack is written in square brackets.

CCG is closely analogous to LIG in the relevant respects: CCG derivations construct trees bottom-up, and allow unboundedly complex categories to be derived through function composition.

3 Minimalist Grammars

A common assumption is that in the abstract configuration in (5), WH_1 prevents WH_2 from moving. Beyond this, however, versions of "Shortest Move" differ. MGs implement a simple conception of Shortest Move, according to which this configuration dooms the derivation, since WH_2 has been prevented from moving to its closest potential attractor. This constraint prevents unchecked featural requirements from accumulating (as they did in Figure 2), and places a bound on the stored information that can condition derivational operations (**Fin**). Therefore after a certain number of traces have been generated (or, after a certain number of wh-phrases have been introduced into their base positions) it becomes impossible to retain the corresponding requirements for wh-phrase surface positions.

(5)



Alternative versions of Shortest Move say that WH_1 can move, which frees up WH_2 for subsequent attractors (e.g. Richards 1997). This involves unbounded storage (¬**Fin**), because there is no limit to how many elements might be waiting to be "freed up" in this manner; the situation remains equivalent in relevant respects to Figure 2. This approach to generating the pattern in Figure 1 is therefore more analogous to LIG than to MG.

MGs can produce a certain variant of the pattern of interest (Gärtner and Michaelis, 2010), namely one where the number of wh-phrases is unbounded but the number of contiguous *clusters* of wh-phrases (i.e. wh-phrases that surface together at the edge of a single clause) remains bounded, by using one of the boundedly many units of derivational state to store each cluster. But in the crucial pattern in Figure 1, where each wh-phrase constitutes its own cluster, there is no bound on the number of clusters. Note also that no operation excorporating a wh-phrase from a cluster is possible: this would require the recovery of information that had been abandoned when the cluster was formed

in order to satisfy **Fin**.

Non-standard variants of MGs that are not restricted in this way include Graf (2012) (which rejects **Ext**) and Gärtner and Michaelis (2005) (which rejects **Fin**). Kobele and Michaelis (2005) showed that the latter has the power of a Turing machine. In contrast, LIG rejects **Fin** in a restricted way: the stack-structured memory generates wh-trace dependencies that are structurally nested, in accordance with the empirically-motivated Path Containment Condition (Pesetsky, 1982). The PDA–CFG equivalence demonstrates a trade-off between stack-based storage (¬**Fin**) and non-edge string-rewriting (a string version of ¬**Ext**); LIG and TAG's differing methods of tree-construction stand in the same relationship (Joshi et al., 1990), so TAG also predicts structurally nested wh-trace dependencies. The position shared by LIG and TAG is the first step on a hierarchy that can be defined either via different relaxations of **Fin** (Weir, 1992) or different relaxations of **Ext** (Rogers, 2003). The different points on this hierarchy will make distinct predictions about the possible patterns of wh-trace configurations in trees, investigation of which we leave for future work.

4 Conclusion

Our main claim here is that a pattern of extractions in languages like Bulgarian — if it extends without bound as indicated in Figure 1, as we have assumed — is incompatible with the conjunction of **Ext** and **Fin**, and therefore incompatible with (standard versions of) the MG formalism. Adequately capturing the relevant pattern requires abandoning either **Ext** (as in TAG) or **Fin** (as in LIG and CCG). In presenting this argument we aim to highlight ways of comparing mildly context-sensitive formalisms that are not based on weak generative capacity.

References

Hans-Martin Gärtner and Jens Michaelis. 2005. A note on the complexity of constraint interaction: Locality conditions and Minimalist Grammars. In *Logical Aspects of Computational Linguistics*, volume 3492 of *LNCS*, pages 114–130, Berlin Heidelberg. Springer.

Hans-Martin Gärtner and Jens Michaelis. 2010. On the Treatment of Multiple-Wh Interrogatives in Minimalist Grammars. In Thomas Hanneforth and Gisbert Fanselow, editors, *Language and Logos*, pages 339–366. Akademie Verlag, Berlin.

Gerald Gazdar. 1988. Applicability of indexed grammars to natural languages. In Uwe Reyle and Christian Rohrer, editors, *Natural Language Parsing and Linguistic Theories*, pages 69–94. Springer.

Thomas Graf. 2012. Movement-generalized Minimalist grammars. In *LACL 2012*, volume 7351 of *Lecture Notes in Computer Science*, pages 58–73.

Aravind K. Joshi, Tilman Becker, and Owen Rambow. 2000. Complexity of scrambling: a new twist to the competence-performance distinction. In Anne Abeillé and Owen Rambow, editors, *Tree Adjoining Grammars*, pages 167–181. CSLI Publications, Stanford, CA.

Aravind K. Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 69–124. Springer, New York, NY.

Aravind K. Joshi, K. Vijay Shanker, and David Weir. 1990. The convergence of mildly context-sensitive grammar formalisms. University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-90-01.

Stephan Kepser and Jim Rogers. 2011. The Equivalence of TAGs and Monadic Linear CF Tree Grammars. *Journal of Logic, Language and Information*, 20(3):361–384.

Gregory M. Kobele and Jens Michaelis. 2005. Two Type-0 Variants of Minimalist Grammars. In James Rogers, editor, *Proceedings of FG-MoL 2005*.

George A. Miller and Noam Chomsky. 1963. Finitary models of language users. In Robert Duncan Luce, Robert R. Bush, and Eugene Galanter, editors, *Handbook of Mathematical Psychology*, volume 2. Wiley and Sons, New York.

David Pesetsky. 1982. *Paths and categories*. Ph.D. thesis, MIT.

Norvin Richards. 1997. *What moves where when in which language?* Ph.D. thesis, MIT.

James Rogers. 2003. Syntactic structures as multi-dimensional trees. *Research on Language and Computation*, 1:265–305.

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

Edward P. Stabler. 2011. Computational perspectives on minimalism. In Cedric Boeckx, editor, *The Oxford Handbook of Linguistic Minimalism*. Oxford University Press, Oxford.

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

David Weir. 1992. A geometric hierarchy beyond context-free languages. *Theoretical Computer Science*, 104(2):235–261.