Some notes on the locally variable complexity of natural language strings

Diego Gabriel Krivochen
University of Reading

Abstract. Proof-theoretic models of grammar are based on the view that an explicit characterization of a language comes in the form of the recursive enumeration of strings in that language (Chomsky & Miller, 1963: 283; Langendoen & Postal, 1984: 18, ff.). That recursive enumeration is carried out by a procedure which strongly generates a set of structural descriptions Σ and weakly generates a set of strings S; a grammar is thus a function that pairs an element of Σ with elements of S (Chomsky, 1965). Structural descriptions are obtained by means of Context-Free phrase structure rules of the general format A → B, and structure is assumed to be *uniform*: binary branching, endocentric trees all the way down. In this work we will analyze instances in which such a rigid conception of phrase structure results descriptively inadequate, and propose a solution for the problem of phrase structure grammars assigning too much or too little structure to natural language strings. We propose that the system can oscillate between levels of computational complexity in local domains (*cycles*), which also yields interesting predictions for locality phenomena.

Keywords: Syntax, Derivations, Mixed Computation.

1 Introduction

One of the basic assumptions in proof-theoretic models of grammar (in the sense of Pullum & Scholz, 2001) is that natural languages *qua* sets of well-formed sentences, and thus structural descriptions for those sentences, fall in a specific level within the Chomsky Hierarchy of formal grammars:

> *Theorem 1: for both grammars and languages, Type 0 ⊇ Type 1 ⊇ Type 2 ⊇ Type 3* (Chomsky, 1959: 143) where Type 0 = unrestricted; Type 1 = Context-Sensitive; Type 2 = Context-Free; Type 3 = regular.

That is an assumption about the strong generative capacity of natural language grammars. Empirically, this view implies *structural uniformity*: the idea that the computational complexity of linguistic dependencies is uniform, and can thus be characterized by a formal system located at a single point in the Hierarchy.

The exact nature of the generative device is not uncontroversial, though. Transformational generative grammar (Chomsky, 1957 and much subsequent work) has argued for a combination of Context-Free phrase structure rules and Context-Sensitive transformations (provided the limitations of pure PSR to generate structural descriptions for all and only grammatical sequences in L; Chomsky, 1957, 1959; Postal, 1964). Generalized / Head-Driven Phrase Structure Grammar (Pollard & Sag, 1994) goes for Context-Free; as does Lexical Functional Grammar (Kaplan & Bresnan, 1982) in its formalization of *c-structure*. In contrast, Joshi (1985) proposes a system in which, by means of an operation *adjunction* in a grammar with preserved links and local distributional constraints, the complexity of the grammar goes up to *mildly context-sensitive* (by virtue of allowing limited crossing dependencies). At the other extreme, Kornai (1985) claims that grammatical sentences in natural languages form regular (finite-state) sets. Crucially, these authors (and many others) assume that a single kind of computational device is sufficient to provide structural descriptions for natural language sentences. The *adequacy* of said structural descriptions, however, is a different problem. It has been long recognized that PSGs assign too rich a structure to P-markers in specific circumstances:

> *a constituent-structure grammar necessarily imposes too rich an analysis on sentences because of features inherent in the way P-markers are defined for such sentences.* (Chomsky, 1963: 298)

Following up on Chomsky (1963) and Postal (1964), Lasnik (2011) acknowledges the problem imposing 'too much structure' on structural descriptions for strings if a uniform 'moving up' in the Chomsky Hierarchy is performed (that is: 'FSGs are inadequate for some substrings, then we proceed to CSGs; these also have limitations, thus we go further up…'):

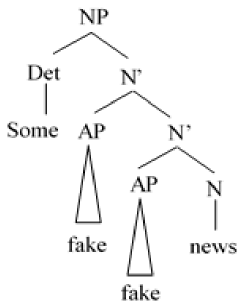
> *In a manner of speaking, what we really want to do is* ***move down*** *the* [Chomsky] *hierarchy. Finite-state Markov processes give flat objects, as they impose no structure. But* ***that is not quite the answer either****.* (Lasnik, 2011: 361)

1.1 Some examples:

Let us illustrate the kind of problem that arises when a single template for structural descriptions is adopted. Consider the following string:

1. Some fake fake news

A PSG of the kind we have been discussing can only assign a single structural description to (1), namely:



Note that, if c-command relations translate into scope at Logical Form (Ladusaw, 1980; May, 1985), the only possible interpretation for (1) is, roughly, (2):

2. Some news which are fake as fake news (i.e., truthful news)

But that is not the only interpretation for (1): a non-scopal interpretation is also available:

3. Some news which sound very fake (i.e., intensifying the meaning of 'fake' via iteration)

The meaning of intensive reduplication is reminiscent of the "rhetorical accent" identified by Stanley Newman in his classic work on English stress (Newman, 1946). The same reading could have been obtained by means of vowel lengthening:

4. Some faaaaaaake news ('Some fa·ke news', in Newman's notation)

It is clear that the structural representation in Fig 1. is not an *adequate* structural description for the string (1), insofar as it is unable to account for both interpretations. There is no scope between both instances of 'fake' in the interpretation in (3), which means that there cannot be a c-command relation between them. The structural description must then be *flat*, but only *locally so*: we still want to keep a scope relation between the quantifier and the noun, which translates into the requirement that the quantifier c-commands the noun. The problem we are facing can be formulated as follows: a Context-Free structural description is locally adequate, but not globally so. It *necessarily* assigns too complex a structure for a substring of (1). Chomsky's solution to the 'too much structure' conundrum (also noted in Chomsky and Miller 1963) was of course to go beyond phrase structure rules and incorporate a transformational component into grammars. There has thus been nothing in the metatheory that leads us to prefer a completely underspecified phrase structure building engine over a transformational model in which $\Sigma \rightarrow F$ rules are supplemented with mapping operations.

2 Towards a solution

Faced with the problem of having a uniform generative system that assigns extra structure to substrings of natural languages, what does a possible solution look like? In past works (Krivochen, 2015, 2016, 2018) we have defended a view of linguistic computation in which the assignment of structural descriptions to strings is a dynamical process, which assigns each substring the *simplest possible structural description which captures semantic dependencies within said substring*. We have referred to such a system as a 'computationally mixed' view of phrase structure. By saying that a system is 'computationally mixed', we mean that it is such that the structural descriptions assigned to substrings in *L(G)*, for *L* a finitely enumerable set of strings and *G* a grammar (which we may identify with finite-state, context-free, or context-sensitive production rules), *need not all be formally identical*. Rather, we maintain, a computationally mixed system assigns a substring *the simplest structural description that captures and represents the formal and semantic relations between syntactic objects in the least entropic way possible (where 'simplest' is to be interpreted in terms of the Chomsky Hierarchy, so that finite-state is computationally simpler than context-free, and the latter is simpler than context-sensitive).* The proposition that we are defending in this paper entails that *structure generation and assignment cannot be independent of semantics*.

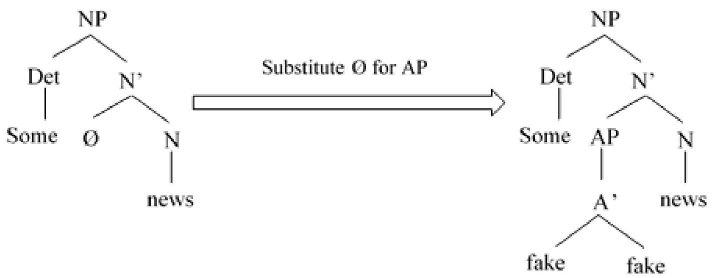
The computational system is based on two processes: *chunking* and *substitution/adjunction*. Chunking implies identifying substrings within a string which are computationally uniform, that is, which display a single level of complexity in their semantic-syntactic dependencies: this identification proceeds 'bottom-up' in the Chomsky Hierarchy, the rationale being that simpler dependencies are to be evaluated first in order to prevent overgeneration of structure (which is precisely the problem we identified with uniform PSGs). In the simple case in (1), for the intensive meaning, we have essentially two chunks:

5. a. Some news
   b. fake fake

The structural dependency assigned to *Some* and *news* needs to capture the fact that the quantifier has scope over the noun, translating into a logical form *some*(*news*). But the relation between both instances of *fake* is of a different kind: it can be expressed in purely finite-state terms without losing information, it is strictly *paratactic* and non-scopal. Locally, then, (1) displays both context-free and finite-state dependencies. But there must be a way to put both chunks together, otherwise, it would be impossible to build a compositional interpretation for the NP. What we need is an operation that can target a rooted tree and insert it in a designated position within another tree, making the former opaque for purposes of operations at the latter. Essentially, we are in presence of a *generalized transformation*, which in the formulation in Chomsky (1995: 189) operates as follows:

<u>*Generalized Transformation:*</u>
i. Target a category α
ii. Add an external Ø to α, thus yielding {Ø, α}
iii. Select category β
iv. Substitute Ø by β, creating {γ, {α, β}}

Roughly speaking, the derivation proceeds as follows:



The Adjective Phrase is a local finite-state unit (in which adjectives establish a purely paratactic relation, without there being scope relations between them), which gets inserted, via *substitution*, into a wider phrasal context displaying Context-Free complexity (the usual state of affairs in PSGs). In this context, the notion of *derivational cycle* can be straightforwardly characterized without making reference to designated non-terminal nodes (S and NP in early generative grammar, C and *v* in more recent incarnations of the theory; see Ross, 1967 and Chomsky, 2000 respectively): *a cycle is a monotonically derived sub-structure*, which displays uniform complexity. The shift in computational dependencies signals the boundaries of cycles, and limits the possibilities of applying further operations (e.g., extraction). The operations applying inside derivational units α and β are usually referred to as 'singulary' transformations; the operation that puts α and β together is a 'generalized' transformation (see Fillmore, 1963). The combination of both defines a *mixed computational system*: singulary transformations over FS sub-units are CF, generalized transformations, if implemented through a TAG, take us up half a notch in the CH, to mild context sensitivity.

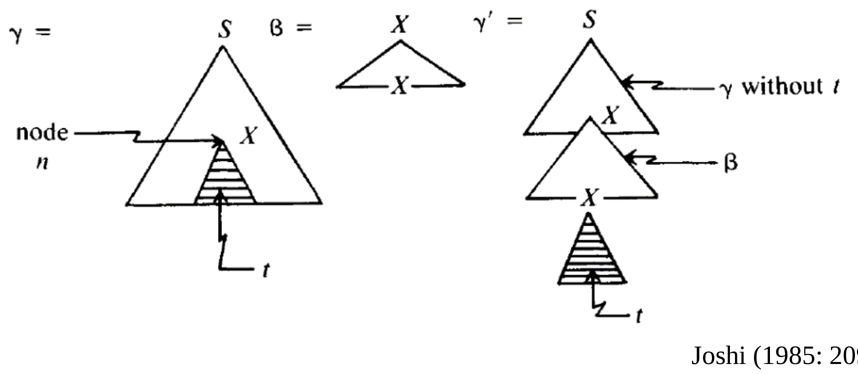
In the theory exposed here, syntactic domains can get 'atomized' and realized in other derivational spaces, thus disrupting the monotonicity of structure building and defining cycles. What we have called 'atomization' is the process by means of which a derivational unit is taken as an internally opaque whole, and inserted in a wider syntactic context by means of an *embedding* transformation (either *adjunction* or *substitution*). Let us consider some examples involving complex predicates:

6. a. Yusei also broke the window into the room and quickly set up his duel disk. (from www.janime.biz/5DS/series054.html) (*Path of Motion*)
   b. The boat crossed the Atlantic to Dover (*Path of Motion*)
   c. John hammered the metal flat (*Transitive Resultative*)

In Krivochen (2018) it was argued that Path of Motion constructions like [NP broke the window into the room] must be analyzed as [NP #bloke the window# into the room], with #broke the window# occupying a terminal position that is usually 'reserved' for a motion verb: cf. *Yusei went into the room*. The same holds for (6 b) [NP #crossed the Atlantic# to Dover], and *modulo* motion, for Resultative Constructions like (6 c) [NP #hammered flat# the metal]. That is, in these particular cases, whole VPs are inserted in a wider syntactic context, by either *substitution* or *adjunction*, thus disrupting the monotonicity of structure building. The operation *adjunction*…

> *… composes an auxiliary tree β with a tree γ. Let γ be a tree with a node labeled X and let β be an auxiliary tree with the root labeled X also. (Note that γ must have, by definition, a node - and only one - labeled X on the frontier.)*

The corresponding diagram is the following:



Joshi (1985: 209)

In Fig. 3, γ is the initial tree which contains a node X that corresponds to the root of the auxiliary tree β. In turn, *t* is a sub-tree in γ dominated by X: when β is adjoined to γ at X, *t* is displaced down, and re-adjoined to a node X in the frontier of β. The result of *adjunction* is γ', with β adjoined to γ and *t* adjoined to a node X in the frontier of β (which is identical to the node *t* was originally dominated by).

The case in which an auxiliary tree targets a node in the frontier of an initial tree (without there being a 'displaced' sub-tree in the IT) is referred to as *substitution* (Joshi & Schabes, 1991: 4); it is only *adjunction* that pushes the strong generative power of the grammar to 'mild-context sensitivity'. *Substitution* pushes the grammar to CF dependencies if manipulating pairs of nonterminals, and stays within finite-state boundaries if manipulating units {terminal, nonterminal} (Greibach, 1965: 44).

When we are dealing with a structure that has been derived non-monotonically (i.e., by joining together separately assembled objects) we can define locality conditions in terms of relations between sub-trees: a singulary transformation in a sub-tree X cannot target an element in a separate sub-tree Y, unless it is ordered *after* a *generalized transformation* relating X and Y, which in turn can be either a *conjoining* or an *embedding* transformation (see Fillmore, 1963); whether either of those allows for extraction of an element from its output is an empirical question yet to be answered. In other words: if adjunction occurs *post-cyclically*, the adjoined tree is rendered opaque for purposes of operations at the target of adjunction, and vice versa. Essentially, this opacity results from having two parallel derivations, as in the model of Uriagereka (2002; 2012): the non-monotonic introduction of structure generates cyclicity effects at the level of the root node in each sub-derivation. Fillmore distinguishes between two kinds of *generalized* transformations: (a) *embedding transformations*, which insert a sequence into another thus generating *hypotactic dependencies*, and (b) *conjoining transformations*, which take A and B and form C containing A and B, generating a *paratactic* dependency between them:

<u>Embedding transformation</u>:
Given P a pre-sentence, A a constant, and a WAY a terminal string,
A → P' in context W…Y

<u>Conjoining transformation</u>: $\left.\begin{matrix} P \\ P' \end{matrix}\right\} \rightarrow P''$

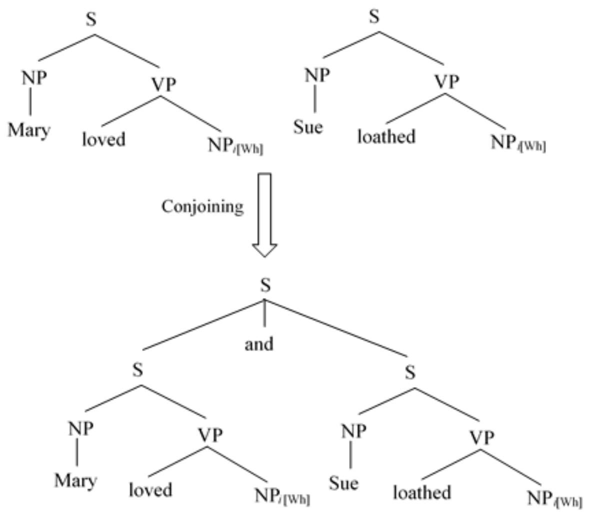
If we diagram these two kinds of transformations, we get:

Conjoining transformation *Embedding transformation*

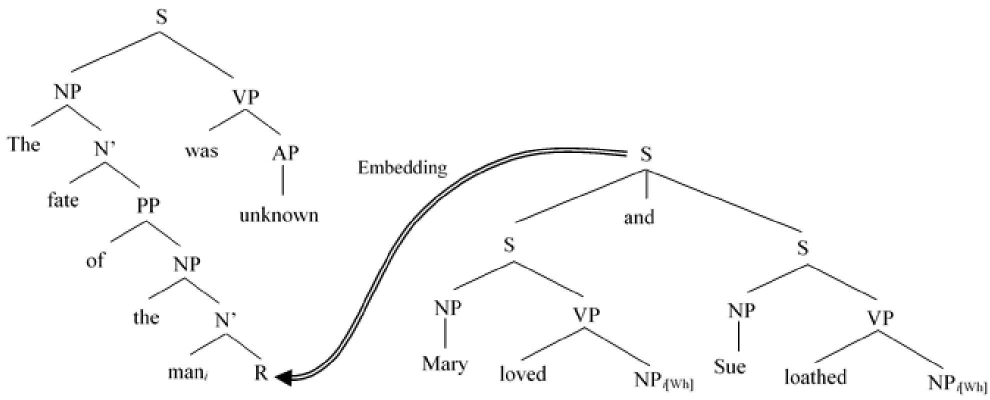
Pre-sentences result from applying all pertinent preliminary simple transformations (including all *copying* and *chopping* transformations). For each symbol in a sequence, an *embedding transformation* specifies the structure of the *pre-sentence* that can be inserted in the structural position occupied by that symbol, as long as contextual conditions (specified by means of variables in the transformation) are met. A system that allows for both conjoining and embedding instantiates the kind of *mixed dependencies* we advocate for automatically: the general formulation of *embedding* transformations is Context-Sensitive, whereas the format of *conjoining* transformations is Context-Free. In a sentence in which both have applied, we will have local units of different computational complexity:

7. The fate of the man who Mary loved and Sue loathed was unknown

Relative clauses are inserted via *embedding*, and symmetric coordination (Schmerling, 1975) is a good example of *conjoining*. The relevant derivation, thus, would go along the following lines (we omit some labelled nodes for simplicity):



The *conjoining* transformation generates a paratactic dependency between the two Ss, which materializes as a conjunction *and*. Now we must get to embedding. The target of Relative Clause adjunction must be an NP with a designated node, call it R (Fillmore, 1963: 223), which will be substituted by the root S that results from *conjoining* under the condition that the adjoined S is *not* interrogative. Thus we get:



The derivation of (7), then, involves both parataxis and hypotaxis, a Context-Free and a Context-Sensitive rule (and a finite-state, flat dependency between *Mary loved NP* and *Sue loathed NP*). Note that (7) could not have been derived monotonically, for we are relating separate preliminary strings (to each of which corresponds an elementary tree) which display dynamically varying computational dependencies (Type 2 for each separate unit *Mary loved NP* and *Sue loathed NP*, Type 3 once these units are symmetrically coordinated, and Type 2 again when the complex phrase marker is flattened and inserted in a node at the frontier of the target IT); the structure does grow all the way, but *not always at the same rate*.

An interesting difference between *embedding* and *conjoining* processes, which seems to establish a relative order between rules is that the result of an *embedding transformation* can be the input for a *singulary transformation*, whereas this is not the case for *conjoining transformations* (unless a further *embedding* transformation has applied to the output of *conjoining*); this is an eminently *empirical* problem (see Fillmore, 1963: 209). In this context, we can formulate the following condition:

8. *Let γ and β be two sub-trees such that γ contains a node X that corresponds to the root of β. A singulary transformation T_S triggered from γ can affect β* ***iff*** *T_S is intrinsically ordered* ***after*** *an embedding transformation that adjoins β to γ at X.*

What singulary transformations *cannot have access to*, we argue, is *elements embedded within β*; only *β* as a whole can be affected by a singulary transformation at γ ordered after *adjunction* of β to γ. We must note that committing ourselves to a model of syntax with multiple cycles does *not* entail committing ourselves to a multi-layered model with several levels of representation and corresponding rules of interpretation. Now consider our proposal about the derivation of *resultative* constructions and complex attributive expressions: in a resultative construction, primary and secondary predication do not coexist in a derivational space *before* adjunction, and after adjunction the Initial Tree is opaque to operations triggered at Auxiliary Tree (the target of adjunction).

Concretely, let us analyze the possibility of having Wh-extraction from an adjoined domain in (6):

9. a. *What did Yusei break into the room?
   b. *What did the boat cross to Dover?
   c. *What did John hammer the metal?

Note that the facts in (9) cannot be accounted for by a theory that commits to uniform motononicity in structure building, because within a derivational current objects should be in principle visible and therefore accessible for purposes of syntactic operations (unless there is some lexically governed rule applying, as in the case of factive islands). But given a cross-derivational constraint as in (8), which appeals to a strong notion of cyclicity, we can provide an explanation. by the ordering constraint formulated above, a singulary transformation like *Wh-movement* would not be able to target a non-root element from the adjoined domain. Thus,

10. *What$_i$ did John burn the toast *e*$_i$?

is adequately excluded, since the extraction site (marked with *e*) is inside an *adjunction-induced island*, a domain that was atomized (computationally 'flattened', in the words of Lasnik & Uriagereka) and inserted in a wider phrasal context. In this context, it is the computationally mixed nature of structural descriptions for natural language strings that yields locality effects. Indeed, reordering transformations (like Wh-movement and Topicalization) *cannot* apply to resultatives targeting either the result or the affected object respectively:

11. a. *What did the river freeze? (answer: 'solid')
    b. *What did Mary shout herself? (answer: 'hoarse')
    c. */%…that / because, the metal, John hammered *t* flat
    d. */%…that / because, the river, *t* froze solid
    e. *…that / because, herself, Mary shouted *t* hoarse

On the other hand, if we are dealing with a monotonic structure (e.g., the uniformly Context-Free *John bought a book* → *What$_i$ did John buy e$_i$?*), extraction should be permitted since we are working within a single derivational space, and structure is preserved in all its complexity. Let us now see the case of garden-variety depictive secondary predication:

12. John drinks his tea hot

There is no reason to propose a multiple derivation structure for (12), thus, (13 a-b) (which feature *Wh-movement* and topicalization in an embedded context) are correctly predicted to be grammatical

13. a. What does John drink *t* hot?
    b. …because his tea, John drinks *t* hot

Changing the perspective on syntactic computation from uniform to locally variable allows us to capture well-established syntactic conditions as properties of the shift in computational complexity and the opacity of adjoined domains. As an example, the impossibility of extracting material from a relative clause, as in **Who$_i$ does the fate of the man that Mary loved t$_i$ and Sue loathed t$_i$ is unknown* (Ross' 1967 Complex NP Constraint) emerges from the dynamics of the computation: relative clauses are non-monotonically introduced in the derivation via substitution, and a transformation triggered at the target of substitution cannot have access to the internal structure of the adjoined domain, which was built separately. It can, however, target the *root* node and, transitively, everything the root dominates: in this sense, 'mixed computation' can straightforwardly account for *strong* islandhood phenomena.

References

Chomsky, Noam 1959. On Certain Formal Properties of Grammars. Information and Control 2, 137-167.

Chomsky, N. 1963. Formal Properties of Grammars. In R. D. Luce, R. R. Bush & E. Galanter (eds.), *Handbook of Mathematical Psychology*, 323–418. New York: John Wiley & Sons

Chomsky, N. 1995. *The Minimalist Program*. Cambridge, Mass.: MIT Press.

Chomsky, N. and Miller, G. 1963. Introduction to the Formal Analysis of Natural Languages. In R. D. Luce, R. R. Bush & E. Galanter (eds.), *Handbook of Mathematical Psychology*, 269–321. New York: John Wiley & Sons.

Fillmore, Charles 1963. The Position of Embedding Transformations in a Grammar. Word 19(2), 208-231.

Greibach, Sheila 1965. A New Normal-Form Theorem for Context-Free Phrase Structure Grammars. Journal of the ACM **12**(1), 42-52.

Joshi, A. 1985. Tree adjoining grammars. In D. Dowty, L. Karttunen & A. Zwicky (eds.) *Natural Language Parsing*, 206-250. Cambridge, Mass.: CUP.

Joshi, A. & Schabes, Y. 1991. Tree-Adjoining Grammars and Lexicalized Grammars. Technical Reports (CIS). Paper 445. http://repository.upenn.edu/cis_reports/445.

Kaplan, R. & Bresnan, J. 1982. Lexical-Functional Grammar: A Formal System for Grammatical Representation. In Bresnan, J. (ed.) *The Mental Representation of Grammatical Relations*, 173-281. Cambridge, Mass.: MIT Press.

Kornai, A. 1985. Natural Languages and the Chomsky Hierarchy. In M. King (ed): Proceedings of the 2nd European Conference of the Association for Computational Linguistics, 1-7.

Krivochen, D. 2015. On Phrase Structure building and Labeling algorithms: towards a non-uniform theory of syntactic structures. The Linguistic Review 32(3). 515-572.

Krivochen, D. 2016. Divide and…conquer? On the limits of algorithmic approaches to syntactic-semantic structure. *Czech and Slovak Linguistc Review* 1/2016. 15-38.

Krivochen, D. 2017. *Aspects of Emergent Cyclicity in Language and Computation*. PhD dissertation, University of Reading.

Ladusaw, W. 1980. *Polarity sensitivity as inherent scope relations*. Bloomington, Indiana: University of Iowa, Indiana University Linguistics Club.

Langendoen, T. & Postal, P. 1984. *The Vastness of Natural Languages*. Oxford: Blackwell.

Lasnik, H. 2011. What Kind of Computing Device is the Human Language Faculty? In A-M. Di Sciullo & C. Boeckx (eds.) *The Biolinguistic Enterprise: New Perspectives on the Evolution and Nature of the Human Language Faculty*, 354-365. Oxford: OUP.

Newman, Stanley 1946. On the stress system of English. Word 2, 171–187.

May, R. 1985. *Logical Form: Its Structure and Derivation*. Cambridge, Mass.: MIT Press.

Pollard, C. & Sag, I. 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press.

Postal, P. 1946. *Constituent Structure*. Bloomington, Indiana: University of Bloomington.

Pullum, G. K. & B. C. Scholz. 2001. On the distinction between model-theoretic and generative-enumerative syntactic frameworks. In P. de Groote, G. Morrill, & C. Retoré (eds.) *Logical Aspects of Computational Linguistics: 4th International Conference* (Lecture Notes in Artificial Intelligence, 2099), 17-43. Berlin: Springer Verlag.

Ross, J. R. 1967. *Constraints on Variables in Syntax*. PhD Thesis, MIT.

Schmerling, S. 1975. Asymmetric Conjunction and rules of Conversation. In P. Cole & J. Morgan (eds.) *Syntax and Semantics, Vol. 3: Speech Acts*, 211-231. New York: Academic Press.