

Phrase Structure and Conjunctivist Interpretation

Tim Hunter*

March 7, 2008

1 Introduction

The research program to which this paper contributes aims to express the formal properties of natural languages in terms of a small number of basic operations. There are two related reasons to attempt this, which have been called Darwin's Problem and the Granularity Mismatch Problem.

Darwin's Problem (Hornstein, to appear; Chomsky, 2005) could also be called "the logical problem of language evolution": given that language seems to have appeared in humans so suddenly (in roughly the last 50,000–100,000 years), why does it appear to show such complex properties? One solution to this problem would be to identify a small number of basic operations (hopefully a single one) that, in combination with other general cognitive faculties, give rise to the characteristic properties of natural languages. In this case we could hypothesise that the addition of these basic operations was the only evolutionary change necessary for language to appear in humans, making the suddenness of this change more plausible.

Darwin's Problem parallels the more familiar logical problem of language acquisition. Solutions to the latter typically involve reducing the various properties of natural languages to a relatively small number of points of variation and enriching the initial state of the language learner, in order to reduce the distance between the start and end states, given the short developmental time in which this distance is covered. Similarly, the solution to the evolutionary problem sought here involves narrowing the gap between the cognitive abilities of linguistic beings (humans) and those of others (non-humans), given the short evolutionary time in which this distance was covered. But the two constraints have the potential to serve as interesting foils for each other, because while solutions to the acquisition problem tend towards relatively complex conceptions of the initial state of the learner (as soon as a property is believed to hold of all natural languages, it is added to the proposed innate knowledge), this initial state is exactly what solutions to the evolution problem must simplify. Indeed, when we consider the state of generative theories in the late 1980s and early 1990s, the culmination of work over a few decades in which the logical problem of language acquisition was used as the main guiding light for choosing amongst otherwise-equal theories, it becomes clear that the rich, modular and heterogeneous but entirely language-specific nature of UG under these theories makes Darwin's Problem appear particularly prickly. A strong way of stating the implications of this problem for linguistic theory would be: in much the same way as we observed, in the 1960s, that it was not scientifically adequate to

*For many of the ideas discussed in this paper I owe, very obviously, many thanks to Norbert Hornstein and Paul Pietroski. In addition I would like to thank John Hale, Bill Idsardi, Greg Kobele, Jeff Lidz, David Poeppel, Ed Stabler, Juan Uriagereka, Amy Weinberg and Alexander Williams for helpful discussion and comments.

provide a catalogue of what a speaker knows of his/her language without any regard for how he/she came to have this knowledge after such a short space of developmental time, we should now observe that it is not scientifically adequate to provide a catalogue of what a newborn infant knows of the possible languages he/she might be required to learn, without any regard for how this information came to be present in such a short space of evolutionary time.

To the extent that we do manage to identify a small number of operations which underlie the characteristic properties of natural languages, as Darwin's Problem suggests that we can and should, this result may go some way towards facilitating interaction between linguistics and neuroscience. Such interaction is currently difficult, according to Poeppel and Embick (2005), because of what they call the Granularity Mismatch Problem. The differing "conceptual granularity" of the primitive concepts in linguistics and neuroscience, they argue, "prevents the formulation of theoretically motivated, biologically grounded, and computationally explicit linking hypotheses that bridge neuroscience and linguistics". They point out a number of neurolinguistic studies which implicitly assume that coarse-grained notions such as "syntax", "semantics" and "phonology" are explicit and monolithic tasks that can be identified with particular brain areas (eg. Broca's area) or particular electrophysiological responses (eg. N400, P600). While associations like "syntax is in Broca's area" and "N400 'is' lexical semantic integration" are "important correlative datapoints, one learns little of explanatory depth about language and little about the brain" from them. For much of the history of generative grammar, the next level of abstraction that neurolinguists might hope to step down to — the level of the components comprising, say, "syntax" — was a very long way further down, concerning detailed notions like c-command, government, binding relations and movement relations. Given that we have independent reason (Darwin's Problem) to think that these detailed notions may emerge from the interaction of a very small number of more basic and ubiquitous operations, there is at least hope that these basic operations may provide "the appropriate level of abstraction to create an interface for linguistics and neurobiology".

Attempting to solve these two problems is, of course, in line with the broad aims of the Minimalist Program as set out in Chomsky (1995). However, in what follows I will argue that the basic operations most likely to form part of solutions to Darwin's Problem and the Granularity Mismatch Problem are not the ones presented in that work and used in much of the work stemming from it: namely, "merge" and "move" (with "move" often decomposed into "copy" and "merge"). Rather, I suggest, there is reason to believe that "merge" itself is separable into two component operations, and that these two component operations are the only primitives necessary. In addition, I aim to illustrate a methodological point: that by taking existing ideas from linguistic theory and expressing them in more explicit, formal terms, we can arrive at a better understanding of the relationships between different theoretical proposals.

In the following section (§2) I introduce some problems posed by taking merge as a primitive operation, as pointed out by Hornstein and Nunes (to appear) and Hornstein (to appear), and informally outline the approach taken in those works to try to resolve this by decomposing merge into two further operations. I then (§3) present a revised version, integrating the "Conjunctivist" theory of semantic composition from Pietroski (2005). This revised version (i) overcomes some technical difficulties concerning the interpretation of Hornstein's novel syntactic structures, (ii) demonstrates that Hornstein's basic syntactic operations and Pietroski's basic semantic operations fit together in an encouragingly natural way, reinforcing the evidence for each, and (iii) suggests an interesting and novel conception of the relationship between movement, copying

and adjunction, discussed in §4. These novel ideas are then explored in relation to other formalised models of grammars, specifically the Bare Grammar framework of Keenan and Stabler (2003), in §5.

2 Decomposing “merge”

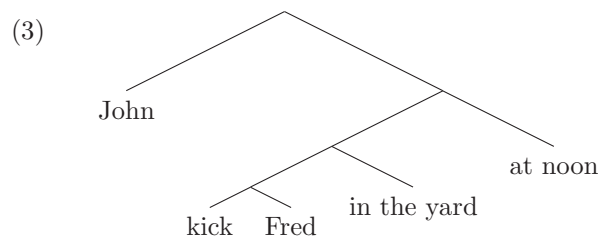
2.1 Problems for “merge”

To introduce the problems facing systems where the conventional merge operation is taken as a primitive, consider the following paradigm:

- (1) John kicked Fred in the yard at noon.
- (2) a. It was *kick Fred* that John did in the yard at noon.
 b. It was *kick Fred in the yard* that John did at noon.
 c. It was *kick Fred at noon* that John did in the yard.
 d. It was *kick Fred in the yard at noon* that John did.
 e. *It was *kick* that John did Fred in the yard at noon.

Italics indicate the part of the sentence in (1) that has been manipulated in each of the variants in (2). The structural description of (1) should therefore indicate that each of these italicised portions is a constituent of a particular type, which is subject to whatever grammatical operation takes place to produce the sentences in (2). Descriptively speaking, the pattern to capture is that if a constituent of this type includes ‘kick’, then (i) it must also include ‘Fred’, and (ii) it may also include zero or more of the modifier phrases ‘in the yard’ and ‘at noon’.

Standard accounts represent the structure of (1) as a tree with the following shape, derived via a sequence of merge operations¹:



The widely-adopted Inclusiveness Condition (Chomsky, 1995) requires that constituent labels can only be lexical items, unadorned with bar-levels in the style of X-Bar Theory (or with anything else). Syntactic operations are understood to be defined on lexical items (only), and when an operation applies to a non-lexical constituent, it does so in virtue of that constituent being equivalent, for syntactic purposes, to the lexical item that is its label. Under the standard assumption that the constituent ‘kick Fred’ should have the label ‘kick’, consider the question of how to label the constituent ‘kick Fred in the yard’. There are two possibilities: either ‘kick’, or something else. If we decide on the label ‘kick’, then the movement of

¹I assume here that any movement (for example, of the subject and object DPs from theta positions to Case positions) is not relevant. I also ignore the tense inflection which is generally agreed to have “hopped” onto the verb from a position which is linearly adjacent to it but not low enough in the tree structure to concern us here.

‘kick Fred’ in (2a) will be a violation of the A-over-A condition². If we decide on any other label, we will be missing the robust generalisation that ‘kick Fred in the yard’ seems to be subject to exactly the same operation as ‘kick Fred’ is (as illustrated in (2b)).

It is worth noting that this problem is not specific to any particular “construction” (eg. clefting), but rather a pervasive problem concerning the very notion of constituency imposed by systems with merge as their only basic operation. The clefting structures above provide one way to probe constituency, but any other test will give the same problematic results, for example, ‘do so’ replacement:

- (4)
- a. John *kicked Fred* in the yard at noon, and Bill did so in the house at one o’clock.
 - b. John *kicked Fred in the yard* at noon, and Bill did so at one o’clock.
 - c. John *kicked Fred* in the yard *at noon*, and Bill did so in the house.
 - d. John *kicked Fred in the yard at noon*, and Bill did so too.
 - e. * John *kicked Fred* in the yard at noon, and Bill did so Peter in the house as one o’clock.

Here we see that ‘did so’ can replace exactly the constituents that can be fronted in (2), further evidence that they should all be subject to the same syntactic operation.

Nominal expressions with multiple adjuncts also display analogous behaviour under ‘one’ replacement:

- (5)
- a. John found the yellow *bottle of water* with the cork, but not the blue one with the cap.
 - b. John found the *yellow bottle of water* with the cork, but not the one with the cap.
 - c. John found the yellow *bottle of water with the cork*, but not the blue one.
 - d. John found the *yellow bottle of water with the cork*, but not the glass.
 - e. * John found the yellow *bottle of water* with the cork, but not the blue one of wine with the cap.

Attempting to label a tree structure for ‘yellow bottle of water with the cork’ will present the same problem mentioned above.

The original data from (2) are used for illustration in what follows, but the matter at hand is constituency, not clefting.

2.2 Hornstein’s proposal

Having observed the difficulty of choosing a label for the constituent ‘kick Fred in the yard’ under standard assumptions, Hornstein (to appear) and Hornstein and Nunes (to appear) suggest resolving the contradiction by not labelling it.³ The operation conventionally known as “merge”, applying to α and β to form a new object with α and β as constituents and either α or β as its label:

(6) $\alpha, \beta \xrightarrow{\text{merge}}$

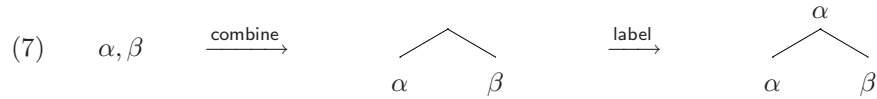
$$\begin{array}{c} \alpha \\ \diagup \quad \diagdown \\ \alpha \quad \beta \end{array}$$

is broken down into two operations: **combine**⁴, which applies to α and β to form a new object with α and β as constituents but no label, and **label**, which adds a label to an object of the type produced by **combine**:

²Hornstein (to appear) shows that the A-over-A constraint can be reduced to a form of relativised minimality, which is the one overarching constraint on syntactic operations in his system. I follow him here in taking this restriction as a given.

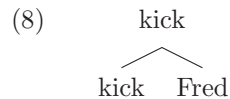
³The idea of using unlabelled structures for adjuncts goes back to Chametzky (1996).

⁴Hornstein calls it “concatenate”.

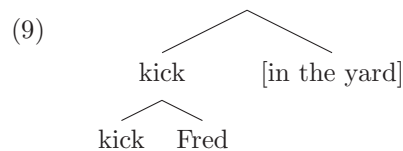


The inputs α and β here can, of course, also be complex objects with α and β as their labels, but crucially only labelled objects can be input to the **combine** operation.

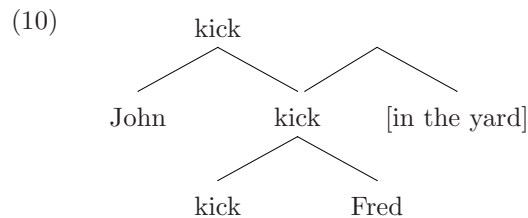
We focus first on how this solves the labelling problem, by considering a sentence like (1) but with only one adjunct: ‘John kicked Fred in the yard’. If we stipulate (for now) that combining with an argument necessarily triggers labelling, whereas combining with an adjunct does not, we find that the pattern in (2) follows naturally. Having combined ‘kick’ with ‘Fred’ and labelled the result ‘kick’ to form:



we combine this object with ‘in the yard’ (the internal structure of which we can ignore), *without* labelling the result, yielding the following:



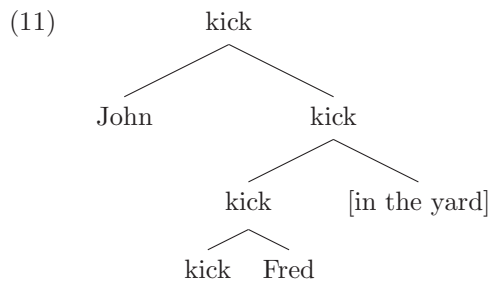
Since this object does not have a label (only parts of it do), it is not equivalent to any lexical item in the eyes of the syntactic/computational component, and therefore is not (as a whole) subject to further applications of **combine**. The object $[\text{kick} \text{ kick Fred}]$, however, is, so the derivation can finish with another application of **combine** and **label** (ignoring tense):



This is the final structure for the sentence ‘John kicked Fred in the yard’, but crucially it allows ‘kick Fred’ to be moved without violating the A-over-A constraint.⁵

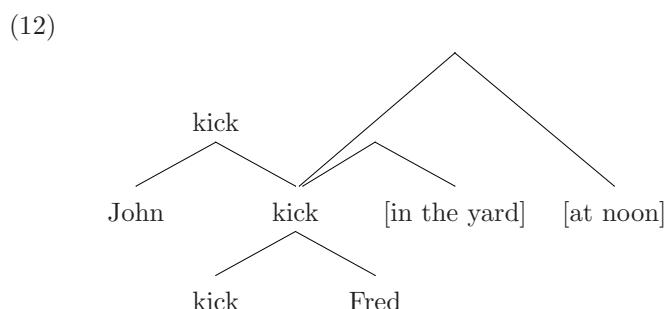
Alternatively, we *may* label the object in (9) before combining with the subject, resulting in the following structure:

⁵Not really: with the current simplified phrase structure, it still violates A-over-A because ‘John kick Fred’ has the label ‘kick’. This will be avoided once a more articulated phrase structure including functional heads like *v* and/or T is introduced, such that the subject is in a separate projection. For now though the point is just to show how movement of ‘kick Fred’ is not blocked by a constituent ‘kick Fred in the yard’ having the label ‘kick’.



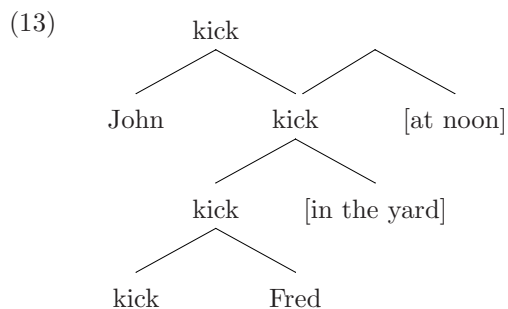
This allows ‘kick Fred in the yard’ to be moved by the same operation as moved ‘kick Fred’ above, in virtue of it also having the label ‘kick’.

For the sentence in (2) with two adjuncts (or the version where only the verb and its argument (‘kick Fred’) are fronted (2a)), we can derive the following structure by combining ‘kick Fred’ twice with adjuncts, before combining it with ‘John’:



Note that in this structure, the two adjuncts are in exactly the same structural relation to the verb they modify, unlike in (3). A constituent can combine with any number of adjuncts like this, but can only have one labelled node immediately dominating it. So we could not, for example, label the parent node of ‘at noon’ if we insist (as we will) on labelling the parent node of ‘John’, because this would cause the middle ‘kick’ node to have two labelled parents.⁶

We can include adjuncts in labelled constituents if necessary, as illustrated in (11), so in cases where some but not all of the adjuncts are targeted, we include only the relevant ones in the labelled “spine” of the tree. For example, to derive (2b) we would label the structure as follows, before applying the final transformation:



From (11) it is also clear that we can build structures where all the adjuncts are included in the constituent to be targeted, as required for (2d).

⁶I keep these restrictions because this conforms with the way derivations are treated in Hornstein (to appear). But in future work I would like to explore the possibility that allowing one node to have two labelled parent nodes could produce “sideways movement”.

The question of *why* labelling applies obligatorily to argument structures but optionally to adjunct structures remains. If we adopt a neo-Davidsonian semantics, and the idea that the default, unmarked semantic correlate of syntactic combination is conjunction (Pietroski, 2005), one possible answer is that we need labels to identify the relation which is to hold between the event variable and the entity described by the argument. So in the sentences above, the entity described by ‘Fred’ is the patient of the kicking event precisely because ‘Fred’ is the sister of the minimal projection of ‘kick’, and the entity described by ‘John’ is the agent of the kicking event precisely because ‘John’ is the daughter of the maximal projection of ‘kick’. These definitions of what are often called the internal and external argument relations are stateable *at the CI interface*, provided that each time an argument is combined, the resulting syntactic object is labelled. They are not affected by the choice between (10) and (11).

Adjuncts like ‘in the yard’, on the other hand, are standardly taken to contribute predicates of the event, so they need only to be combined in order to be interpreted (conjunctively); labelling is not required. The result of combining with an adjunct *may* be labelled if it is required for other reasons; for example, so that it can undergo focus movement or whatever sort of movement applies when we front ‘kick Fred in the yard’ in (11). The result of combining an argument, on the other hand, must *always* be labelled for interpretation to be possible, which explains why ‘kick’ alone cannot be fronted in (2); ‘kick Fred’ is necessarily labelled, so moving only the verb itself will always violate the A-over-A condition.

2.3 Problems to be solved

The basic ideas from this proposal will be adopted throughout the rest of this paper, but there are some problems which require some modifications to the details.

2.3.1 Derivations are not tree-shaped

While it’s obvious how we want structures like (10) to be interpreted, some non-standard form of compositionality will be required to execute this. Interpretation usually happens compositionally over a tree structure, either the derived structure (as in much work within Chomskyan transformational grammar (Heim and Kratzer, 1998)) or the derivation structure (as in much work in categorial grammar (Steedman, 2000) and mathematical studies of compositionality (Hodges, 2001)), but in this case neither of these is a tree structure. Recursively determining a semantic value for each node based on the daughters of that node and their mode of combination, in either structure, will not yield one final semantic value, because there is no single root node.

2.3.2 Adjunction relations are sometimes labelled, sometimes not

Given a structure like (11), we need the adjunct to be interpreted conjunctively, even though the labelling required for the movement has effectively put it into an argument configuration.

3 A revised version

3.1 Progression of a derivation

In Chomsky (1995) a “stage” of a derivation is basically an intermediate set of objects out of which the final syntactic object will be built. A derivation is a sequence of stages, each of which follows legitimately from the previous one, such that the final stage in the sequence contains exactly one syntactic object. To progress from one stage of the derivation to a next valid stage, we choose some subset of items from the current stage, apply a syntactic operation (such as merge) to these items, and replace them with the resulting object:

$$(14) \quad \{\alpha, \beta, \gamma\} \xrightarrow{\text{merge}} \left\{ \begin{array}{c} \alpha \\ \alpha \quad \beta \end{array} \right\}, \gamma \xrightarrow{\text{merge}} \left\{ \begin{array}{c} \alpha \\ \gamma \quad \alpha \\ \alpha \quad \beta \end{array} \right\}$$

We can’t do things quite like this in the new system, because the output of one operation can be the input to two different future operations. For example, in deriving (10), after combining ‘kick Fred’ with ‘in the yard’, ‘kick Fred’ must still be available for further applications of `combine`, so when moving from one stage of a derivation to the next, we allow ourselves to apply operations to any item in the stage, plus any “constituents” of these items. Taking `combine` to be a function over syntactic objects, we have the following:

$$(15) \quad \text{combine} \left(\text{John}, \begin{array}{c} \text{kick} \\ \text{kick} \quad \text{Fred} \end{array} \right) = \begin{array}{c} \text{John} \quad \text{kick} \\ \text{kick} \quad \text{Fred} \end{array}$$

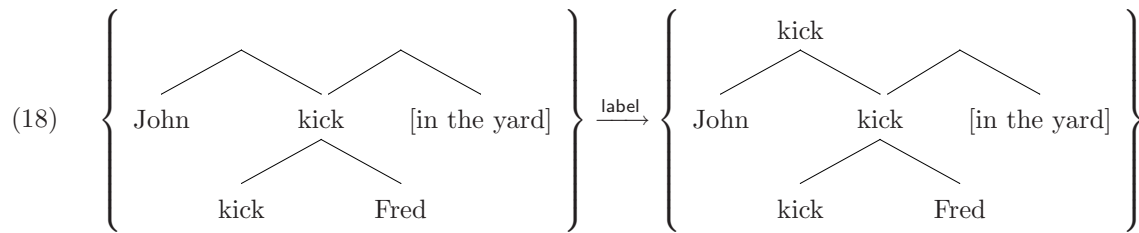
and therefore a valid step in the derivation is:

$$(16) \quad \left\{ \text{John}, \begin{array}{c} \text{kick} \quad [\text{in the yard}] \\ \text{kick} \quad \text{Fred} \end{array} \right\} \xrightarrow{\text{combine}} \left\{ \begin{array}{c} \text{John} \quad \text{kick} \quad [\text{in the yard}] \\ \text{kick} \quad \text{Fred} \end{array} \right\}$$

Similarly, the next step is to apply the label which establishes ‘John’ as the external argument of ‘kick’, which does not operate on the entire structure thus derived but rather only on the output of the previous `combine` operation⁷:

$$(17) \quad \text{label} \left(\begin{array}{c} \text{John} \quad \text{kick} \\ \text{kick} \quad \text{Fred} \end{array} \right) = \begin{array}{c} \text{kick} \\ \text{John} \quad \text{kick} \\ \text{kick} \quad \text{Fred} \end{array}$$

⁷The `label` operation is presented in (17) as if it were a function of a single argument. This is not accurate because the label of either of the immediate constituents may project, so the input does not uniquely determine the output. We return to this issue in §5.1.



We therefore desire a notion of “constituent” such that the object labelled here is a “constituent” of the entire structure.

With this view of the way derivations must proceed, in order to accommodate the phrase structures described above, we now turn to look more closely at the structure-building operations themselves.

3.2 The combination operation

In a sentence such as (19) with a large number of adjuncts, we would like each adjunct to be in an identical structural relation to the verb it modifies:

(19) John ran quickly [in a t-shirt] [on Monday] [from here] [to there] [because of the parrot].

Since the number of adjuncts which can be adjoined without “making the structure any deeper” is unbounded, we need to allow for an unbounded number of adjuncts all “equally close” to the verb.⁸ Strict concatenation is not adequate for this because the verb could be immediately adjacent to at most two adjuncts using that operation. Concatenation, as usually formulated, is also not commutative, whereas the operation *combine* above is.

Let us represent the result of combining two objects α and β with simply the (unordered) pair $\{\alpha, \beta\}$. If we apply the *combine* operation six times, in order to combine the verb ‘ran’ with each of the six adjuncts in (19), progressing through the derivation as shown below (ignoring for now the internal structure of the phrasal adjuncts), we end up with six unordered pairs:

- (20) a. {John, ran, quickly, [in a t-shirt], [on Monday], [from here], [to there], [because of the parrot]}
- b. {John, {ran, quickly}, [in a t-shirt], [on Monday], [from here], [to there], [because of the parrot]}
- c. {John, {ran, quickly}, {ran, [in a t-shirt]}, [on Monday], [from here], [to there], [because of the parrot]}
- d. ...
- e. {John, {ran, quickly}, {ran, [in a t-shirt]}, {ran, [on Monday]}, {ran, [from here]}, {ran, [to there]}, {ran, [because of the parrot]}}

This notation looks unnatural but we require something like this to indicate clearly that ‘ran’ bears the same structural relation to each of the adjuncts; this would be difficult in a tree diagram.

Each of the six unordered pairs “says” that the lexical item ‘ran’ has entered into a relation with a particular other syntactic object. Frequently one syntactic object participating in a relation with two other syntactic objects is known as a “movement” configuration, requiring a separate “copy” to be produced; but until any of these extra notions are shown to be necessary, I will assume they can be dispensed with, and we will just say that a single instance can appear in two sets (why not?) and leave it at that.

⁸For the moment we ignore the distinctions between adjuncts which are typically thought to adjoin to T, *v*, V, etc.; these distinctions are independent of the matter at hand and can be easily maintained once these functional heads are included.

3.3 The labelling operation

To continue the derivation of (19), ‘John’ must also enter into a relation with ‘ran’, so we apply **combine** a seventh time and produce a seventh unordered pair. This stage of the derivation is:

(21) { {ran, John}, {ran, quickly}, {ran, [in a t-shirt]}, {ran, [on Monday]}, {ran, [from here]}, {ran, [to there]}, {ran, [because of the parrot]} }

We want to maintain the idea that something extra must be established between ‘ran’ and ‘John’, which is not required between ‘ran’ and its adjuncts. In particular, an *asymmetric* relation must be established between them, whereas a symmetric relation suffices for adjuncts. The unordered pairs we have constructed so far state only symmetric relations. We can represent the effect of applying the “labelling” operation to the syntactic object {ran, John} as turning it into the ordered pair ⟨ran, John⟩, where the first coordinate of such an ordered pair is understood to be the label. The derivation progresses to the following stage:

(22) { ⟨ran, John⟩, {ran, quickly}, {ran, [in a t-shirt]}, {ran, [on Monday]}, {ran, [from here]}, {ran, [to there]}, {ran, [because of the parrot]} }

This actually completes the derivation for the sentence in (19). Unlike the use of stages in Chomsky (1995), we need not end up with a single syntactic object. Interpretation of this derived structure will be explained below.

Consider now sentences with two arguments, such as:

(23) John kicked Fred in the yard.

Note that for this sentence it will not suffice to combine ‘kicked’ with each of ‘John’, ‘Fred’ and ‘in the yard’, and “orient” the relationship between ‘kicked’ and each of its arguments, because the resulting object would place ‘John’ and ‘Fred’ in exactly the same relation to the verb:

(24) { ⟨kicked, John⟩, ⟨kicked, Fred⟩, {kicked, [in the yard]} }

This does not allow the CI interface to infer the required internal and external argument relations in order to identify the agent and patient of the kicking event.

In straightforward adaptation of the conventional tree structures, let us say that once an *ordered* pair (corresponding to a labelled constituent) has been formed, it is equivalent to the lexical item labelling it (its first coordinate) for the purposes of the **combine** operation. Thus having reached, in the obvious way, the following stage in the derivation of (23):

(25) { John, ⟨kicked, Fred⟩, {kicked, [in the yard]} }

we establish the relation between ‘kicked’ and ‘John’ by applying **combine** to the ordered pair:

(26) { {John, ⟨kicked, Fred⟩}, {kicked, [in the yard]} }

and then “label” the resulting unordered pair:

(27) { ⟨⟨kicked, Fred⟩, John⟩, {kicked, [in the yard]} }

This structure establishes ‘Fred’ as the internal argument, and ‘John’ as the external argument, of ‘kicked’. However, we need to generalise the idea of an ordered pair’s label: instead of the label being defined as the first coordinate, it must now be defined recursively as the label of the first coordinate, where an instance of a lexical item is taken to be the label of itself. Thus the label of ⟨⟨kicked, Fred⟩, John⟩ is ‘kicked’.

3.4 Interpretation at the interfaces

We now need to provide a mapping from the structures to the relevant interpretations for each of the following two sentences:

- (28) a. John ran quickly in a t-shirt on Monday from here to there because of the parrot.
 b. { {ran, quickly}, {ran, [in a t-shirt]}, {ran, [on Monday]}, {ran, [from here]}, {ran, [to there]}, {ran, [because of the parrot]}, ⟨ran, John⟩ }
- (29) a. John kicked Fred in the yard.
 b. { ⟨kicked, Fred⟩, John, {kicked, [in the yard]} }

3.4.1 Semantic interpretation

We stipulate for now that there must be exactly one ordered pair in the final stage of the derivation. This ordered pair may, of course, contain other ordered pairs, as in (29). The interpretation of this one pair, corresponding to a well-formed tree, can proceed according to conventional compositional interpretation of tree-shaped derived structures; in particular, we adopt the “Conjunctivist” proposal of Pietroski (2005). On this view, every syntactic object signifies a predicate, and the default method of interpreting complex object is via conjunction⁹:

$$(30) \text{Val}(O, \alpha \frown \beta) \iff \text{Val}(O, \alpha) \wedge \text{Val}(O, \beta)$$

In some cases, however, this method of interpretation is not appropriate; the canonical example of such a situation is the combination of a verb and one of its arguments, where a binary relation is established between the event satisfying the predicate signified by the verb, and the entity satisfying the predicate signified by the argument.

$$(31) \text{Val}(O, \alpha \frown [\text{ARG-}\beta]) \iff \text{Val}(O, \alpha) \wedge \text{Val}(O, \text{ARG-}\beta)$$

$$(32) \text{Val}(O, \text{ARG-}\beta) \iff \exists X [\text{Argument}(O, X) \wedge \text{Val}(X, \beta)]$$

$$(33) \text{Val}(O, \alpha \frown [\text{ARG-}\beta]) \iff \text{Val}(O, \alpha) \wedge \exists X [\text{Argument}(O, X) \wedge \text{Val}(X, \beta)]$$

Recall that the motivation behind requiring labelling for arguments was to ensure that the appropriate structural configurations can be established for determining thematic roles. It is therefore natural to take the first axiom in (30) to apply to “unlabelled” constituents (i.e. unordered pairs), and use the second axiom (31) when interpreting “labelled” constituents:

$$(34) \text{Val}(O, \{\alpha, \beta\}) \iff \text{Val}(O, \alpha) \wedge \text{Val}(O, \beta)$$

$$(35) \text{Val}(O, \langle \alpha, \beta \rangle) \iff \text{Val}(O, \alpha) \wedge \exists X [\text{Argument}(O, X) \wedge \text{Val}(X, \beta)]$$

To introduce these axioms we have used ARG to mark the constituent being interpreted as an argument and Argument to denote the binary relation. These describe the general form of the axioms but in fact we must

⁹The unusual Val notation here is a result of treating expressions as signifying plural predicates; details aside though, $\text{Val}(O, \alpha)$ can be treated as roughly equivalent to $\llbracket \alpha \rrbracket(O) = \top$ or “O satisfies (the predicate signified by) α ”.

distinguish between internal and external arguments, using INT and EXT in place of ARG and Internal and External in place of Argument, as appropriate. Crucially, this distinction is available only at the interface where semantic interpretation takes place, and is not visible to the syntactic/computational component.

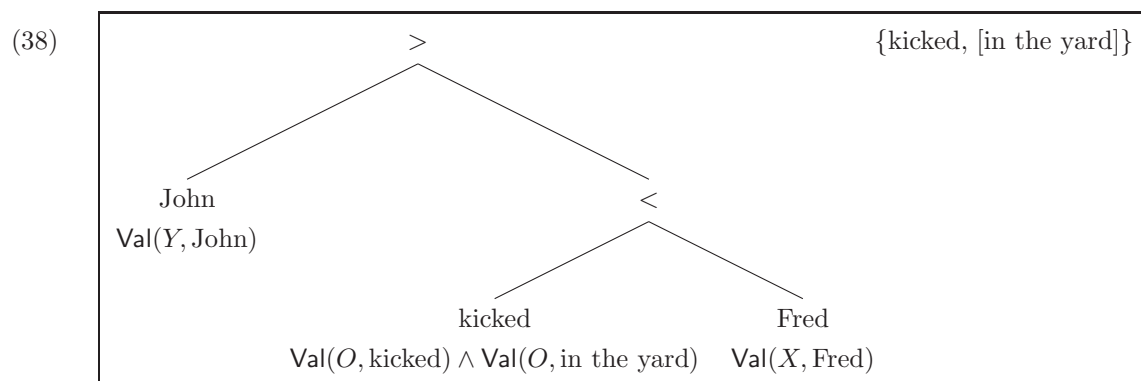
Applying these axioms, we can interpret the “main tree” of (29) as follows:

- (29) a. John kicked Fred in the yard.
 b. { ⟨(kicked, Fred), John⟩, {kicked, [in the yard]} }

(36) $\text{Val}(O, \langle \text{kicked}, \text{Fred} \rangle)$
 $\iff \text{Val}(O, \text{kicked}) \wedge \text{Val}(O, \text{INT-Fred})$
 $\iff \text{Val}(O, \text{kicked}) \wedge \exists X[\text{Internal}(O, X) \wedge \text{Val}(X, \text{Fred})]$

(37) $\text{Val}(O, \langle \langle \text{kicked}, \text{Fred} \rangle, \text{John} \rangle)$
 $\iff \text{Val}(O, \text{EXT-John}) \wedge \text{Val}(O, \langle \text{kicked}, \text{Fred} \rangle)$
 $\iff \exists Y[\text{External}(O, Y) \wedge \text{Val}(Y, \text{John})] \wedge \text{Val}(O, \text{kicked}) \wedge \exists X[\text{Internal}(O, X) \wedge \text{Val}(X, \text{Fred})]$

It remains to say how the semantic contribution of the phrase ‘in the yard’ is integrated. Clearly we want to end up conjoining the requirement $\text{Val}(O, \text{in the yard})$ with the predicate above (again, ignoring the internal structure of such adjuncts for now). The intuition to be pursued here is that ‘in the yard’ imposes the restriction on the variable O , rather than X or Y , because it is the O variable that is directly modified by the verb ‘kicked’, and it is with the verb ‘kicked’ that ‘in the yard’ has been associated, as recorded by the syntactic object {kicked, [in the yard]}. In other words, the pair {kicked, [in the yard]} determines where in the tree the semantic contribution of ‘in the yard’ is included. To state this more formally, the contribution of each “node of the tree” is not only the semantic contribution of the lexical item at that node, but the result of conjoining the contribution of this lexical item with the contributions of all the other syntactic objects with which it is associated by the additional unordered pairs. Thus, to illustrate the interpretation of the entire derived structure in (29), consisting of main tree interpreted above plus the unordered pair {kicked, [in the yard]} “hanging off to the side”, we might draw the following, where the “arrow” at each internal node of the tree points to the label of the corresponding constituent (adapting notation from Stabler (1997)):



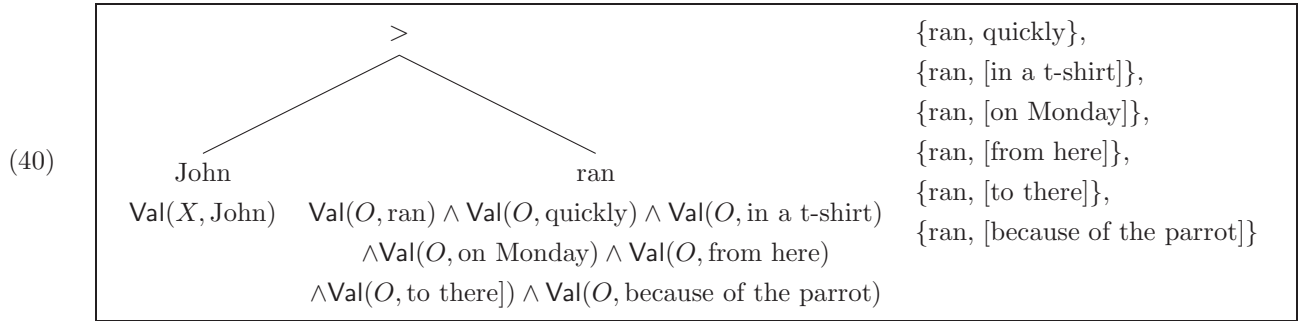
where one variable is chosen arbitrarily for each node, here O , X and Y , and the relations between these

variables remain to be stated on the basis of grammatical relations such as internal and external argument:

$$(39) \text{ Val}(O, \text{John kicked Fred in the yard}) \\ \iff \exists Y[\text{External}(O, Y) \wedge \text{Val}(Y, \text{John})] \wedge \text{Val}(O, \text{kicked}) \wedge \text{Val}(O, \text{in the yard}) \\ \wedge \exists X[\text{Internal}(O, X) \wedge \text{Val}(X, \text{Fred})]$$

Similarly, the interpretation of the structure in (28) could be illustrated as follows:

- (28) a. John ran quickly in a t-shirt on Monday from here to there because of the parrot.
 b. { {ran, quickly}, {ran, [in a t-shirt]}, {ran, [on Monday]}, {ran, [from here]}, {ran, [to there]}, {ran, [because of the parrot]}, ⟨ran, John⟩ }



$$(41) \text{ Val}(O, \text{John ran quickly in a t-shirt on Monday from here to there because of the parrot}) \\ \iff \exists X[\text{External}(O, X) \wedge \text{Val}(X, \text{John})] \wedge \text{Val}(O, \text{ran}) \wedge \text{Val}(O, \text{quickly}) \wedge \text{Val}(O, \text{in a t-shirt}) \\ \wedge \text{Val}(O, \text{on Monday}) \wedge \text{Val}(O, \text{from here}) \\ \wedge \text{Val}(O, \text{to there}) \wedge \text{Val}(O, \text{because of the parrot})$$

3.4.2 Phonological interpretation

To illustrate the phonological interpretation of these structures we consider the effect of the fronting transformation in (2). Recall that whatever structure we assign to adjuncts, it should allow for any number of adjuncts (zero or more) to be moved along with the verb and its argument. Assuming that displaced chunks of the sentence must be labelled causes a problem, as noted in §2.3.2, because it forces adjuncts into an argument configuration. But with the revised conception of phrase structure, consisting of a single main tree structure and various pairs independently indicating adjunction relations, a new solution to this problem is possible.

We take the movement in (2) to be occurring “in the main tree”, establishing a relation between the verb ‘kick’ and some functional projection high in the tree which can go by the name of F. From the stage illustrated in (29), then, we continue by combining F, labelling the result:

$$(42) \{ \langle F, \langle \langle \text{kicked}, \text{Fred} \rangle, \text{John} \rangle \rangle, \{ \text{kicked}, [\text{in the yard}] \} \}$$

then associating ‘kicked’ with F by combining F with ‘kicked Fred’, in accordance with (an equivalent of) the A-over-A condition (but see footnote 5):

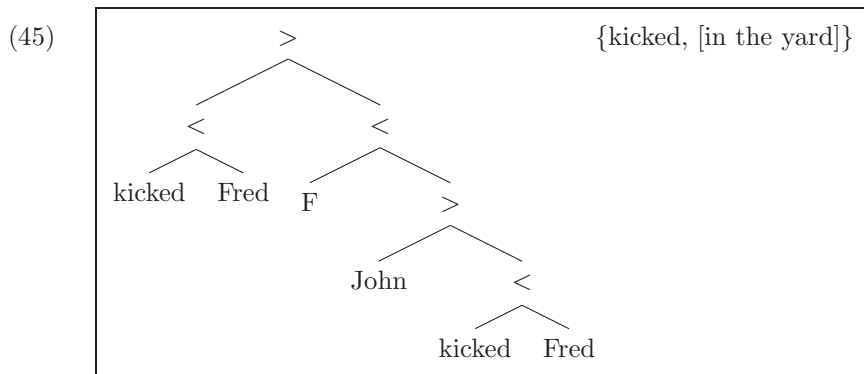
(43) $\{ \{ \langle \text{kicked}, \text{Fred} \rangle, \langle \text{F}, \langle \langle \text{kicked}, \text{Fred} \rangle, \text{John} \rangle \rangle \}, \{ \text{kicked}, [\text{in the yard}] \} \}$

and finally establish the presumably-required asymmetric relation between ‘kicked’ and F by “labelling”:

(44) $\{ \langle \langle \text{F}, \langle \langle \text{kicked}, \text{Fred} \rangle, \text{John} \rangle \rangle, \langle \text{kicked}, \text{Fred} \rangle \rangle, \{ \text{kicked}, [\text{in the yard}] \} \}$

The one syntactic object ‘kicked Fred’ now “appears twice” in the “main tree” of the derivation, once to indicate a relation with ‘John’ and once to indicate a relation with F; in really just the same way as ‘kicked’ already appeared twice in the derivation, once to indicate a relation with ‘Fred’ and ‘John’, and once to indicate a relation with ‘in the yard’. Unless any more complex notions such as copying and/or movement turn out to be beneficial, we will assume that the unremarkable ability of two sets to have a non-empty intersection gives us all the machinery we need to represent these structures.

The final stage of the derivation, then, can be illustrated more clearly as follows:



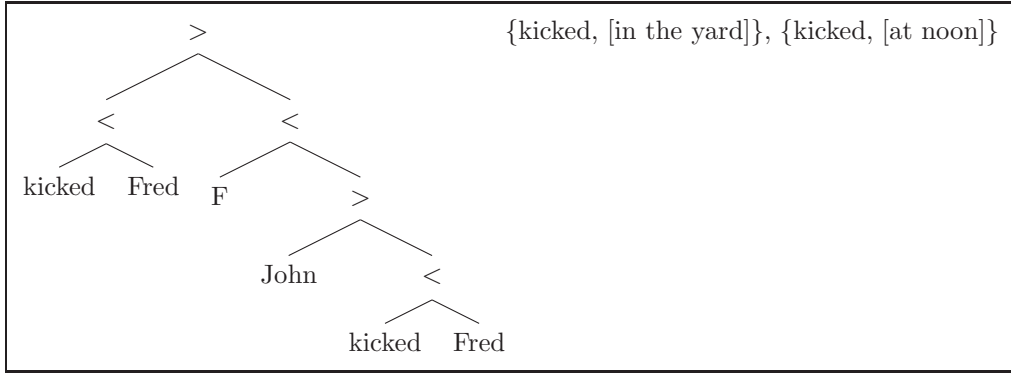
As was the case for semantic interpretation, phonological interpretation of the “main tree” of the derivation proceeds fairly conventionally: by something like the LCA, and abstracting away from complications due to the separation of tense from the lexical verb, we produce the string “kick Fred, John did” for the main tree. The pronunciation of the adjunct needs to be included somehow. As for semantic interpretation, we pursue the intuition that the recorded association between ‘kicked’ and ‘in the yard’ determines “where in the tree” the adjunct is to be integrated. But here, because of the “movement” of the VP, there are two candidate positions in which the adjunct could be linearised, one for each of the positions in the main tree where ‘kicked’ appears. As it turns out, given the assumption that each instance of a lexical item is to be interpreted exactly once at the phonological interface (at least in English), these two candidate positions yield the pattern observed in (2)¹⁰:

- (46) a. It was kick Fred that John did in the yard.
 b. It was kick Fred in the yard that John did.

With more than one adjunct, we simply take the choice of where to pronounce each adjunct to be independent; this produces the full range of possibilities seen in (2):

¹⁰ Assuming that whatever idiosyncratic property of English requires verbs to be directly adjacent to their direct objects, is not our problem

(47)



(48)

- a. It was kick Fred that John did in the yard at noon.
- b. It was kick Fred in the yard that John did at noon.
- c. It was kick Fred at noon that John did in the yard.
- d. It was kick Fred in the yard at noon that John did.
- e. *It was kick that John did Fred in the yard at noon.

Fronting only the verb is still a violation of some equivalent of the A-over-A constraint; but furthermore, any number of the adjuncts can be pronounced in the focus position, without placing them in what would appear to be an argument configuration, as observed in §2.3.2.

3.4.3 Back to semantics

Having introduced the transformed structures to illustrate phonological interpretation, we now turn to semantic interpretation of these structures. §3.4.1 addressed only the semantic interpretation of structures without the focus projection.

The basic intuition, as with phonological interpretation, is that the element that is “moved” to the focus projection is (obligatorily) interpreted in that higher position. Without committing to exactly what it means to be interpreted “with focus”, we can assume that while the untransformed sentence ‘John kicked Fred’ signifies the simple conjunction of three predicates:

$$(49) \text{Val}(O, \text{John kicked Fred}) \iff \mathcal{K}(O) \wedge \mathcal{J}(O) \wedge \mathcal{F}(O)$$

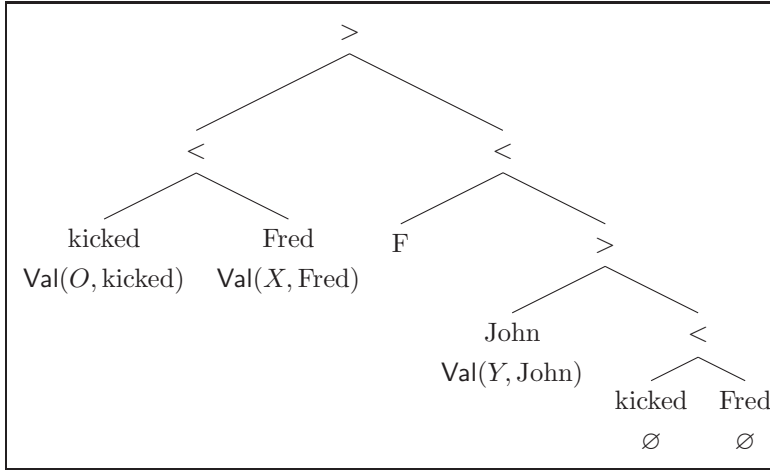
$$\mathcal{K}(O) \iff \text{Val}(O, \text{kicked})$$

$$\mathcal{J}(O) \iff \exists Y[\text{External}(O, Y) \wedge \text{Val}(Y, \text{John})]$$

$$\mathcal{F}(O) \iff \exists X[\text{Internal}(O, X) \wedge \text{Val}(X, \text{Fred})]$$

the sentence illustrated in (50) signifies a conjunction including slightly modified versions of \mathcal{K} and \mathcal{F} , which we can call \mathcal{K}' and \mathcal{F}' respectively, in virtue of the fact that ‘kicked’ and ‘Fred’ are interpreted in the projection of F:

(50) a.

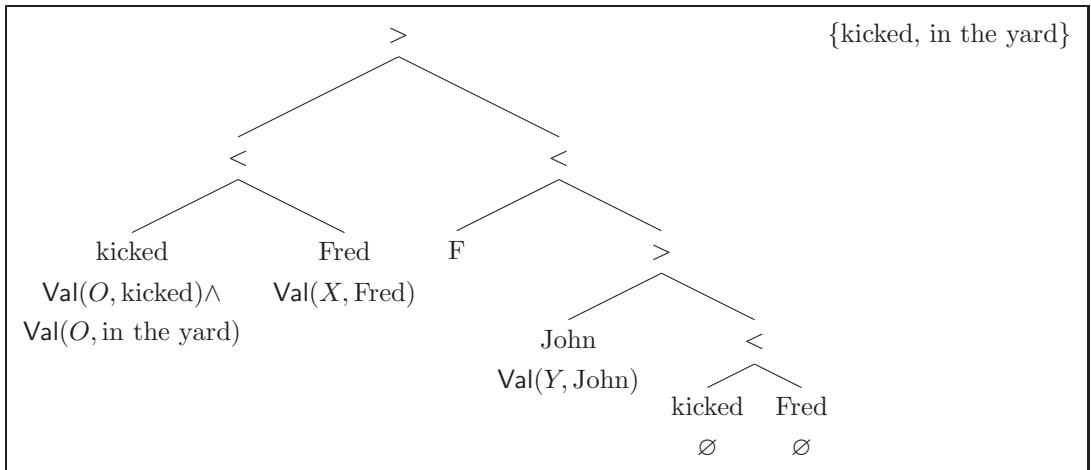


b. $\text{Val}(O, \text{'Kick Fred, John did'}) \iff \mathcal{K}'(O) \wedge \mathcal{J}(O) \wedge \mathcal{F}'(O)$

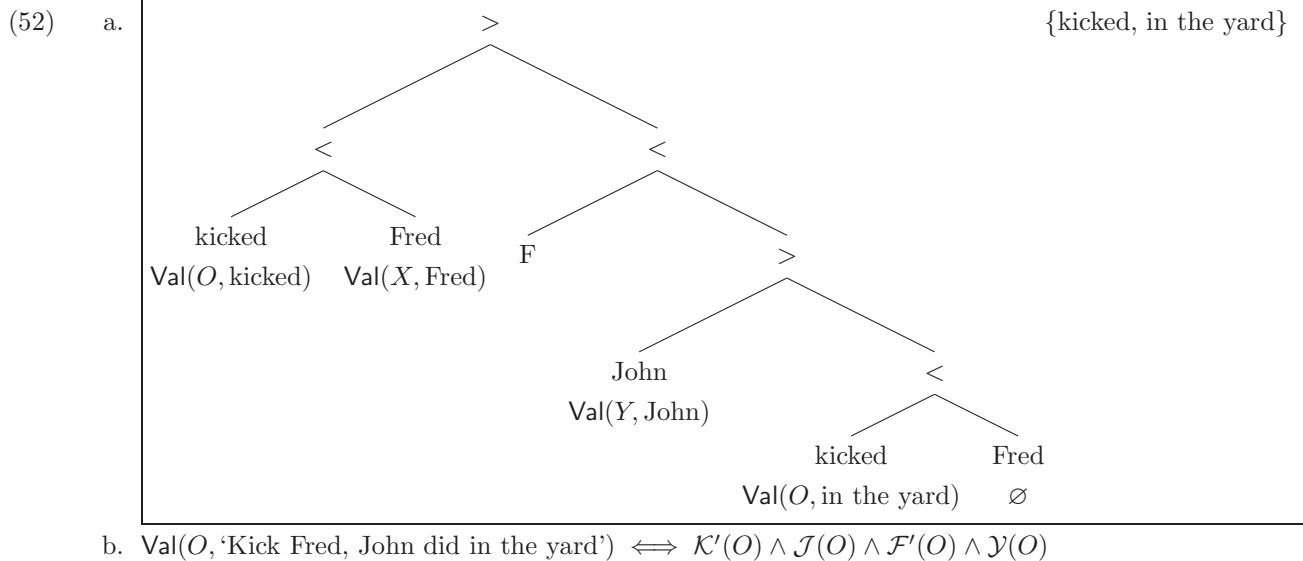
We can think of \mathcal{J} as the predicate “has John as external argument” and \mathcal{F} as the predicate “has Fred as internal argument”. As in §3.4.1, the tree in (50) shows the “lexical semantic contribution” of each node underneath it. The lower nodes representing ‘kicked’ and ‘Fred’ have no such contribution because these lexical items are interpreted in exactly one position, namely the higher one, just as they were at the phonological interface. In particular, note that the lack of *lexical* semantic content at the lower node labelled ‘kicked’ does not preclude ‘John’ from being understood as an external argument. Admittedly, it is not at all clear how to ensure that \mathcal{K}' and \mathcal{F}' end up modifying the same event variable O as \mathcal{J} does, but this is a problem that needs to be solved to allow for these constructions in any event-based semantics.

Consider now the corresponding structure with an adjunct added, (45). As with phonological interpretation, two possibilities arise as a result of the fact that the lexical item with which the adjunct ‘in the yard’ is associated, ‘kicked’, appears at two positions in the tree. These two possible semantic interpretations can be represented as follows, where $\mathcal{Y}(O) \iff \text{Val}(O, \text{in the yard})$ (and \mathcal{Y}' is the corresponding “focussed” version):

(51) a.



b. $\text{Val}(O, \text{'Kick Fred in the yard, John did'}) \iff \mathcal{K}'(O) \wedge \mathcal{J}(O) \wedge \mathcal{F}'(O) \wedge \mathcal{Y}'(O)$



Note that interpretation still consists of associating each lexical element with exactly one node of the tree, conjoining the corresponding lexically-contributed predicates, and then relating the variables (one for each node) to each other via **Internal** and **External** relations. The first of these three steps is trivial for lexical elements which, like ‘John’ in (50), are syntactically related to exactly one position in the tree anyway. Other elements may be syntactically related to multiple positions in the tree, like ‘kicked’ in (50), in which case all but one of these positions are ignored; or syntactically related to zero positions in the tree, like ‘in the yard’ in (50), in which case it is associated with the node of the element with which it has combined.

This account leaves two questions unanswered: firstly, what exactly the “focussed version” of a predicate is (i.e. what the relationship between \mathcal{K} and \mathcal{K}' is), and secondly, how it is that the focussed versions of predicates apply to the same variable as they would if they were in their non-focussed position. But crucially, answers to these two questions are required in order to account for the semantics of a simple focussed sentence like (50), which the current proposal assigns a completely standard syntactic structure. Any solutions which work in that case will straightforwardly carry over to the novel structures for adjuncts: whatever an account of focus does to the condition $\text{Val}(O, \text{kicked})$ in (50), we can do to the condition $\text{Val}(O, \text{kicked}) \wedge \text{Val}(O, \text{in the yard})$ in (51).

One problem which arises with this novel view of interpretation is that we have no explanation for the correlation between when an adjunct is interpreted semantically with focus, and when it is pronounced fronted. If all we say is that each interface is free to interpret an adjunct in either of the tree positions it’s attached to, no correlation is expected between position of pronunciation and focus. This problem could be resolved by integrating some notion of phases into the theory. Juan Uriagereka (personal communication) points out that if each adjunct is interpreted in a single phase, and the focus position is in a different phase from the non-focus position, then the desired correlation is enforced. In particular, the proposal for interpretation of adjuncts in Lasnik and Uriagereka (2005, ch 7.8) implements this with an even “looser” relationship between adjuncts and the phrase marker than the present proposal: adjuncts are “activated simply by ‘being there’” in a particular derivational cycle. This is attractive because if the only sense in which an adjunct is part of a derivation is by being associated with one particular phase or another, with no

further relationship to the derived object itself (the phrase marker), then it is natural that its interpretation should be restricted to one phase only.

3.5 Potential extensions to more complex structures

The revised version of the theory presented in §3 provides a solution for the problems listed in §2.3: we have extended conventional algorithms for interpretation at each interface to deal with the non-tree structures, and done so without the need to ever place an adjunct into an argument-like configuration with a labelled “mother node”.

In this section I briefly explore the compatibility of the system presented above with more complex syntactic structures. The aim is not to provide accounts for these phenomena as explicit as the discussion above, but rather to briefly sketch out some reasons to believe that this theory can extend to cover these cases at least as well as more standard theories, and perhaps even better.

3.5.1 Extending to include v

Many approaches capture the implication from (53a) to (53b) by assigning these sentences meanings along the lines of (54a) and (54b), respectively.

(53) a. Chris raised the flag.

b. The flag rose.

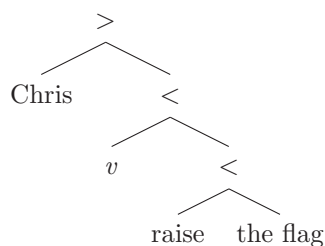
(54) a. $\llbracket \text{Chris raised the flag} \rrbracket = \top \iff \exists e[\text{Agent}(e, c) \wedge \exists e'[\text{Rising}(e') \wedge \text{Theme}(e', f) \wedge \mathcal{R}(e, e')]]$

b. $\llbracket \text{The flag rose} \rrbracket = \top \iff \exists e[\text{Rising}(e) \wedge \text{Theme}(e, f)]$

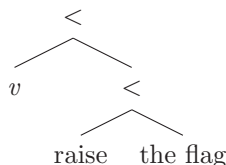
Exactly what the relation \mathcal{R} should be here is complicated. Here I will simply address the question of how a meaning of this form could be derived, whatever the specifics of the relation \mathcal{R} .

Typically v is thought to take the VP as its complement and the canonical subject as its specifier, so after a series of **combine** and **label** operations we will derive the structures in (55). To abstract away from the details of determiners for now, we can treat ‘the flag’ as an atomic syntactic object, just like the name ‘Chris’.

(55) a. $\{ \langle \langle v, \langle \text{raise}, [\text{the flag}] \rangle \rangle, \text{Chris} \rangle \}$



b. $\{ \langle v, \langle \text{raise}, [\text{the flag}] \rangle \rangle \}$



Blindly applying the Conjunctivist axioms of composition, we derive the following from (55b):

$$\begin{aligned}
(56) \quad & \text{Val}(O, \langle v, \langle \text{raise}, [\text{the flag}] \rangle \rangle) \\
& \iff \text{Val}(O, v) \wedge \exists O' [\text{Internal}(O, O') \wedge \text{Val}(O', \langle \text{raise}, \text{the flag} \rangle)] \\
& \iff \text{Val}(O, v) \wedge \exists O' [\text{Internal}(O, O') \wedge \text{Val}(O', \text{raise}) \wedge \exists X [\text{Internal}(O', X) \wedge \text{Val}(X, \text{the flag})]]
\end{aligned}$$

We would expect, given the previous treatment of simple transitive sentences, that the relations expressed by Theme and Agent (at least) in (54) would be expressed by relations such as **Internal** and **External** now; and indeed, while Theme states the relation between e and f in (54b), **Internal** correspondingly states the relation between O' (the raising event) and X (the flag) in (56). This is consistent with the existing idea that the internal participant of an event is its theme. The structure of this expression also dictates that the **Internal** relation holds between the “causing” event and the “result” event¹¹, so we can hypothesise that the internal participant of a causing event is the corresponding result event; this does not introduce any conflict with the existing hypothesis about themes if we just notice that it is the “lower” event, the causing event, which has the theme as its internal participant, and since these causing events are disjoint from result events, the **Internal** relation is free to relate causing events to their result events, and result events to their themes.

Also note that the standard compositional axiom for interpretation of argument configurations has introduced the existential closure of the “lower event”, O' . Whereas this same axiom applies to both the v -‘raise’ argument configuration and the ‘raise’-‘the flag’ argument configuration in (56), these are treated differently in (54b), with existential closure applying only to the former. This results from treating nominal expressions like ‘the flag’ as predicates of individuals.

Continuing with the axioms unchanged to consider the transitive version, from (55a) we derive:

$$\begin{aligned}
(57) \quad & \text{Val}(O, \langle \langle v, \langle \text{raise}, [\text{the flag}] \rangle \rangle, \text{Chris} \rangle) \\
& \iff \exists Y [\text{External}(O, Y) \wedge \text{Val}(Y, \text{Chris})] \wedge \text{Val}(O, \langle v, \langle \text{raise}, [\text{the flag}] \rangle \rangle) \\
& \iff \exists Y [\text{External}(O, Y) \wedge \text{Val}(Y, \text{Chris})] \wedge \text{Val}(O, v) \\
& \quad \wedge \exists O' [\text{Internal}(O, O') \wedge \text{Val}(O', \text{raise}) \wedge \exists X [\text{Internal}(O', X) \wedge \text{Val}(X, \text{the flag})]]
\end{aligned}$$

This clearly suggests that the **External** relation relates the higher “causing” events to their agents. Once we incorporate this idea, the derived meanings of these two expressions include all the important properties of those in (54): the implication holds, and the meanings have been constructed in a manner which treats each of the three argument configurations (‘Chris’- v , v -‘raise’, and ‘raise’-‘the flag’) in the same way. In (57), however, we have ended up with one extra component, the conjunct $\text{Val}(O, v)$. We could take v to signify the trivial predicate satisfied by everything, or alternatively it may signify the predicate satisfied by the kind of event that can be a causing event (those which are related to their agents by the **External** relation, and to their result events by the **Internal** relation).

3.5.2 Quantification

Initially, it seems that such a minimal view of syntax, with no notion of a trace or a chain, will struggle to allow for the interpretation of quantificational relations. Typically the “top of the chain” is taken to

¹¹Pietroski (2005) calls the relation between these two events the Terminator relation.

contribute the operator and the “foot of the chain” or trace the variable bound by this operator. If possible we would like to find a way to derive these operator-variable relations while maintaining the simple view of syntax as establishing a network of relations amongst objects, without copies or traces or chains, where each object is interpreted once. The idea we can pursue here is that a quantificational expression does not need to contribute a variable to be bound, because the compositional axiom for interpreting argument relations introduces this variable anyway; an expression like ‘every boy’ can contribute the “quantificational force” of ‘every’ and the restriction of ‘boy’ in its raised position, and let the *argument structure* it helped build contribute the variable, without any reference to the lexical content of the quantificational expression.

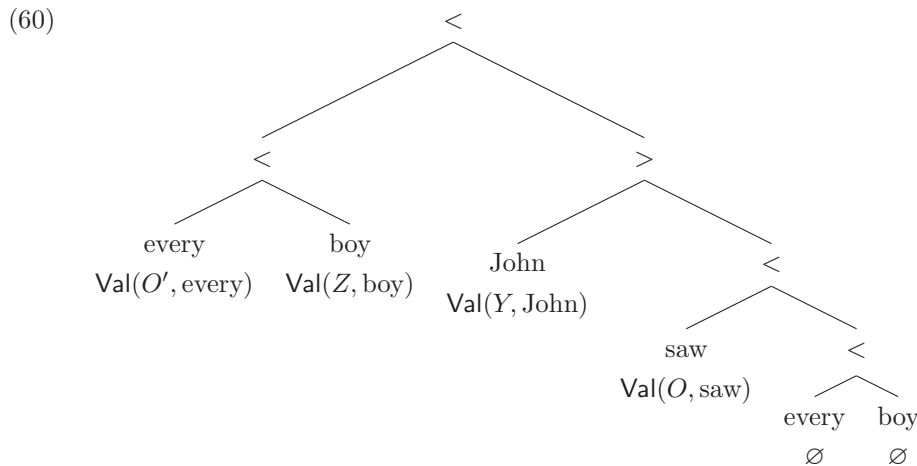
The details of quantification under Conjunctivist semantics are beyond the scope of this paper (see Pietroski (2005, ch.2)), but suffice it to say that in (58), where a standard approach such as Heim and Kratzer (1998) requires the external argument of the determiner to evaluate to the expression in (59), we also need the external argument to signify, very roughly speaking, the things that John saw.

(58) John saw every boy.

(59) $\lambda x.$ John saw x

With no notion of traces or copies, how can we introduce a variable corresponding to x in this version?

Consider what will happen if we try to apply the standard Conjunctivist axioms to the standard tree structure for this sentence (at least, the portion of it representing the external argument of ‘every’), continuing the idea that each lexical element contributes at exactly one position in the tree (and that the raised phrase ‘every boy’ does so in its higher position):



Recall that what is represented underneath the nodes of these trees is the *lexical* semantic content at those nodes, and the idea that the lack of *lexical* semantic content in the lower position of ‘kick Fred’ in (51) did not preclude ‘John’ from being interpreted as an external argument. Of what verb it was an external argument does not matter on Conjunctivist views; what mattered is that it contribute the predicate \mathcal{J} . Similarly, the fact that no lexical semantic content is in the lower position of ‘every boy’ here, does not change that fact that ‘saw’ has an internal argument. Having observed this, and recalling the Conjunctivist axiom of (35), it does not seem unreasonable to associate something like the following predicate (of O) with the external argument of the determiner:

(61) $\text{Val}(O, \text{saw}) \wedge \exists Y[\text{External}(O, Y) \wedge \text{Val}(Y, \text{John})] \wedge \exists X[\text{Internal}(O, X)]$

The result of the fact that no lexical semantic content does not appear in the lower position of ‘every boy’ is that no restriction is applied to the variable X that is introduced as a result of that particular argument relation. The crucial property of the Conjunctivist approach here is that no *expression* ever “introduces a variable”; all variables are introduced by the labelling of the tree structure. Expressions contribute to semantic interpretation only by restricting one or another of these variables. To say that ‘every/some boy’ is not interpreted in its lower position is simply to say that this expression does not restrict the variable introduced there.

Clearly, this is far from a comprehensive account of quantificational structures. A number of questions remain.¹² But the main point to be noted here is that the lack of traces or copies, which are traditionally used to introduce variables in quantificational structures, will not necessarily make it impossible to introduce the required variable, given the Conjunctivist view that expressions introduce monadic predicates, which restrict variables introduced by argument structure.

3.5.3 Reconstruction

Speaking in conventional terms, when a “moved” *wh*-phrase includes other nominal expressions, these nominal expressions can sometimes satisfy their binding properties in either the “operator position” or the “variable position” of the *wh*-phrase, yielding ambiguity. For example, in (62), ‘himself’ can have either ‘John’ or ‘Bill’ as its antecedent:

(62) John_{*i*} wondered [[which picture of himself_{*i/j*}] Bill_{*j*} saw ___]

In its surface position, ‘himself’ can be bound by ‘John’, but not by ‘Bill’. Inside the thematic position of the *wh*-phrase, however, ‘himself’ can be bound by ‘Bill’. In systems where the *wh*-phrase has been copied in order to enter into two different relations, this has prompted the suggestion that the two structures corresponding to the two distinct interpretations are those in (63), where part of each copy has been deleted:

(63) a. John_{*i*} wondered [[which picture of himself_{*i*}] Bill_{*j*} saw ~~which picture of himself~~]
 b. John_{*i*} wondered [[~~which picture of himself~~] Bill_{*j*} saw ~~which~~ picture of himself_{*j*}]

This covers the facts, but raises two conceptual questions. Firstly, why should the deleted parts of the two copies of the *wh*-phrase be the particular parts that they are? Secondly, why should the deleted parts of the two copies be exactly complementary?

Much of the discussion of reconstruction focusses on the first question. For example, the fact that in some languages the operator (‘which’) alone can be overtly displaced, with the restrictor (‘pictures of himself’) apparently left in its proposition-internal position(s), is presented as evidence that the particular “split” illustrated in (63b) is feasible (Chomsky, 1995).¹³ This issue is orthogonal to the question of whether two

¹²Perhaps the most striking is the question of how the quantification introduced by ‘every’ gets access to the variable X , which is existentially closed in (61); a possible answer to this is to assume that the existential closure that we have taken to be part of the Conjunctivist axioms, is in fact a default binding that occurs if a variable remains unbound when the entire structure has been interpreted, but this will not work in sentences with two quantifiers because we would have no way of tracking which quantifier should bind which variable. The answer to this question could conceivably relate to the question raised in §3.4.3 about how focussed predicate “connect up with” the right variable from lower down in the structure.

¹³The phrase ‘which pictures of himself’ can be taken to contribute three elements to semantic interpretation: an operator, a restrictor, and a variable. Most of the details of how these elements are introduced will be irrelevant to what follows; we require only that ‘himself’ is interpreted in the same place as the restrictor (which seems uncontroversial), but for concreteness the structures in (63) suppose that ‘which’ introduces the operator and that the variable is introduced implicitly, roughly along the lines described in §3.5.2.

separate copies are created, complementary parts of which are then deleted, or just a single copy exists, different parts of which are interpreted in different positions.

The question of why complementary pieces of the two “copies” should be deleted, however, disappears if a syntactic object establishes multiple relations without copying. Under this account, there is one *wh*-phrase, and each of its semantic contributions (operator, restrictor, and variable) must be made in exactly one place. For a coherent semantic interpretation to be possible, the operator (‘which’) will have to be interpreted in the “high” position, and the variable will appear in the “low” position, and the restrictor (‘picture of himself’) can be interpreted in either position. So the two options in (63) follow naturally.

A proponent of a copying theory might use the same argument, based on coherence of semantic interpretation, to argue that the high copy can not contribute a variable purely in virtue of its position, and that, independently, the low copy similarly can not contribute an operator. It is not clear, however, that any similar reason can be put forward for the deletion of the restrictor (‘picture of himself’) from either copy. Leaving the restrictor in both places will not cause any problems for semantic interpretation. Note that economy arguments, along the lines that vacuous “extra” restriction should be avoided, will struggle to account for this if interpreted as economy requirements on derivations, because in a copying theory avoiding vacuous restriction requires more operations (deletions) than allowing it. If interpreted representationally, such requirements seem to argue against copying. To the extent that the two copies of the *wh*-phrase are independent, we might ask why either of the restrictors should be deleted when neither of the restrictors is deleted in a sentence like ‘I wonder which boy ate which apple’; to the extent that they aren’t, we might ask why calling them “two copies” makes sense.¹⁴

4 Movement, adjunction and copying

As noted briefly above, in deriving the “focussed” sentences of (2), the syntactic object ‘⟨kicked, Fred⟩’ ends up appearing twice in the “main tree” of the derived object. The structures before and after the relevant steps are repeated here:

- (64) a. { ⟨⟨kicked, Fred⟩, John⟩, {kicked, [in the yard]} }
 b. { ⟨⟨F, ⟨⟨kicked, Fred⟩, John⟩⟩, ⟨kicked, Fred⟩⟩, {kicked, [in the yard]} }

Of course, the progression from (64a) to (64b) is precisely what is commonly known as “movement”: an object combines with an object of which it is a part. Typically movement configurations are taken to require copying of the moved object. But given the conception of a syntactic derivation as establishing relations

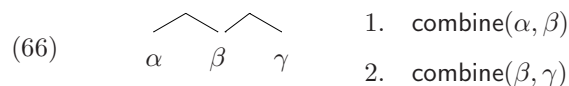
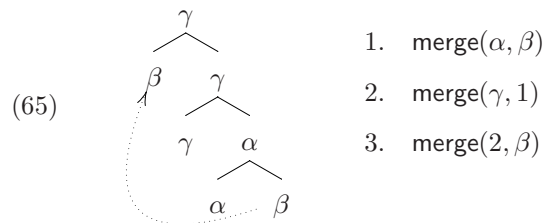
¹⁴Freidin (cited in Chomsky, 1995, p.204) has pointed out an interesting asymmetry between the reconstruction of complement clauses (arguments) and relative clauses (adjuncts):

- (i) which claim [that John was asleep] was he willing to discuss
 (ii) which claim [that John made] was he willing to discuss

In (i), reconstruction is obligatory and Principle C forbids ‘he’ and ‘John’ from being coreferential; in (ii), reconstruction is optional and ‘he’ and ‘John’ may or may not be coreferential. Lebeaux (cited in Chomsky, 1995, p.204) accounted for this difference by proposing that the adjunct clause can be inserted as a generalised transformation at any point in the derivation and interpreted at the corresponding position, whereas the complement clause must be present at D-Structure, and must be interpreted in its corresponding position. This does not account for the optional reconstruction of other complements, however, such as those in (62). The present account of the argument/adjunct distinction doesn’t account straightforwardly for the distinction between (i) and (ii), but the fact that significantly different phrase structures are involved makes the possibility of an explanation more plausible than in many Minimalist theories.

between elements by combining them, there was no obvious need to copy ‘kicked’ to derive (64a), even though it has already entered into relations with ‘Fred’, ‘John’ and ‘in the yard’ by this stage. If no copying were required to establish these three relations, why should copying be required to establish a fourth relation with F? Or, if copying is required to establish the relation with F, why was it not required to establish the previous relations?

The approach to adjuncts presented here centres on the idea that an object may independently enter into multiple syntactic relations, but this is really what “syntactic movement” has always sought to capture. Consider the following two derivations:



Given the graphical representations of the derived objects, it seems that the intuition that copying is required in (65) stems from the fact that we needed to write β twice on the page. There was no need to do so to represent (66) using the original “multiple domination” notation used in Hornstein (forthcoming) and in §2.2, so no talk of copying arose. But once the sequence of operations involved in each derivation is specified, it becomes clear that there is not necessarily any real distinction between the way β participates in two derivations; in each case, it is an operand of two different applications of the same operation. The familiar notions of movement and adjunction can be defined as patterns of re-use in the derivation: broadly speaking, movement is combining with something of which you are already a part, and adjunction is combining with two separate things, neither of which is a part of the other.¹⁵ This insight was hiding only just beneath the surface in Hornstein’s original work, but was obscured by the informal presentation grounded in the conventional “tree” notation.

One conclusion that one might initially draw from this observation would be that we had overlooked a genuine need for copying when the novel adjunction structures were introduced, and that the explicit comparison between (65) and (66) alerts us to this genuine need for copying. This would presumably be grounded in the belief that an extra copy of a syntactic object is required every time that object enters into a new syntactic relation. But this then suggests that we should consider copying to be required when one head takes two arguments, as γ does in (65) in order to establish relations with both α and β . This idea that projection can be identified with movement/copying has been mentioned in Boeckx (2008) (and references therein), and we will see below (§5.1) that under one formalisation of the labelling operation this connection becomes very natural.

¹⁵Some fine-tuning of these definitions is necessary to allow for more exotic patterns of re-use we might want to allow such as adjunct movement and sideways movement.

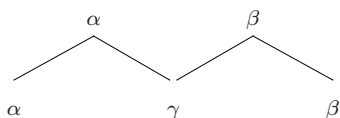
Alternatively, one might draw the conclusion that the need for copying was not genuine even in the case of conventional “movement” configurations. This amounts to trusting the less “geometrically guided” intuition appealed to in §3, that there is no reason to doubt the ability of one object to enter into multiple relations without any notion of “copying” being required. The view that movement can be implemented without copying, but rather simply by “internal merge”, has been expressed in Epstein et al. (1998) and Chomsky (2004), among others.¹⁶

My aim here is not to argue in favour of one or the other of these two conclusions. Rather it is to point out that the assumption that copying is required for movement configurations, but not for projection or for the labelless adjunction structures discussed above, is arbitrary. Surely two more natural null hypotheses that should be considered first are that copying is required in all of these cases, or (probably more natural still) in none.

What sort of evidence, then, *does* bear on the question of whether to include some notion of copying in the syntactic/computational component of grammars for natural languages? I think the observations above suggest that this question is much more subtle than it appears. The discussion of reconstruction in §3.5.3 illustrates that the same sound-meaning pairs can be generated by assuming that *wh*-movement produces copies, parts of which are deleted, as by assuming copying is not required and allowing the *wh*-phrase to contribute to semantic interpretation in each of the places where it is linked in to the tree. Only a more subtle argument about the complementary pattern of deletion provided any evidence in favour of one view or the other. That argument is not intended to settle the deal. Other similarly delicate arguments may provide evidence in the other direction. But the simple fact that an anaphor embedded in a *wh*-phrase can display behaviour characteristic of either of the linearly separate positions with which the *wh*-phrase is associated, does not provide strong evidence that copying should be implicated in movement more than it should in projection or Hornstein’s adjunction.

Similarly, phenomena where we observe a particular portion of a sentence being “pronounced twice” in separate linear positions (Bošković and Nunes, 2007), do not necessarily require any notion of copying of phrase structure. There is nothing incoherent about a linearisation algorithm (mapping from derived objects to strings) that duplicates the sections of output corresponding to portions of structures that are linked in to the tree in two positions. Of course, this proposal needs to be supplemented with an account of why some but not all movement results in the moved element being pronounced multiple times, but this is exactly the same question that copying accounts must answer to account for the fact that some but not all chains have all but one of their copies deleted. Kobele (2006) emphasises the fact that a number of ways of accounting for this phenomenon turn out to be notational variants. Again, simple facts about the sound-meaning relation that the grammar generates do not seem to provide strong evidence for or against a copying operation.

¹⁶Citko (2005) uses multiple domination not only to implement movement configurations, but also to allow one element to be an argument of two separate heads which are *not* in a domination relation:



This differs from the multiple domination structures presented in this paper in that both α and β project over γ (differentiating it from the treatment of adjuncts above), and no domination relation holds between α and β (differentiating it from “internal merge” treatments of movement).

One area where evidence for or against a copying operation may be discovered is in the interaction with *derivational* constraints (given that representationally, there is nothing to distinguish copying approaches from “multiply-linked” approaches). In systems where some operations are allowed to apply at certain points in the derivation but not at others, it becomes possible to have the derivational step of applying the copy operation affect the possible continuations of the derivation.

Hornstein (to appear, ch. 5.7) presents an argument of this sort. A basic property of the complete system presented there is a derivational constraint along the lines of relativised minimality, restricting the application of the combine operation, as a function of the sets of nodes dominating the two potential combiners. Under a natural (but not logically necessary) conception of the interaction between the copy operation and this derivational constraint, two elements whose dominating nodes currently conspire to forbid them from combining, can get around this restriction if one of the elements is first copied, on the logic that this new copy has no dominating nodes. The range of structures that can be built will then differ, depending on whether one includes a copy operation or not.

It is worth emphasising that only a copy operation with exactly the required interaction with Hornstein’s particular notion of the derivational minimality constraint will produce this result. It relies crucially on the fact that derivational constraints must be Markhovian with a context of one step, and that whatever looks at the newly-generated copy to determine whether derivational constraints are satisfied cannot “see” the properties of the original copy that precluded combination. An argument in a setting of different derivational constraints, will necessarily be an argument for a copy operation interacting in the required way with *those* particular derivational constraints, and not in any obvious sense an argument for the same operation.

5 Relation to formal models of grammar

Keenan and Stabler (2003) introduce “Bare Grammar” (BG), a framework in which a large variety of grammar formalisms can be expressed. The BG approach requires the structure-building operations to be functions over the set of expressions. The language generated by a particular grammar is the closure of a set of lexical expressions under these structure-building operations. This permits a derivation to be uniquely identified by a derivation tree with lexical expressions at the leaves, and one structure-building operation at each internal node.

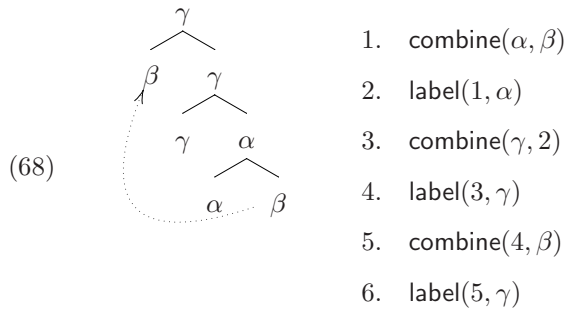
In this section I consider ways to express the theory presented above within this framework. In the process of doing this we are forced to sharpen our conception of the theory under consideration, and otherwise-unnoticed insights can emerge, either relating ideas within the theory to each other, or relating the theory to others that have already been expressed in this framework.

5.1 The label operation

The label operation has so far been treated implicitly as a unary operation. On this view, however, it is not a function, because its output is not uniquely determined by the object being labelled: an object $\{\alpha, \beta\}$ may lead to either $\langle \alpha, \beta \rangle$ or $\langle \beta, \alpha \rangle$. One way to resolve this is to construe label as a binary operation, with a second argument indicating the label to be applied. The operation will only be defined if the second argument meets certain restrictions: it must be a lexical item (not a complex object), and it must be the label of one of the elements of the unordered pair given as the first argument.

- (67)
- a. $\text{label}(\{\alpha, \beta\}, \alpha) = \langle \alpha, \beta \rangle$
 - b. $\text{label}(\{\alpha, \beta\}, \beta) = \langle \beta, \alpha \rangle$
 - c. $\text{label}(\{\alpha, \beta\}, \langle \alpha, \gamma \rangle)$ is undefined because $\langle \alpha, \gamma \rangle$ is not lexical
 - d. $\text{label}(\{\alpha, \beta\}, \gamma)$ is undefined because γ is not a valid label for $\{\alpha, \beta\}$

This is the idea which brings projection into line with the other kinds of re-use we have seen (movement and adjunction). Consider again the derived structure illustrating a simple movement configuration in (65); using **combine** and **label** instead of **merge**, this object will now be derived as follows:



Now we can see that when an element projects in the familiar derived structure (on the left), the correlate in the derivation itself is a re-use of that element as an argument to **label**. (It will necessarily be a *re-use* because for an item to be a valid label, it must be a constituent of the thing it is labelling, which means it must have already been an argument to **combine**.) In derived structure terms, β moves, α projects once and γ projects twice, but these notions can (like adjunction) be defined purely in terms of particular re-use patterns: movement is being passed to **combine** together with something of which you are a part, and projection is being passed to **label** together with something of which you label an immediate part.

A different way to formulate the desired set of operations as functions is to introduce a “compiled” operation, **combineAndLabel**, which takes two expressions as arguments and produces a labelled combination of them, just like the conventional **merge** operation. This new asymmetric operation, and the symmetric **combine** operation, construct all and only the same expressions as when labelling occurs separately, because it was already the case that an unordered pair $\{\alpha, \beta\}$ could never be labelled α and then later labelled β ; labelled constituents were subject to multiple applications of **combine**, but unlabelled constituents were never subject to multiple applications of **label**.

- (69)
- a. $\text{combine}(\alpha, \beta) = \text{combine}(\beta, \alpha) = \{\alpha, \beta\}$
 - b. $\text{combineAndLabel}(\alpha, \beta) = \langle \alpha, \beta \rangle$

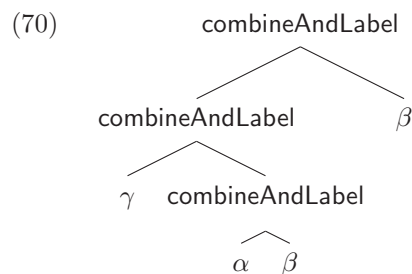
This formulation is sympathetic to the intuition that, given the semantic interpretation of the two kinds of complex objects (labelled and unlabelled), there is something unnatural about first combining, say, a verb and its argument in a symmetric relationship with a conjunctive interpretation, only to change the relationship to one with a completely different interpretation. It also may be preferable given that heavy re-use of syntactic objects, while perhaps instructive in revealing similarities between movement, adjunction and projection as described above, poses its own problems in the BG framework; see §5.2.

At first the compiled `combineAndLabel` operation might seem to conflict with one of the central points of Hornstein (to appear): he suggests that the addition of the labelling operation to a system already equipped with `combine` “supplies the necessary ingredient to get from a flat beads-on-a-string system to a hierarchical nesting system”; this is “the central innovation of UG, the change that enables the peculiar architecture of natural language to emerge”. The idea of identifying one grammatical operation with this consequence is obviously attractive in light of questions about how language might have evolved, so a grammar with `combine` and `combineAndLabel` as its primitives might not seem to have all the appeal of one with `combine` and `label`. But as Hornstein already points out, the claim that the added special ingredient in human language is the cognitive ability to give labels to complex objects produced by `combine`, is not incompatible with the idea that “compiled” or “composite” operations like `merge` or `move` could “be treated as a primitive within a native speaker’s grammar, even though the compiled operation is not itself a primitive of UG”. In other words, the attractive argument about adding the ability to label complex objects can be played out at the lower level of abstraction where grammatical operations are *implemented*.

5.2 Re-using expressions

A prominent feature of the system presented above is the ability to “re-use” syntactic objects. In the BG framework, however, we require that every derivation have the form of a tree, which means that any syntactic object can only be input to a single operation (that which leads to the object represented by its parent in the derivation tree).

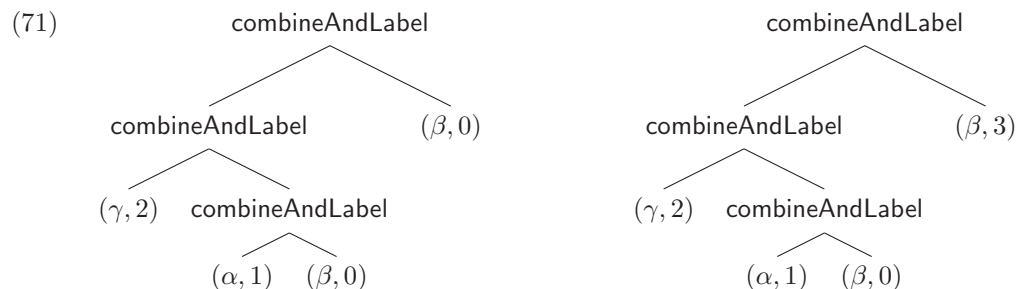
An initially-appealing way to work around this conflict is to allow one lexical item to appear at more than one leaf of the derivation tree. Thus, in representing the derivation of the structure in (65) we would draw a derivation tree like the following (adopting the `combineAndLabel` strategy purely for succinctness, where the “left” operand is that one that determines the label of the result):



There are (at least) two drawbacks to this approach. Firstly, although it has not been addressed in the theory above, we are eventually going to need to represent, distinctly from the derivation in (65), the derivation where *two distinct instances* of β are used; in the conventional informal notation, this would be represented by not drawing the arrow on the tree in (65). We would like to reserve the derivation tree in (70) for this second derivation, with two instances of β each participating independently in the derivation. The second drawback is that whenever a complex (non-lexical) element is to be re-used, we will be forced to duplicate large chunks of the derivation tree.

One sufficiently expressive way to represent the re-use of syntactic objects is to attach an index to each leaf of the derivation tree; then two leaves labelled with the same lexical item represent the same instance of that lexical item if they also share an index, and different instances otherwise. The distinction that (70)

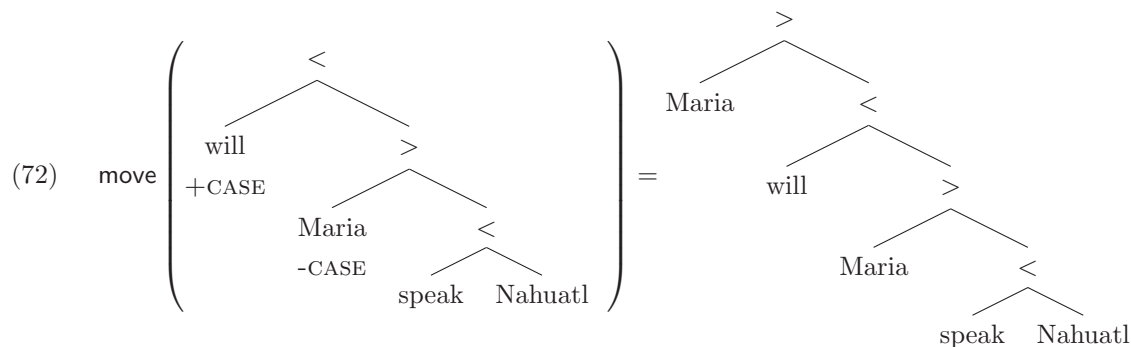
failed to capture can then be indicated as in the two trees in (71):



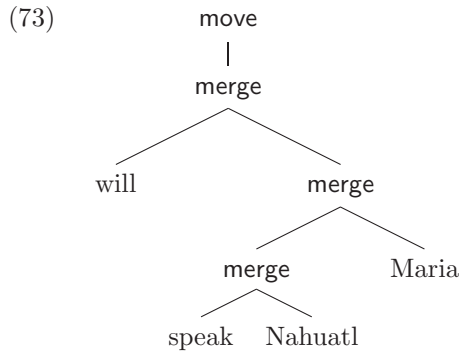
However, this strategy does depart from the constraints of the BG framework.

To head towards a way to express this movement in a BG-compatible way, we can note that we currently have the power to express many, many more movement operations than we are actually going to need in writing natural language grammars. When describing the last step in the “movement” derivation of (71), we provide $(\beta, 0)$ as an operand to `combineAndLabel` to indicate that this is the object that is “moving”, or at least, “re-merging”. Note that this means that we can express the movement of *any* subpart of the tree to merge with the root. But movement in natural languages is, of course, severely restricted. If we formulate restrictions on movement such that, given any structure τ , at most one subpart of τ is allowed to re-merge with the root of τ , movement can be formulated as a one-place operation.

This is the approach taken in the Minimalist Grammars (MGs) of Stabler (1997). These grammars adopt the idea that every operation is driven by the need to check features of lexical items. Since the order in which a particular lexical item’s features must be checked is predetermined, the subtrees which are candidates for movement are restricted to those that can check the currently-outstanding feature of the root. An added “economy” condition stipulates that the movement operation can only occur when exactly one such candidate subtree exists. This allows a one-place operation `move` to be defined, as demonstrated in the following example (from Stabler, 1997):



From the input tree it is clear that the next operation must check the `+CASE` feature of ‘will’. The only subtree capable of checking this feature is ‘Maria’, so this movement can be specified by a unary operation; there is no need to specify the moving subtree as an operand. The derivation of which (72) illustrates the final step is unambiguously represented by the following derivation tree:



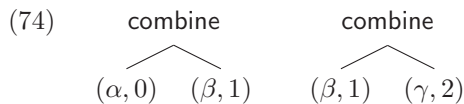
Note that this allows ‘Maria’ to appear at just a single leaf of the derivation tree. Its position in the derivation tree completely determines its role throughout the derivation, including the fact that it is eventually moved to check the +CASE feature of ‘will’.¹⁷

Considering these properties of MGs, it appears that (at least part of) the problem of specifying how objects are re-used could be overcome by explicitly relating syntactic structure-building to something like feature-checking. The focus in §3 was on the mechanics of the operations and the structures they build, putting aside questions of when these operations must, may and may not apply. The need to establish certain relations between syntactic objects did guide the exploration of structures it was possible to build, but the formal machinery massively overgenerates. Future work restricting this generative capacity to something more in line with the true properties of natural language grammars, by driving derivations with feature-checking, will hopefully lead to a formulation compatible with the BG framework the way it does for MGs.

5.3 Derivations do not have a unique “root operation”

As mentioned in §2.3.1, derivations of the sort presented here are not tree-shaped, for two reasons: firstly because (on the most transparent formulations, at least) some nodes have multiple parents, and secondly because there is no unique root node. The previous section on representations of “re-use” addresses the former, but for completeness we should note that the latter remains inconsistent with the BG framework.¹⁸

Consider, for example, the derivation represented in (66). The corresponding derivation structure (adopting the indexing approach to the re-use problem, for temporary expository purposes) would seem to be:



This structure does not have a single root node because the final derived object is not the output of any one application of a syntactic operation.

¹⁷MGs differ from the grammars explored above in that they distinguish between “external merge” and “internal merge”, representing these by the `merge` and `move` operations respectively. In this respect the related Sideward Movement Grammars (SMGs) of Stabler (2006) more closely resemble the present proposal. In these grammars both kinds of merge are represented with a single unary `merge` operation, and just as in MGs `move` is restricted to displacing only certain subtrees of its operand, SMGs’ `merge` is restricted to affecting only elements which meet certain requirements, but these requirements are independent of whether or not the “moving” element has already been “merged into the tree”. This restriction again comes from limiting the configurations in which features can be checked.

¹⁸In §2.3.1 it was noted that the non-tree shape of the derivations was not compatible with conventional methods of computing compositional semantic values. This question of how to interpret the structures has been answered with the unconventional interpretation procedures in §3.4, but the formal inconsistency with the BG framework, for what it is worth, has not.

6 Conclusion

One might reasonably ask how a system of operations, basic though they may be, that are fairly clearly not carried out in real time in the course of language use, can be proposed as a solution to the two motivating problems presented earlier: Darwin’s Problem and the Granularity Mismatch Problem. It might appear that a solution to Darwin’s Problem must tell us what it is that humans came to be able to *do* that their ancestors couldn’t, and that a solution to the Granularity Mismatch Problem will necessarily tell us what one should look for signs of the brain *doing*. How can an abstract derivational system, albeit one composed of very meagre resources, really shed any light on these questions?

One possible interpretation of proposals of the kind presented here is that they are claims about the representations employed in linguistic computation. The idea is that the labelling operation encapsulates all that is distinctive about syntactic representations by describing all that is distinctive about the *structure of the space* of such representations which is “navigated” in real-time computation. With respect to the logical problem of language evolution, the solution put forward is not that humans gained the ability to carry out some procedure they previously couldn’t, but rather to represent a new type of object. A relevant analogy might be the “creative step” required for a device to move from only having representations of the real numbers, to having the ability to represent complex numbers: the function $f(a, b) = a + bi$ encapsulates the structure of the space of complex numbers — it is what makes the complex numbers “what they are”, and characterises a sensible way to think of how they are “arranged” relative to each other and to the real numbers (i.e. the Argand plane) — even though this function need not play any distinguished role in computations (addition, multiplication, etc.) over these new objects. The labelling operation could do likewise for syntactic representations, but with the crucial difference that it “closes” the space of representations such that the new ones (labelled syntactic objects) are subject to a pre-existing operation (combination); while the ability to represent complex numbers must be supplemented with definitions of, say, addition and multiplication over them in order to relate them to each other in any interesting way, the pre-existing combination operation, by stipulation, fills this role for labelled syntactic objects.

Furthermore, with respect to the questions raised by evolution, it is at least logically possible that a characteristic space of representations could be *all* that needs to be added to domain-general and even non-human reasoning skills, to produce the linguistic behaviour we observe. Recent work on models of language learning (Yang, 2004) has emphasised that a domain-general learning mechanism (for example, Bayesian updating) could be employed to search through a hypothesis space that is unique to language. Similarly, it is possible that while we have a structured space of representations that is unique to language, the process of determining, for example, which structural description corresponds to the sequence of phonemes being perceived is carried out by domain-general reasoning abilities; in much the same way that it’s possible to build parsers by supplementing general problem-solving mechanisms with only *declarative statements* about the representations to be built. As Chomsky (1995) notes, “Contrary to common belief, assumptions concerning the reality and nature of I-language (competence) are much better grounded than those concerning parsing”.

Turning to implications for Poeppel and Embick’s Granularity Mismatch Problem, parallels may be drawn between the representational consequences of the syntactic operations proposed here and those of theoretical phonological rules. (The original “ancestors” of current phonological rules and those of syntactic operations were explicitly presented as different levels within one formal system in Chomsky (1975).) Like

syntactic operations, the rules of phonological theory are not generally considered to be carried out in real time (Bill Idsardi, personal communication), so it is not generally expected that an area of the brain will be identified that performs, say, word-final devoicing. However, the fact that the relevant rules require representations characterised by certain distinctive features has allowed researchers to investigate where in the brain, and where in the time course of processing, computations appear to be sensitive to these particular representational distinctions (Poeppel et al., 2008).

Less directly, there are also implications for the real time processes themselves that must be carried out in language use. The minimal nature of the abstract structure-building operations imposes a degree of homogeneity on syntactic structural descriptions (phrase markers) and on the kinds of well-formedness conditions on them that the theory can express. For example, the overarching aim of the proposal in Hornstein (to appear) is to discover a system in which *all* relations among lexical items are established under the ubiquitous combine operation, which is subject to one specific minimality restriction, rather than, say, some relations being established under government, some under binding, some under A-movement and some under A-bar-movement. If the well-formedness conditions have this more homogeneous character, then one might expect a smaller number of coarser-grained real-time procedures to be sufficient in syntactic processing: instead of a procedure for detecting well-formed binding relations, another for detecting well-formed A-movement relations, and so on, one procedure simply for detecting well-formed relations would suffice.¹⁹ If current associations like “syntax is in Broca’s area” are too coarse-grained, and notions like government and binding too fine-grained, for interesting connections between linguistics and neuroscience to be made, then a notion like combination may be closer to the correct granularity.

References

- Boeckx, C. (2008). Bare Syntax. Oxford University Press, Oxford.
- Bošković, Z. and Nunes, J. (2007). The copy theory of movement: A view from PF. In Corver, N. and Nunes, J., editors, The Copy Theory of Movement, pages 13–74. John Benjamins, Amsterdam/Philadelphia.
- Chametzky, R. A. (1996). A Theory of Phrase Markers and the Extended Base. State University of New York Press, Albany, NY.
- Chomsky, N. (1975). The Logical Structure of Linguistic Theory. Plenum Press, New York, NY.
- Chomsky, N. (1995). The Minimalist Program. MIT Press, Cambridge, MA.
- Chomsky, N. (2004). Beyond explanatory adequacy. In Belletti, A., editor, Structures and Beyond. Oxford: Oxford University Press.
- Chomsky, N. (2005). Three factors in language design. Linguistic Inquiry, 36(1):1–22.
- Citko, B. (2005). On the nature of merge: External merge, internal merge, and parallel merge. Linguistic Inquiry, 36(4):475–496.

¹⁹The talk of procedures here is orthogonal to the question of whether anything specifically procedural was part of the evolutionary step leading to language. Even if declarative knowledge was all that was added, the interaction of this knowledge with whatever makes use of it may give rise to identifiable sequences of steps that, while derivative, are repeatedly used in language processing.

- Epstein, S. D., Groat, E., Kawashima, R., and Kitahara, H. (1998). A Derivational Approach to Syntactic Relations. Oxford University Press, Oxford.
- Heim, I. and Kratzer, A. (1998). Semantics in Generative Grammar. Blackwell, Oxford.
- Hodges, W. (2001). Formal features of compositionality. Journal of Logic, Language and Information, 10(1):7–28.
- Hornstein, N. (to appear). A theory of syntax: Basic operations in the minimalist program. Manuscript.
- Hornstein, N. and Nunes, J. (to appear). Some thoughts on adjunction. Manuscript.
- Keenan, E. L. and Stabler, E. P. (2003). Bare Grammar. CSLI Publications, Stanford, CA.
- Kobele, G. (2006). Generating copies. PhD thesis, UCLA.
- Lasnik, H. and Uriagereka, J. (2005). A Course in Minimalist Syntax. Blackwell, Malden, MA.
- Pietroski, P. M. (2005). Events and Semantic Architecture. Oxford, New York, NY.
- Poeppl, D. and Embick, D. (2005). The relation between linguistics and neuroscience. In Cutler, A., editor, Twenty-First Century Psycholinguistics: Four Cornerstones. Lawrence Erlbaum Associates.
- Poeppl, D., Idsardi, W. J., and van Wassenhove, V. (2008). Speech perception at the interface of neurobiology and linguistics. Philosophical Transactions of the Royal Society B: Biological Sciences, 363(1493):1071–1086.
- Stabler, E. (1997). Derivational minimalism. Lecture Notes in Computer Science, 1328:68–95.
- Stabler, E. (2006). Sideways without copying. In P. Monachesi, G. Penn, G. S. and Wintner, S., editors, Proceedings of the 11th Conference on Formal Grammar.
- Steedman, M. (2000). The Syntactic Process. MIT Press, Cambridge, MA.
- Yang, C. (2004). Universal grammar, statistics, or both? Trends in Cognitive Sciences, 8:451–456.