Basic Operations: Minimal Syntax-Semantics

Norbert Hornstein
Paul Pietroski
University of Maryland

Abstract

In this programmatic paper, we articulate a minimalist conception of linguistic composition, syntactic *and* semantic, with the aim of identifying fundamental operations invoked by the human faculty of language (HFL). On this view, all complex expressions are formed via the operation COMBINE(A, B). But this operation is not primitive: COMBINE(A, B) = LABEL[CONCATENATE(A, B)]. We take labeling to be a computationally simple but perhaps distinctively human operation that converts a mere concatenation of expressions, like A^B, into a more complex unit like [$_{A}$ A^B], with the subscript indicating a copy of the dominant constituent. We discuss several virtues of this spare conception of syntax. With regard to semantics, we take instances of COMBINE(A, B) to be instructions to build concepts. More specifically, we claim that concatenation is an instruction to conjoin monadic concepts, while labeling provides a vehicle for invoking thematic concepts, as indicated by the relevant labels.

Key words: basic operations, concatenate, label, copy, conjoin, close.

Table of Contents

1. Introduction

We take it as given that humans have a faculty of language, stable states of which can be described as I-languages in Chomsky's (1986) sense. Idealizing, one can describe HFL as a biologically implemented capacity to acquire and use (in normal conditions of experience) one or more recursive *procedures* that associate phonological instructions to articulatory/perceptual systems with semantic instructions to conceptual/intentional systems, by means of a constrained syntax; cp. Chomsky (1995).

Abstracting from phonology, we assume that expressions of an I-language *are* semantic instructions whose structural properties are determined by (i) principles

governing the combination of I-language elements, and (ii) constraints on how HFL can naturally "interface with" other aspects of cognition. From this perspective, one joint task for syntacticians and semanticists is to specify the relevant elements and structural properties, along with a biologically implementable algorithm for generating complex expressions, which can be treated as instructions to construct mental representations of some kind from language-independent cognitive resources.

In developing a specific proposal, we also want to stress two dovetailing points. First, while "poverty of stimulus" arguments reveal the inadequacy of empiricist accounts of language acquisition, appealing to innate principles is only a first step towards explaining how and why children acquire the languages they do. Second, morals from the study of vision remain relevant for the study of language; see Poeppel and Embick (2005), discussed below.

Regarding the first point, we take it as given that HFL respects logically contingent language-specific constraints, which can be described as an innate Universal Grammar (UG) that all children use in acquiring their sundry I-languages. This raises familiar questions about language variation. But even if nativists can account for such variation, say in terms of lexical and parametric differences, getting beyond a mere *description* of the child's innate endowment requires some account of *how* constraints of UG are enforced by HFL and *why those* constraints are the ones enforced. Could a different faculty—one that imposed alternative constraints on acquirable languages—have emerged from the available biological building blocks. Or is HFL the only linguistic system that creatures like us could have had? Or was there a small range of possibilities, with a combination of selective pressure and contingency leading to the actual result?

Theorists should address such questions, in accounts of why language acquisition is constrained in the ways it is, and providing answers requires thought about how constraints of UG might be implemented. The space of "possible language faculties" depends on which operations can be carried out by a biology like ours, and how these operations can be recruited to recursively combine lexical correlates of concepts. Moreover, the natural history that led to HFL was presumably one in which some new but modest capacity was added to a primate mind. In which case, if one wants to explain how a certain line of primates acquired a capacity to acquire human languages, one needs an account of how this talent could have been unleashed by a small biological change.

This brings us to the second point above. We view the task of getting beyond mere description of innate linguistic constraints, and moving towards an explanation of how and why HFL has just these constraints, as akin to the task described by Marr (1982) in the context of studying vision. Given a description of the input-output profile for some posited cognitive system, one can and should ask *how* that system computes the relevant function, bearing in mind that nature has somehow realized the actual algorithm. And at least initially, it can be useful to abstract from various details of the input-output profile, in order to identify some fundamental operations—perhaps corresponding to an important subsystem—that are especially good candidates for realization. The study of how animals represent distal geom-

etry was advanced by focusing on how a computational system might detect edges, given certain information, and then asking how the relevant computations might be implemented by available wetware. In linguistics, syntacticians and semanticists share the question of how humans associate strings of words with complex meanings. We propose to address this question by investigating how a simple system, equipped to perform just a few basic operations, might recursively combine lexical correlates of concepts in a way that approximates what humans do in this regard. Ideally, discrepancies between the simple model and observations will be due to complicating interactions between the core system and other cognitive systems deployed in language use.

This is not to deny or even ignore the salient differences between (the studies of) language and vision. On the contrary, and in particular, we take it to be an explanandum that I-languages are distinctly human procedures for generating expressions. Other animals do not—and apparently cannot—associate the sounds of English (or signs of ASL) with all and only the meanings that children can naturally associate with these distinctively human signals. Yet the procedures that children easily acquire are somehow implemented with biological circuitry that is presumably not different in kind from that of our evolutionary cousins; cp. Poeppel and Embick (2005). So our working hypothesis is that I-language composition *adds something* to more ancient cognitive operations that are available to other animals. But other things equal, any posited "human twist" should be minimal—both methodologically, and in terms of any novel circuitry required.[1]

Many theorists will not share our particular biolinguistic conception, adopted here without argument, of the theoretical project that syntacticians and semanticists share. But however one thinks about the target of inquiry in this domain, it is all too easy to simplify an extant theory of syntax by positing a sophisticated (and perhaps unimplementable) mapping from the allegedly simple grammatical forms to the corresponding mental representations. Likewise, one might purchase simplicity in semantics at the cost of grammatical forms that are not generable by a

1. Contrary to what some critics of minimalism have suggested, one can pursue this program while assuming, as we do, that Government and Binding proposals are roughly correct. We view Minimalism as an attempt to simplify GB proposals, abstracting away from certain details concerning the best descriptions of particular languages, with the aim of connecting the theoretical core of such proposals with the broader and ultimately biological questions gestured at above. Explaining how humans (came to have a capacity to) acquire languages, and why these languages are governed by the principles that linguists discover, requires theoretical work of a sort familiar in other sciences. In addition to seeking more refined principles, which permit better and more detailed descriptions of particular languages, one must also look for underlying elementary principles that may be far removed from the observations and generalizations that initiated the inquiry. Because the GB program was so successful, in providing a sketch of what the human faculty of language must be like, one can and should ask which basic operations this faculty employs. Mature language users presumably supplement these operations with many cognitive and communicative strategies that have been learned or otherwise acquired. But even if full linguistic competence requires more than mere facility with the basic operations employed by the language faculty, narrowly described, identifying these operations remains a central theoretical task in linguistics; see Hauser, Chomsky, and Fitch (2002).

procedure that children can naturally come to implement. But this merely shifts explanatory burdens in unhelpful way. There are various ways to do better. And our aim here is not to argue against any. Instead, we want to illustrate the potential virtues of adopting a particular minimalist strategy: offer very spare conceptions of syntax *and* semantics, *both* of which might initially seem to be descriptively inadequate; then show how the conceptions dovetail in ways that make the theoretically attractive package empirically plausible.

To advertise, we think that combining expressions in an I-language is a uniform operation—COMBINE(A, B)—applicable to diverse pairs of expressions that include lexical atoms, functional items, and endlessly many phrases. This operation is somehow asymmetric, since at least many complex operations are "headed," as opposed to being mere concatenations of constituents. But in our view, COMBINE is not a primitive operation. It is rather the composition of two operations, CONCATENATE and LABEL;

$$\text{COMBINE(A, B)} = \text{LABEL[CONCATENATE(A, B)]}$$

where CONCATENATE(A, B) just is the concatenation of A with B, either or both of which may already be labeled.[2] The idea is that LABEL is a computationally simple but perhaps distinctively human operation that converts a mere concatenation like A^B into something more, like $[_{A}$ A^B], with the subscripted copy of A indicating that this concatenate (as opposed to B) is the label/head of the new expression; see Hornstein (2009). Concatenating the labeled expression with a third expression C would thus yield $[_{A}$ A^B]^C, which might in turn be labeled with A or C, as opposed to the mere concatenation A^B^C. In section two, we discuss several virtues of this spare conception of I-language syntax.

Abstracting from phonology, we take each instance of COMBINE(A, B) to be a semantic instruction. So let's say that **SEM**{COMBINE(A, B)} is the semantic instruction created by combining the simple expressions A and B. On our view, then, the meaning of any complex expression is the meaning of a labeled concatenation.

$$\textbf{SEM}\{\text{COMBINE(A, B)}\} = \textbf{SEM}\{\text{LABEL[CONCATENATE(A, B)]}\}$$

Concatenation is presumably an ancient operation, widely available in animal cognition.[3] One might well expect its semantic correlate to be equally basic and not specifically linguistic. In any case, we suggest that concatenation is an instruction to *conjoin monadic concepts*, and that labeling provides a vehicle for *invoking thematic relations* when (and only when) the labels of concatenates *conflict*; see Pietroski (2005, 2006).

2. By contrast, one might follow Chomsky (1995, 2000) in positing an asymmetric operation—MERGE(A, B)—that is not decomposed into suboperations.
3. See, e.g., Gallistel (1993).

The idea is that while **SEM**{[$_A$ A^B]} is always an instruction to invoke a concept via A and conjoin it with another, there are two possibilities for how the second concept is related to B. In expressions where the label of B is compatible with A, the second concept is invoked directly via B itself. (Think of adjuncts, as in 'blue armadillos', corresponding to a concept of things that are both armadillos and blue.) But in cases where the label of B conflicts with A—think of arguments, as in 'eat beets'—the second monadic concept is complex and partly thematic, like EAT(E) & ∃X[PATIENT(E, X) & BEETS(X)], which has *a thematic constituent that includes* a concept invoked via B.

2. Concatenate, Copy, and Label

As biologically instantiated cognitive systems, natural language grammars (NLGs) seem special in several ways. Perhaps most distinctively, they generate an unbounded number of objects with unique hierarchical properties (e.g. endocentric X' structures). In addition, they exploit relations like c-command, and obey locality conditions like Minimality, in generating expressions that respect generalizations like Structure Preservation (among others). These features of NLGs have no obvious analogues in animal minds/brains. One important minimalist question asks to what degree this apparent distinctness is real, and how the genuinely distinctive aspects of NLGs interrelate with other cognitive operations to yield grammatical structures. Recall that the more exceptional grammatical operations are, the harder it is to explain their emergence and their physical embodiment. This suggests that the apparent distinctness is largely superficial. One way to vindicate this thought is to decompose operations like Merge and Move into simpler more generic basic components and to derive the attested grammatical properties from their interactions. As such, we take Move and Merge to be complex operations: Move is the combination of Copy and Merge, while Merge is the combination of Concatenate and Label. Neither of these assumptions is especially novel. But in our view, some important consequences of this decomposition have not yet been adequately explored or appreciated.[4]

2.1. Is C-Command Primitive?

As an illustration, consider the following question: why are chains subject to c-command? Why does the head of a chain c-command its foot? The GB answer is that chains are grammatical objects *defined* in terms of c-command. This effectively builds the c-command condition into UG, thereby making the c-command requirement a primitive fact about NLGs. But one can answer the question differently, inviting a substantive and explanatory reduction of the fact (revealed by GB) that chains are subject to c-command. Perhaps chains are *constructed* in a way that

4. The program outlined here is developed in Hornstein 2009. It bravely goes where many have gone before; most prominently Chomsky's myriad minimalist papers, Robert Berwick 1997 and Samuel Epstein 1999.

requires the head to move to a position in which it c-commands the foot. This kind of proposal can differ from the first, in interesting ways, depending on the details of how chains are constructed.

Here is one story. Chains are formed by Move, which is composed of simpler operations, Copy and Merge (Chomsky 1995, Nunes 2004). Merge is subject to the Extension Condition (EC), which only allows elements to be added to phrases at the top. The operation in (3) meets this condition. The one in (4) does not.

(3) A MERGE $[_C$ …C X B….] → $[_C$ A $[_C$ …C X B…]]

(4) A MERGE $[_C$ …C X B….] → $[_C$ …C A X B…]

If Move has Merge as a subpart, then Move is subject to EC. Consequently, Move results in chains where each link c-commands its predecessors. Consider the chain that results from moving X in (5).

(5) $[_B$ A $[_C$ …C X B…]] MOVE X → $[_B$ X $[_B$ A $[_C$ …C X B…]]]

If moving X is matter of copying X and merging it (in an EC-obeying way) with an expression of which X is a constituent, then the copy of X must merge at the top. In which case, the head of the X-chain must c-command its foot. In this way, we can explain/derive the fact that chains are subject to c-command.[5]

This approach immediately invites another question: why does Merge obey EC? One possible answer is that Merge conforms to EC because EC is a "nice" computational property. Perhaps it is somehow good for computational systems like the language faculty to lose no information, in the sense that inputs of a Merge operation are *recoverable* in the output. Note that in (3), each of the inputs—'A' and '$[_C$ …C X B…]]' —is a constituent of the output expression, '$[_C$ A $[_C$ …C X B…]]'. This is not the case in (4). Thus, EC enforces a monotonicity requirement on structure-building operations: they can add but not substract information. (This kind of requirement is characteristic of certain subprocesses in vision.) If it is good to be able to recover a phrase's parts—say, to compute some aspects of its phonological or logical form—then EC would be a good property for structure building operations like Merge to respect. So perhaps EC applies to structure building operations because the language faculty is well designed, and well designed systems have "nice" computational properties like monotonicity.

We prefer a less teleological account. But sketching our proposal requires a brief detour. Consider again the fine structure of Merge. It does two separate (though related) things. First, it is an operation that combines two elements into a new object. Second, the object so formed can be subsequently combined with other elements to form further new combinable objects. This invites an analysis according to which Merge is a complex operation with two subparts:

5. The relation of Merge and c-command has been noted in Epstein 1999.

one that simply combines objects; and one that ensures that any output of this first operation can be (an input to this operation, and thereby be) combined with other objects. It is useful, developing this analysis, to contrast Merge with Concatenate.

Concatenation is, computationally, an elementary mode of combination.[6] Like any such operation, it is defined over a set of atoms. What one gets from concatenation depends on which atoms are being manipulated. Concatenating the letters *t, h, e, c, a, t* can yield the complexes *t^h^e^c^a^t* or *t^c^h^a^e^t*, among many others; while concatenating the words *the, cat* can yield (only) *the^cat* or *cat^the*. These complexes are weakly similar—*t^h^e^c^a^t* and *the^cat* have the same orthographic string order—but strongly distinct; *t^c^h^a^e^t* is a possible output of concatenating letters but not words. Specifying the relevant atoms is thus crucial for determining the complexes that can be formed by concatenation. Though correlatively, the operation Concatenate is not itself domain specific. Relevant atoms can include phonemes, syllables, sentences, actions, plans, flowers, galaxies, electrons, or whatever. The mental life of non-verbal beings surely involves concatenating some elements (though not others) into larger sequential objects. So in this sense, Concatenate is presumably not specific to the human language faculty, but instead a much older operation to which something distinctively human could have been added.

It is often assumed that Concatenate is not the operation that knits together the lexical atoms of a sentence as concatenating words/morphemes would appear to yield flat "beads on a string" expressions. Concatenating *the, dog, barks* can yield *the^dog^barks*, but not [[*the dog*] *barks*]; where bracketing indicates genuine linguistic composition. This can make it seem that phrases and sentences, which exhibit hierarchical structure, cannot be concatenations of lexical atoms. But the reasoning here relies on a premise that we reject—viz., that complexes like *the^dog* are not atoms for concatenation, even though their lexical parts are. We agree that phrases and sentences cannot be mere concatenations of lexical items: an additional operation **O**, which effectively marks the result of concatenating two items as a third potential concatenate, is required. But if complexes built via Concatenate can somehow be treated as atomic inputs to subsequent instances of this operation, and so [[*the dog*] *barks*] is reducible to **O**(*the^dog*)^*barks*, then Concatenate does suffice to yield hierarchy given the additional operation **O**.

The idea, familiar in one sense, is that [[*the dog*] *barks*] differs from the^dog^barks in the following respect: in the former but not the latter, *barks* was concatenated with *the^dog*, which was (despite having parts) treated as an atom in the sense of being available for concatenation. From this perspective, [[*the dog*] *barks*] is hierarchical because one of its two concatenates was generated via the operation Concatenate. This frames, without yet answering, the real question: what operation ensures that a complex linguistic thing is a potential input to Concatenate, given that (i) this domain-general operation can be applied to lan-

6. In a Turing machine, concatenating α and β amounts to writing α and β in adjacent tape cells.

guage only via some specification of relevant atoms, and (ii) this specification is presumably given in terms of lexical elements? Or put another way, what makes *the^dog* atomic in the relevant sense, despite its obvious complexity? Our proposal is that Labels do the trick, and that this is one way in which adding a relatively modest operation to a very simple system can have a dramatic effect. Let's see how.

2.2. Labels

Chomsky (1995) analyzes phrase building as consisting of two operations. The first operation Merge, surveyed in (5), takes a pair of atoms and combines them. If this aspect of Merge just *is* Concatenate, as we suggest, it is reflected with (6a). The second operation is labeling. This operation "names" the resulting linguistic object via one of the inputs as in (6b).

(6) a. Concatenate A,B → A^B

b. Label A^B → $[_{A}$ A^B]

The square brackets, with subscript 'A', should be read as saying that a certain object (viz., the result of concatenating A with B) has label A. Note that while Concatenate takes two inputs, each of which may be simple, the operation Label takes an output of Concatenate and returns a similar complex object (with an additional copy of something). The latter operation, we suggest, makes complexes like A^B into potential concatenates—i.e., things "visible" to Concatenate as atoms that can be combined with others.[7]

This is, in some ways, a return to an old idea. In Chomsky (1955), labels on phrase markers were understood to define the 'is-a' relation. For example, (7) identifies V^NP as a VP.

(7) a. VP → V NP

b. $[_{VP}$ V NP]

7. Chomsky has recently discussed this recombinative power in terms of "Edge Features." These are features that primitive lexical atoms have and that combinations of lexical atoms have. The proposal is that labeling is the operation whereby complexes inherit their edge features from those of their sub-elements. What labeling does (e.g., turn a^b into $[_{a}$ a^b]) is allow the complex to be treated as an atom as regards further combination. Note, without something like labeling or edge features there is no well defined notion of the combination operation. What labeling does is extend a more primitive operation of combination/merge/concatenate *given* a primitive one; given edge features then hierarchy follows if complexes inherit these features from their parts in some way. However, the Merge operation is only defined over a class of inputs with some feature; be it an edge feature or a label. Absent this, there is no operation defined at all.

Applied to (6b), the labeling says that A^B is (an) A. So understood, the formal effect of labeling is to generate a *closure* of concatenation within the domain of lexical atoms by mapping each concatenated complex to one of its atomic parts.[8] On this view, the labeling in (6b) implies that A^B *is as concatenable as* A itself. Any system that can generate A^B treats A as a concatenatable item; and given a capacity to use A as a label, a simple system should treat A^B, so labeled, as equally concatenable. One could, of course, design a machine that detects and refuses to concatenate any "type-A expressions that have parts." But absent substantive and encodable assumptions to the contrary, a naturally emerging system that can concatenate A with B should treat anything that counts as (an) A as a concatenatable item. In which case, labeling A^B as (an) A would make A^B an atom for purposes of subsequent concatenation. Thus, by simply adding the operation Label to a system that allows atoms to concatenate, one can get a system that allows unbounded structured hierarchies of the X'-variety.[9]

For labels to work their hierarchy-inducing magic, labels must be understood in a bare phrase structure (BPS) manner. There are no bar levels on the label in (6b): A^B is *not* labeled *A'*, or *AP*, but simply *A*. This is required for the derived object to be further concatenable. For unlike A' and AP, only A is a lexical primitive, and only it can enter into concatenations. This comports with the BPS idea, following Muysken (1982), that bar-levels are (at most) relational properties of a phrase. There is no *intrinsic* syntactic difference between an A, A' and AP. So if Concatenate is blind to relational properties, then as far as this operation is concerned, a labeled concatenate simply is an atom and so available for further concatenation. Put another way: if there is endocentric labeling, and labels are interpreted as defining the "is a" relation in a BPS manner, and Concatenate only recognizes the intrinsic features of items (not relational ones), then this simple operation can yield nested structures.[10]

We were, recall, asking why Merge respects the Extension Condition (EC). And we detoured, to motivate the following claim: *given labels* and the corresponding *derived* labeled "atoms," Merge can be identified with Concatenate. For if Merge is indeed a species of concatenation, and labeling returns atoms, then Merge must be "at the root." The operation Concatenate, which always applies to atoms, cannot see anything but roots. Once the complex A^B is labeled A, as in

8. Pietroski (2006b) explores a Fregean analogy, often noted, that may run deep: one can characterize the natural numbers in terms of zero and the relation *less than*, which is the transitive closure or "ancestral" of the predecessor relation.
9. Given a system that strings atoms of a certain kind together, adding the operation LABEL will yield a system that can generate structured versions of these atoms by the same operation. If the atoms are lexical items, then the structured elements will look a lot like phrases.
10. If this is correct, then Bare Phrase Structure is not only methodologically prized on minimalist grounds, but is required to achieve hierarchy from concatenation. Note that labeling does not turn a phrase into a word, rather it endows it with the combinatorial powers of a word. Nonetheless, phrases are distinct elements, as can be seen by their interpretive properties. Words may assign theta roles, phrases do not. This, however, is a fact about how complex objects are interpreted, not how the syntax treats them.

(6b), the internal structure of [$_A$ A^B] is unavailable for subsequent concatenation; this structure is hidden inside the concatenative atom A. The only eligible target of concatenation is the A-labeled structure. So this is what Merge targets. Overlap is blocked on the assumption that A and B are atomic *at this point in the derivation*, just as "orthographic overlap" is blocked when concatenating words. (Combining *the* and *cat* can yield *the^cat*, but not *thec^at* or *th^ecat*.)

In short, EC applies to Merge because Merge is Concatenate-plus-Label; and EC applies to Concatenate, which is always "at the root," because this elementary operation always applies to atoms. So if labeling is what marks a complex linguistic object a potential concatenate, then nothing "internal" to a complex concatenate will be a possible target of concatenation. Indeed, this is what it is for an expression to be an atom with regard to Concatenate: the expression has no relevant internal structure. This explanation is purely formal, in contrast with the more teleological account of EC sketched above. An operation **O** that converts complexes into atoms via labeling obeys EC *because* all **O** does is combine atoms, and atoms can only be combined at their roots. Of course, if EC turns out to be computationally attractive, for the reasons mentioned above, then a Concatenate-plus-Label system will inherit the relevant computational virtues (like monotonicity). It wouldn't follow, however, that the Concatenate-plus-Label system was somehow selected or favored for its "nice" computational properties. Our account does not require or posit a natural history with alternative language faculties that were somehow less useful. On the contrary, our suggestion is that simple operations make it possible to generate endlessly many expressions, and these operations interact in ways that (necessarily) have certain consequences like EC.[11]

Let us recapitulate. We have been describing, as opposed to stipulating, potentially explanatory relations among several grammatically important notions: c-command, EC, and labeling. We have done this by treating Move as a composite operation, with Merge as a subpart, and then treating Merge itself as a species of Concatenate. Labeling leverages hierarchy from Concatenate by mapping complex concatenates into "derived" atoms. This requires understanding labels in a BPS fashion. Thus, by "decomposing" Merge and Move into Copy, Concatenate, and Label, we are able to relate the c-command property of chains to the hierarchical nature of phrases.

Can this be extended to account for other GB generalizations? We hope so. One promising area of inquiry concerns the structural requirements on rules of construal. In particular, why are bound anaphors and controlled PROs c-commanded by their antecedents? The c-command requirement on antecedents reduces to more basic facts if binding is coded by movement. This supposition has some empirical backing (c.f. Bowers 2008, Hornstein 1999, 2001, 2003, Polinsky and Potsdam

11. If this is correct, then Move is not a species of Merge (i.e. ReMerge) but involves some non-structurally sensitive operation. See Hornstein 2009 for discussion where the Copy operation is suggested to be this non-structurally sensitive operation. In effect, Copy need not be sensitive to constituency because Merge/Concatenate already is. This is more fully discussed in Hornstein 2009 and alternatives are considered which relax this atomicity assumption.

2002, Zwart 2002, Lidz and Idsardi 1997 etc).[12] If bound pronouns are also products of movement (c.f. Kayne 2002) then principle B effects start falling into line as well. A natural project would be to show that movement underlies all binding phenomena, and that this is the source of the corresponding c-command constraint(s).

C-command also plays a role in computing minimality, characterized in (10).

(10) Minimality Condition:
A movement operation cannot involve X^1 and X^3 over an X^2 which is relevantly identical to X^3 if X^2 c-commands X^3:…..X^1….X^2….X^3….

So we can once again raise the question of *why* c-command is relevant.

Here is a suggestion. The intuition behind Rizzi's original proposal is that grammars prefer shorter dependencies to longer ones. In other words, minimality codes a preference for relations among elements/positions that are as *short* as possible (see Rizzi 1990, Chomsky and Lasnik 1993). On the assumption that dependencies incur a computational cost, *minimizing* dependency length is what we expect.[13] So the next question is: how do grammars evaluate distance? How is the length of a dependency determined, such that shorter ones trump longer ones? Or echoing a Marr-style question, how is distance grammatically computed?

One obvious thought is that grammars measure distance by the nodes intervening between related expressions, i.e. a *path* (see Pesetsky 1982, Kayne 1983, May 1985), and that minimality amounts to the injunction to minimize path length.

Paths are the natural measures of distance within hierarchically structured objects, especially if those objects have no sequential order. Indeed, it is hard to see how else to code proximity between items in a hierarchically structured object like a phrase marker in which elements have not been linearized.[14] So given that phrases are hierarchically ordered, one might well expect paths to be the measures of linguistic distance.

Consider an example to fix ideas. In (11), the path of the *what* targeting C^0 is the set of nodes {VP, vP, TP, CP}. These are the maximal projections that dominate the launch site of *what* (in VP) and its landing site (in CP).

(11) $[_{CP}$ what C^0 $[_{TP}$ who T^0 $[_{vP}$ v $[_{VP}$ buy what]]]]

These paths provide a natural measure of the distance between *what* and Spec C. The preference for short dependencies can be recast as the maxim that path length

12. This is clearly related to the project first outlined in Koster 1984 and many papers since.
13. The cost of longer dependencies makes sense on a variety of natural assumptions—e.g., that dependencies require various kinds of memory resources to be coded, and that the longer the dependency the greater the resource requirements. The exact specification of memory cost is not relevant, so long as it has some nonneglible cost.
14. This is the standard current assumption for any theory in which Spell Out is what sequentially orders terminals in a phrase.

should be minimized. For example, if some expression moves to check a feature of the target—like the WH feature on C^0 in (11)—the grammar accomplishes this with the shortest possible move. This raises the question of how grammars *compare* paths.

Assume that NLGs are restricted to Boolean measures[15] and so the relative size of two finite sets is fixed whenever one is a proper subset of the other. In the example above, the *who*-path {CP, TP} is a proper subset of the *what*-path {CP, TP, vP, VP}. So the former path is shorter in the relevant sense. If we assume that grammars compare path lengths by computing the relevant subset relations (i.e. that path comparison is restricted to Boolean comparisons), it follows that minimality is constrained by c-command, as (10) requires.

This is illustrated by considering a structure like (12) and comparing it with (11).

(12) $[_{CP}\ C^0\ [_{TP}\ [_{DP} \ldots \text{Wh2}]\ T^0\ [_{vP}\ t_1\ [_{VP}\ V\ \text{Wh1}]]]]$

The paths of Wh1 and Wh2 to C^0 are as follows: P(Wh1) = {VP, vP, TP, CP}; and P(Wh2) = {DP, TP, CP}. Neither is a proper subset of the other. So neither path is longer than the other, in the sense of length that matters here. In general, paths will be comparable with respect to length just in case they meet the c-command condition in (10). If grammars prize shorter dependencies over longer ones, and the language faculty uses Boolean resources to evaluate grammatical options, then (10) will be a *correct but not basic* description of the facts—and it will follow that only c-commanding elements are relevant to minimality.

This conception of minimality has many further interesting consequences that we can only mention here.[16] For example, given this conception, the A-over-A condition becomes a special case of minimality. Another consequence is that multiple specifiers of a common head are equidistant from a target; that is, the equidistance principle in Chomsky 1995 is a special case of minimality so understood. Moreover, combined with the BPS interpretation of labels noted above, we can explain why only MaxPs count in computing paths. Last, this way of understanding minimality comes close to deriving structure preservation conditions—why DPs move, D's don't, and D^0s move only in limited ways. We leave the derivation of these conclusions as an exercise for the reader.[17]

Minimalist considerations suggest that basic grammatical operations and principles are not (largely) *sui generis*. We have proposed one way of decomposing some standardly posited operations and principles in terms of simpler ones that are more cognitively generic. And we have shown how a system that combined these simpler operations, in the ways outlined above, would appear to have many distinctive properties that characterize NLGs (e.g. hierarchy, headedness, structure preservation, c-command etc). In the next section, we argue that our basic opera-

15. This is plausibly the weakest assumption one can make and, given minimalist reasoning, the favored conclusion.
16. For further elaboration see Hornstein 2009.
17. One derivation is outlined in Hornstein 2009.

tions also make good semantic sense in the context of neo-Davidsonian theories of meaning.

3. Conjoin, Thematically Relate, and Close

We have suggested three basic grammatical operations: concatenation, a certain kind of labeling, and copying.[18] One wants to know how these operations are related to meaning (and what this implies for lexicalization). Moreover, absent a specific semantic proposal, one might worry that the envisioned syntax is less minimal than it initially appears to be.

To take an extreme example, imagine a theory according to which concatenation signifies function-application, each label is an opportunity for type-shifting, and copies are bindable variables that can be targets of an unrestricted abstraction operation. Given a suitable lexicon, such a theory might be descriptively adequate, in the sense of letting theorists compositionally represent any meaning that any speaker (young or old) can assign to any string. But the cost would be massive overgeneration, in the sense of permitting endlessly many <string, meaning> pairs that speakers cannot naturally generate. In our view, this kind of explanatory failure reflects overly powerful description of the faculty that makes I-languages possible, as opposed to insufficient description of independent cognitive constraints (e.g., memory) on human use of linguistic composition.

Correspondingly, our goal is to associate basic grammatical operations with basic semantic operations that have the following properties: they are, together, *just* powerful enough for descriptive adequacy (given the proposed syntax); yet they are still natural operations, in the sense of being plausible candidates for implementation by children—and at least largely, by our recent nonhuman ancestors. To the extent possible, we also want any novel twists that make human syntax and semantics *distinctive* to be two sides of one coin. In the context of our proposal that labeling is the key to making human syntax out of mere concatenation, we suggest that labeling is also the vehicle for making human semantics out of some ancient operation associated with concatenation.

Unsurprisingly, then, we suggest that concatenation has a simple and uniform interpretation across all expressions. In particular, we think that concatenation signifies conjunction of monadic concepts, and hence that cases of elementary *adjunction*—as in 'red ball'—are paradigms of semantic composition. The flip side of this proposal will be that *labeling mismatches* (as with heads and *complements*) invoke thematic concepts. This inverts a more traditional picture (discussed in 3.2) according to which concatenation signifies saturation of one concept by another.

18. Copying is mentioned in note 11. It is more seriously discussed in Hornstein 2009, to which the interested reader is referred for details.

3.1. Concatenation Signifies Conjunction

If only for simplification, let's say that (i) an atomic expression is an instruction to *fetch* a concept, of some appropriate type, that was linked to the expression in the course of lexicalization, and (ii) combining expressions is an instruction to *combine*, via some operation, concepts that have been fetched or formed via the expressions. Let **CONCAT** be the operation signified by concatenation. Then A^B, the mere concatenation of A with B, would be the following complex instruction: apply **CONCAT**, whatever operation that is, to concepts obtained by executing the subinstructions (subexpressions) A and B. But it is an open question whether there are any grammatical expressions of the unlabeled form A^B. Our hypothesis is that **CONCAT** is an operation of monadic concept conjunction, and correlatively, that atomic expressions are instructions to fetch monadic concepts.[19] And our view allows for unlabeled concatenations; perhaps cases of pure adjunction are examples (cp. Chametsky [1996]). But for these purposes, it will be simpler and more illustrative to suppose that every expression has a label, with one caveat.

Our view is compatible with the idea that labels are functional formatives, which combine with unlabeled lexical *roots*, and hence that a word belonging to a lexical category is already a complex expression. For example, the noun 'stab' might be the result of concatenating a root √stab with a nominalizing element that also labels the resulting expression—$[_N$ √stab^*N*], a.k.a. stab_N —while the homophonous verb $[_V$ √stab^*V*], a.k.a. stab_V, has a different covert constituent; see Halle and Marantz (1993), Borer (2005). This leaves room for the idea that phrases like 'red ball' are labeled concatenations of an unlabeled *root* with a labeled head, as in $[_N$ √red^ball_N], where the noun may itself be complex. We can accommodate this idea.

$[_N$ √red^ball_N] =
LABEL[CONCATENATE(√red, ball_N)] =
LABEL[CONCATENATE(√red, LABEL[CONCATENATE(√ball, *N*)]]

Note that in such cases, if such there be, there is no "choice" about how to label the concatenations. In the example above, *N* is the only label, and hence the only projectable label. One can go on to speculate, drawing on Baker (2005), that *N* is an instruction to fetch a formal concept like INDEXABLE(_), while *V* is an instruction to fetch a formal concept like TENSABLE(_); see Pietroski (forthcoming).

Nonetheless, $[_N$ √red^ball_N] is labeled, and thus not a mere concatenation of the form A^B. So absent a hypothesis about the role of labeling, a hypothesis about **CONCAT** doesn't determine the semantic character of $[_N$ √red^ball_N]. Put another way, let **COM** be the operation signified by COMBINE. Then on our view, **COM** differs from **CONCAT**, much as COMBINE differs from CONCATENATE.

19. See Pietroski (2007, forthcoming) for discussion and independent defense of the idea that lexical items are instructions to fetch monadic concepts, even if the concepts lexicalized are not monadic. The idea is that lexicalization may be a partly creative process that involves abstraction of a monadic concept from a nonmonadic concept lexicalized.

In many cases, **COM**(A, B) ≠ **CONCAT**(A, B). For example, [$_V$ stab$_V$^Caesar$_N$] is not an instruction to conjoin concepts fetched via stab$_V$ and Caesar$_N$. So we need to say how **COM** differs from **CONCAT**, in terms of a spare but descriptively adequate semantic role for labels.

One way of encoding this idea is by treating constituents of *labeled* expressions as *relativized* instructions. For example, [$_N$ √red^ball$_N$] can be treated as a *label-relativized* instruction: execute the subinstruction √red-as-*N*, execute the subinstruction ball$_N$-as-*N*, and conjoin the two resulting concepts. Similar remarks apply to [$_V$ √stab^*V*], [$_V$ stab$_V$^Caesar$_N$], and [$_D$ every$_D$^ball$_N$]. Formally, this introduces a degree of freedom.[20] But the interpretive effect of this formal freedom may be minimal.

The phrase [$_N$ √red^ball$_N$] is explicitly labeled as an instruction of the *same type* as the word ball$_N$. And a plausible constraint is that relativizing a label to itself has no semantic effect. In which case, ball$_N$-as-*N* just *is* the instruction ball$_N$. So if concatenation signifies conjunction, [$_N$ √red^ball$_N$] is plausibly an instruction to execute the subinstruction ball$_N$ and conjoin the result with a concept formed by executing the subinstruction √red-as-*N*. If relativizing an unlabeled root to a label also has no semantic effect, then √red-as-*N* just *is* the instruction √red, and the labeled phrase [$_N$ √red^ball$_N$] is interpreted as the mere concatenation √red^ball$_N$ would be interpreted.[21]

COM(√red, ball$_N$) = **CONCAT**(√red-as-*N*, ball$_N$-as-*N*)
= **CONCAT**(√red, ball$_N$)

Likewise, if ball$_N$ = [$_N$ √ball^*N*], this would be a simple instruction to conjoin concepts fetched via the root √ball and label *N*, with the result that [$_N$ √red [$_N$ √ball^*N*]] is an instruction to construct a concept like RED( ) & BALL( ) & INDEXABLE( ).

More generally, labeling might have a semantic effect that *distinguishes* a labeled expression from the corresponding unlabeled concatenation *only* when the constituents/concatenates have *competing* labels. The instruction **COM**(A, B) might differ from **CONCAT**(A, B), in terms of concepts constructed, only when A and B differ in ways that require resolution about the *kind* of instruction that the resulting phrase— LABEL[CONCATENATE(A, B)]—is to be. In section 3.2, we'll contrast this idea with a structurally similar but more traditional view that associates concatenation with function-application. But first, let us sketch our neo-Davidsonian

20. This is a highly restricted version of the more descriptively powerful relativization introduced, for different reasons, by Higginbotham (1986). But it mirrors Higginbotham's (1985) distinction between theta-linking and theta-marking.

21. We suspect that 'red ball' may exhibit a kind of ambiguity, as between [$_N$ √red^ball$_N$] and a more complex expression—perhaps [$_N$ [$_N$ red$_A$^*one*]^ball$_N$], with a covert pronominal coindexed with 'ball'—that has roughly the following meaning: ball that is red for a ball; cp. 'big ant'; cp. Higginbotham (1985), Pietroski (2005, 2006b). But we cannot here consider the voluminous literature on comparative constructions; see, e.g., Kennedy (200x) and references there.

proposal about [$_V$ stab$_V$^Caesar$_N$], assuming for now that stab$_V$ is an instruction to fetch a monadic concept of events (stabs), and that Caesar$_N$ is an instruction to fetch a monadic concept—like CAESARIZER(X), or perhaps CALLED(X, 'CAESAR') & INDEXED(X, *1*)—that applies to exactly one person.[22]

Labeling introduces grammatical relations, like *being the internal argument of a predicate*, that are plausible vehicles for a limited range of "participation" relations like *being the Patient of an event*. This suggests that certain grammatical relations are devices for invoking thematic relations. There are many ways of encoding this idea. But suppose that being an internal argument of a predicate is a two-step "macro" instruction: execute the subinstruction issued by the argument expression itself, thereby obtaining some monadic concept C(X); and immediately form the corresponding complex monadic concept ∃[C(X) & INTERNAL(_, X)]. We assume that the relevant conjunction and (existential) closure operations are not distinctly human.

This treats the relativized instruction Caesar$_N$-as-V as an instruction to create a concept of things that have Caesar as their internal participant. We assume that while stabs are distinct from ordered pairs of individuals, like {Brutus, {Caesar}}, stabs still have internal participants. And one can say that internal participants of stabs are Patients.

$$\forall E\{\text{STAB}(E) \supset \forall X[\text{INTERNAL}(E,X) \equiv \text{PATIENT}(E, X)]\}$$

Knowing this may be part of knowing what 'stab' means. Other verbs may be used to fetch concepts of states or other "eventualities" whose internal participants are unaffected Themes.[23] But in any case, for verbs of any given thematic class, the formal concept INTERNAL(E, X) can be replaced with a more specific thematic content. And whatever one says about the role associated with being the direct object of 'stab', one can say that **COM**(stab$_V$, Caesar$_N$) = **CONCAT**(stab$_V$, Caesar$_N$-as-V); where **CONCAT**(A, B) is a macro instruction to execute A, execute B, and conjoin the two resulting concepts.

Likewise, suppose the external argument (specifier) of a verb phrase is grammatically distinguished from the internal argument (complement): by being the argument of a covert "small verb," as in [$_v$ Brutus$_N$^[$_v$ *v*^[$_V$ stab$_V$^Caesar$_N$]]]; or because being combined with a complex V, as in [$_v$ Brutus$_N$^[$_V$ stab$_V$^Caesar$_N$]], is recognizably different from being combined with a lexical V. Either way, one can say that the external argument (Brutus$_N$-as-external) is the following relativized macro: execute the subinstruction issued by Brutus$_N$ itself, thereby obtaining some monadic concept B(X); and immediately form the corresponding concept, ∃X[B(X) & EXTERNAL(_, X)].Then executing the external-argument instruction will yield a concept of things that have Brutus as their external participant. So given lexical information to the effect that external participants of stabs are agents,

22. See, e.g., Burge (1973), Katz (1994), Longobardi (1994).
23. For relevant discussion, see Dowty (1991), Pesetsky (1982), Kratzer (1996), Baker (1997) and references there; see also Pietroski (2005), Williams (2007).

$$\forall E\{\text{STAB}(E) \supset \forall X[\text{EXTERNAL}(E,X) \equiv \text{AGENT}(E, X)]\}$$

'Brutus stab Caesar' can serve as an instruction to build a concept like the following:

$$\exists X[\text{BRUTUSIZER}(X) \,\&\, \text{AGENT}(\_, X)] \,\&\, \text{STAB}(\_) \,\&\, \exists X[\text{CAESARIZER}(X) \,\&\, \text{PATIENT}(\_, X)].$$

In cases where the verb is tensed, the tense morpheme can be treated as an instruction to fetch a temporal monadic concept, and existential closure (on the only open variable) will yield a truth-evaluable claim like (13).

(13) $\exists\{\exists X[\text{BRUTUSIZER}(X) \,\&\, \text{AGENT}(\_, X)] \,\&\, \text{STAB}(\_) \,\&\, \text{PAST}(\_) \,\&\, \exists X[\text{CAESARIZER}(X) \,\&\, \text{PATIENT}(\_, X)]\}$

Alternatively, the untensed clause may be the internal argument of a larger verb phrase as in 'see Brutus stab Caesar',

$$\text{SEE}(\_) \,\&\, \exists X\{\text{INTERNAL}(\_, X) \,\&\, \exists X'[\text{BRUTUSIZER}(X') \,\&\, \text{EXTERNAL}(X, X')] \,\&\, \text{STAB}(X) \,\&\, \exists X'[\text{CAESARIZER}(X') \,\&\, \text{INTERNAL}(X, X')]\}$$

which can be treated semantically on a par with 'see a tree'. [24]

$$\text{SEE}(\_) \,\&\, \exists X[\text{INTERNAL}(\_, X) \,\&\, \text{TREE}(X)]$$

Initially, it might seem odd to treat both grammatical predicates *and* grammatical arguments as instructions to make conjoinable monadic concepts, which can be converted into complete thoughts by means of existential closure. But this might have been a workable strategy for the minds that initially faced the task of interpreting I-language expressions. Social primates can surely represent individuals as event participants. There is an ancient tradition of taking monadic concepts (and subject-predicate thoughts) to be logically special. Existential closure is an especially simple way to convert a monadic concept into a truth-evaluable thought. And while Frege viewed sentences as names for truth values, Tarski showed how to view them as predicates (satisfied by everything or nothing). So it is at least possible that I-languages are procedures for generating instructions to systematically create conjunctive monadic concepts that are easily converted into truth-evaluable thoughts. This does treat "macro instructions" as restricted type-shifters, which create concepts of things with participants from concepts of things that can be participants; where the shifting is induced by the need to treat concatenation *uniformly* as a sign for predicate-conjunction. But as discussed in 3.2 below, this may be the simplest kind of shifting that accommodates both adjunction and complementation.

24. See Higginbotham (1983), Parsons (1990). Prima facie, the basic significance of combining declarative sentences in a discourse is also conjunctive. But the "final" closure of a variable, yielding a truth-evaluable object, may reflect a cognitive interface with the language system—as opposed to a grammatical formative.

Prepositional phrases can also be treated as instructions to construct complex monadic concepts conjoinable with others. The nounish concept fetched with Rome_N may not be coherently conjoinable with the verbish concept constructed by executing $[_V \text{Brutus}_N \text{^}[_V \text{stab}_V \text{^Caesar}_N]]$. But $[_P \mathbf{in}_P \text{^Rome}_N]$—the result of labeling $\text{in}_P \text{^Rome}_N$ as a phrase headed by the functional item $\mathbf{in}_P$—can be an instruction of the same type as $\text{stab}_V \text{^}[_V \text{stab}_V \text{ Caesar}_N] \text{^}[_V \text{Brutus}_N [_V \text{stab}_V \text{ Caesar}_N]]$; and likewise for $[_P \mathbf{in}_P \text{ March}_N]$. Think of $\mathbf{in}_P$ as playing two roles, much like a verb: it imposes a constraint of its own, to the effect that values of its variable be "containables;" but more importantly, it takes a complement like Rome_N, and thereby creates a relativized instruction to form a monadic concept like $\exists$X[CONTAINER(_, X) & ROME(X)]. In which case, executing $[_P \mathbf{in}_P \text{ Rome}_N]$ can yield a concept like CONTAINEE(_) & $\exists$X[CONTAINER(_, X) & ROME(X)], which can be conjoined with a concept likw STAB(_) & $\exists$X[PATIENT(_, X) & CAESARIZER(X)].

On this view, $\mathbf{in}_P$ serves to *convert* an instruction to form a concept of a potential container *into* an instruction to form a concept of something as contained. Similar remarks apply to other prepositions. Indeed, one might even think of head-complement and head-specifier *relations* as "dedicated prepositions." This would explain the near synonymy of 'Brutus stabbed Caesar' and 'There was a stabbing of Caesar by Brutus'.

This assumes that the labels 'P' and 'V' do not conflict: relativizing a prepositional instruction to a verbal label, as in $[_V [_V \text{stab}_V \text{^Caesar}_N]] \text{^}[_P \mathbf{in}_P \text{^Rome}_N]]$, does not affect interpretation. The idea is that $[_P \mathbf{in}_P \text{^Rome}_N]$-as-V just *is* the instruction $[_P \mathbf{in}_P \text{^Rome}_N]$; the adjunct is understood as an instruction to create a concept of things contained in Rome, *not* a concept whose external participants are contained in Rome. But this assumption is not *ad hoc*. For on our view, the preposition is itself a remedy for the mismatch between the verbish instruction $[_V [_V \text{stab}_V \text{^Caesar}_N]]$ and the nounish instruction Rome_N. There is no mismatch, calling for further remedy, between $\mathbf{in}_P$ and $[_V [_V \text{stab}_V \text{^Caesar}_N]]$. In this sense, we endorse the old idea that verbs and nouns have conflicting features—[+V, -N] vs. [-V, +N], tensable vs. indexable—and that prepositions [-V, -N] are neither nounish or verbish, and hence *compatible* with both kinds of instructions.[25] Prepositional phrases can modify nouns, as in $[_N \text{man}_N \text{^}[_P \mathbf{in}_P \text{^Rome}_N]]$ or $[_N \text{stab}_N \text{^}[_P \mathbf{in}_P \text{^Rome}_N]]$. So we assume that relativizing such phrases to nominal labels is also semantically inert.

To recapitulate, **COM** differs from **CONCAT**. Labels are not irrelevant. On the contrary, they provide vehicles for introducing dyadic concepts like INTERNAL(E, X), which can be replaced with more specific thematic contents given lexical information. But **COM** differs from **CONCAT** only where it needs to—i.e., in cases where concatenates have *competing* labels. Correlatively, the device of labeling makes it possible to have competing labels, which can be interpreted as vehicles for introducing nonconjunctive aspects of meaning. The derivation below

25. Verbs may require selection of V+ (tensable) concepts, while nouns require selection of N+ (indexable) concepts, with prepositions imposing no further constraints along this dimension. Adjectives/adverbs [+N, +V] may require a choice of one or the other, with a further phrasal constraint ruling out any "mixed" choice of the form +N^+V (noun/adjective^verb/adverb).

illustrates the proposed tight connection between syntactic and semantic operations.

A. Concatenate(stab_V, Caesar_N) → stab_V^Caesar_N
B. Label(A) → $[_V\ \text{stab}_V\ \text{Caesar}_N]$ *Stab(e) & Patient(e, Caesar)*
C. Concatenate(Brutus_N, B) → Brutus_N^$[_V\ \text{stab}_V\ \text{Caesar}_N]$
D. Label(C) → $[_V\ \text{Brutus}_N\ [_V\ \text{stab}_V\ \text{Caesar}_N]]$
Agent(e, Brutus) & Stab(e) & Patient(e, Caesar)
E. Concatenate($\textbf{in}_P$, March_N) → $\textbf{in}_P$^March_N
F. Label(E) → $[_P\ \textbf{in}_P\ \text{March}_N]$ *Containee(e) & Container(e, March)*
G. Concatenate(D, F) → $[_V\ \text{Brutus}_N\ [_V\ \text{stab}_V\ \text{Caesar}_N]]$^$[_P\ \textbf{in}_P\ \text{March}_N]$
H. LABEL(G) → $[_V\ [_V\ \text{Brutus}_N\ [_V\ \text{stab}_V\ \text{Caesar}_N]]\ [_P\ \textbf{in}_P\ \text{March}_N]]$
Agent(e, Brutus) & Stab(e) & Patient(e, Caesar) & Containee(e) & Container(e, March)

3.2. Comparison with a More Familiar View

Given any theory that assigns a uniform interpretation to concatenation, either adjunction or complementation (or both) must be treated as something other than mere concatenation. We have opted for treating complementation as the special case, calling for extra machinery in terms of labeling. This fits with the idea that adjunction is the truly recursive part of natural language grammar. But even if concatenation is associated with a powerful recursive operation like function-application, with lexical items signifying appropriate functions and domain elements, appeals to "type-shifting" seem unavoidable; see, e.g., Parsons (1970), Kamp (1975), Partee and Rooth (1983), Chierchia (1998), Jacobson (1999). In short, complementation ends up looking complicated, no matter what.[26] And we prefer to keep at least one part of the grammar minimal. Still, it may be useful to compare our view—about the relations among **COM**, **CONCAT**, and labels—with a structurally similar but more traditional idea.

The idea of treating labeled expressions as relativized instructions is independent of any particular hypothesis about **CONCAT**. For this is just one way of saying that given a phrase of the form $[_A\ A\text{^}B]$, the nondominant concatenate B may need to be adjusted for purposes of semantic composition with A. The specific *kind* of adjustment called for (if any) will depend on **CONCAT** and A. But one can hypothesize that **CONCAT** is the operation of saturation (or function-application), and that simple predicate-argument combination involves no further operation, while

26. This leads many theorists to abandon the idea that concatenation has a uniform interpretation across constructions, in favor of hypotheses according to which the significance of combining expressions depends—one way or another—on the expressions being combined; see, e.g., Higginbotham (1985), Larson and Segal (1995), Heim and Kratzer (1998), Chung and Ladusaw (2003), Culicover and Jackendoff (2005). Though such pluralism makes it easier to achieve descriptive adequacy, it comes at the cost of positing a more sophisticated language faculty, one that associates grammatical combination with multiple kinds of semantic composition.

predicate-adjunct combination invokes type-lifting. From this perspective, an expression that by itself calls for a concept of type <e, **t**> can be used as part of a macro instruction to create a corresponding concept of type <<e, **t**>, <e, **t**>>. For example, if 'red' by itself calls for the concept RED(_), then 'red'-as-adjunct would call for the higher-order concept λC.C(x) & RED(x). On this view, relativizing an argument to a predicate makes no difference to the instruction associated with the argument, but relativizing an adjunct to a predicate is an instruction to type-lift.

Let's continue to underline grammatical arguments. In 3.1, we construed underlining as an instruction to relativize *non*vacuously, thus signaling a substantive "label relativization." But one might instead think of underlining as an instruction to relativize vacuously, thus *precluding* substantive relativization of the constituent semantic instruction. The phrase [$_V$ stab$_V$^Caesar$_N$] would then be a macro instruction to perform saturation on concepts obtained by executing the following two vacuously relativized instructions: stab$_V$ relative to itself (i.e, stab$_V$), and Caesar$_N$ relative to stab$_V$ (i.e., Caesar$_N$). By contrast, [$_N$ red$_A$ ball$_N$] would be an instruction to perform saturation on concepts obtained by executing the following two instructions, one of which is nonvacuously relativized: red$_A$ relative to ball$_N$, and ball$_N$ relative to itself; where red$_A$ relative to ball$_N$ is a macro instruction concepts to execute red$_A$ and type-lift. The net result of executing [$_N$ red$_A$ ball$_N$] would be to saturate λC.C(x) & RED(x) with BALL(x), yielding BALL(x) & RED(x).

One can get the same net result by simply conjoining BALL(x) with RED(x). As sketched above, neo-Davidsonians can treat verb-complement and verb-specifier combinations as interaction effects involving both concatenation and labeling. So especially given that verbs are associated with thematic roles, one way or another, it is no simpler to treat complementation and adjunction in terms of saturation and type-lifting. But one might think that determiners, which can also take internal/external arguments, tell in favor of the idea that concatenation at least sometimes signifies saturation. In which case, one might argue that Minimalist reasoning suggests that concatenation always signifies saturation. So we conclude by briefly noting that even determiners can be viewed as instructions to fetch monadic concepts, albeit *plural* monadic concepts. The idea is that determiner-complement and determiner-specifier combinations are also interaction effects involving both concatenation and labeling.

Given the singular arguments 'Brutus' and 'Caesar', it does no harm to view the event variable in (13) as singular.

(13) ∃{∃x[BRUTUSIZER(x) & AGENT(_, x)] & STAB(_) & PAST(_) &
∃x[CAESARIZER(x) & PATIENT(_, x)]}.

But given plural arguments, as in (14), plural event variables are required; see Schein (1993, 2006) and Pietroski (2005, 2006a), among others.

(14) The doctors stabbed the oranges with knives on Monday

The idea, which we cannot develop here in detail, is that (14) is true iff there were some event*s* that satisfied each of the following conditions: their Agents were the doctors; they were stabbings; their Themes were the oranges; they were done with knives; and they occurred on Monday. Following Boolos (1998), one can treat a plural variable as a variable that has *more than one* value relative to each assignment of values to variables. This provides an independently attractive treatment of "essentially plural" predicates, like the verb phrases in (15).

(15) The rocks rained down on the huts that surrounded the lakes

Given plural variables, labeling, and copying, one can also accommodate determiners. The details illustrate why and how it can be useful to hypothesize that certain expressions exhibit semantic effects that reflect interactions of our posited basic operations. Consider (16). If the grammatical structure is simply as in (16a),

(16) Brutus stabbed every senator

(16a) $[_T$ **past** $[_V$ Brutus$_D$ $[_V$ stab$_V$ $[_D$ every$_D$ senator$_N$]]]]

the determiner phrase is a constituent of a phrase labeled 'V', raising well-known difficulties for semantic composition. But if lexical properties of 'every' demand that 'every senator' combine with something to form a complex expression of the same type as 'every', and the operation of copying is available, then the relevant grammatical structure is arguably as shown in (16b);

(16b) $[_D$ $[_D$ *every$_D$ senator$_N$*] $\exists$:$[_T$ **past** $[_V$ Brutus$_D$ $[_V$ stab$_V$ $[_D$ every$_D$ senator$_N$]]]]]

where '$\exists$:' indicates existential closure of the event predicate, introduced by stab$_V$, and italics indicates a copy of an expression previously concatenated. As noted above, one can think of this closure (not as a long-distance dependency encoded in the syntax, but rather) as a reflex of how the language faculty—which cyclically generates complex *monadic predicates*—interfaces with cognitive systems that traffic in *complete thoughts* that are truth evaluable.

Suppose that only the higher copy of the determiner phrase in (16b) is interpreted, perhaps because the lower copy of every$_D$ fails to have an external argument. Then (16b) will be interpreted as an expression in which 'every' has an "existentially closed tense phrase" as its external argument, where this external argument has a gap: $\exists$^$[_T$ **past** $[_V$ Brutus$_D$ $[_V$ stab$_V$ ...]]]. A "gappy" expression of this sort can, like the sentence 'Brutus stabbed him', be interpreted as having a truth value relative to an assignment of a value to variable. Put another way, an open sentence can be viewed as an assignment-relative predicate of truth values: **true** if there was a stab by Brutus of the thing assigned to the variable, and **false** otherwise. In fact, while appealing to truth values is familiar and expositorily convenient, it is not required; see Pietroski (forthcoming) for presentation in a sparer framework that

eschews truth values, expressions of type **<t>**, propositional conjunction, and requires only a highly restricted version of existential quantification. But for present purposes, let's go with expository convenience.

If open sentences are predicates of truth values, then 'every' takes a predicate of individuals as its internal (nominal) argument and an assignment-relative predicate of truth values as its external (sentential) argument. But our suggestion is not that the direction of explanation is from lexical demands imposed by the determiner to the resulting syntax/semantics.

On the contrary, we think of $every_D$ as a device that lexicalizes a quantificational concept in accord with demands imposed by the syntax/semantics. So if $every_D$ demands an external argument, yet ends up in a position where its external argument is an open sentence, then (given our proposed combinatorics) the determiner must be understood as a predicate of things whose external participants are the values of open sentences. If these values are classical, then determiners are predicates of things whose external participants are **true** or **false**.

Initially, this might seem odd. But on our neo-Davidsonian account of verbs, $stab_V$ is understood as a predicate of events, even if the concept lexicalized is relational; see Pietroski (2007, forthcoming). From this perspective, lexicalizers may well *abstract* the monadic concept STAB(E) from the relational concept STAB(X, Y) or STAB(X, Y, E), by drawing on thematic concepts: STAB(X, Y) ≡ ∃E[STAB(X, Y, E)]; STAB(X, Y, E) ≡ AGENT(E, X) & STAB(E) & PATIENT(E, Y). Think of these biconditionals as providing a contextual definition of STAB(E), not a decomposition of STAB(X, Y). In this way, $stab_V$ can be an instruction to fetch a monadic concept—perhaps abstracted in the course of lexicalization—of things whose external participants are Agents.

Likewise, $every_D$ can be an instruction to fetch a monadic concept of things with participants (because of where $every_D$ appears in interpreted structures), even if the concept lexicalized is a genuine second-order relation (as it surely is with $most_D$). In this sense, we take the idea of quantifier raising seriously, but without positing QR as a basic operation: if a determiner phrase raises from a sentential clause (for whatever reason), it remerges with that clause, which remains *sentential*. This turns out to be an empirically attractive approach.

Other semantic theories often mask the "open sentence" character of a raised determiner's external argument, by hypothesizing some kind of type adjustment according to which the relevant sentential constituent—an expression of type **<t>**—behaves like a noun or relative clause of type **<e, t>**. Prima facie, this is implausible, since determiners cannot take overt relative clauses as external arguments. Note that while 'Every senator who arrived is a liar' is a fine sentence, 'Every senator who arrived who is a liar' is not, and 'Every senator who is a liar' cannot mean that every senator is a liar. Of course, *if* 'every' signifies a function of type **<<e, t>, <<e, t>, t>>**—from functions of the sort indicated by nouns, onto functions that map functions of the sort indicated by relative clauses onto truth values—then one must posit some kind of type adjustment. But this raises the difficult question, discussed by Barwise and Cooper (1981) among others, of why easily conceivable though "nonconservative" functions cannot be signified with deter-

miners. In short, the side-effects of adjustment may be worse than the initial problem. And one can adopt a different semantics for determiners.

We embrace the idea that 'every' takes a *sentential* external argument. If 'every' can take a predicate of individuals as its internal argument, and an assignment-relative predicate of truth values as its external argument, then perhaps 'every' is satisfied by things that have individuals as their internal participants and truth values as their external participants. There is no mystery about what such things might be: ordered pairs of the form {{e}, **t**} will do the job; where e is an entity, like an individual senator, and **t** is a truth value. Call these "Frege-Pairs," and note that one is already committed to such pairs if one appeals to functions of type <e, **t**>. So any posited "expansion" of the usual domain is minimal: in addition to entities and truth values, we appeal to ordered pairs of these, with truth values as the external elements—mirroring the grammatical fact that the external arguments of determiners are sentential. Suppose that 'every' is satisfied by one or more Frege-Pairs iff every one of them associates its entity with the value **true**. The idea is that 'every' can be satisfied plurally, by many Frege-Pairs "at once," so long as each of those Frege-Pairs associates its entity with truth. Likewise, suppose that some Frege-Pairs satisfy 'some' iff at least one of them associates its entity with **true**; they satisfy 'no' iff none of them associates its entity with **true**; and so on.

If every$_D$ is understood as a predicate of Frege-Pairs, and senator$_N$ is understood as a predicate of individuals, these expressions cannot be simply conjoined. But the lexical instruction senator$_N$, which bears no essential relation to any determiner, can be distinguished from the relativized instruction senator$_N$–as-D (*i.e.,* senator$_N$ *as complement of* a determiner). In this phrasal guise, the noun can be understood as semantically subordinate, with a meaning that manifests as part of a macro instruction to create a concept of things with the senators as internal participants. With this is mind, recall (16c).

(16c) $[_{D1}\ [_D\ \textit{every}_D\ \textit{senator}_N]\exists\hat{}[_T\ \textbf{past}\ [_V\ \text{Brutus}_D\ [_V\ \text{stab}_V\ \ldots_{\underline{1}}]]]]$

The index, which indicates the "gap" corresponding to the lower copy of the determiner phrase, can be interpreted like a pronoun. But the important idea is that in (16c), every$_D$ and its arguments can be understood as instructions to create concepts of Frege-Pairs, much as 'stab' and its arguments can be understood as instructions to create concepts of of events. The internal argument of every$_D$ can be an instruction to create a concept of those Frege-Pairs whose internal participants are the senators. (Half will be of the form <e, **true**>, where e is a senator, and these Frege-Pairs will also satisfy every$_D$.) The external argument—the existentially closed tense phrase with an indexed gap—can be an instruction to create a concept of those Frege-Pairs that meet the following condition: the external participant is **true** iff Brutus stabbed the internal participant; or more explicitly, the Frege-Pair's external participant is **true** iff its internal participant was the Patient of a stabbing whose Agent was Brutus. A pair of the form <e, **true**> meets this condition iff Brutus stabbed the entity e in question.

The details, including those concerning multiply quantified expressions, are provided in Pietroski (2005, 2006a). The leading idea is that given an assignment Ω that associates some variable with a value, a Frege-Pair <e, **t**> determines a variant assignment Ω' that is like Ω except for associating the variable with e; where **t** is a potential (truth) value of a sentence relative to the variant assignment. Unsurprisingly, a Tarskian semantics for (16c) does not require a type-adjustment that implausibly blurs the distinction between embedded sentences and relative clauses. But technicalities aside, the result is that (16c) can be interpreted as a complex predicate—built from lexical elements via Concatenate, Label, and Copy—that is satisfied by things that meet the following three conditions: every one of them associates an entity with **true**; they associate the senators with truth values; and they associate entities with truth values in accord with the rule "**true** iff the entity was stabbed by Brutus."

While (16c) is a predicate on this view, a final existential closure—corresponding to the claim that *there are one or more* Frege-Pairs that meet each of these three conditions—yields the right truth condition for (16): **true** iff every senator was stabbed by Brutus. If this is correct, no special explanation is needed for why determiners cannot indicate nonconservative functions of type <<e, **t**>, <<e, **t**>, **t**>>. If determiners are instructions to create concepts of Frege-Pairs, they *never* indicate functions of this higher type. Put another way, determiners do not indicate *relations* between predicates. So trivially, determiners do not indicate nonconservative relations. And while actual determiners can be described as predicates of Frege-Pairs, words that indicate nonconservative relations (like equinumerosity) cannot. The reason—which also helps explain why determiner phrases are understood as *restricted* quantifiers, and why they cannot take relative clauses as external arguments—is obvious: only the internal/nominal argument of a determiner corresponds to a set of individuals; the external/sentential argument does not, *pace* standard theories, specify a second set. Taking our spare syntax seriously has semantic payoffs.

4. Concluding Remarks

Since our introduction can also serve as a summary, we will end briefly. By decomposing the operation COMBINE (or MERGE) into the simpler operations CONCATENATE and LABEL, one can simplify extant conceptions of syntax, while also extending their explanatory reach. One can also view grammatical combination in human languages as the result of adding a new twist—labels—to the more cognitively ancient operation of concatenation. By decomposing the significance of COMBINE into simpler operations, corresponding to the semantic roles of CONCATENATE and LABEL (conjunction of monadic concepts, and a restricted kind of thematic shifting), one can also simplify extant conceptions semantics while extending their explanatory reach. In short, applying minimalist reasoning to syntax and semantic simultaneously has payoffs for each, as well as the overall theory of grammar.

References

Baker, Mark (1997). "Thematic Roles and Grammatical Categories". In L. Haegeman, (ed.) *Elements of Grammar*. Dordrecht: L Kluwer, p. 73-137.

Baker, Mark (2005). *Lexical Categories*. Cambridge: Cambridge University Press.

Barwise, Jon and Cooper, Robin (1981). "Generalized Quantifiers and Natural Language". *Linguistics and Philosophy* 4:159-219.

Bickerton, Derek and William H. Calvin (2000). *Lingua ex Machina*. Cambridge, MA: MIT.

Berwick, R. (1997). "Syntax facit saltum". *Journal of Neurolinguistics*. 10(2/3): 231-49.

Boolos, George (1998). *Logic, Logic, and Logic*. Cambridge, MA: Harvard University Press.

Borer, Hagit (2005). *Structuring Sense*. Oxford: Oxford University Press.

Bowers, John (2008). "On reducing control to movement". *Syntax*, 11:125-143.

Burge, Tyler (1973). "Reference and Proper Names". *Journal of Philosophy* 70: 425-39.

Carruthers, Peter (2002). "The Cognitive Functions of Language". *Behavioral and Brain Sciences* 25: 6.

Chametsky, Robert (1996). *A Theory of Phrase Markers and the Extended Base*. Albany: SUNY Press.

Chierchia, Gennaro (1998). Reference to kinds across languages. *Natural Language Semantics* 6: 339-405.

Chomsky, Noam (1955) [printed 1975]. *The Logical Structure of Linguistic Theory*. New York, NY: Plenum

Chomsky, Noam (1955). *Aspects of the Theory of Syntax*. MIT Press. Cambridge, MA.

Chomsky, Noam (1986). *Knowledge of Language*. New York: Praeger.

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

Chomsky, Noam and Howard Lasnik (1993). "The theory of principles and parameters". In Chomsky 1995, p. 13-127.

Chomsky, Noam (2000). "Minimalist Inquiries". In R. Martin et.al. (eds.) *Step by Step: Essays in Honor of Howard Lasnik.* Cambridge, MA: MIT Press, p. 89-155.

Chung, Sandy and Ladusaw, William (2003). *Restriction and Saturation.* Cambridge, MA: MIT.

Culicover, Peter and Jackendoff, R. (2005). *Simpler Syntax*. Oxford: Oxford University Press.

Dowty, David (1991). "Thematic proto-roles and argument selection". *Language* 67: 547-619.

Epstein, Samuel (1999). "Un-principled syntax: the derivation of syntactic relations". In S. Epstein and N. Hornstein (eds.) *Working Minimalism*. Cambridge, MA: MIT Press, p. 317-45.

Fodor, Jerry (1975). *The Language of Thought*. New York: Crowell.

Fodor, Jerry (2000). *The Mind Doesn't Work That Way*. Cambridge, MA: MIT Press.

Gallistel, Charles R. (1993). *The Organization of Learning*. Cambridge, MA: MIT.

Halle, Morris and Alec Marantz (1993). "Distributed Morphology and the Pieces of Inflection". In K. Hale and S. Jay Keyser (eds.), *The View from Building 20, Essays in Honor of Sylvain Bromberger*. Cambridge, MA: MIT Press, p. 111-186.

Hauser, Marc, Chomsky, Noam, and Fitch, Tecumseh (2002). "The faculty of language: what is it, who has it, and how did it evolve?". *Science* 298: 1569-1579.

Heim, Irene and Kratzer, Angelika (1998). *Semantics in Generative Grammar*. Oxford: Blackwell.

Higginbotham, James (1983). "The Logical Form of Perceptual Reports". *Journal of Philosophy* 80:100-27.

Higginbotham, James (1985). "On Semantics". *Linguistic Inquiry* 16: 547-93.

Higginbotham, James (1986). "Linguistic Theory and Davidson's Program". In E. Lepore (ed.), *Inquiries into Truth and Interpretation*. Oxford: Oxford University Press, p. 29-48.

Hornstein, Norbert (1999). "Movement and control". *Linguistic Inquiry* 30, 69-96.

Hornstein, Norbert (2001). *Move! A Minimalist Theory of Construal*. Oxford: Blackwell.

Hornstein, Norbert (2003). "On control". In R. Hendrick (ed.). *Minimalist Syntax*. Oxford: Blackwell, p. 6-81.

Hornstein, Norbert (2009). *A Theory of Syntax: Minimal Operations and UG*. Cambridge: CUP.

Jackendoff, Ray (2002). *Foundations of Language*. Oxford: Oxford University Press.

Jacobson, Pauline (1999). "Towards a Variable-Free Semantics". *Linguistics and Philosophy* 22: 117-184.

Johnson, Kyle (2001). "What VP ellipsis can do, what it can't, but not why". In *The Handbook of Contemporary Syntax*. M. Baltin and C. Collins (eds.). Oxford: Blackwell, p. 439-479.

Kamp, Haus (1975). "Two Theories of Adjectives". In *Formal Semantics in Natural Language*. E. Keenan (ed.), 123-55. Cambridge: Cambridge University Press, 1975.

Katz, Jerrold (1994). "Names without Bearers". *Philosophical Review* 103: 1-39.

Kayne, Richard (1983). "Connectedness". *Linguistic Inquiry*, 14, 93-133.

Kayne, Richard (2002). "Pronouns and their antecedents". In S.D. Epstein and T.D. Seely (eds.). *Derivation and Explanation in the Minimalist Program*. Oxford: Blackwell.

Kennedy, Christopher (1999). "Projecting the adjective: The syntax and semantics of gradability and comparison". Garland Press, New York.

Koster, Jan (1984). "On binding and control". *Linguistic Inquiry* 15; 417-59.

Kratzer, Angelika (1996). "Severing the External Argument from its Verb". In J. Rooryck and L. Zaring (eds.), Phrase Structure and the Lexicon. Dordrecht: Kluwer Academic Publishers, p. 109-137.

Kripke, Saul (1980). *Naming and Necessity*. Cambridge, MA: Harvard University Press.

Larson, Richard and Segal, G. (1995). *Knowledge of Meaning*. Cambridge, MA: MIT.

Lidz, Jeff and William J. Idsardi (1997). "Chains and phonological form". In A. Dimitriaidis, H. Lee, C. Moisset, and A. Williams (eds.) *Proceedings of the 22nd annual Penn Linguistics Colloquium.* 109-125.

Longobardi, Giuseppe (1994). "Reference and Proper Names". *Linguistic Inquiry* 25: 609-65.

May, Robert (1985). *Logical Form*. Cambridge, MA; MIT.

Muysken, Peter (1982). "Parametrizing the notion of 'head'". *Journal of Lingusitic Research* 2, 57-75.

Nunes, Jairo (2004). *Linearization of Chains and Sidewards Movement*. Cambridge, MA: MIT.

Parsons, Terrance (1970). "Some problems concerning the logic of grammatical modifiers". *Synthese* 21: 320-334.

Parsons, Terrance (1990). *Events in the Semantics of English*. MIT. Cambridge, MA.

Partee, Barbara and Rooth, Mats (1983). "Generalized Conjunction and Type Ambiguity". In R. Bäuerle *et. al*. (eds.), *Meaning, Use, and Interpretation of Language.* Berlin: de Gruyter, p. 334-356.

Pesetsky, David (1982). *Paths and categories*. PhD thesis. MIT. Cambridge, MA.

Pietroski, Paul (2005). *Events and Semantic Architecture*. Oxford: Oxford University Press.

Pietroski, Paul (2006a). "Interpreting Concatenation and Concatenates". *Philosophical Issues* 16: 221-45.

Pietroski, Paul (2006b). "Induction and Comparison". *University of Maryland Working Papers in Linguistics* 15: 157-90.

Pietroski, Paul (2007). *Croatian Journal of Philosophy* 7: 343-374.

Pietroski, Paul (forthcoming). "Minimal Semantic Instructions". In C. Boeckx, ed., *Oxford Handbook of Linguistic Minimalism*. Oxford: OUP.

Pinker, Steven (1997). *How The Mind Works*. New York: Norton.

Poeppel, David and Embick, David (2005). "The relation between linguistics and neuroscience". In A. Cutler (ed.), *Twenty-First Century Psycholinguistics: Four Cornerstones*. Lawrence Erlbaum, p. 103-118.

Polinsky, Maria and Eric Potsdam (2003). "Backward Control". *Linguistic Inquiry* 33, 245-282.

Richards, Norvin (2001). *Movement in Language*. Oxford: OUP.

Rizzi, Luigi (1990). *Relativized Minimality*. Cambridge, MA: MIT.

Schein, Barry (1993). *Plurals*. Cambridge, MA: MIT Press.

Schein, Barry (2006). "Plurals". In E. Lepore and B. Smith (eds.), *The Oxford Handbook in the Philosophy of Language*. Oxford: Oxford University Press.

Spelke, Elizabeth (2002). "Developing knowledge of space: Core Systems and New combinations". In S. Kosslyn and A. Galaburda (eds.), *Languages of the Brain*. Cambridge, MA: Harvard Univ. Press.

Szczegielniak, Adam (2004). *Relativization and Ellipsis*. PhD thesis. Harvard University. Cambridge, MA.

Williams, Alexander (2007). "Patients in Igbo and Mandarin". In J. Dölling and T. Heye-Zybatow (eds.) *Event Structures in Linguistic Form and Interpretation.* Berlin: de Gruyter.

Zwart, Jan-Wouter (2002). "Issues relating to a derivational theory of binding". In S.D. Epstein And T.D. Seely (eds.). *Derivation and Explanation in the Minimalist Program*. Oxford: Blackwell.