A minimalist theory of feature structure*

David Adger

Queen Mary, University of London

May 2008

Final Version

1 Introduction

This chapter has two intertwined aims. One is to outline, within a broadly minimalist framework, a fairly explicit proposal for a theory of feature structures, that is, the structure of features and the syntax of how features combine to form lexical items; the other is to explore the consequences of the idea that structure embedding in human language is only ever syntactic (that is, that there is a single engine for the generation of structure and that engine is the syntax—see Marantz 1997, Borer 2005; also compare the ideas in Hauser, Chomsky, and Fitch 2002, Fitch, Hauser, and Chomsky 2005). The two aims are connected, since if structure embedding is only syntactic, then the feature structures that are the basic atoms of syntax (i.e. lexical items) cannot involve embedding of one feature inside another. This approach contrasts rather starkly with work in other approaches which take lexical items to have rich featural structure (e.g the chapters using HPSG, LFG and FUG in this volume).

Before launching in on features, a few words about what a grammar in Minimalism looks like. Much like categorial grammars, Minimalist grammars can be seen as lexically

driven combinatory systems, involving the repeated application of very basic operations to lexical items and to the operators' own outputs (e.g. Stabler 1997, Stabler 1998, Frampton and Gutmann 1999, Retoré and Stabler 2004). Unlike Categorial Grammars, the operations build and alter structure, so that the grammar can be used to define legitimate syntactic objects (rather like TAGs, Kroch and Joshi 1987, Frank 2002). The core operation is Merge, which essentially defines what legitimate (non-atomic) syntactic objects are. I give the version of Merge defined in Chomsky (1995, 243) here, but we will change this directly:

(1) a. lexical items are syntactic objects.
     b. If A is a syntactic object and B is a syntactic object, then Merge of A and B, K = {C, {A, B}}, is a syntactic object.

C in K is known as the **label** of K. Chomsky argues that the label is always identical to one of the two arguments of Merge, thus (1) can be reformulated as:

(2) a. lexical items are syntactic objects.
     b. If A is a syntactic object and B is a syntactic object, then Merge of A and B, K = {A, {A, B}}, is a syntactic object.

We return to the isue of which of the two subconstituents of the output of Merge is chosen as the label in section 3.1.1. K is more usually represented as a labeled bracketing or tree (although such a representation imposes an order which the sets do not have):
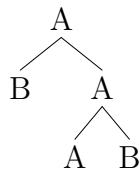
(3) [$_A$ A B]

(4)

$$
\begin{array}{c}
A \\
\diagup\diagdown \\
A \quad B
\end{array}
$$

A generalized version of this operation (usually called Move, Remerge, or Internal Merge) allows Merge of an element that has already been Merged:

(5)

```
            A
          /   \
         B     A
              /  \
             A    B
```

This can be thought of in two ways: either the higher B is type-identical to the lower B, or it is token identical. Both approaches have been pursued in the literature (under the 'copy' theory (Chomsky 1993), or the 'remerge' theory of Move (e.g. Gärtner 2002)).

These structures are built by the syntax and are then interpreted by the interfaces that map the structures to sounds and meanings. Minimalism assumes a number of interface constraints that dictate how the structures are so mapped. For example, on the sound side, some statement needs to be made about the linearization of the structures, while on the meaning side, something needs to be said about how each part of the structure's interpretation relates to the interpretation of the other parts of the structure. These interface constraints are maximally general (for example, a constraint will require that only one of the two B's in (5) be pronounced—see, e.g. Nunes 1995, Bobaljik 1995, while a similar constraint will require only one to be semantically interpreted—e.g. the Theta Criterion essentially has the effect; see also Adger and Ramchand's (2005) Interpret Once Under Agree condition).

2 Types of Feature System

In Adger (2006), in response to Asudeh and Toivonen (2006), I noted that feature systems could be thought of as increasing in complexity given the kinds of rules required to generate lexical items in the system (see Gazdar, Pullum, Carpenter, Klein, Hukari, and Levine (1988) for a more formal discussion which defines categories via a recursive definition on sets). In the remainder of this section I extend this line of thinking, running through various types of feature system, from privative to category recursive, evaluating each in turn.

A simple privative system can be given by two statements which define features and lexical items as follows:

(6) a. An atomic symbol drawn from the set F = {A, B, C, D, E, ...} is a feature

   b. A Lexical Item (LI) is a set of features, for example LI_i = {A, C, E}

I take the word privative here to characterize systems where atomic features may be present or absent, but have no other properties.

We are interested in only syntactic features here, rather than in the features which distinguish lexical items phonologically and semantically. One could therefore think of LI in (6) as being close to the traditional notion of a syntactic category as a bundle of features. Note that since LI is defined as a set, the same feature cannot appear twice. Examples of LIs in such a system might be:

(7)   a.   {T, past, plural}

       b.   {D, definite, accusative, plural}

However, a system like this suffers from the problem of how to state syntactic dependency relations. Since the features have no properties, there is no obvious way of encoding that two lexical items must have the same specification for a certain feature in a syntactic structure, a core explanandum of natural language, reducing to the fact that syntactic dependencies exist. To state such dependencies with purely atomic, unstructured, features, one would need a separate system of rules that explicitly state the relevant constraint:

(8)   If a subject has the feature [plural] then the verb which it is a subject of must also have this feature.

It's unclear, within the Minimalist framework, where one could state such a constraint, since the framework allows only lexical items, syntactic operations and interface constraints relating syntax to interpretation.

One could perhaps hypothesize an interface constraint requiring the interpretation of the number feature on the subject and the interpretation of the number feature on the verb to be identical (e.g. the semantic theories of agreement defended by, for example, Dowty and Jacobson 1988), but that doesn't obviously capture the syntactic nature of the dependency. Moreover, it's difficult to see how one might extend such a system to, for example, structural case matching between adjectives and nouns (see Svenonius (2003) for discussion of this point).

In any event, a system where features have no properties requires some syntax external theory of syntactic dependency formation, which seems rather problematic.

We can sidestep this problem by introducing some minimal complexity into our theory

of privative features, allowing them to have exactly one property which will ensure that there is a matching feature elsewhere in the syntactic structure. This is the intuition behind the notion of uninterpretability introduced by Chomsky (1995). We add a statement to our theory specifying the form each $F_i \in$ F takes, giving (9) as our new feature theory:

(9)   a.   An atomic symbol drawn from the set F = {A, B, C, D, E, ...} is a feature

       b.   An atomic symbol drawn from the set F and prefixed by u is a feature

       c.   A Lexical Item (LI) is a set of features, for example LI_i = {A, uC, E}

With this definition in place, we can now make our syntactic structure building rules sensitive to the presence of the u prefix, ensuring that when a feature bears such a prefix, there must be another feature in the structure which is exactly the same, but lacks the prefix. This implements Chomsky's notion of **checking**. We can now ensure that a subject will agree with its verb by endowing the verb with a set of features which bear the u prefix. Unless the subject bears exactly the same set of features unprefixed, a general constraint stipulating that all u-prefixed features must be checked will rule the structure out:

(10)   {D, definite, plural} ... {T, past, $u$plural}

(11)   *{D, definite, singular} ... {T, past, $u$plural}

Note that the prefix is doing purely formal work in this system. Chomsky proposed that the uninterpretability property of features not only implemented syntactic dependency in the way just outlined, but furthermore could be connected to the interpretation of the elements on which the features were specified. The idea is that a feature like [plural] only has an interpretation when specified on a category which can be potentially interpreted as plural (e.g. on a noun), otherwise an instance of this feature will be uninterpretable: interpretability is detectable from a feature's syntactic/semantic context. The formal property of features (the u prefix) which enables them to enter into dependency relations is thus linked to the interpretation of features by the (semantic) interface (see the discussion of Full Interpretation in section 3.1.1 below).

This kind of system, where features have just one property (the 'match me' property of uninterpretability) appears to be the minimal system that is needed to capture the notion of syntactic dependency. The syntactic operations are sensitive to this property

and will generate structures that satisfy its needs.[1]

However, one issue with this kind of privative theory is that it makes capturing morphological and semantic commonalities between features difficult. For example, there is no formal connection between the features [singular] and [plural] in such a privative system, yet there is a clear semantic connection between them and in many languages a clear morphological connection (for example, the corresponding morphemes are in complementary distribution).

One might think that this problem could be sidestepped by taking the semantic and morphological interpretation of absence of the feature [plural] to be whatever the semantics and morphology of singularity is. That is, the interface rules would take absence of a [plural] feature to be equivalent to an explicit statement of a [singular] feature. This would then connect the semantic and morphological notions of plurality and singularity with a single syntactic features presence or absence, this solving the problem. However, this will not do.

To see this, take the following examples:

(12)   a.   The owl hoots.
        b.   The owls hoot.

Under the analysis just sketched, we take the feature [plural] to have the following interpretations in English on a noun and on T (ignoring pronominal subjects and using capitals to signify whatever the relevant semantics is—see below for a more serious suggestion):

(13)   a.   PLURAL ← {N, plural} → -/z/
        b.   SINGULAR ← {N} → -/0/
        c.   PRESENT ← {T, present, $u$plural} → -/0/
        d.   PRESENT ← {T, present} → -/z/

However, under such a system, nothing stops the combination of (a) and (d):

(14)   ... {N, plural} ... {T, present}

Neither item bears a feature with the 'match-me' property of uninterpretability. The

---

[1]One can imagine other ways of implementing the 'match me' property, but they are no less complex than the system just described.

semantics works out correctly, but the predicted morphology is wrong. We predict -/z/ on both the N and on T, giving:

(15)   *The owls hoots

The problem here is that agreement is obligatory, but, given the assumptions that the features are atomic and privative, and that the semantics of singularity follows from the absence of [plural], there's just no way of encoding the obligatoriness of agreement. Under this system of assumptions, to avoid generating (15), we are forced to posit a [singular] feature in the syntax:

(16)   a.   PLURAL ← {N, plural} → -/z/
        b.   SINGULAR ← {N, singular} → -/0/
        c.   PRESENT ← {T, present, $u$plural} → -/0/
        d.   PRESENT ← {T, present, $u$singular} → -/z/

This will now capture what we want captured, but suffers from the problem of that the two features [singular] and [plural] are completely independent. I take this conclusion to be incorrect.

For these reasons, one may wish to enrich the system so as to allow such a link to be made. The simplest way of doing this is to allow features to have values. We can then arrange the theory so that an unvalued feature on T gets its value from the subject.

There are many kinds of feature systems where features have values. We can rank them in richness depending on what vocabulary the values are drawn from. One simple system is a binary system, where the values are drawn from the vocabulary {+, −}. Richer systems allow an expansion of the value vocabulary to other atomic symbols, to features, to lexical items and to complex syntactic objects. We'll discuss these in turn.[2]

---

[2]Although I begin here with binary systems, there is actually a simpler system, which allows only one value (say +), with an unvalued feature being interpreted as − by the interfaces. This will capture the obligatoriness of agreement, since T can bear an unvalued [plural] feature which will be valued by the [+plural] feature on the noun, so capturing the agreement pattern. An unvalued plural feature will then be interpreted by the morphology and semantics as singular. However, for features like structural case, this isn't obviously sufficient, since standard analyses take case features to require valuation, rather than to be spelled out with a default. There are clearly alternative theories of case one might pursue were one to take this general direction, and it may make some sense of the fact that languages often do have default case mechanisms. One advantage of such a 'one-value' system is that syntax does not seem to

For a binary system, we change our specification of the form of features as follows:

(17)   a.   a feature is an ordered pair $\langle$Att, Val$\rangle$ where

          b.   Att is drawn from the set of attributes, {A, B, C, D, E, ...}

          c.   and Val is drawn from the set of values, {+, −}

Following linguistic tradition, we write the ordered pair $\langle$A, +$\rangle$ as either [A:+] or [+A]. This definition of course gives us a finite set of LIs (categories) on the assumption that there is a finite set of features. Our previous examples might now look as follows:

(18)   a.   {T:+, past:+, plural:+}

          b.   {D:+, definite:+, plural:+}

This approach has an advantage over the privative approach in that it immediately allows one to capture the fact that the entities that plural marked elements denote are the complement set of those denoted by singular marked elements. It simplifies the interface to interpretation, since the form of the feature does the work of establishing the semantic and morphological connection rather than the syntax external systems.

Since features have values, we can encode the 'match me' property via a special value rather than via the diacritic u. A simple way of doing this is to allow the empty set to be the value in such cases: $\langle$A:$\emptyset\rangle$. We can now rewrite our checking situation as follows:

(19)   { T:+, past:+, plural:$\emptyset$} ... {D:+, definite:+, plural:+}

Rather than checking, we define a valuing operation that ensures that the values of the singular feaures are the same (see section 3.1.2), where the empty set value is replaced by a 'true' binary value. Lack of value, then, does the same work as the presence of the u prefix did in the privative system described above:

(20)   { T:+, past:+, plural:+} ... {D:+, definite:+, plural:+}

This binary perspective also leads to a fracturing of the set of features. For example, I just noted that the system entails that the denotation of plural NPs is the complement set of the denotation of singular NPs. But of course there are languages with duals too, where

---

care overly about minus values: there don't seem to be processes that, for example, move all non-wh phrases to a specific position, and, in a binary system, this is a fairly easy thing to state. Thanks to Peter Svenonius and Daniel Harbour for much discussion on this point.

this statement would appear to be false. Since our features are binary, for a language with three number categories, we need to introduce another feature. This will predict, apparently wrongly, that we should have four numbers, rather than three.

Is this a disadvantage of the system? It depends on the actual interpretations we give to the features. For example, if we assume a semantics where we have a set of individuals { a, b, c, ...} and a set of plural individuals { {a, b}, {b, c}, {a, {b, c}}, ...} etc (where each plural individual is a structured set (or lattice: see Link 1983)) and we take D (the domain of all individuals) to be the union of these (minus the empty set), then we can define number features as follows (cf Noyer 1992, Schlenker 2003, Harbour 2007):

(21) a. [+atomic] partitions D into atomic and non-atomic members

  b. $\lambda f : \forall x \in f, atom(x).f$

(22) a. [+augmented] partitions D into a set whose elements have proper subsets, and one that doesn't.

  b. $\lambda f : \forall x \in f, \exists y, y \subset x.f$

In (22), the nature of the proper subsets is further determined by the other features in the LI.

This will now elegantly capture the three numbers we find phenomenologically (see again Noyer 1992 and Harbour 2007 for detailed discussion).

(23) a. [+atomic, +augmented]: this will require the relevant NP to be found in the domain of atomic individuals which have proper subsets. However, as atomic individuals cannot, by definition, have subsets, this feature specification will be contradictory

  b. [+atomic, –augmented]: this will require the relevant NP to be found in the domain of atomic individuals which do not have proper subsets, this straightforwardly giving us the set of singulars

  c. [–atomic, +augmented]: this will require the relevant NP to be found in the domain of non-atomic individuals which have non-atomic proper subsets; this entails only non-atomic individuals with a cardinality greater than or equal to 3.

  d. [–atomic, –augmented]: this will require the relevant NP to be found in

the domain of non-atomic individuals which do not have non-atomic proper subsets (i.e which have only atomic proper subsets). This is of course simply the set of duals.

A further nice consequence of such a system is that it captures certain natural classes in a way that is familiar from binary features in phonology. If we look at the feature specifications of a language with a dual number, we have the following:

(24)

	[±singular]	[±augmented]
1	+	–
2	–	–
≥3	–	+

Note that there is a way of referring to the class of singulars and duals together ([–augmented]) and of duals and plurals together ([–singular]), but no way of referring to the class of singulars and plurals to the exclusion of duals. This is the correct result: languages with duals do not treat singular and plural as a natural class to the exclusion of dual.

To capture these kinds of effects in a privative system, it's necessary to organize the privative features into a geometry (see, e.g. Harley and Ritter 2002). In a binary system, the geometry emerges as an epiphenomenon of the possible compositions of the meanings of the features (see, especially, Harbour 2007).

A further enrichment of the system would be to allow a larger set of values, giving a multi-valent feature theory (e.g. the system in Adger 2003, essentially the same as the view espoused in *Aspects*, Chomsky 1965, 171):

(25)   a.   a feature is an ordered pair ⟨Att, Val⟩ where

   b.   Att is drawn from the set of attributes, {A, B, C, D, E, ...}

   c.   and Val is drawn from the set of values, {+, −, a, b, c, ...}

Our example LIs now might look as follows, with a mix of binary and multivalent features of other sorts, some of which are lexically valued while others are not:

(26)   a.   {T:+, tense:past, case:nom, number:∅}

   b.   {D:+, definiteness:+, case:∅, number:pl}

This is similar to a privative system in that there is no notion of binary opposition built in, but similar to the binary system in that syntactic commonalities are expressed in the syntactic feature system, so that, for example, the notion of 'number' is syntactically represented.

One use that has been made of such a system is the analysis of English auxiliary inflection outlined in Adger (2003). I proposed there that each auxiliary bore a feature [Infl] which could take as its value elements from the set {past, present, perf, prog, pass}. Each auxiliary bears a valued Infl feature specifying its category, and an unvalued one. The unvalued feature receives its value from a higher Infl feature.

To see how such an analysis works, take a sentence like:
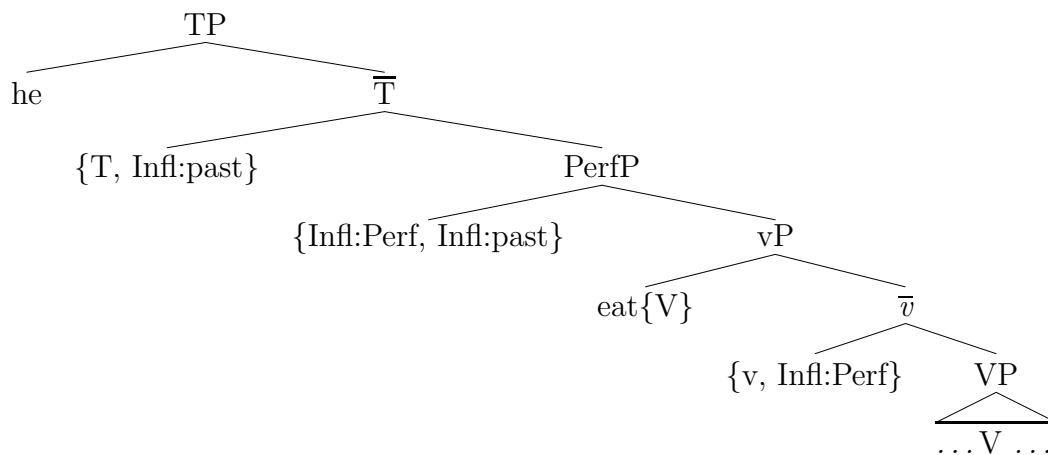
(27)   He had eaten.

The syntactic features bundled together to make the relevant lexical items are:

(28)   {D, pro:+, case:nom, num:sing, pers:3, gender:masc}, {T, Infl:past, ...}, {Infl:Perf, Infl:∅ ...}, {v, Infl:∅ ...}, {V}

Simplifying somewhat, the structure is built up by Merge of the verb *eat* and the category v, followed by Move (Internal Merge) of *eat* to the left of v. Then the auxiliary bearing the feature [Infl:Perf] is Merged, and the unvalued Infl feature on v receives Perf as its value. When T is then Merged, the unvalued Infl feature on the auxiliary receives past as its value, giving the final output (after the subject is Merged) in (29) (see Adger 2003 for further details—this derivation is somewhat simplified, lacking a vP internal subject trace, and having the V in the specifier of v rather than adjoined, although see Matushansky 2006 for the latter):

(29)

```
                    TP
         ┌───────────┴───────────┐
        he                       T̄
                     ┌───────────┴───────────┐
               {T, Infl:past}              PerfP
                              ┌──────────────┴──────────────┐
                       {Infl:Perf, Infl:past}              vP
                                              ┌─────────────┴─────────────┐
                                         eat{V}                          v̄
                                                            ┌─────────────┴─────────────┐
                                                      {v, Infl:Perf}                   VP
                                                                               ┌────────┴────────┐
                                                                               . . . V . . .
```

The LI {v, Infl:Perf} is pronounced as *-en*, while the auxiliary {Infl:Perf, Infl:past ...} is the past form of the perfect auxiliary, and is hence pronounced as *had*. The subject pronoun is pronounced as *he*, and the verb as *eat*. Putting all of these together we get *He had eat-en*. This analysis essentially treats affix-hopping as a feature-valuation phenomenon, rather than as a post-syntactic lowering rule.

Whether this is the correct analysis or not, it requires a complexity of feature values that goes beyond simple binary values.

However, such a multi-valent attribute value system will also, for natural languages, require some kind of typing, to ensure that only appropriate values are given for the relevant attributes. For example, we do not want *past* to be a possible value of *number*, nor do we want *singular* to be a value of *mood*. There are two possibilities here: we can appeal to the interface, and suppose that no interpretation is available for these features, or we can rule out these feature-value pairs within the specification of the feature system itself. For example, we can specify the syntax of Val in our definition of feature in something like the following way:

(30)   a.   Val → **e** / **a**:__

      b.   Val → **f** / **a**:__

      c.   Val → **f** / **c**:__

      d.   Val → **g** / **d**:__

This set of rules constrains the possible values of **a** to be just **e** and **f**. The set of LIs (i.e. syntactic categories/permissible syntactic feature structures) is once again finite. Of course, such constraints are not necessary if the values are only $\{+,-\}$ and if all features

have just these as values.[3]

A richer system yet would allow attributes to themselves be the value of another attribute, introducing recursion into the feature structures, as is common in Functional Unification Grammar, LFG, HPSG etc. We could do this by having the rule for Val expand as follows:

(31)   Val → SynObj

This has the effect of allowing an entire category to be a possible value of another attribute, giving feature structures like those familiar from GPSG (Gazdar, Klein, Pullum, and Sag 1985). For example, it is now possible to have a GPSG-style category-valued [SLASH] feature which can be used to encode long distance dependencies:

(32)   {cat:V, tense:past, SLASH:{N, bar:2, case:acc, number:plural}}

An alternative allows values to expand directly as features, giving the Attribute Value Matrices (AVMs) familiar from work in HPSG, LFG and FUG:

(33)   Val → F (or F → Att F)

(34)   [a:[b:[c:[d:z]]]]

(35)   [a:[^b_c]]

As I noted in Adger (2006), it is not possible to implement feature typing by simple context-sensitive rules now, since non-adjacent information is required: Val expands as F, and then F as [Att:Val], so there is no possible context-sensitive rule which allows us to say that an attribute that is a value of a higher attribute can be keyed to the category of that higher attribute, as the higher attribute is never its sister (mutatis mutandis for the alternative expansion given above). Typing must then be specifically stated separately from the mechanism that specifies what a possible feature structure can look like. Again see Gazdar, Pullum, Carpenter, Klein, Hukari, and Levine (1988) for discussion. In a system like this, there is the possibility of an infinite number of syntactic categories with

---

[3]Annabel Cormack points out to me that this is essentially an issue about the learnability of the lexicon, since the child will determine the range of values on the basis of features of the input. This is the tack I took in Adger 2006 in response to criticism from Asudeh and Toivonen 2006. (30) would then just be an abbreviation of knowledge rather than anything more insightful.

a single root feature unless there are additional constraints that prohibit recursion of a particular category within its own value. Not only would we need such constraints to rule out feature structures like A:A, but also, more problematically A:B:...:A (see Shieber 1986, 20). This is not true in non-embedding feature systems: the number of lexical items with different specifications is finite on the assumption that the number of features is finite.

We have seen a range of theories allowing different levels of complexity of feature structures, from simple atomic features to rich AVM structures. We have not been concerned with the theory of feature content, only of structure. In the next section I argue that a general constraint that follows from the architecture of Minimalism puts an upper bound on the complexity of the feature structures that constitute lexical items, disallowing feature embedding.

3 A Minimalist Feature System

In the remainder of this chapter, I'm going to explore the impact on feature structures of a basic Minimalist idea, that the only source of embedding in language is Merge (Hauser, Chomsky, and Fitch 2002, Fitch, Hauser, and Chomsky 2005). Since feature structures are substructures of lexical items, and lexical items are the input to Merge, it follows that feature structures cannot involve the kind of complexity that comes from embedding one structure inside another. Hence we have:

(36)   No Complex Values Hypothesis: Features cannot embed other features in a lexical item.

This hypothesis leads to a 'flat' structure for categories: they are simply sets of ordered pairs, as in (25). Adhering to the hypothesis means that the richest theory we can adopt is the multi-valent theory. Note that this places an upper bound on the complexity of LIs, not a lower bound. It may, in fact, be the case that the 'one-value' system outlined in footnote 2 is sufficient, and is, in fact, the lower bound. I won't address this question in the remainder of the chapter, but simply assume that multi-valent structures are available, as I did in Adger (2003).

The fact that there is so little richness of structure in lexical entries gives the theory its particular flavour: the structure required by the phenomena of human languages ends up being syntactic rather than lexical structure. In a sense, then, Minimalism is unlike other 'lexicalist' theories in that almost all the interesting structure is syntactic, although the information which leads to the building of that structure is entirely lexical.[4]

We have said nothing as yet about the distribution of featural information in a larger syntactic structure. That is, when two items Merge, is there some constraint on what they can/must be? Empirically, we clearly want to say that there is, to distinguish, for example *the many men* from **many the men*, so the question is how this kind of distributional constraint is modeled theoretically.

This brings us to the topic of category features, and specifically functional categories. In Adger (2003), following Grimshaw (1991) and much subsequent work, the latter are seen as loci for the instantiation of features and are assumed to be ordered in a number of (universal) scopal hierarchies (the Hierarchies of Projection (HoPs), also known as, Extended Projections (Grimshaw 1991) or Functional Sequences (Starke 2001)). Each hierarchy is rooted in a specific kind of category (what we traditionally think of as the lexical categories N, V, A, P etc) and can terminate in a specific functional category (D, C, Deg, etc). The hierarchies directly specify the distribution of featural information in the clause, noun phrase etc. The motivation for these hierarchies is primarily empirical (see, for example, Cinque (1999), Julien (2002), or the papers in Cinque and Kayne (2005)). This architecture now gives us two types of feature:

(37)   a.   Category features (C, T, V, N, D, ...)

        b.   Morphosyntactic (MS) features (case, number, person, finiteness, definiteness, ...)

From this perspective, the HoPs organize the category features into syntactic hierarchies, while the distribution of the morphosyntactic (MS)-features is not so constrained. The general programme outlined here has some latitude here, however, in that it would be consistent with the leading ideas of Minimalism to simply have one type of feature, with

---

[4]One of the few recent works which takes providing a restrictive theory of lexical items as a major aim is Emonds 2000 (especially chapters 2 and 8). Emonds proposes formal limits on the internal structure of lexical items. This theory, although it posits structures somewhat richer than the view I'll defend here, shares the same programmatic goals.

the HoPs specifying the distribution of all features. This leads to the 'nano-syntactic' research programme pursued by Michal Starke and some of his colleagues (Caha 2007). I will argue here for a more conservative position that distinguishes between categorial features, organized into hierarchies by the HoPs, and MS-features which essentially subclassify the categories.

This theoretical move requires us to expand our rule for lexical items. Rather than (25), we have:

(38)   a.   A lexical item is a set $\{K, F_1, ..., F_n\}$,

            (i)   K is an ordered pair $\langle$Cat, N$\rangle$

            (ii)   Cat is drawn from the set {C, D, T, Num, Asp, ...}-$\emptyset$, and N is drawn from the set of natural numbers above 0

    b. F_i is a pair $\langle$Att, Val$\rangle$, where Att is drawn from a finite set of MS-features and Val from a finite set of values

    c.   Hierarchies of Projections: these are sequences of Ks whose second member is ordered by the relation $<$.

Two such hierarchies are offered in Adger (2003), essentially defining 'extended projections' (Grimshaw 1991):

(39)   a. $\langle$V, 1$\rangle$$<$$\langle$v,2$\rangle$ $<$$\langle$Pass(ive),3$\rangle$$<$$\langle$Prog,4$\rangle$$<$$\langle$Perf,5$\rangle$$<$$\langle$Mod(al),6$\rangle$$<$$\langle$Neg,7$\rangle$$<$$\langle$T,8$\rangle$ $<$$\langle$Fin,9$\rangle$$<$$\langle$C,10$\rangle$

    b. $\langle$N,1$\rangle$$<$$\langle$n,2$\rangle$$<$$\langle$Poss,3$\rangle$$<$$\langle$Num,4$\rangle$$<$$\langle$D,5$\rangle$$<$$\langle$Q,6$\rangle$

For example, we will have the following items:

(40)   a.   {T:8, tense:past, num:sing, case:nom}

    b.   {D:5, num:sing, case:acc, def:+}

We can now answer the question of how HoPs constrain the building of structure. We define a well formed structure as follows (assume an LI is a trivial syntactic object):

(41)   If α and β are syntactic objects and

        $\exists g \in \alpha \wedge \exists g' \in L(\beta)$

        such that $g, g' \in K_H$(Categorial Features in hierarchy H)

        $\wedge \, val(g) > val(g')$

then $\{\alpha, \{\alpha, \beta\}\}$ is a syntactic object.

L here is a function which returns the label of β.[5] (41) can be seen as incorporating an interface condition that allows only structures respecting HoPs, since only such structures are scopally interpretable. It is a variant of Merge, in that it builds structure. Let us call it HoP-Merge.

The function *val* returns the value of a feature (i.e. the second element of the pair). We will take the ordering relation $>$ to be undefined if the value returned by *val* is the empty set (by assuming that the relation $>$ does not order sets but rather numbers). HoP-Merge is then only possible between elements that bear valued categorial features.

To see how this works, take our previous example of *the many men* and concentrate just on categorial features:

(42)   a.   F(men) = $\{\langle N,1\rangle\}$

            b.   F(many) = $\{\langle Num,4\rangle\}$

            c.   F(the) = $\{\langle D,5\rangle\}$

HoP-Merge licenses a structure that looks as follows, since *many* bears a feature (Num) whose value is higher than the feature that *men* bears:

(43) $\{\{\langle Num, 4\rangle\}, \{\{\langle Num, 4\rangle\}\{\langle N, 1\rangle\}\}\}$

To derive (43), we take α to be *many* and β to be *men*. Our definition of HoP-Merge requires that there be some feature $g \in \alpha$ and some feature $g' \in L(\beta)$ such that they are both in the same hierarchy. In this case $g \in \alpha = \langle Num,4\rangle$ and $g' \in L(\beta) = \langle N,1\rangle$, which are both part of the same HoP, as defined above. The next condition is that the values be strictly ordered, which is true, since $4 > 1$. This licenses us to build a new syntactic object which is a set whose elements are α and the set $\{\alpha, \beta\}$.

We also allow *the men*, in a similar fashion:

(44) $\{\{\langle D, 5\rangle\}, \{\{\langle D, 5\rangle\}\{\langle N, 1\rangle\}\}\}$

and *the many men*:

(45) $\{\{\langle D, 5\rangle\}, \{\{\langle D, 5\rangle\}, \{\{\langle Num, 4\rangle\}, \{\{\langle Num, 4\rangle\}\{\langle N, 1\rangle\}\}\}\}\}$

---

[5]If we take a system where labels are eliminated (as in Collins 2002) then we need to ensure that g' is in the head of β, however that is defined in such a system.

Given the relevant HoP, *many the men* is ruled out: since the value of Num is not higher than the value of D, no structure is licensed which is labelled by Num but contains D:[6]

(46) $*[_{\{\langle \mathrm{Num},4\rangle\}}\{\langle \mathrm{Num}, 4\rangle\}[_{\{\langle \mathrm{D},5\rangle\}}\{\langle \mathrm{D}, 5\rangle\}\{\langle \mathrm{N}, 1\rangle\}]]$

From this operation on syntactic structures, it also follows that each LI must have a valued categorial feature, or else it cannot be ordered by HoP; given this, it is possible to remove the requirement in the specification of lexical items that they must contain a valued categorial feature, but I will leave this redundancy in place in the definition of LI for explicitness' sake.

The system outlined here raises a conceptual issue: the order of the scopal hierarchies is captured by associating categories with elements of the set of natural numbers which, themselves, follow a sequence. Ideally, one would want the ordering of the elements in a HoP to follow from semantic properties of the categories, rather than to be simply stipulated, as I have done here (see, e.g., Nilsen 2003). However, I'll stick with this technology here, as, once again, it allows us to be explicit.

3.1   Features Driving Syntax

3.1.1   Triggering Merge

In the system outlined here, features drive the various syntactic operations. We can think about this in an essentially Fregean way: the features have some property that needs to be satisfied. The property is satisfied by being in certain structural configurations (which are all basically extensions of sisterhood). So satisfying this property of a feature forces syntactic operations to apply so as to achieve the requisite configurations.

With this much in place, we now turn to Merge. In Adger 2003 I suggested that one variety of Merge is licensed by a matching of categorial features on the selector and the selectee (see also Collins 2002 and more recently, Pesetsky and Torrego 2007). We can implement this by taking the feature on the selector to be unvalued (i.e have the empty set as its value). For example, take the verb *devour* and a simple pronominal object *it*:

---

[6]I substitute labeled square brackets here for set brackets to aid readability. For the proper functioning of HoP-Merge, the labeled square bracket notation needs to be interpreted as the appropriate set brackets.

(47)   devour[V:1, D:∅] it [D:5]

We want to define the operation so as to ensure that *devour* will Merge with *it*, or with any other syntactic object bearing a valued D-feature. We also want to ensure that the operation will not allow *devour* to combine with a syntactic object that does not bear an appropriate category feature:

(48)   *devour[V:1, D:∅] [that[C] he was there]

Finally, we want the selectional feature, once it has done its job, to no longer be accessible to syntactic operations.

Let us call this variant of Merge Sel(ect)-Merge. We can define the triggering context for Sel-Merge as follows to achieve this:

(49)   If α and β are syntactic objects and

   a. $\exists g \in \alpha \wedge \exists g' \in L(\beta)$

   b.   such that $Cat(g) = Cat(g')$

   c. $\wedge \; val(g) = \emptyset \wedge val(g') \neq \emptyset$

The various parts of this definition have specific terminology in minimalism. (a) specifies the **probe** and the **goal** of the operation; (b) ensures that the appropriate features **match**; (c) is one version of what is usually termed the **activity condition**, requiring at least one of the relevant syntactic objects to bear an unvalued feature.[7]

Our definition of HoP-Merge involved a triggering context and an output, but we have not yet specified an output for Sel-Merge. Following a suggestion by Ash Asudeh (pc), we can take the valuing operation to be unification of values (Shieber 1986), where the empty set will unify with any other value, but no two other values can unify. That is, we specify token identity of feature value in a structure which is the output of Merge. More concretely:

---

[7](c) doesn't actually encompass the full range of effects of the Activity Condition, which specifies that both the probe and goal categories bear an unvalued feature. This version just requires the probe to bear one. The other half of the condition, which is essentially a version of the GB Case Filter, specifies that the goal category (i.e. β) should have an unvalued feature if it is to enter into a syntactic operation. An alternative, compatible with the implementation offered here, might be to take case to be the morphological realization of a tense or aspect feature (e.g. Pesetsky and Torrego 2001, Svenonius 2002). Other possibilities are mentioned in the literature (Řezáč 2004).

(50)   a.   If $val(g) = \emptyset$ and $val(g') = +$, then $val(g) \sqcup val(g') = +$

       b.   If $val(g) = \emptyset$ and $val(g') = -$, then $val(g) \sqcup val(g') = -$

       c.   If $val(g) = \emptyset$ and $val(g') = \emptyset$, then $val(g) \sqcup val(g') = \emptyset$

Similar proposals within minimalism have been made by Frampton and Gutmann 2002, Pesetsky and Torrego 2004, Bhatt 2005 among others.[8] I'll present some evidence that this interpretation of valuation is correct in section 3.1.2. We then complete (50) as follows:

(51)   If α and β are syntactic objects and

       a. $\exists g \in \alpha \wedge \exists g' \in L(\beta)$

       b.   such that $Cat(g) = Cat(g')$

       c. $\wedge\ val(g) = \emptyset \wedge val(g') \neq \emptyset$

       d.   then $\{\alpha - g, \{\alpha - g, \beta\}\}$ is a syntactic object where $val(g)$ and $val(g')$ are replaced by $val(g) \sqcup val(g')$[9]

Taking our example with *devour* and *it*, we have the LI $\{\langle V,1\rangle, \langle D,\emptyset\rangle\}$ for the former and $\{\langle D,5\rangle\}$ for the latter, so it is true that there are features g and g' where $Cat(g) = Cat(g')$, and where one of these is valued and the other unvalued. We are therefore licensed to apply Merge, resulting in the following representation (I again use labeled square brackets in place of set brackets for readability):

(52) $[_{\{\langle V,1\rangle\}}\{\langle V, 1\rangle, \langle D, 5\rangle\}\{\langle D, 5\rangle\}]$

The label on the new object lacks the selectional D-feature of its head, partly capturing the Fregean intuition that this feature is satisfied. This ensures that this valued feature cannot serve as the goal for any higher operation. We could simplify the theory here by taking the label and the head to be identical (as in Brody's (2000) Mirror Theory). I

---

[8]If this is the right interpretation of valuation, then we may be able to remove the activity condition entirely, since the only pairs of matching features which will be able to unify their values will be those pairs of matching features (at least one of) which will have an empty-set value. However, this will entail that valued features will be able to be 'linked' by the Agree operation, and it's not at all clear that this has only beneficial effects.

[9]It might seem odd that the values of g and g' are unified and then the whole feature g is removed from the structure, but the definition does no harm here, and allows maximal uniformity when we come to other syntactic operations.

return to this issue in the discussion of Agree, below.

It is not possible to generate the example where we attempt to create a syntactic object by Merging *devour* and a CP, since Sel-Merge will not license this as a legitimate syntactic object (and nor will HoP-Merge).

We now need to return to our definition of lexical item and revise it. We previously stated the following:

(53)   a.   A lexical item is a set $\{K, F_1, ..., F_n\}$,

           (i)   K is an ordered pair $\langle$Cat, N$\rangle$

           (ii)   Cat is drawn from the set {C, D, T, Num, Asp, ...}-$\emptyset$, and N is drawn from the set of natural numbers above 0

    b. F_i is a pair $\langle$Att, Val$\rangle$, where Att is drawn from a finite set of MS-features and Val from a finite set of values

Given the theory of selection we have just espoused, we now need to allow a lexical item to also contain unvalued categorial features, which we notate as S (for selectional features):

(54)   a.   A lexical item is a set $\{K, S, F_1, ..., F_n\}$,

           (i)   K is an ordered pair $\langle$Cat, N$\rangle$

           (ii)   S is an ordered pair $\langle$Cat, $\emptyset\rangle$

           (iii)   Cat is drawn from the set {C, D, T, Num, Asp, ...}-$\emptyset$, and N is drawn from the set of natural numbers above 0

    b. F_i is a pair $\langle$Att, Val$\rangle$, where Att is drawn from a finite set of MS-features and Val from a finite set of values

We noted above that there was a certain redundancy between HoP-Merge and our definition of lexical item: to build a structure using HoP-Merge, any lexical item will have to have a valued categorial feature, or else it cannot be 'placed' in the structure. Sel-Merge as defined immediately above is a second way in which elements come to be in structure. It requires an unvalued categorial feature on one of the two syntactic objects. Note that the selectional feature is not obligatory, unlike the categorial feature, since the categorial feature is necessary and sufficient to allow the lexical item to enter the syntactic computation, while the selectional feature need not play any role. Selectional features are therefore not obligatory. A further question is whether we allow more than one (see, e.g. Stabler

1997 who has no limit on the number of selectional features). We will simply stipulate in our definition of what makes a well formed lexical item that at most one categorial feature and one selectional feature is possible.[10]

With this definition in place we have a typology of lexical items:

(55)   a. $\{K, S, F_1, ..., F_n\}$ an item with both selectional and MS-features

      b. $\{K, F_1, ..., F_n\}$ an item bearing no selectional features, only MS-features

      c. $\{K, S\}$ an item bearing no MS-features, only an S-feature

      d.   *$\{S, F_1, ..., F_n\}$ impossible, no valued categorial feature

      e.   *$\{F_1, ..., F_n\}$ impossible, no valued categorial feature

      f.   *$\{S\}$ impossible, no valued categorial feature

      g. $\{K\}$ an item bearing only a categorial feature

This typology, together with the two kinds of Merge, partly gives us a reconstruction of the intuitive difference between functional and non-functional lexical items. Lexical items like (b) are what we usually term functional categories—they cannot select their complements and so can never project in a Sel-Merge operation, only in a HoP-Merge one. Lexical items lacking a valued categorial feature (d, e, f) are impossible, since they can never be entered into a HoP. Both functional and non-functional lexical items can bear selectional features, although only non-functional ones can select their complements, while functional ones can only select their specifiers, so (a), (c) and (g) can be either lexical or functional.[11]

The final thing to note about both HoP-Merge and Sel-Merge is that they both create local tree structures on the basis of matching categorial features: there is a deep commonality between them. One could abstract out a general operation (Merge) as follows:

(56)   If α and β are syntactic objects and

      a. $\exists g \in \alpha \wedge \exists g' \in \beta$

---

[10]I see no obvious way of deriving this stipulation at the moment. The intuition that motivates it is as follows: since hierarchy is given in HoP, we would like to avoid it in the lexical entries, but with two selectional features it seems we need it. Two selectional features will require some order on their checking, but no order is available inside a set. This redundancy is avoided if only one selectional feature is permitted on a lexical item.

[11]A recent trend in Minimalism (Marantz 1997, Borer 2005) has been to take the lowest level of HoPs to bear no features at all, denying any syntactic selectional properties to lexical items.

b. such that $R(g, g')$,

c. then $\{\alpha, \{\alpha, \beta\}\}$ is a syntactic object

The two different types of Merge would then depend on the nature of some featural relation R between the two syntactic objects concerned: if the projecting object bears a lexical category feature (N, V, A), then it needs to have a selectional feature that matches the categorial feature of its complement; if it is a functional category, then its complement must occur lower in the relevant HoP. This, in turn, is related to the fact that the lexical categories are the lowest elements of the HoPs and so can never Merge with a complement via HoP-Merge.

There are various alternative ways of expanding or eliminating R, leading to related but different versions of the approach. For example, one might assume that each category in a HoP selects its sister via an unvalued feature, reducing HoP-Merge to Sel-Merge (e.g. Abney 1987). Alternatively, one might deny the existence of syntactic selectional features and require all Merge to be via HoP-Merge (e.g. Borer 2005). One could eliminate the requirement that there is some R, and assume that all constraints on Merge reduce to semantic (interface) constraints, so that the syntactic system allows anything to Merge with anything. We leave these alternatives aside here in favour of the HoP-Merge/Sel-Merge distinction.

If we take Merge to be triggered, as I have suggested here, then we need some way of ensuring that it actually takes place. For example, nothing we have said so far disallows (57):

(57)   *Anson hit

The verb *hit* is specified as {V:1, D:∅} and by virtue of its valued categorial feature can be the lowest item in the 'verbal' HoP. It follows that we can build further structure above it, introducing the subject and the inflectional material. However, we have so far said nothing which forces the selectional feature it bears to be 'satisfied'; that is, we haven't yet quite made good on the intuition we began with that features could have a Fregean property, that they somehow need to be satisfied.

The way we do this is by specifying an interface constraint that ensures that all features have a value at the interface to interpretation. This constraint is usually called

Full Interpretation, and here's the version I'll adopt:[12]

(58)   Structures containing unvalued features are marked as deviant at the interfaces.

On the usual derivational interpretation of Minimalism, we can further constrain this by specifying that the output of Merge is submitted to the interface at certain points in the derivation. The most stringent requirement would be that each output is immediately submitted to the interface (giving a directly compositional interpretation of Minimalism); a weaker constraint is that chunks of the derivation are submitted to the interfaces at specified points (perhaps at certain points within each HoP, or at the end of a HoP), giving what Chomsky calls 'derivation by phase' (Chomsky 2001).

Chomsky suggests that phases are propositional-like units of structure: verb phrases which have all of their arguments (vPs) and clauses. Alternative suggestions abound: Svenonius (2004) proposes that phases are simply chunks of the derivation which have no unvalued features. Adger (2003) suggested that CPs, PPs and DPs are phases, essentially making the notion of phase identical to that of a full HoP.

3.1.2 Triggering Agreement

We have seen how selection proceeds: Merge, triggered by the need to satisfy unvalued categorial features, builds structures where two elements bearing the relevant features become sisters, with the whole structure bearing the same features as the 'selector', minus the selectional feature itself. We can also use the unvaluedness property to tackle agreement and (case-)government effects. Again, lack of value for a feature triggers a syntactic operation, although this is not in this instance a structure building operation; it merely alters properties of matching features. The operation is called Agree and we define it as follows:

(59)   In a structure $\{\alpha, \{\alpha, \beta\}\}$,

       If $\exists g \in \alpha \land \exists g' \in \bigcup \Delta(\beta)$

       such that $att(g) = att(g')$

       $\land \ val(g) = \emptyset \land val(g') \neq \emptyset$

       then $\{\alpha - g, \{\alpha - g, \beta\}\}$ is a syntactic object where $val(g)$ and $val(g')$ are replaced

---

[12]See Frampton and Gutmann 2004 for discussion of a system where syntax is 'crash-proof' and so filters on derivations like Full Interpretation are automatically met by the system.

by $val(g) \sqcup val(g')$

Here Δ is a function which returns the set of terms strictly dominated by its argument (it is basically the inverse of L, which returns the labels—see Chomsky 1995, p247.), so $\bigcup \Delta(\beta)$ is simply the union of all the subsets of β that have sets as members. The function *att* is analogous (possibly identical) to *cat*, returning the attribute of a MS-feature. C-command is built into the triggering configuration via Δ.

A further property of Agree, shared with HoP-Merge, is that the unvalued feature is always 'higher' in the structure. This immediately precludes analyses like the Adger (2003) account of English verbal inflection outlined above. See section 3.1.3 for discussion.

The decision to remove the feature g from the head has an implication for morphology. If g is not present when morphological interpretation of the syntax takes place, then we expect that the morphology is not sensitive to g, contrary to what is usually the case. The literature contains a number of ways of tackling this issue: Chomsky (1995) suggests a distinction between deletion of the feature and erasure, where a deleted feature is not available for further syntactic computation but is available to the morphology, while Chomsky (to appear) suggests that the valuing operation (i.e. Agree in (59)) takes place as part of the transfer of information to the morphological component, so that the feature g is visible to the morphology, but not visible to further syntactic computation. For concreteness here, I'll simply assume that a morphological interface rule takes as its input the information $\{\alpha - g\}$, so g is available to condition the application of the rule, while as far as the syntax is concerned, $\{\alpha - g\}$ is actually computed, so g is no longer available in the syntactic derivation.

For example, take a sentence like (60):

(60)   *We asks you

Under this system, *asks* (or rather the tense node associated with it) bears the unvalued features [number:$\emptyset$, person:$\emptyset$], while the pronoun bears [number:plural, person:first]. Assuming that *asks* c-commands *we* at some stage of the derivation, the structural relation required by Agree holds. This will unify the values, so that the features on (the T associated with) *ask* are also specified as [number:plural, person:first]. The morphological interface simply has no way of spelling out such a feature bundle as *asks* in Standard English, and so (60) is never generated. Full Interpretation requires that the features on

the verb receive a value before the morphological interpretation of these features takes place.

Some evidence that agreement requires unification comes from Hindi long distance agreement. Bhatt (2005) shows that, in Hindi, a higher verb can agree with the object of a lower verb, as in (61):

(61) Firoz-ne   rotii   khaa-nii chaah-ii
      Firoz-ERG bread.F eat-INF.F want-PF.3FSG
      'Firoz wanted to eat bread.'

Here we see the infinitive in the embedded clause agrees with its object, and that, in such a case, the higher selecting verb also agrees with the object. The higher verb exhibits this kind of agreement when its own subject is ergatively marked. Such agreement is optional (with some pragmatic effects):

(62) Firoz-ne   rotii   khaa-naa chaah-aa
      Firoz-ERG bread.F eat-INF.M want-PF.3MSG
      'Firoz wanted to eat bread.'

Bhatt also shows that the infinitival verb agrees with its object only if the matrix verb does, giving the following contrast:

(63) Shahrukh-ne   [ṭehnii   kaaṭ-nii] chaah-ii   thii
      Shahrukh-ERG branch.F cut-INF.F want-PF.3FSG be.PST.FSG
      'Shahrukh had wanted to cut the branch.'

(64) *Shahrukh-ne   [ṭehnii   kaaṭ-nii] chaah-aa   thaa
      Shahrukh-ERG branch.F cut-INF.F want-PF.3MSG be.PST.MSG
      'Shahrukh had wanted to cut the branch.'

(64) shows infinitival agreement with no finite agreement, while (65) shows finite agreement with no infinitival agreement. Both are bad:

(65) *Shahrukh-ne   [ṭehnii   kaaṭ-naa] chaah-ii   thii
      Shahrukh-ERG branch.F cut-INF.M want-PF.3FSG be.PST.FSG
      'Shahrukh had wanted to cut the branch.'

One might assume from this that the agreement on the matrix verb is some sort of copy of the agreement on the embedded verb, and it is the latter that is the core phenomenon. However, Bhatt shows that this will not explain the following example:

(66) *Shahrukh [ṭehnii   kaaṭ-nii] chaah-aa   thaa
      Shahrukh branch.F cut-INF.F want-PF.3MSG be.PST.MSG
      'Shahrukh had wanted to cut the branch.'

Since *Shahrukh* is not ergative, and is masculine, the matrix verb should be able to agree with it. But, on the hypothesis that the long-distance agreement on the matrix verb is parasitic on the independently occurring local agreement on the infinitive, (66) should be perfectly well formed.

Bhatt, following, e.g. Shieber (1986), Frampton and Gutmann (2002) suggests a unification type account. The idea is that the infinitival agreement is blocked from happening in the lower clause for some reason (Bhatt suggests that this is because the features on the infinitive are not a 'full agreement' set, a suggestion which, following Chomsky 2001 adds a further condition to Agree such that only a full ϕ-feature set satisfies the case requirements of the lower DP; I'm skeptical about this analysis of case, and so do not adopt this idea in the theory outlined here. There are other possibilities, such as the features not being in the appropriate c-command relationship). However, the finite element creates an agreement relation between its features and the as yet unvalued features of the infinitive. Bhatt calls this co-valuation, and we can think of it as unification. I notate it with the index 1 in the derivation below. The finite verb then sets up an agreement relationship with the object, and this values the verb's features, as a side effect valuing the infinitive's gender feature:

(67)   branch[fem:+, sg:+] cut[fem:∅] want[fem:∅, sg:∅] →
        branch[fem:+, sg:+] cut[fem:1] want[fem:1, sg:∅] →
        branch[fem:+, sg:+] cut[fem:+] want[fem:+, sg:+]

The definition of Agree here is very similar to the definition of Sel-Merge. Both set up a requirement that the two elements entering into the operation have matching features and both require unification of those features. Furthermore, the feature matching operation is defined in asymmetric terms, such that one of the features 'projects'. The difference is that Merge builds structure while Agree operates on pre-built structure. A further difference is that Merge is a local operation, looking at the immediate labels of the two Merged elements, while Agree looks into one of the structures via the Δ function. We will see in the next section that the operation of Move involves a cross-cutting of these

properties.

3.1.3 Triggering Movement

The final important property of features in the Minimalism is their role in triggering Movement. In Chomsky (1995) and Adger (2003) this is dealt with via the concept *strength* (notated by an asterisk after the feature [F:val*]).

However, given the bifurcation between categorial and MS-features we have introduced, we can rethink strength as simply being a notation for an unvalued categorial feature: that is, since categorial features can only be valued in a Merge configuration (as we have already seen), Movement takes place to ensure the appropriate Merge configuration for valuation of a selectional feature. We define Move, then, by cross-cutting our definitions of (Sel-)Merge and Agree:

(68)   In a structure $\{\alpha, \{\alpha, \beta\}\}$,   Like Agree

           If $\exists g \in \alpha \wedge \exists g' \in \bigcup \Delta(\beta)$ Like Agree

           such that $Cat(g) = Cat(g')$ Like (Sel-)Merge

           $\wedge val(g) = \emptyset \wedge val(g') \neq \emptyset$ Like both

           then $\{\alpha - g, \{\tau(g'), \alpha - g\}\}$ is a syntactic object where $val(g)$ and $val(g')$ are replaced by $val(g) \sqcup val(g')$ Like both.

Here τ is a function which returns some constituent that dominates its argument. Note, just as in Sel-Merge, the new object created lacks the categorial feature that triggered the operation.

Note how this triggering configuration conflates aspects of the previous configurations for Merge and Agree. Like Agree, it operates on a pre-built structure, it searches into that structure. Like Merge, however, it operates on categorial, rather than non-categorial, features, and it triggers the creation of a new structure. Like both, it requires a probe-goal relation, matching of the feature, the unvaluedness of at least one of the features, and the unification of their values.

Let's see how the system runs with a simple sentence like *he kissed him*. I have not discussed case features in any great detail as yet, and I'll simply assume here that they are unvalued on the assigner and valued on the assignee.[13]
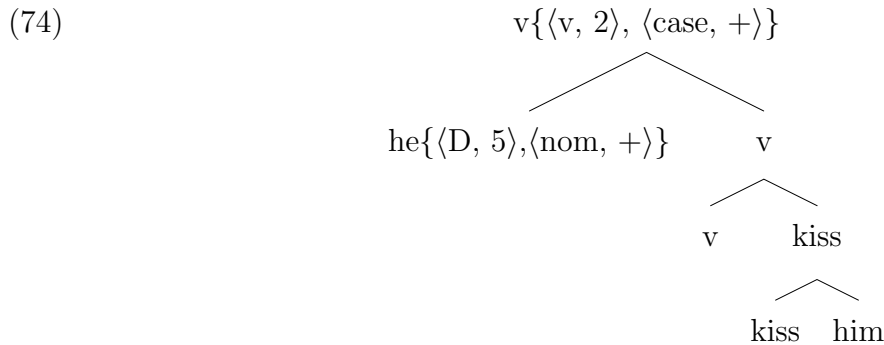
---

[13]This is a different analysis to that in Adger 2003 as the theory developed there allowed the unvalued

(69)   (a) him = {⟨D, 5⟩, ⟨acc, +⟩, ...}

(b) kiss = {⟨V, 1⟩, ⟨D, ∅⟩, ...}

(c) Since both bear a categorial feature D, but D on the verb is unvalued, we Merge the two syntactic objects, projecting the verb (minus its selectional feature), and we value the verb's D-feature

(70)   kiss{⟨V, 1⟩}

                          kiss   him{⟨D, 5⟩, ⟨case,∅⟩}
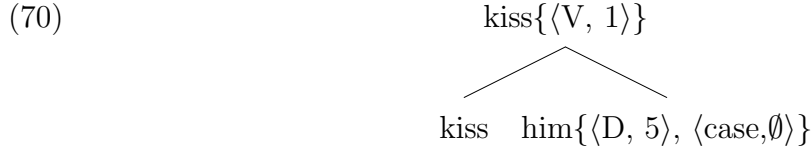
We then take another lexical item, v:

(71)   v = {⟨v, 2⟩, ⟨D, ∅⟩, ⟨acc, ∅⟩}

(72)   (a) Since the value of the categorial feature of v is higher than that of V, we can HoP-Merge these projecting the former.

(b) In the resulting structure, we have a trigger for Agree, since the case feature on v matches that on *him.* We therefore value the former.

(c) The projection of v is now: {⟨v, 2⟩, ⟨D, ∅⟩, ⟨acc, +⟩}

(73)   (a) he = {⟨D, 5⟩, ⟨nom, +⟩}

(b) Since the attributes D are the same on *he* and on v, but D on the v is unvalued, we Merge the two, projecting v.

(74)   v{⟨v, 2⟩, ⟨case, +⟩}

                  he{⟨D, 5⟩,⟨nom, +⟩}   v

                                          v   kiss

                                              kiss   him

The next step for this simple sentence is to Merge T:

(75)   LI = {⟨T, 8⟩, ⟨D, ∅⟩, ⟨nom, ∅⟩, ⟨tense, past⟩}

feature to be either higher or lower than the valued one. The theory in the present chapter is stricter, and closer to the primary literature, and rules out such an analysis.

29

(76)   Since the value of the Cat attribute of (75) is higher than that of v, we Merge (74) with (75) projecting T.

(77) $T\{\langle T, 8\rangle, \langle D, \emptyset\rangle, \langle nom, +\rangle, \langle tense, past\rangle\}$

$$
\begin{array}{c}
\text{T} \qquad v\{\langle v, 2\rangle, \langle acc, +\rangle\} \\
he\{\langle D, 5\rangle, \langle nom, +\rangle\} \qquad v \\
v \qquad kiss \\
kiss \qquad him
\end{array}
$$
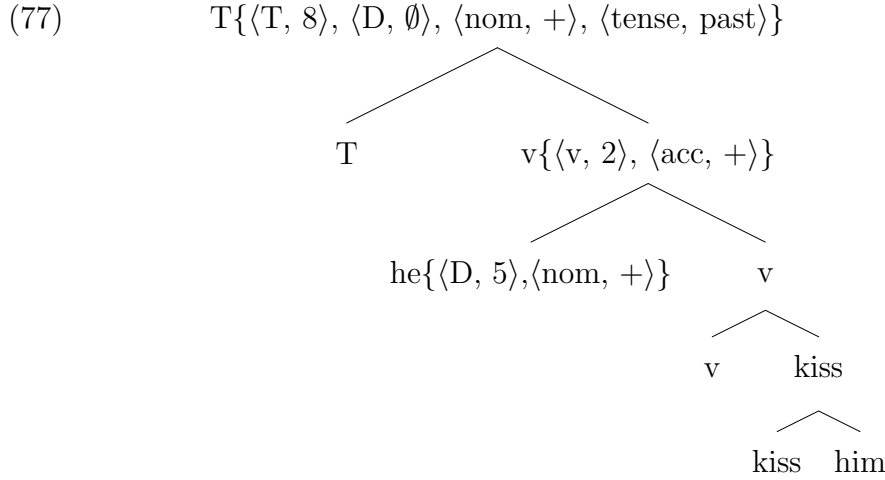
(78)   (a) The resulting structure is a trigger for Agree, and values the nominative case feature analogously to the valuing of the accusative one lower down.

(b) This structure is also a trigger for Internal Merge, since the D feature on T is unvalued. It matches with the interpretable D feature of the subject pronoun, and licenses an application of Merge of a term containing the D-feature of the subject pronoun with T. In this case we take the subject pronoun itself.

(79) $T\{\langle T, 8\rangle, \langle nom, +\rangle, \langle tense, past\rangle\}$

$$
\begin{array}{c}
he\{\langle D, 5\rangle, \langle nom, +\rangle\} \qquad \text{T} \\
\text{T} \qquad v\{\langle v, 2\rangle, \langle acc, +\rangle, \langle tense, past\rangle\} \\
he\{\langle D, 5\rangle, \langle nom, +\rangle\} \qquad v \\
v \qquad kiss \\
kiss \qquad him
\end{array}
$$
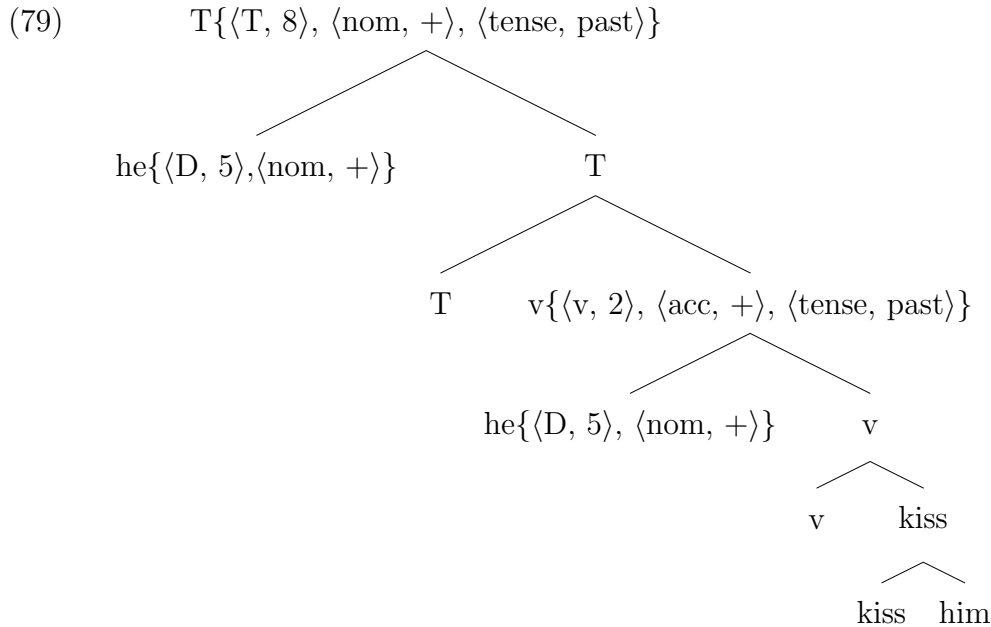
In this final structure all of the unvalued features have now been valued via Agree or Merge. I have neglected to move the verb to the left of *v* in this tree. In Adger (2003), I took inflectional morphology to be dealt with via (head) movement. I'm now more

inclined to take it to be read off the functional spine of the clause, as in Mirror Theory (Brody 2000). The idea is that the morphology simply mirrors the functional spine of the clause, so that the verb is just the pronunciation of V+v+T with the relevant features. This will account for the morphological position of the past tense on the verb.

Two major questions have not been answered yet: why does the function τ choose the pronoun to move, rather than some higher term? And why does T's D feature match with that of the subject rather than the object?

The answer to the second question is the so called *Minimal Link Condition.* This is a condition on closeness which is added into the triggering configurations for Merge and Agree. It essentially ensures that the two features in a matching relation have no other matching feature 'between' them. It is relevant only for Agree and Internal Merge (as for external Merge the features can never, by definition, have any other feature between them). We can specify the MLC as a constraint on all syntactic operations. It is redundant for External Merge. I give the revised versions of the configurations for Agree and Internal Merge:

(80)   In a structure $\{\alpha, \{\alpha, \beta\}\}$,

        If $\exists g \in \alpha \wedge \exists g' \in \bigcup \Delta(\beta)$ Probe-Goal

        $\wedge \neg \exists g'' \in \gamma, \beta \in \Delta(\gamma)$ MLC

        such that $att(g) = att(g') = att(g'')$ Match

        $\wedge\ val(g) = \emptyset \wedge val(g') \neq \emptyset$ Activity Condition

        then $\{\alpha, \{\alpha, \beta\}\}$ is a syntactic object where $val(g)$ and $val(g')$ are replaced by $val(g) \sqcup val(g')$ Value

(81)   In a structure $\{\alpha, \{\alpha, \beta\}\}$,

        If $\exists g \in \alpha \wedge \exists g' \in \bigcup \Delta(\beta)$ Probe-Goal

        $\wedge \neg \exists g'' \in \gamma, \beta \in \Delta(\gamma)$ MLC

        such that $Cat(g) = Cat(g') = Cat(g'')$ Match

        $\wedge val(g) = \emptyset \wedge val(g') \neq \emptyset$ Activity Condition

        then $\{\alpha - g, \{\tau(g'), \alpha\}\}$, is a syntactic object where $val(g)$ and $val(g')$ are replaced by $val(g) \sqcup val(g')$ Merge and Value

The question of which constituent τ picks out is more vexed, and is basically the question of pied-piping. For the case we have here, it is straightforward to simply define τ to return

the largest constituent that contains all instances of the feature that is τ's argument. However, it is well known that there are important problems with this type of solution when we come to wh-movement. The theory developed here basically predicts that what is pied-piped will always be a projection of the element which introduces the relevant feature, since the definition of Merge involves the projection of the features of the head. For the classic Ross examples, such as (82), this is obviously problematic:

(82) a. Letters, which I specified the height of, were on the cover of the report.
      b. Letters, [the height of which] I specified, were on the cover of the report.

Which is selected by the preposition *of*. The categorial feature [Wh] is Merged in the projection of the DP, presumably below the highest functional projection in DP (which we'll assume is D). In any event, this feature is certainly no higher than D, so the representation of the (a) example is (83), where the D is null:

(83) Letters, [Wh which] I specified the height of [D ⟨[Wh which]⟩], were on the cover of the report.

However, the (b) example is potentially problematic, since the Wh-feature is embedded in the structure that is moved. One possibility here is that the categorial feature which drives the movement can be Merged higher in the DP projection, some distance from its apparent morphological locus. For example, in (84), if the feature triggering Internal Merge is g, then some α containing g must be able to Merge at various levels in the higher HoP. The Wh-feature would then be checked under Agree rather than being the trigger for Move :

(84) Letters, [C[g:∅] I specified [α[f:∅,g:8] the height of which[f:+]], were on the cover of the report. →
      Letters, [α[f:+,g:8] the height of which[f:+]] [C[g:8] I specified, were on the cover of the report.

See Cable (2007) for a theory of pied-piping with these characteristics.

One final issue is that in the definitions of Agree and Internal Merge here I have taken the unvalued feature to be the higher one. This is standard (see, e.g. Chomsky 2001). However, in Adger 2003, I proposed a number of analyses that relied on the lower feature

being unvalued. If these analyses turn out to be on the right lines then the activation condition must be specified as a disjunction, allowing either or both features to lack a value.

4   No Complex Values Revisited

A recurrent issue in feature systems which allow embedded feature structures is that nothing disallows a selectional feature having as a value a category which itself bears a selectional feature. For example, in HPSG, without further constraints, it's possible to write such structures as the following:

(85)   V[SUBCAT⟨ N[SUBCAT ⟨ P ⟩] ⟩]

This kind of specification will allow a lexical entry for a verb which subcategorizes for a noun with a PP complement. However, natural languages do not seem to work like this: they allow only one level of subcategorization (see Sag's chapter in this volume for extensive discussion).

The feature system developed above does not allow complex values, so a feature specification like (85) is impossible. There is no legitimate LI that looks as follows:

(86)   *[V:1, N:P:∅].

The nearest we can get is to specify a selectional feature on the verb for both an N and a P:

(87)   [V:1, N:∅, P:∅]

This will allow us to Merge with a noun but it also requires us to Merge with a P. However, this merely specifies a V that selects an N and a P, not one that selects an N that selects a P. Moreover, the specification of lexical item we gave above blocks multiple selectional features, so even (87) is not a well formed lexical item.

Rich feature structures also allow us to specify properties of the complement. For example, in a system that allows complex categories, without further stipulation, it's possible to write a lexical item which looks as follows:

(88)   V[SUBCAT⟨ S[DAUGHTER [CAT N] animate] ⟩]

This would be a verb which subcategorizes for a clausal complement which has an animate nominal daughter. It's impossible to write such a lexical item in the system developed above, as there simply isn't enough structure to the features. The impossibility of (88) follows from the theory of feature structures itself.

However, this restriction might get us into trouble. Although it's true that we don't seem to have verbs which look as follows:

(89)   *V where V selects CP containing an animate DP

(90)   *V where V selects CP containing a perfect auxiliary

we also now predict that we can't have verbs which specify a PRO subject:

(91)   *V where V selects CP containing a PRO subject

There's simply no way to write a lexical item in our feature theory which will directly have this effect:

(92)   try [V:1, C[PRO:∅]:∅]

However, we do seem to have such verbs, as the following contrast shows:

(93)   a. *I tried him to be there
        b.   I tried to be there.
        c. *I tried that you are here

One approach to this problem would be to appeal to semantic selection. That is, we specify the verb *try* as [C:∅], so that both (b) and (c) in (93) are syntactically impeccable (as in Grimshaw 1979, Stowell 1983). The problem with (c) is that a finite clause with indicative mood and the complementizer *that* is semantically incompatible with *try*.

The ungrammaticality of the (a) example would then follow if *try* selects C, but the null C in English can't license an accusative. This analysis receives support from the fact that it is possible to have an overt subject in the complement of *try*, but it requires the accusative licensing complementizer *for*:

(94)   ?I tried (very very hard) for him to be there.

The only other parse available for the (a) example above is that *him to be there* is a TP, but this will of course mean that the [C:∅] feature on *try* is not valued.

An alternative approach to this is to say that there must be a category distinction, rather than a featural distinction at play here (e.g Rizzi 1997's suggestion that the C layer includes a Fin(iteness)P, distinct from the top of the relevant HoP). We could then have *try* select for that category, while *that* would realize another category, ruling out (93-c):

(95)   { V:1, Fin:∅ }

Under this analysis, both the zero element that licenses PRO and *for* would be versions of Fin. For an argument that the apparent case licensing correlations between finiteness and overt subject should be distinguished from the semantic category of finiteness, see Adger (2007).

A related issue arises for verbs which select TP rather than CP (such as raising and ECM class verbs). There's a strong generalization that such verbal complements in English can't be finite:[14]

(96)   a. *He seems is here

        b.   He seems to be here

(97)   a. *I took him/he is a priest

        b.   I took him to be a priest.

(98)   Generalization: selected TP is never finite (raising, ECM)

This generalization is a surprise if a complex value like [V:1, T[finite:−]:∅] or its equivalent is possible. There's no reason not to write a lexical item which will license such structures.

To capture the generalization, I argued in Adger 2003 that finite T bears a clause-type feature which has to be valued by C (this is the feature that is responsible for T-to-C movement in questions as opposed to declaratives). This unvalued feature needs to be valued, but in the absence of C there is no element bearing a matching feature which is local enough (the T of the higher clause, in this analysis, does not enter the derivation in time and the structure is interpreted with an unvalued clause-type feature; I suggested a

---

[14]I'm assuming that cases like *I believe he is here* involve at least Fin, if not a full CP. I use the example of the ECM verb *take* here as it does not allow a corresponding finite complement, even with an overt *that*.

different route to the generalization in Adger 2003, one which is not open to us under the unification interpretation of valuation).

A further issue which raises similar questions comes from the apparently very detailed selectional requirements of prepositions. For example, the verb *depend* seems to require the preposition *on*:

(99)   I depend on/*in/*to/*under ... cocaine

Other verbs seem to require classes of prepositions:

(100)   I put the book on, in, under, ..., *to the table.

In a system where we can specify complex values, this is easy to capture:

(101)   depend {V:1, P[on]:∅}

(102)   put {V:1, P[location:+]:∅}

Under the theory developed in this chapter, these are not possible LIs.

I think that the solution to this problem is that there is no category P. Rather P is a cover term for a whole range of category labels involving semantic notions like location, path, direction etc (see Svenonius 2007). (102) is really just:

(103)   put {V:1, Loc:∅}

Of course it now follows that the feature [Loc] is a category feature (since it enters into selection); since it is not an MS-feature, it shouldn't enter into long distance in situ agreement, which seems correct. It should also be possible to single out locative prepositions for movement, arguably what happens in Locative Inversion.

The idiosyncratic P-selection of verbs like *depend* is a little trickier, but note that *depend* also allows *upon*. It may very well be the case that {*on*, *upon*} form a category, but I leave this for future work.

A yet harder problem is *tough*-constructions, where certain adjectives really do seem to select for a complement which contains a trace:

(104)   Anson is easy/*possible to please.

It's straightforward to analyse these in a system which allows complex features, for example by specifying that the complement contain a gap, perhaps via a GPSG-style SLASH feature:

(105)   easy {A:1, C[SLASH:NP]:∅}

However, the system we have developed above makes such an analysis unstateable. One option would be to propose a category which is interpreted as a function from propositions to predicates (a certain kind of a relative clause), and to have the adjective select for that category.

A complex values system would also allow us to state generalizations like the following:

(106)   Plural DPs move to Spec TP.

However, on the assumption that plural is not a categorial feature (an assumption motivated by the fact that it enters into the Agree relation), such a generalization cannot be easily captured in the system outlined here. Merge, whether Internal or External, operates only on categorial features, rather than on MS-features. Given this we cannot write a lexical item for T which will embody the generalization in (106):

(107)   *[T:8, D[plural]:∅]

The closest we can get is:

(108)   [T:8, D:∅, plural:∅]

But this won't do the job for us: there is no dependency between the value of the number feature [plural] and the selectional feature [D], and so no way to ensure that only plural DPs move to the specifier of T. The only way to capture such a generalization would be to show that [plural] is a categorial feature rather than an MS-feature. However, that would entail that [plural] was a feature that could be selected for, and that it should not enter into Agreement. Neither of these claims is true. The way we have set up the system, movement and selection should always correlate.

The No Complex Values hypothesis restricts the theory of selection in an interesting way, blocking certain analyses of phenomena, and, at the same time, ensuring locality of selectional properties. Given the way we have defined Merge, Move and Agree, so

that they are sensitive to the categorial/MS-feature distinction, we also make predictions about correlations in languages between what is selectable, what is moveable, and what enters into long distance agreement. These various predictions need to be worked out on a case by case basis, but the prospects look interesting, to say the least.

5 Conclusion

In this chapter I've attempted to develop a fairly explicit proposal for what a feature theory in Minimalism might look like, which attempts to resolve the inevitable tension that arises between the simplicity of theories and their restrictiveness. I've argued that one can maintain a fairly simple theory of features by adhering to the No Complex Values hypothesis, and have shown how that leads to a certain restrictiveness in the theories of selection and movement.

I should stress that the theory is certainly not as 'minimal' as one might like: I have made stipulations about the structure of lexical items (for example, the stipulation that any lexical item can contain at most one categorial and one selectional feature), which one would hope to derive from more fundamental properties of the system; I have allowed a multi-valent feature system, which is probably too rich, and whose semantics is not as transparent as a simple binary system; I have stipulated a version of the activation condition, although it seems almost derivable, if we take feature valuation to be unification; I have stipulated the deletion (subtraction) of triggering features in the projection line, something which should probably follow from an interface condition (Brody 1997, Adger and Ramchand 2005). Still, I hope that the explicitness of the proposals outlined here will be helpful in the further development of feature theory within Minimalism, and will also be useful to those who come from different perspectives.

References

Abney, S. P. 1987. The English noun phrase in its sentential aspect. Ph.D. thesis, MIT, Cambridge MA.

Adger, David. 2003. *Core Syntax*. Oxford: Oxford University Press.

Adger, David. 2006. Remarks on minimalist feature theory and move. *Journal of*

Linguistics 42:663–673.

Adger, David. 2007. Three domains of finiteness: a Minimalist perspective. In Irina Nikolaeva, ed., *Finiteness: all over the clause*, 23–58, Oxford: OUP.

Adger, David and Ramchand, Gillian. 2005. Move and Merge: *Wh*-dependencies revisited. *Linguistic Inquiry* 36:161–194.

Asudeh, Ash and Toivonen, Ida. 2006. Symptomatic imperfections. *Journal of Linguistics* 42:395–422.

Bhatt, Rajesh. 2005. Long distance agreementi n Hindi-Urdu. *Natural Language and Linguistic Theory* 23:757–807.

Bobaljik, Jonathan. 1995. Morphosyntax: On the syntax of verbal inflection. Ph.D. thesis, MIT.

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

Brody, Michael. 1997. Perfect chains. In Liliane Haegeman, ed., *Elements of Grammar: Handbook of Generative Syntax*, 139–167, Dordrecht: Kluwer.

Brody, Michael. 2000. Mirror theory: syntactic representation in perfect syntax. *Linguistic Inquiry* 31:29–56.

Cable, Seth. 2007. Q-particles and the nature of wh-fronting, ms. MIT.

Caha, Pavel. 2007. The shape of paradigms, paper presented at GLOW XXX, Tromsø.

Chomsky, Noam. 1965. *Aspects of the Theory of Syntax*. Cambridge, MA: MIT Press.

Chomsky, Noam. 1993. A minimalist program for linguistic theory. In Kenneth Hale and Samuel Keyser, eds., *The view from Building 20*, 1–52, Cambridge, MA: MIT Press.

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

Chomsky, Noam. 2001. Derivation by phase. In Michael Kenstowicz, ed., *Ken Hale: A Life in Language*, 1–52, Cambridge, MA: MIT Press.

Chomsky, Noam. to appear. On phases, to appear in Carlos P. Otero et. al. eds., *Foundational Issues in Linguistic Theory* (MIT).

Cinque, Guglielmo. 1999. *The Syntax of Adverbs*. Oxford: Oxford University Press.

Cinque, Guglielmo and Kayne, Richard. 2005. *The Oxford Handbook of Comparative Syntax*. Oxford: Oxford University Press.

Collins, Chris. 2002. Eliminating labels. In Samuel David Epstein and T. Daniel Seely, eds., *Derivation and Explanation in the Minimalist Program*, 42–64, Oxford: Blackwells.

Dowty, David and Jacobson, Pauline. 1988. Agreement as a semantic phenomenon. In John Powers and Franscisca de Jong, eds., *Proceedings of the Fifth ESCOL*, 95–101, Ithaca: Cornell Linguistics Club.

Emonds, Joe. 2000. *Lexicon and Grammar: The English Syntacticon*. Berlin: Mouton de Gruyter.

Fitch, W. Tecumseh, Hauser, Marc D., and Chomsky, Noam. 2005. The evolution of the language faculty: Clarifications and implications. *Cognition* 97:179–210.

Frampton, John and Gutmann, Samuel. 1999. Cyclic computation, a computationally efficient minimalist syntax. *Syntax* 2:1–27.

Frampton, John and Gutmann, Samuel. 2002. How sentences grown in the mind, unpublished manuscript, Northwestern University.

Frampton, John and Gutmann, Samuel. 2004. Crash-proof syntax. In Samuel David Epstein and T. Daniel Seely, eds., *Derivation and Explanation in the Minimalist Program*, 90–105, Oxford: Blackwell.

Frank, Robert. 2002. *Phrase structure composition and syntactic dependencies*. MIT Press.

Gärtner, Hans-Martin. 2002. *Generalized Transformations and Beyond: Reflections on Minimalist Syntax*. Berlin: Akademie-Verlag.

Gazdar, Gerald, Klein, Ewan, Pullum, Geoff, and Sag, Ivan. 1985. *Generalized Phrase Structure Grammar*. Oxford: Blackwells.

Gazdar, Gerald, Pullum, Geoffrey K., Carpenter, Robert, Klein, Ewan, Hukari, Thomas E., and Levine, Robert D. 1988. Category structures. *Computational Linguistics* 14:1–19.

Grimshaw, Jane. 1979. Complement selection and the lexicon. *Linguistic Inquiry* 10:279–326.

Grimshaw, Jane. 1991. Extended projections, ms, Brandeis University.

Harbour, Daniel. 2007. *Morphosemantic Number*. Dordrecht: Kluwer.

Harley, Heidi and Ritter, Elizabeth. 2002. Person and number in pronouns: A feature-geometric analysis. *Language* 78:482–526.

Hauser, Marc D., Chomsky, Noam, and Fitch, W. Tecumseh. 2002. The faculty of language: What is it, who has it, and how did it evolve? *Science* 298:1569–1579.

Julien, Marit. 2002. *Syntactic Heads and Word Formation*. New York: Oxford University Press.

Kroch, Anthony and Joshi, Aravind. 1987. Analyzing extraposition in a tree adjoining grammar. In Geoffrey Huck and Almerindo Ojeda, eds., *Discontinuous constituency*, Syntax and semantics 20, 107–149, New York: Academic Press.

Link, G. 1983. The logical analysis of plurals and mass terms: A lattice-theoretical approach. In R. Bäuerle, C. Schwarze, and A. von Stechow, eds., *Meaning, Use and Interpretation of Language*, 302–323, Berlin: Walter de Gruyter.

Marantz, Alec. 1997. No escape from syntax: Don't try morphological analysis in the privacy of your own lexicon. In Alexis Dimitriadis, Laura Siegel, Clarissa Surek-Clark, and Alexander Williams, eds., *University of Pennsylvania Working Papers in Linguistics vol. 4.2*, 201–225, University of Pennsylvania.

Matushansky, Ora. 2006. Head movement in linguistic theory. *Linguistic Inquiry* 37:69–109.

Nilsen, ò ystein. 2003. Eliminating positions. Ph.D. thesis, LOT, Utrecht Institute of Linguistics OTS.

Noyer, Rolf. 1992. Features, positions and affixes in autonomous morphological structure. Ph.D. thesis, MIT.

Nunes, Jairo. 1995. The copy theory of movement and linearization of chains in the Minimalist Program. Ph.D. thesis, University of Maryland.

Pesetsky, David and Torrego, Esther. 2001. T to C movement: causes and consequences. In Michael Kenstowicz, ed., *Ken Hale: A Life in Language*, 355–426, Cambridge, MA: MIT Press.

Pesetsky, David and Torrego, Esther. 2004. Tense, Case and the nature of syntactic categories. In Jacqueline Gueron and Jacqueline Lecarme, eds., *The Syntax of Time*, Cambridge, MA: MIT Press.

Pesetsky, David and Torrego, Esther. 2007. The syntax of valuation and the interpretability of features. In Vida Samiian Simin Karimi and Wendy Wilkins, eds., *Phrasal and Clausal Architecture: Syntactic Derivation and Interpretation*, 262–294, Amsterdam: John Benjamins.

Retoré, Christian and Stabler, Edward. 2004. Generative grammars and resource logics. *Research on Language and Computation* 2:3–25.

Rizzi, Luigi. 1997. The fine structure of the left periphery. In Liliane Haegeman, ed., *Elements of Grammar: Handbook of Generative Syntax*, 281–337, Dordrecht: Kluwer.

Schlenker, Philippe. 2003. Indexicality, logophoricity, and plural pronouns. In Jacqueline Lecarme, ed., *Research in Afroasiatic Grammar II (Selected Papers from the Fifth Conference on Afroasiatic Languages, Paris, 2000)*, 409–428, Amsterdam: Benjamins.

Shieber, Stuart. 1986. *An Introduction to Unification-Based Approaches to Grammar*. Stanford: CSLI.

Stabler, Edward. 1998. Acquiring languages with movement. *Syntax* 2:72–97.

Stabler, Edward P. 1997. Derivational minimalism. In *LACL '96: Selected papers from the First International Conference on Logical Aspects of Computational Linguistics*, 68–95, London, UK: Springer-Verlag.

Starke, Michal. 2001. Move dissolves into merge. Ph.D. thesis, University of Geneva.

Stowell, Tim. 1983. Subjects across categories. *The Linguistic Review* 2:285–312.

Svenonius, Peter. 2002. Icelandic case and the structure of events. *Journal of Comparative Germanic Linguistics* 5:197–225.

Svenonius, Peter. 2003. Interpreting uninterpretable features. *Linguistic Analysis* 33:375–413.

Svenonius, Peter. 2004. On the edge. In David Adger, Cécile de Cat, and George Tsoulas, eds., *Peripheries: Syntactic Edges and their Effects*, 259–287, Dordrecht,

The Netherlands: Kluwer Academic Publishers.

Svenonius, Peter. 2007. Projections of p, lingBuzz/000484.

Řezáč, Milan. 2004. Elements of cyclic syntax: Agree and Merge. Ph.D. thesis, University of Toronto.