# CHAPTER 7

# Minimalist Grammars with Adjunction

Sections 7.1 to 7.8 of this chapter are based on a published paper, Fowlie (2014).

## 7.1 Introduction

Adjuncts are optional,[1] meaning the sentence is grammatical without them. For example, in (1-a), *red* is optional. They are transparent to selection in that the selector seems to select for the features of the head, not those of the intervening adjunct. For example, in (1-b), the gender of *boek* 'book' is neuter. The intervening adjective does not have gender agreement, so *het* selects *boek* for its gender, regardless of the intervening adjunct.

(1) a. The (red) rose **Optionality**

b. Het mooi-e boek
the.NEU beautiful-DEF book
'The beautiful book' (Dutch) **Transparency**

Many languages have a default order for adjuncts, with unmarked intonation and without special scopal meaning. For example, English has ordered adjectives.

(2) a. Wear the enormous ugly green hat

*Wear the hat that is enormous, ugly, and green*

[1] Arguments have been made for (rare) obligatory adjuncts (Grimshaw and Vikner, 1993), which I will address in section 7.7.3 below.

b. #Wear the ugly enormous green hat

*Of your enormous green hats, wear the ugly one.*

Adjuncts differ from arguments primarily in the direction of their dependencies. Arguments and their selectors are mutually dependent: it is the fact that the argument is needed by the selector that licenses the presence of the argument, and it is the presence of the argument that makes the selector complete. For example in the sentence *She slept*, the verb *slept* requires a sleeper and is not allowed unless it has one, and the DP *she* is allowed in the sentence because *slept* requires it. Adjuncts, on the other hand, require a phrase to adjoin to, but are not required by anything. For example, in *She slept soundly*, the adverb *soundly* requires a VP (or something along those lines) to adjoin to, but the VP *slept* does not require an adverb. It is for this reason that in a classic phrase structure grammar, the internal structure of XPs can be determined entirely by the requirements of the X head, but phrase structure rules (or some equivalent) are still required to describe the distribution of adjuncts. In an MG, the requirements of the head are modelled with the feature stack preceding the category feature; however, no mechanism equivalent to phrase structure rules for adjuncts exists in MGs.

To replace this missing mechanism, I propose a separate function that I dub *Adjoin*, along with a listing in the grammar of what categories can adjoin to what categories. This listing, together with the action of Adjoin that builds the right structure, is equivalent to the missing phrase structure rules. For example, if in a PSG we had the rule NP→AP NP, we could just as easily say *A's are among the adjuncts of Ns* if we also had an understanding that adjuncts always adjoin after the requirements of the head are fulfilled and if we knew what side of the NP the AP should appear on.

I argue that without such a separate mechanism we are forced to treat adjunction as selection, which makes wrong predictions.

On top of this selection/adjunction distinction, there is a second, more difficult issue: that of adjunct ordering. We have already seen that adjuncts tend to follow quite a strict order cross-linguistically, and that this order cannot always be explained by semantic factors. This chapter supposes that we do in fact need to account for at least some of these ordering facts in the syntax. The intention is to take a careful look at what kinds of mechanisms the grammar would have to include for this to be possible.

I propose a grammar that uses a mechanism of numerical indexing of hierarchy levels. I do not argue that this is necessarily the right model, but rather that any syntactic model of hierarchies requires something to do the work of the numerical indices.

## 7.2 Minimalist Grammars

This section provides a brief overview of minimalist grammars. For a more complete picture, please see chapter **??**.

I formulate my model as a variant of *Minimalist Grammars* (MGs), which are Stabler (1997)'s formalisation of Chomsky (1995)'s notion of feature-driven derivations using the functions Merge and Move. MGs are mildly context-sensitive, putting them in the right general class for human language grammars. They are also simple and intuitive to work with. Here I will briefly run over the definition of MGs.

I will give derived structures as strings as Keenan and Stabler (2003) grammar would generate them.[2]

[2]Keenan & Stabler's grammar also incorporates an additional element: lexical items are triples of string, features, and lexical status, which allows derivation of Spec-Head-Complement order. I will leave this out for simplicity, as it is not really relevant here, as our interest is in spec/adjunct placement, which will always be on the left. For convenience of English reading, I will give sentences in head-spec-complement order, but the formal definition I give here always puts the selected on the left and the selector on the right.

**Definition 7.2.1.** A ***Minimalist Grammar*** is a five-tuple $G = \langle \Sigma, \mathbf{sel}, \mathbf{lic}, Lex, M\rangle$. $\Sigma$ is the *alphabet*. $\mathbf{sel} \cup \mathbf{lic}$ are the *base features*. Let $F = \{\texttt{+f}, -\texttt{f}, \texttt{=X}, \texttt{X} | \texttt{f} \in \mathbf{lic}, \texttt{X} \in \mathbf{sel}\}$ be the *features*. $Lex \subseteq \Sigma \times F^*$, and $M$ is the set of operations **Merge** and **Move**. The language $L_G$ is the closure of $Lex$ under $M$. A set $C \subseteq F$ of designated features can be added; these are the types of complete sentences.

Minimalist Grammars are *feature-driven*, meaning features of lexical items determine which operations can occur and when. There are two finite sets of features, ***selectional*** features **sel** which drive the operation **Merge** and ***licensing*** features **lic** which drive **Move**. **Merge** puts two derived structures together; **Move** operates on the already built structure. Each feature has a positive and negative version. Positive **sel** and **lic** features are `=X` and `+f` respectively, and negatives are `X` and `-f`. Intuitively, negative **sel** features are the categories of lexical items. Merge and Move are defined over ***expressions***: sequences of pairs ⟨derived structure, feature stack⟩. The first pair can be thought of as the "main" structure being built; the remaining are waiting to move. Positive **sel** features are not unlike the "uninterpretable" category features of classical minimalism.

An MG essentially works as follows: **Merge** takes two expressions and combines them into one if the first structure displays `=X` and the second `X` for some $\texttt{X} \in \mathbf{sel}$. The `X` features are deleted, after which the second structure may still have features remaining, meaning the second structure is going to move. It is stored separately by the derivation until the matching positive licensing feature comes up later in the derivation, when the moving structure is combined again; this is **Move**. **Move** also carries the requirement that for each $\texttt{f} \in \mathbf{lic}$ there be at most one structure waiting to move. This is the ***shortest move constraint (SMC)***.[3]

[3]The SMC is based on economy arguments in the linguistic literature (Chomsky, 1995), but it is also crucial for a type of finiteness: the valid derivation trees of an MG form a regular tree language (Kobele et al., 2007). The number of possible movers must be finite for the automaton to be finite-state. The SMC could also be modified to allow up to a particular (finite) number of movers for each $\texttt{f} \in \mathbf{lic}$.

For example, we can put *saw* and *him* together to make *saw him* if the features of *saw* are `=D =D V` and the features of *him* are just `D` (Figure 7.1a).

If we Merge *saw* with *who*, we don't put them together since *who* is going to wh-move anyway. Instead we store it for later (Figure 7.1b).


(a) Merging a non-mover

(b) Merging a mover

Figure 7.1: Merge

Merge is defined formally as follows:

**Definition 7.2.2** (*Merge*)**.** For $\alpha, \beta$ sequences of features, $s, t$ derived structures, $\text{mvrs}_{s,t}$ expressions:[4]

$$\textbf{Merge}(s : \texttt{=X}\alpha :: \text{mvrs}_s, t : \texttt{X}\beta :: \text{mvrs}_t) = \begin{cases} ts : \alpha :: \text{mvrs}_s \cdot \text{mvrs}_t & \text{if } \beta = \epsilon \\ (s : \alpha) :: (t : \beta) :: \text{mvrs}_s \cdot \text{mvrs}_t & \text{if } \beta \neq \epsilon \end{cases}$$

The function Adjoin is based on Merge, so I will not go into detail on Move here. See Chapter **??** for a thorough introduction. Here I give just the definition.

**Definition 7.2.3** (*Move*)**.** For $\alpha, \beta, \gamma$ sequences of features, $s, t$ derived structures, mvrs an expression, suppose there is a unique $\langle t, \beta \rangle \in$ mvrs such that $\beta = \texttt{-f}\gamma$. Then:

$$\textbf{Move}(s : \texttt{+f}\alpha :: \text{mvrs}) = \begin{cases} ts : \alpha :: \text{mvrs} & \text{if } \gamma = \epsilon \\ s : \alpha :: t : \gamma :: \text{mvrs} - t : \beta) & \text{if } \gamma \neq \epsilon \end{cases}$$

In this chapter I will make use of *derivation trees*, which are trees describing the derivation. They may also be annotated: in addition to the name of function, I (redundantly) include for clarity the derived expressions in the form of

[4] :: adds an element to a list; $\cdot$ appends two lists.

strings and features. For example, figure 7.2 shows derivation trees (annotated and unannotated) of *the wolf* with feature D.


Figure 7.2: Annotated and unannotated derivation trees

## 7.3 Cartography

The phenomena this model is designed to account for are modifiers and other apparently optional projections such as those in (3). (For more data and discussion, please see chapter **??**). We will see, though, that it is more successful with adjectives and adverbs than functional projections. Section 7.12 improves on the adjunction model.

(3) a. The small ancient triangular green Irish pagan metal artifact was lost.

b. *The metal green small artifact was lost. **Adjectives**

c. Frankly, John probably once usually arrived early.

d. *Usually, John early frankly once arrived probably. **Adverbs**

e. [Il premio Nobel]$_{\text{top}}$, [a chi]$_{\text{wh}}$ lo daranno?
[the prize Nobel]$_{\text{top}}$, [to whom]$_{\text{wh}}$ it give.fut
The Nobel Prize, to whom will they give it? **Left periphery**

f. [$_{\text{DP}}$ zhe [$_{\text{NumP}}$ yi [$_{\text{ClP}}$ zhi [$_{\text{NP}}$ bi]]]
[$_{\text{DP}}$ this [$_{\text{NumP}}$ one [$_{\text{ClP}}$ CL [$_{\text{NP}}$ pen]]]
'this pen' **Functional projections**

These three phenomena all display *optionality, transparency to selection*, and *strict ordering*. By *transparency* I mean that despite the intervening modifiers, properties of the selected head are relevant to selection. For example, in a classifier language, the correct classifier selects the noun even if adjectives intervene.

(4) a. te' mexha
$\text{CL}_{\text{WOOD}}$ table
'the table' Q'anjob'al[5]

b. te' yalixh mexha
$\text{CL}_{\text{WOOD}}$ small table
'the small table'

c. *no' yalixh mexha
$\text{CL}_{\text{ANIMAL}}$ small table

The hypothesis that despite their optionality these projections are strictly ordered is part of *syntactic cartography* (Rizzi, 2004). Cinque (1999, 2010), in particular proposes a universal hierarchy of functional heads that select adverbs in their specifiers, yielding an order on both the heads and the adverbs. He proposes a parallel hierarchy of adjectives modifying nouns. These hierarchies are very deep. The adverbs and functional heads incorporate 30 heads and 30 adverbs.

Cinque argues that the surprising universality of adverb order calls for explanation. For example, Italian, English, Norwegian, Bosnian/Serbo-Croatian, Mandarin Chinese, and more show strong preferences for *frankly* to precede *unfortunately*. These arguments continue for a great deal more adverbs.[6]

(5) Italian

a. **Francamente** ho *purtroppo* una pessima opinione di voi.
**Frankly** have *unfortunately* a bad opinion of you
'Frankly I unfortunately have a very bad opinion of you.'

b. *Purtroppo ho francamente una pessima opinione di voi.
*Unfortuately* have **frankly** a bad opinion of you

(6) English

a. **Frankly**, I *unfortuately* have a very bad opinion of you

b. ?*Unfortunately* I **frankly** have a very bad opinion of you

[5]Data from Bervoets et al. (2011)
[6]Data from Cinque (1999)

(7) Norwegian

a. Per forlater [**rerlig talt**] *[heldigvis]* [nil] selskapet.
Peter leaves [**honestly spoken**] *[fortunately]* [now] the.party.
'Frankly, Peter is fortunately leaving the party now.'

b. *Per forlater *[heldigvis]* [**rerlig talt**] [nil] selskapet.
Peter leaves *[fortunately]* [**honestly spoken**] [now] the.party.

(8) Bosnian/Serbo-Croatian

a. **lskreno**, ja *naialost* imam jako lose misljenje o vama
**Frankly**, I *unfortunately* have very bad opinion of you.
Frankly, I unfortunately have a very bad opinion of you.'

b. **Naialost*, ja **iskreno** imam jako lose misljenje o varna.
*unfortunately* I **frankly** have very bad opinion of you.

(9) Mandarin Chinese

a. **laoshi-shuo** wo *buxing* dui tamen you pian-jian.
**Frankly**, I *unfortunately* to them have prejudice
'Honestly I unfortunately have prejudice against them.'

b. **buxing* wo **laoshi-shuo** dui tamen you pian-jian.
*unfortunately* I **Frankly** to them have prejudice

## 7.4 Desiderata

In addition to these three main properties, an account of adjuncts should ideally also account for the following: a lack of adjunction to multiple categories at once (Graf, 2014), selectability of adjunct categories, adjuncts of adjuncts, unordered adjuncts, so-called obligatory adjuncts, and adjunct islands. These phenomena are discussed in more detail in Chapter **??**.

Graf (2014) points out that there are no attested structures in which an adjunct adjoins to two phrases in the same structure. For example, *big* is a modifier of nouns such as *horse* and *ship*, yet we never see structures such as that in Figure 7.3.

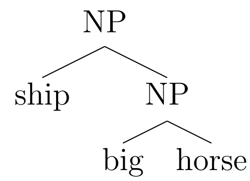



Figure 7.3: An adjective adjoining to two nouns

We also have the following data:

(10) Mary is **tall** — *tall* is selected by *is*

(11) The **surprisingly short** basketball player — *surprisingly* modifies *short*

(12) a. The alliance officer shot Kaeli **in the cargo hold** *with a gun.*
b. The alliance officer shot Kaeli *with a gun* **in the cargo hold**.English PP adjuncts are unordered

(13) a. He makes a **good** father — *good* is an adjunct but is not optional
b. *He makes a father
c. She worded the letter **carelessly**.
d. ...and Marc did so **carefully**. — *carefully* is an adjunct
e. *She worded the letter. — yet it is not optional

(14) a. He left [because she arrived]$_{\text{adjunct}}$.
b. *Who did he leave because ___ arrived? — (some) adjuncts are islands
c. He thinks [she arrived]$_{\text{object}}$.
d. Who does he think ___ arrived? — Embedded object CPs are not islands; islandhood is a property of adjuncts, not embedded clauses in general.

In sum, an account of adjuncts in minimalist grammars should ideally have the following properties:

**Desiderata**

1. **Optionality**: sentences should be grammatical with or without adjuncts
2. **Transparency to selection**: If a phrase P is normally selected by head Q, when P has adjuncts Q should still select P.
3. **One at a time**: adjuncts adjoin to only one phrase at a time.
4. **Order**: there should be a mechanism for forcing an order on adjuncts
5. **Selectability** (10)
   (a) **Efficiency:** All adjectives are possible arguments of the same predicates, so there should be as few distinct categories of adjectives as possible (ideally just one, say `A`).
6. **Adjuncts of adjuncts** (11)
   (a) **Efficiency:** Similarly to selection, there are large classes of adjuncts that adjoin to the same things; their members should not be listed separately. For example, most or all English adverbs are adjuncts of adjectives. There should be as few separate categories of adverbs as possible.
7. **Unordered** Some adjuncts are unordered with respect to each other. (12)
8. **Obligatory adjuncts** Occasionally, adjuncts are obligatory. (13)
9. **Islands** Some adjuncts cannot be extracted from (Huang, 1982). (14)

Figure 7.4: Desiderata

## 7.5 Previous Approaches to Adjunction

This section provides a brief overview of four approaches to adjunction. The first two are from a categorial grammar perspective and account for the optionality and, more or less, transparency to selection; however, they are designed to model unordered adjuncts. The next is an MG formalisation of the cartographic approach. Since the cartographic approach takes adjuncts to be regular selectors, unsurprisingly they account for order, but not easily for optionality or transparency to selection. Finally, I will briefly outline an earlier (published) version of my own solution.

### 7.5.1 Traditional MG solution

To account for the optionality and transparency, a common solution is for a modifier to combine with its modified phrase, and give the result the same category as the original phrase. Traditionally in MGs, an `X`-modifier has features `=XX`: it selects an `X` and the resulting structure has category feature `X`. Similarly, in categorial grammars, an X-modifier has category X/X or X\X. As such, the properties of traditional MG and CG models of adjunction are the same.[7]

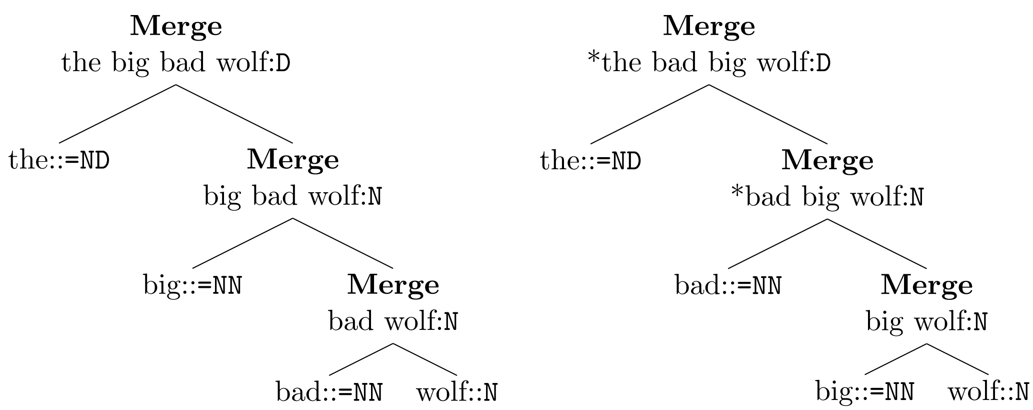



Figure 7.5: Traditional MG approach

This is the same as the traditional categorial grammar approach, where an N-modifier is given the slash category N/N (meaning it needs an N on the right to make an N) or N\N (meaning it needs an N on the left to make an N).

[7]This is not the only possible solution using the MG architecture, but rather the traditional solution. Section 7.5.3 gives a model within MGs that accounts for order.

An anonymous reviewer of Fowlie (2014) suggested a different solution, with a set of silent, meaningless heads that turn categories into selectors of their adjuncts, for example $\epsilon$::`=N =Adj =N`. Such a solution does much better on desiderata 4 and 5 than the one given here, but shares with the cartographic solution given in section 7.5.3 the problem of linguistic undesirability of silent, meaningless elements.

This system makes adjuncts truly optional, and allows for unordered adjuncts, but does not capture anything else. It cannot account for ordering. This is because the category of the new phrase is the same regardless of the modifier's place in the hierarchy. That is, the very thing that accounts for the optionality and the transparency of modifiers (that the category does not change) is what makes strict ordering impossible. Moreover, the modifier is not truly transparent to selection: the modifier in fact becomes the new head; it just happens to share a category with the original head. This can be seen in tree-generating grammars such as Stabler (1997) (figure 7.6).

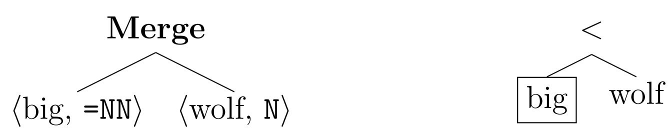



Figure 7.6: Derivation tree and derived *bare tree*. The < points to the head, *big*.

1. **Optionality**: ✓the original category is kept

2. **Transparency to selection**: **Sort of:** in Fig.7.5, *the* selects `N`, but the `N` it checks is the one introduced by *bad*, not the one on *wolf*. This makes *big* the head of the noun phrase *big wolf*, but it should be *wolf*, as shown in Fig. 7.6.

3. **One at a time**: ✗ If we allow adjuncts to have categories `=X=XX` rather than just `=XX`, we erroneously predict derivations like in Figure 7.3. There is no reason not to allow such features, as, intuitively, such an item would still be an X-adjunct (Graf, 2014).

4. **Order**: ✗The original category is kept so any adjunct may adjoin at any time.

5. **Selectability** Adjuncts need two versions, one for being adjuncts and the other for being selected. For example, *bad*::`=NN` cannot be selected by any-

thing until it has itself selected an `N`. We need a second version of *bad*, with, say, category `A`, so that it can be selected by such words as *is* (Fig. 7.7).

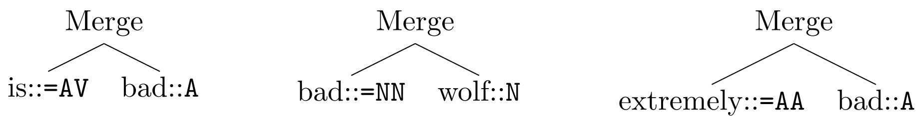



Figure 7.7: Selected adjective, modifier adjective, adjoined-to adjective

   (a) **Efficiency:** ✗ Adjuncts have two versions.

6. **Adjuncts of adjuncts:** ✗ Since adjunction is selection in this model, we have the same problem, but with the same solution: the feature for selection is also the feature for being adjoined to (Fig. 7.7).

   (a) **Efficiency:** The homophony for selection covers adjunction too.

7. **Unordered:** ✓ All adjuncts are unordered in this model

8. **Obligatory adjuncts:** ✗ There is no way to distinguish between an phrase with an adjunct and one without.

9. **Islands:** ✗ There is no way to distinguish between an phrase with an adjunct and one without.

### 7.5.2 Frey & Gärtner

Frey and Gärtner (2002) propose an improved version of the categorial grammar approach, one which keeps the modified element the head, giving true transparency to selection. They do this by asymmetric feature checking. To the basic MG formalism a third polarity is added for **sel**, ≈`X`. This polarity drives the added function Adjoin. Adjoin behaves just like Merge except that instead of cancelling both ≈`X` and `X`, it cancels only ≈`X`, leaving the original `X` intact. This allows the phrase to be selected or adjoined to again by anything that selects or adjoins to `X`.

This model accounts for optionality and true transparency, but it is not designed to capture ordered adjuncts. Also, since adjuncts don't have categories of their own (just ≈`X`), it is not clear how best to model selection of and adjunction to adjuncts.

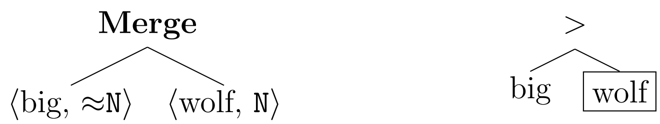



Figure 7.8: Frey & Gärtner: derivation tree and derived *bare tree*. The > points to the head, *wolf*.

1. **Optionality**: ✓the original category is kept

2. **Transparency to selection**: ✓For example, in the example in Figure 7.9, *the* selects `N`, which is introduced by *wolf*.

3. **One at a time:** ✓The adjunct's one and only ≈`X` is cancelled.

4. **Order**: ✗ The original category is kept so any adjunct may adjoin at any time (Fig. 7.9).

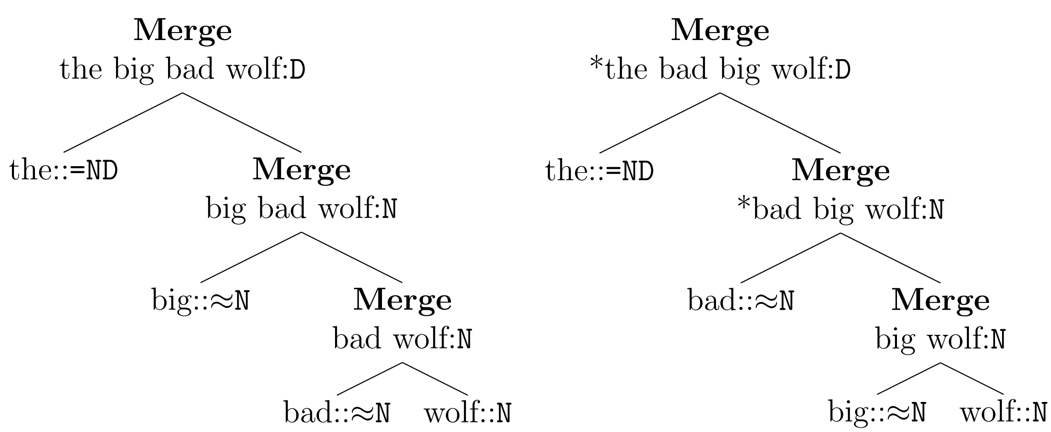



Figure 7.9: F & G derivations of *the big bad wolf* and **the bad big wolf*

5. **Selectability** ✓Frey & Gärtner allow (in one version at least) ≈`X` to be selected, yielding `=≈X` features for items that select adjuncts of `X`.

Merge

is::$\mathtt{=\approx NV}$ bad::$\mathtt{\approx N}$

(a) **Efficiency: ✓**No homophony required

6. **Adjuncts of adjuncts** Not clear: F& G do not give an account of adjuncts of adjuncts, but their account could presumably be analogously extended to include a limited number of $\approx$ symbols, say up to 3 or 4.

Adjoin

extremely::$\mathtt{\approx\approx N}$ bad::$\mathtt{\approx N}$

(a) **Efficiency:** ✓

7. **Unordered** ✓All adjuncts are unordered in this model

8. **Obligatory adjuncts ✗** There is no way to distinguish between a phrase with an adjunct and one without.

9. **Islands** ✓Since Adjoin is a separate operation, it can be defined so that there is no case for adjuncts with movers.

The models seen so far are summarised in Table 7.1.

| **Model** | Trad | F&G |
|---|---|---|
| **Section** | 7.5.1 | 7.5.2 |
| Optional | ✓ | ✓ |
| Efficiency (opt) | ✓ | ✓ |
| Transparent | ? | ✗ |
| 1-at-a-time | ✗ | ✓ |
| Order | ✗ | ✗ |
| Selectability | ✓ | ✓ |
| Efficiency (Sel) | ✗ | ✓ |
| Adj of adj | ✓ | ? |
| Efficiency (Adj of adj) | ✗ | ✓? |
| Unordered | ✓ | ✓ |
| Oblig | ✗ | ✗ |
| Island | ✗ | ✓ |

Table 7.1: Summary of models re: desiderata

### 7.5.3 Selectional approaches

A third approach is to treat adjuncts just like any other selector. This is the approach taken by syntactic cartography. Such an approach accounts straightforwardly for order, but not for optionality or transparency; this is unsurprising since the phenomena I am modelling share only ordering restrictions with ordinary selection.

The idea is to take the full hierarchy of modifiers and functional heads, give each their own category names, and have each select the one below it; for example, *big* selects *bad* but not vice versa, and *bad* selects *wolf*. (See Figure 7.10.) However, here we are left with the question of what to do when *bad* is not present, and the phrase is just *the big wolf*. *big* does not select *wolf* (Fig. 7.10). I will give two solutions, one in which the full structure is always present and one in which it is not.

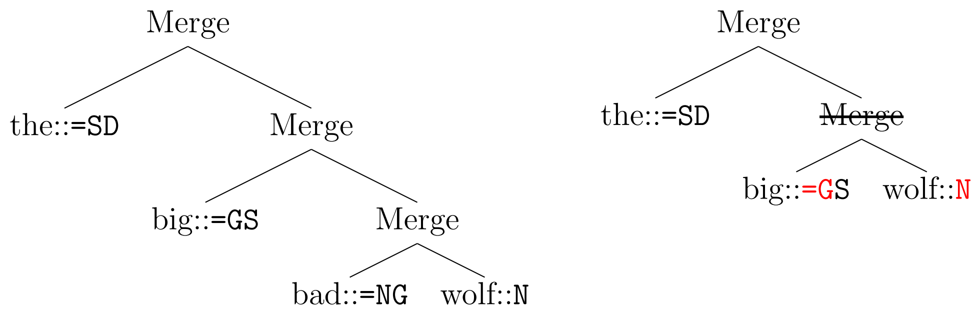



Figure 7.10: Selectional approach

#### 7.5.3.1 Silent, meaningless heads

We give each modifier and functional head a silent, meaningless version that serves only to tie the higher modifier to the lower, like syntactic glue. For example, we add to the lexicon a silent, meaningless "size" modifier that goes where *big* and *small* and other LIs of category `S` go.

$\langle$ the, `=S D` $\rangle$ $\quad$ $\langle$ $\epsilon$, `=S D` $\rangle$ $\quad$ $\langle$ wolf, `N` $\rangle$

$\langle$ big, `=G S` $\rangle$ $\quad$ $\langle$ $\epsilon$, `=G S` $\rangle$ $\quad$ $\langle$ bad, `=N G` $\rangle$ $\quad$ $\langle$ $\epsilon$, `=N G` $\rangle$

This solution doubles substantial portions of the lexicon. Doubling is not computationally significant, but it does indicate a missing generalisation: somehow, it just happens that each of these modifiers has a silent, meaningless doppelganger. Relatedly, the ordering facts are epiphenomenal. There is no universal principle predicting the fairly robust cross-linguistic regularity. Moreover, normally when something silent is in the derivation, we want to say it is contributing something semantically. Here these morphemes are nothing more than a trick to hold the syntax together.

In favour of these heads lacking in both phonetic and semantic content is the efficiency gains silent elements can give a grammar. For example, to remove a silent element from a context-free grammar and still generate the same language, we find every category that could end up with a silent pronunciation (*nullable* category), and then for every rule in which it occurs, a new rule that lacks that nullable category is added to the grammar. For example if we have a rule A$\rightarrow\epsilon$ and another C$\rightarrow$A B, then we add a new rule C$\rightarrow$B. Thus silent elements can in fact make grammars *more* efficient.

However, the silent elements that can increase efficiency of grammars need not necessarily be meaningless. These silent heads in human language could just as easily be universally meaningful. For instance, suppose proper names are always `N`s, and in Italian they are selected by a `D` *il* or *la* to make a DP, while in English are selected by a silent `D`, also forming a DP. The silent English `D` could be argued to carry the same meaning as the overt Italian one.

(15) a. il Paolo
the$_{\text{M}}$ Paul
'Paul'

b. la Paola
the$_{\text{F}}$ Paula
'Paula'

Moreover, items that have nothing interpretable at either interface run counter to minimalist principles.

1. **Optionality**: ✓Choose the right version of an LI. Note this is **inefficient**.

2. **Transparency to selection**: ✗ Selection is of the adjunct, not the head. For example, in the lexicon above, *the* selects the (possibly empty) adjunct with features `=G S`, not the noun.

3. **One at a time**: ✗Since adjuncts are not distinct from other items, and we want some lexical items to be able to select two things (such as $v$ selecting a VP complement and a DP specifier), we could find adjuncts that select two elements as well; this would be indistinguishable from adjoining to two elements.

4. **Order**: ✓This is Merge, so order is determined by the particular lexical items' feature stacks.[8]

5. **Selectability** ✓This is ordinary Merge, so selection proceeds as usual.

   (a) **Efficiency:** ✗We need to add selectable versions of adjuncts; eg big::`=GS` needs to be paired with big::`S`. Selectors of adjectives and adverbs also need multiple versions, for example be::`=SV`, be::`=GV`, etc.

6. **Adjuncts of adjuncts** ✓Adjuncts of adjuncts are simply selectors of adjuncts.

   (a) **Efficiency:** ✗Adjuncts need to select all the adjuncts they adjoin to. eg: very::`=S Int`, very::`=G Int` etc.

7. **Unordered** ✓ Here we use the traditional features, `=XX`, but we might need a version for each category of, say, adjectives, depending on where we think unordered adjuncts adjoin.

8. **Obligatory adjuncts:** ✗ The same thing that makes adjuncts optional makes it impossible to require them.

9. **Islands:** ✗ Not without additional constraints. Again, see Graf (2013a) for constraints that may work here.

If we are willing to accept a model with large amounts of structure that contribute nothing to either interface, we can stop here. This model does succeed in capturing the ordering facts, and the other unmet desiderata are arguably not fatal. If not, we must look further.

[8] Note that there is a missing generalisation here: nothing forces a particular order cross-linguistically, but rather the order is an epiphenomenon of the particular lexical items. This shortcoming is an example of a general problem for MGs: nothing forces, say, verbs to select DP.

#### 7.5.3.2 Massive homophony

A second selectional approach is for each morpheme in the hierarchy to have versions that select each level below it. For example, *the* has a version which selects `N` directly, one that selects "goodness" adjectives like *bad*, one that selects "size" adjectives like *big*, and indeed one for each of the ten or so levels of adjectives.

$\langle$the, `=SD`$\rangle$ $\langle$the, `=GD`$\rangle$ $\langle$the, `=SD`$\rangle$ $\langle$the, `=ND`$\rangle$

$\langle$big, `=GS`$\rangle$ $\langle$big, `=NatS`$\rangle$ $\langle$big, `=NS`$\rangle$

$\langle$bad, `=NatG`$\rangle$ $\langle$bad, `=NG`$\rangle$

$\langle$Canadian, `=NNat`$\rangle$

$\langle$wolf, `N`$\rangle$

This second solution lacks the strangeness of silent, meaningless elements, but computationally it is far worse. To compute this we simply use Gauss's formula for adding sequences of numbers, since an LI at level $i$ in a hierarchy has $i$ versions. For example, in the model above, *the* is at level 4 (counting from 0), and there are 4 versions of *the.* For a lexicon Lex without these duplicated heads, and a language with $k$ hierarchies of depths $l_i$ for each $1 \leq i \leq k$, adding the duplicated heads increases the size of the lexicon. The increase is bounded below by a polynomial function of the depths of the hierarchies as follows:[9]

$$|\mathrm{Lex}'| \geq \sum_{i=1}^{k} 1/2(l_i^2 + l_i) + |\mathrm{Lex}|$$

This approach has the similar properties as the silent meaningless heads approach in terms of our desiderata. Table 7.2 summarises the models seen so far.

[9] I say "bounded below" because this formula calculates the increase to the lexicon assuming there is exactly one LI at each level in the hierarchy. If there are more, each LI at level $i$ of a hierarchy has $i$ versions as well.

| **Model** | Trad | F&G | Syn. glue | Homoph |
|---|---|---|---|---|
| **Section** | 7.5.1 | 7.5.2 | 7.5.3.1 | 7.5.3.2 |
| Optional | ✓ | ✓ | ✓ | ✓ |
| Efficency (opt) | ✓ | ✓ | ✗ | ✗✗ |
| Transparent | ? | ✗ | ✗ | ✗ |
| 1-at-a-time | ✗ | ✓ | ✗ | ✗ |
| Order | ✗ | ✗ | ✓ | ✓ |
| Selectability | ✓ | ✓ | ✓ | ✓ |
| Efficiency (Sel) | ✗ | ✓ | ✗ | ✗ |
| Adj of adj | ✓ | ? | ✓ | ✓ |
| Eff (Adj of adj) | ✗ | ✓? | ✗ | ✗ |
| Unordered | ✓ | ✓ | ✓ | ✓ |
| Oblig | ✗ | ✗ | ✗ | ✗ |
| Island | ✗ | ✓ | ✓ | ✓ |

Table 7.2: Summary of models re: desiderata

### 7.5.4 Fowlie (2013)

In an earlier paper I proposed a way to handle adjuncts that combined some of the benefits of Frey & Gärtner's approach with those of Cinque. The grammar I will propose in this paper is an improvement on my previous work, which I will briefly outline here for comparison. It is also the version Graf (2014) analyses (see Section 7.11), though the results are the same for the new model.

Like Frey & Gärtner, I add to MGs an operation **Adjoin**. Unlike their proposal, Fowlie (2013) does not add a polarity but rather a function **Ad** that maps categories to their adjuncts. It also changes categories from single category names to pairings of the category of the head with the category of the last adjunct adjoined to the phrase: `[X,A]`. For example, in 7.11, the category of *wolf* is `[N,N]`, but when the category `G` adjunct *bad* adjoins to it, the resulting phrase is of category `[N,G]`: a noun phrase whose last adjunct was of category `G`.

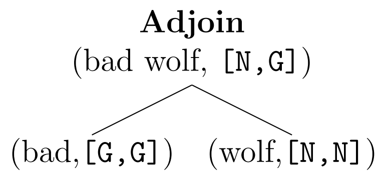



Figure 7.11: Valid derivation of *bad wolf*

Two additional structures are added to the grammar: a partial order on the category features and a partial function **Ad** from category features to sets of their adjuncts. The validity of an application of **Adjoin** is determined by two factors: whether the adjunct is an adjunct of the phrase it is adjoining to, and whether the category of the adjunct is higher in the partial order than the last adjunct adjoined, which is encoded as the second element of the pair of category features. For example, suppose G,S$\in$**Ad**(N), and suppose S$\geq$G$\geq$N. Then the first derivation in 7.12 is valid because G is an adjunct of N and G$\geq$N. The second derivation is bad because the second adjunct is too low (G>S).

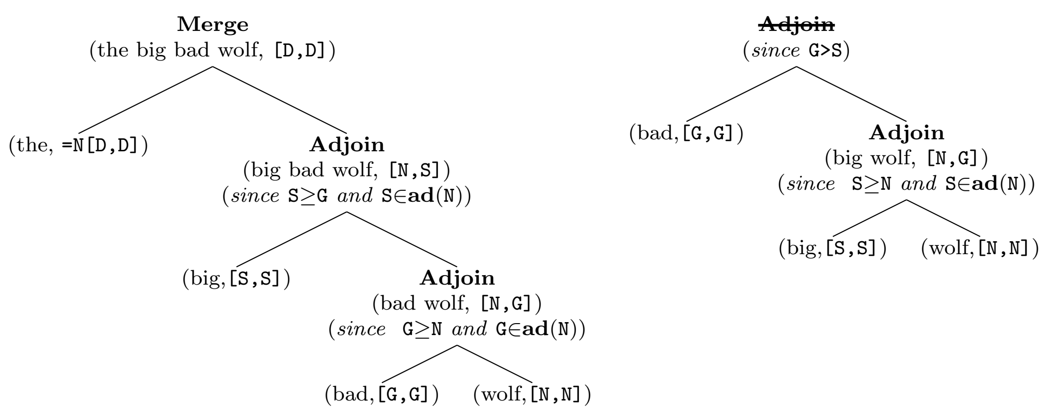



Figure 7.12: Valid derivation of *the big bad wolf* and attempted derivation of **the bad big wolf*

1. **Optionality**: ✓the original category is kept as the first element of the category pair

2. **Transparency to selection**: ✓the original category is kept

3. **One at a time**: ✓The adjunct's one and only category feature is cancelled by Adjoin, so it can't adjoin to anything else

4. **Order**: ✓ The second element of the category is the last adjunct adjoined. There is an order on the adjunct categories and the Adjoin rule requires that the adjunct be higher in the order than that second element of the category pair.

5. **Selectability** ✓Adjuncts have regular categories

   Merge

   is bad:`[V,V]`

   is::`=G[V,V]`    bad::`[G,G]`

   (a) **Efficiency:** ✗ Each adjunct has its own category so an LI that selects all, say, adjectives, needs a version for each adjective category, for example is::`=S V`, is::`=G V`, is::`=Nat V`, is::`=M V`, ...

6. **Adjuncts of adjuncts** ✓Adjunct categories are ordinary categories so they can have adjuncts too.

   Adjoin

   extremely::`[Ex,Ex]`    bad::`G`

   (a) **Efficiency:** ✗ Each adjunct has its own category so the adjunct sets will be inefficiently listed, missing the generalisation that, say, all adverbs modify all adjectives. e.g. $\mathbf{Ad}(\mathtt{S})=\mathbf{Ad}(\mathtt{G})=\mathbf{Ad}(\mathtt{Nat})=\mathbf{Ad}(\mathtt{M})=\ldots$ $=\{\mathtt{Adv}_1,\mathtt{Adv}_2,\mathtt{Adv}_3\ldots,\mathtt{Int(ensifier)}\}$.

7. **Unordered** ✓An extension of this model handles unordered adjuncts. It will be recapitulated for the new proposal in section 7.7.2 below.

8. **Obligatory adjuncts: Maybe.** Until a modified phrase is selected, it is distinguishable from an unmodified phrase. For example, an unmodified noun has category `[N,N]` while a modified noun has category `[N,A]` for some $\mathtt{A}\neq\mathtt{N}$. This might be exploitable for a special case of Merge. This question will be raised again in the new proposal, and the shape of any solution would be similar (Section 7.7.3).

9. **Islands** ✓Since Adjoin is a separate operation, it can be defined so that there is no case for adjuncts with movers.

Where this approach falls short is in adjunction to and selection of categories that are usually adjuncts. While it is possible, a generalisation is missed. The problem is that there is no unifying feature for adjectives or adverbs, since the distinctions are accounted for by giving them different categories. To define adjuncts of Adjectives, we have to give **Ad** as assigning the same adjunct set to all adjectives:

$\mathbf{Ad}(\mathtt{S})=\mathbf{Ad}(\mathtt{G})=\mathbf{Ad}(\mathtt{Nat})=\mathbf{Ad}(\mathtt{M})=...=\{\mathtt{Adv}_1,\mathtt{Adv}_2,\mathtt{Adv}_3\ldots,\mathtt{Int(ensifier)}\}$.

Similarly, selection of adjectives must be specified for each adjective. For example, *be* selects all adjectives, as in:

(16) She is silly/Canadian/brown-haired/tall/ugly...

As it stands, *be* must be cross-classified across adjective categories, multiplying the lexicon and missing a generalistion:

(17) $\langle$is, `=S V`$\rangle$, $\langle$is, `=G V`$\rangle$, $\langle$is, `=Nat V`$\rangle$, $\langle$is, `=M V`$\rangle$, ...

The proposal in this chapter fixes this shortcoming. Table 7.3 summarises the models seen so far.

| **Model** | Trad | F&G | Syn. glue | Homoph | Fowlie13 |
|---:|---|---|---|---|---|
| **Section** | 7.5.1 | 7.5.2 | 7.5.3.1 | 7.5.3.2 | 7.5.4 |
| Optional | ✓ | ✓ | ✓ | ✓ | ✓ |
| Eff (opt) | ✓ | ✓ | ✗ | ✗✗ | ✓ |
| Transparent | ? | ✗ | ✗ | ✗ | ✓ |
| 1-at-a-time | ✗ | ✓ | ✗ | ✗ | ✓ |
| Order | ✗ | ✗ | ✓ | ✓ | ✓ |
| Select | ✓ | ✓ | ✓ | ✓ | ✓ |
| Eff (Sel) | ✗ | ✓ | ✗ | ✗ | ✗ |
| Adj of adj | ✓ | ? | ✓ | ✓ | ✓ |
| Eff (Adj) | ✗ | ✓? | ✗ | ✗ | ✗ |
| Unordered | ✓ | ✓ | ✓ | ✓ | ✓ |
| Oblig | ✗ | ✗ | ✗ | ✗ | ? |
| Island | ✗ | ✓ | ✓ | ✓ | ✓ |

Table 7.3: Summary of models re: desiderata

## 7.6 Proposal 1: Minimalist Grammars with Adjunction

I propose a solution, which I will call Minimalist Grammars with Adjunction (MGAs),[10] which accounts for ordering by indexing phrases according to the hierarchy level of the last adjunct adjoined to them.

A given adjunct phrase $P$ needs four pieces of information: $P$'s category, what $P$ is an adjunct of, what level adjunct $P$ is, and what level the last adjunct that adjoin to $P$ was. We need to know what a category is an adjunct of because that will determine whether, say, an adjective can adjoin to a noun phrase. I include in the grammar a set of adjuncts for each category. The hierarchy level of the adjunct (encoded as a number) is needed for when it acts as an adjunct. If the phrase it is adjoining to already has a adjunct, we need to check that the new adjunct is higher (or at the same level) in the hierarchy. For this purpose, every phrase carries with it an additional number, indexing the level of its last adjunct. The two numbers are kept separate so that adjuncts can have adjuncts,

[10] My earlier paper Fowlie (2013) used this name as well; this model is designed to improve on it.

as in *bright blue*. *Bright blue* has an adjunct *bright*, which may affect what further adjuncts can adjoin to it, but which does not affect what the phrase *bright blue* can adjoin to.

To index hierarchy levels I use natural numbers and the usual order $\leq$ on $\mathbb{N}$.[11] Any index set would do, and in fact I claim these numbers correspond to semantic classes of adjuncts such as "Size" and "Goodness". This reflects the similarity of adjunct ordering across languages; without semantic classes, the universality of Cinque's hierarchies would be accidental. For example, level 6 categories might be the Size class, and level 4 the Goodness class. This hierarchy level−semantic class correspondence is universal across languages, making, for example, *good* and *bad* belong to level 6 in English, and *goed* 'good' and *slecht* 'bad' belong to level 6 in Dutch.

To track hierarchy level, each category feature is expanded into a triple consisting of the category feature, the level of the hierarchy of adjuncts the head is at, and the level of hierarchy the whole phrase is at. These numbers are lexically specified; for example *bad*::`[A,4,0]` would be in the lexicon.

By splitting the category into its category and its level as adjunct, we can allow all, say, adjectives, to have the same category. This extends the efficiency gains in Fowlie (2013) to selection of adjuncts and adjuncts of adjuncts.

When adjunct `[Y,n,m]` adjoins to something of category `[X,i,j]`, the resulting phrase is of category `[X,i,n]`, $\mathtt{i}, \mathtt{j}, \mathtt{n}, \mathtt{m} \in \mathbb{N}$. The second number is what tracks the level of the hierarchy the phrase is at; it is the only thing that can change.

[11] $\mathbb{N}$ is simply acting as an index set, and the maximal depth of hierarchies in a language bounds the actual index set for the grammar.


Figure 7.13: Adjoin. The category feature of the new phrase is the first two elements of the adjoined-to phrase followed by the second element of the adjunct

### 7.6.1 Example

Before I give the full formal definition I will present an example. Suppose we have a grammar in which the adjunct sets are defined as follows:

Ad(`N`)={`Adj, P, C`}, Ad(`Adj`)={`Adv, Int`}, Ad(`Adv`)={`Int`}, Ad(`V`)={`Adv,T`}

We can derive *Apparently, John very often sang* as in figure 7.14. *very* adjoins to *often* since *often* is at level 0 and *very* is at level 3, and $3 \geq 0$. The whole phrase adjoins to *sang* since it's at level 18 and *sang* is at 0. `T` Merges to the VP, yielding a phrase at level 25. *Apparently* is at level 26, so it can adjoin.

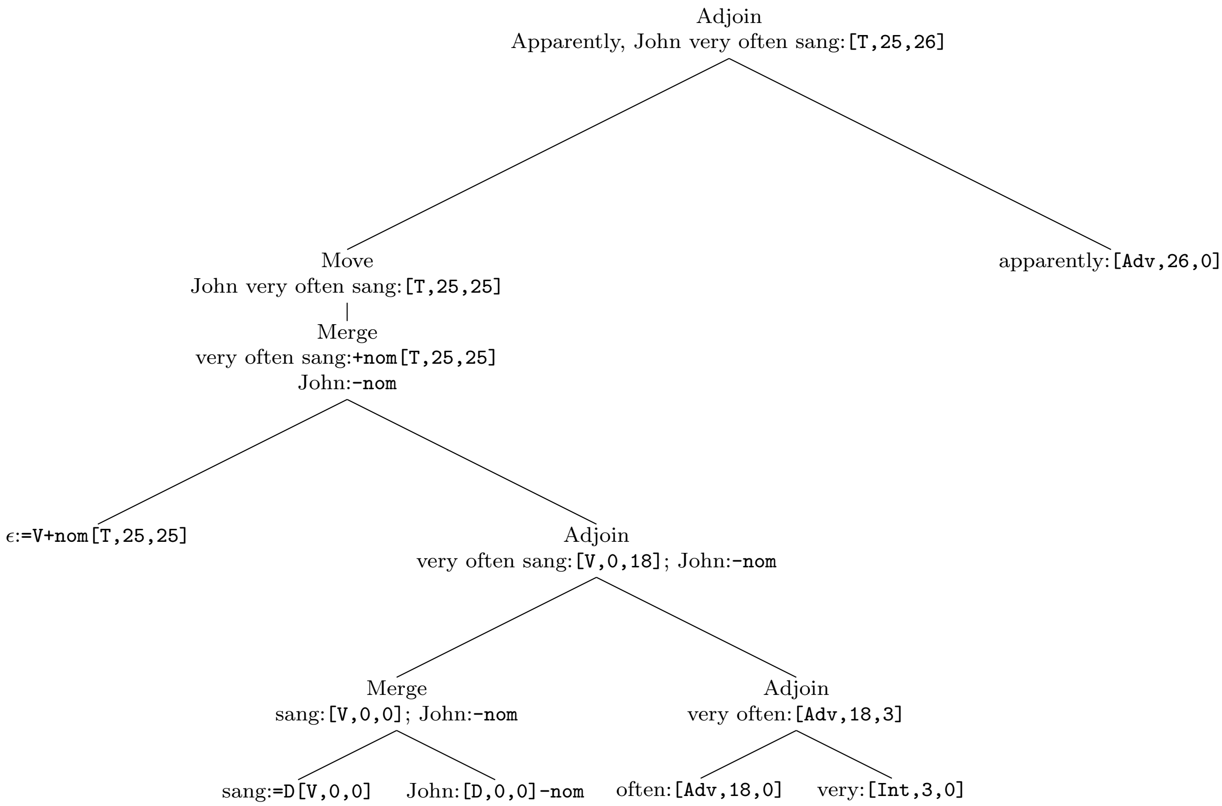



Figure 7.14: Adjunct of adjunct; functional head merge; adjunction after functional head merge

To get order, we require that the first number of the adjunct be at least as high as the second number of the adjoined-to phrase. For example, in Figure 7.15, the derivation of *the big bad wolf* works because Adj$\in$**Ad**(N), and $6 > 4 > 0$. The derivation of **the bad big wolf* fails because the category of *big wolf* is [N,0,6]. *bad*::[Adj,4,0] can't adjoin to it because *bad* is a level-4 adjunct, but *big wolf* is already at level 6, and $4 < 6$.

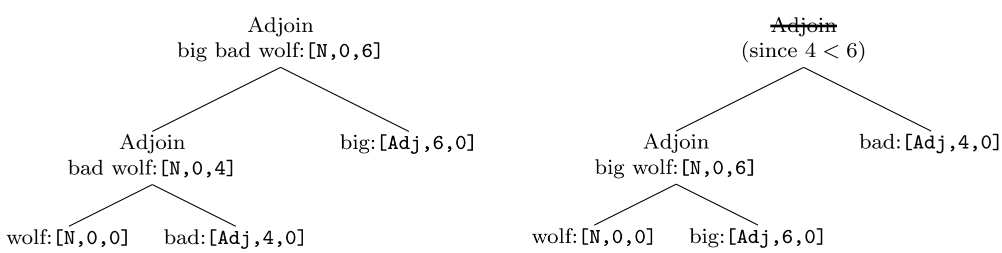



Figure 7.15: Adjunct ordering: valid and invalid derivations

### 7.6.2 Definition

**Merge** must be trivially redefined for categories as triples. **Merge** only cares about category, so it looks to match the positive selectional feature with the first element of the triple. (**Move** is unchanged.)

**Definition 7.6.1** (Merge)**.** For $\alpha, \beta \in F^*$; $s, t$ strings, $mvrs_s, mvrs_t \in (\Sigma^* \times F^*)^*$:

$$\mathbf{Merge}(\langle s, \texttt{=X}\alpha\rangle :: \mathrm{mvrs}_s, \langle t, [\texttt{X}, \texttt{i}, \texttt{j}]\beta\rangle :: \mathrm{mvrs}_t) = \begin{cases} \langle st, \alpha\rangle :: \mathrm{mvrs}_s \cdot \mathrm{mvrs}_t & \text{if } \beta = \epsilon \\ \langle s, \alpha\rangle :: \langle t, \beta\rangle :: \mathrm{mvrs}_s \cdot \mathrm{mvrs}_t & \text{if } \beta \neq \epsilon \end{cases}$$

**Adjoin** applies when the category of the adjunct is an adjunct of the category it is adjoining to, and if the adjunct is a $k$-level adjunct then the level of the phrase it is adjoining to is no higher than $k$. **Move** works as expected: if the adjunct has negative licensing features left after it has had its category feature checked by **Adjoin**, it is added to the list of movers.

**Definition 7.6.2** (Adjoin)**.** Let $s, t \in \Sigma$ be strings, $\texttt{Y},\texttt{X} \in \mathbf{sel}$ be categories, $i, j, n, m \in \mathbb{N}$, $mvrs \in (\Sigma^* \times F^*)^*$ be a mover list, and $\alpha, \beta \in F^*$.

$$\mathbf{Adjoin}(\langle s, [\texttt{X}, \texttt{i}, \texttt{j}]\alpha :: mvrs\rangle, \langle t, [\texttt{Y}, \texttt{n}, \texttt{m}]\beta\rangle)$$
$$= \begin{cases} \langle ts, [\texttt{X}, \texttt{i}, \texttt{n}]\alpha\rangle :: mvrs & \text{if } \texttt{n} \geq \texttt{j} \;\&\; \texttt{Y} \in \mathbf{Ad}(\texttt{X}) \;\&\; \beta = \epsilon \\ \langle s, [\texttt{X}, \texttt{i}, \texttt{n}]\alpha\rangle :: \langle t, \beta\rangle :: mvrs & \text{if } \texttt{n} \geq \texttt{j} \;\&\; \texttt{Y} \in \mathbf{Ad}(\texttt{X}) \;\&\; \beta \neq \epsilon \end{cases}$$

Notice that for Merge, there may be a mover list with both arguments ($movers_s$ and $movers_t$). Island constraints for adjoin are implemented by simply leaving out the mover list that would come with the adjunct. Adjoin is not defined when the adjunct has a mover.[12] This is not necessarily as stipulative as it sounds: Graf

[12] This is possible only because Adjoin and Merge are separate operations, as they are in Frey and Gärtner (2002). A close look at the definitions of Merge and Adjoin reveals that there is nothing formally stopping Adjoin from being a case of Merge, one defined when both phrases display a category feature. I have chosen to keep them as separate operations so that Adjoin may have different properties from Merge, such as island effects, and to maintain a certain type of locality for Merge, discussed in chapter **??**.

(2013a) puts forth that for adjuncts to be truly optional they cannot have movers, or else the derivation tree without the adjunct would have an unchecked positive licensing feature. My definition of Adjoin is simply a way of conforming to this constraint.

**Definition 7.6.3** (MGA)**.** A ***Minimalist Grammar with Adjunction*** is a six-tuple

$G = \langle \Sigma, \mathbf{sel}, \mathbf{lic}, \mathbf{Ad}, Lex, M \rangle$. $\Sigma$ is the *alphabet*. $\mathbf{sel} \cup \mathbf{lic}$ are the *base features*. Let $F = \{+\mathtt{f}, -\mathtt{f}, \mathtt{=X}, [\mathtt{X}, \mathtt{n}, \mathtt{m}] | \mathtt{f} \in \mathbf{lic}; \mathtt{X}, \mathtt{Y} \in \mathbf{sel}; \mathtt{m}, \mathtt{n} \in \mathbb{N}\}$. $\mathbf{Ad} : \mathbf{sel} \to \mathcal{P}(\mathbf{sel})$ maps categories to their adjuncts. $Lex \subseteq_{\text{fin}} \Sigma \times F^*$, and $M$ is the set of operations **Merge, Move,** and **Adjoin**. The language $L_G$ is the closure of $Lex$ under $M$. A set $C \subseteq \mathbf{sel}$ of designated features can be added; $\{[\mathtt{c}, \mathtt{i}, \mathtt{j}] | \mathtt{c} \in C; i, j \in \mathbb{N}\}$ are the types of complete sentences.

### 7.6.3 Adverbs and Functional Heads

Contra Cinque (1999), I model adverbs as separate from functional heads. Adverbs and adjectives differ from functional heads in two ways. First, they are not themselves adjoined to, while adverbs and adjectives are (*very blue*). Second, functional heads are sometimes required and sometimes optional. For example, English requires `T`, but not, perhaps, $\mathtt{Mod}_{\text{epistemic}}$ in every sentence. To model this, I give adjectives and adverbs category triples with their second number set to 0. This allows adjuncts to adjoin to them, starting at the bottom of that hierarchy. Functional heads, on the other hand, will start with their second number equal to their first number. This means that when they Merge, the resulting phrase is at the right level in the hierarchy, preventing low adjuncts from adjoining after the merger of a high functional head.[13]

[13]There is nothing in this formalism that prevents adjunction to a functional head. If the function **Ad** assigns adjuncts to a functional head, then it has adjuncts. They just behave a little oddly: e.g. `[F,3,3]` requires adjuncts above level 3.

For example, in Figure 7.14, *very* adjoins to *often*, which is possible since the second number of *often* is 0. Later, functional head T Merges to the VP. Its second number is 25. This is important because we want to say that *apparently* can only adjoin here because its first number is 26, which is higher than 25. A low adverb such as *again*::`[Adv,3,0]` cannot adjoin to T.

Cinque's adverbs and functional heads, on the other hand, are connected very directly: functional heads optionally select adjuncts in their specifiers. In MGAs, since all adverbs can have the same category, adverbs as specifiers are not possible if we want to preserve the notion that adverbs all have the category *adverb*. However, this is not a bad thing: Cinque observes that a language, or even a sentence in a language, tends to present only one of the functional head and the adverb overtly. In other words, he observes that generally either the functional head or the adverb is (at least phonetically) null. Arguably, since the semantics of the heads and the adverbs match, the phonetically null one could also be semantically null. If something is both phonetically and semantically null, it would be more parsimonious if we could say it was not there at all.

Under the MGA approach, a functional head and adverb that Cinque pairs would be at the same place in the hierarchy, but neither would select the other. For example, we might have (given as triples of *string::meaning::features* since the functional heads in English are phonetically null): briefly::BRIEFLY::`[Adv,10,0]` and $\epsilon$::ASP-DURATIVE::`[F,10,0]`, and usually::USUALLY::`[Adv,21,0]` and $\epsilon$::ASP-HABITUAL::`[F,21,0]`. If we tried to model the first pair instead as a head-specifier, we would have $\epsilon$::ASP-DURATIVE::`=Adv [F,10,0]`. Note, however, that $\epsilon$::ASP-DURATIVE::`=Adv [F,10,0]` can also erroneously select *usually* in its specifier since it too is of category `Adv`. We would need a separate category for each adverb, since they are being selected via Merge by functional heads.

Thus efficient MGAs do not model adverbials as specifiers of functional heads.

#### 7.6.3.1 Problem

A shortcoming of the present model is that while Merge of a high functional head will prevent later adjunction of a low adverb, nothing prevents a low functional head that selects, say, `V`, from merging after the adjunction of a high adverb. For example, in Figure 7.16, the merger of the low aspectual head resets the second number to 2, allowing the adjunction of *soon*, even though it is lower in the hierarchy than *perhaps*.

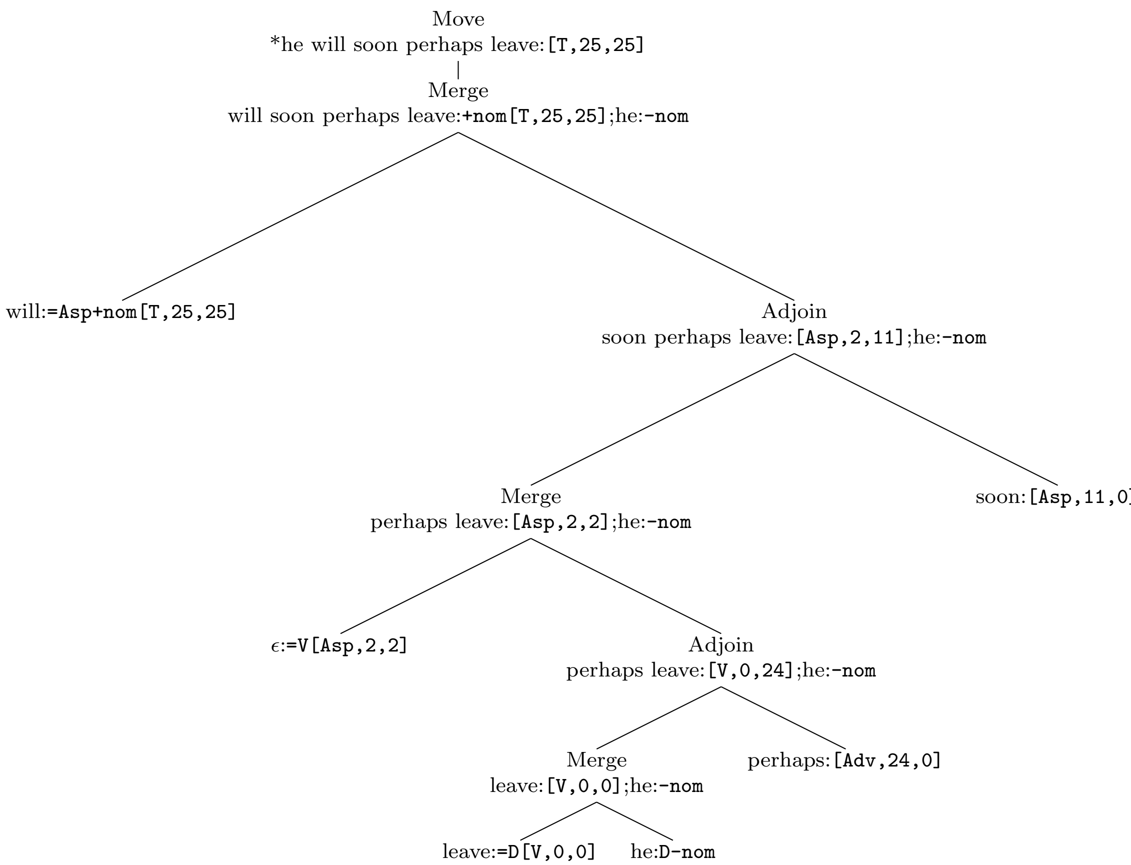



Figure 7.16: Derivation of ungrammatical sentence due to Merge resetting the hierarchy level

This problem will be addressed in section 7.12.

### 7.6.4 Properties

Let us consider the desiderata laid out in section 7.4.

1. **Optionality**: ✓the original category is kept as the first element of the category triple

2. **Transparency to selection**: ✓the original category is kept

3. **One at a time**: ✓The adjunct's one and only category feature is cancelled by Adjoin, so it can't adjoin to anything else

4. **Order**: ✓The third element of the category is the level of the last adjunct adjoined. The Adjoin rule requires that the adjunct be higher in the order than that third element of the category triple.

5. **Selectability** ✓Adjuncts have regular categories.

   Merge

   is bad:`[V,0,0]`

   is::`=Adj[V,0,0]`    bad::`[Adj,4,0]`

   This is of particular importance for functional categories that are in fact required in a given sentence. It is important that we have the mechanism available to treat some functional categories in some cases as optional, and others, for example T in English, as required. Required categories are modelled as per usual, with Merge. For example, in figure 7.17, the requirement of `T` is enforced by `C`'s selecting it. If it were optional, `C` could select `V` directly, with or without `T` intervening. Since `T` selects `V`, rather than adjoining to it, the category of the phrase changes from `V` to `T`, allowing `C` to require `T`'s presence.

   (a) **Efficiency:** ✓Many adjuncts have the same category, so they have the same adjuncts. For example, $\mathbf{Ad}(\texttt{Adj}) = \{\texttt{Adv},\texttt{Int}\}$

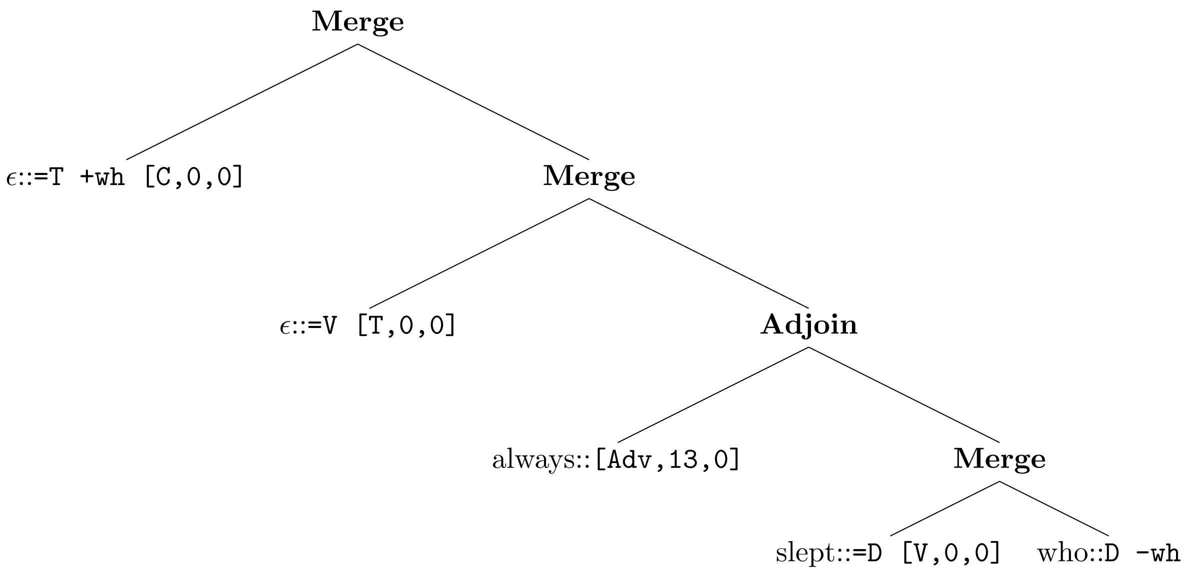



Figure 7.17: *Who always slept?*: required functional categories

6. **Adjuncts of adjuncts** ✓Adjunct categories are ordinary categories so they can have adjuncts too (Figure 7.14).

   (a) **Efficiency:**✓Many adjuncts have the same category, so they are selected by the same LI. For example, in the derivation of *is bad* above, *is* selects anything of category `Adj`.

7. **Unordered** ✓See section 7.7.2 below.

8. **Obligatory adjuncts: Maybe.** See Section 7.7.3

9. **Islands** ✓Since Adjoin is a separate operation, it can be defined so that there is no case for adjuncts with movers.

## 7.7 Discussion and Extensions

This model captures both the strict ordering of the merge-only models and the optionality and transparency to selection of the categorial approaches. Cinque's observation that there is a hierarchy of functional heads and adverbs is modelled directly by defining a hierarchy in the grammar itself. The strict linear order falls out of the order imposed on the selectional features and the definition of **Adjoin**: adjunction is only defined when the hierarchy is respected. Optionality is the result of the transitivity of orders: intervening adjuncts are not necessary for a higher one to be adjoined. Transparency to selection is modelled by the pairing of the selectional features: the original category of the modified element is preserved, and **Merge** can see only that feature. The adjuncts are literally ignored.

The cross-linguistic consistency of the orders is accounted for by the claim that all human languages assign the same order to the same semantic classes of adjuncts. As such, it does not have to be learned, but rather comes with the grammar.

Computationally, this approach has an advantage over the merge-only model with homophony as the latter increases the size of the lexicon by a polynomial function in the depths of the hierarchies of adjuncts, but the former does not.

### 7.7.1 Islandhood

Adjuncts have another classic property: islandhood. Movement is not possible out of certain types of adjuncts (Huang, 1982).

(18) a. You left [because your ex showed up]$_{\mathrm{Adj}}$

b. *Who did you leave [because ___ showed up]$_{\mathrm{Adj}}$?

Any approach that keeps Adjoin separate from Merge introduces the option of stipulating the AIC, either as a separate constraint on Adjoin, as Frey & Gärtner do, or by simply not including movers$_{\mathrm{s}}$ in the definition of Adjoin, making the partial function undefined when the adjunct carries movers. This is not very satisfying, though: better perhaps would be to derive it, as Graf (2013b) does. On the other hand, not all adjuncts are islands.

(19) Who are you sitting [beside __]$_{\text{Adjunct}}$?

As always, islands must remain a matter for further research.

### 7.7.2 Unordered Adjuncts

As it stands, adjuncts such as PPs can be modelled as adjuncts, but they must all adjoin at the same level of the hierarchy, or else be cross-classified for each level of the hierarchy you want them to adjoin at. The former allows them to be freely ordered with respect to each other; the latter gives them freedom with respect to all adjuncts.

An expansion of this model[14] could add a non-number to the set of possible indices, call it $\emptyset$, and Adjoin could be defined to disregard the hierarchy and asymmetrically check the features for $\emptyset$-indexed adjuncts. Any distinct index also opens the door to adjoining on a different side of the head than other adjuncts; the definition I will give here models Engish PPs, which are post-head, unlike adjectives and many adverbs.

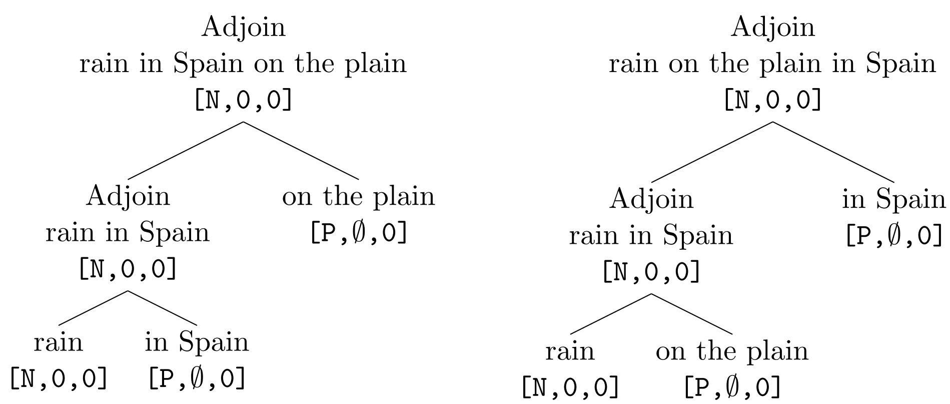



Figure 7.18: Unordered English PPs

In definition 7.7.1, the first and third cases are for adjuncts with number indices, and the second and fourth are for adjuncts with $\emptyset$ indices.

[14] I thank an anonymous reviewer for this suggestion.

**Definition 7.7.1** (Adjoin 2)**.** Let $s, t \in \Sigma$ be strings, $\mathtt{Y,X}\in\mathbf{sel}$ be categories, $i, j, n, m \in \mathbb{N}$, $mvrs \in (\Sigma^* \times F^*)^*$ be a mover list, and $\alpha, \beta \in F^*$.

$$\mathbf{Adjoin}(\langle s, [\mathtt{X},\mathtt{i},\mathtt{j}]\alpha\rangle, \langle t, [\mathtt{Y},\mathtt{m},\mathtt{n}]\beta\rangle :: mvrs)$$
$$= \begin{cases} \langle ts, [\mathtt{X},\mathtt{i},\mathtt{m}]\alpha\rangle :: mvrs & \text{if } \mathtt{m} \geq \mathtt{j} \;\&\; \mathtt{Y} \in \mathbf{Ad}(\mathtt{X}) \;\&\; \beta = \epsilon \\ \langle st, [\mathtt{X},\mathtt{i},\mathtt{j}]\alpha\rangle :: mvrs & \text{if } \mathtt{m} = \emptyset \;\&\; \mathtt{Y} \in \mathbf{Ad}(\mathtt{X}) \;\&\; \beta = \epsilon \\ \langle s, [\mathtt{X},\mathtt{i},\mathtt{m}]\alpha\rangle :: \langle t, \beta\rangle :: mvrs & \text{if } \mathtt{m} \geq \mathtt{j} \;\&\; \mathtt{Y} \in \mathbf{Ad}(\mathtt{X}) \;\&\; \beta \neq \epsilon \\ \langle s, [\mathtt{X},\mathtt{i},\mathtt{j}]\alpha\rangle :: \langle t, \beta\rangle :: mvrs & \text{if } \mathtt{m} = \emptyset \;\&\; \mathtt{Y} \in \mathbf{Ad}(\mathtt{X}) \;\&\; \beta \neq \epsilon \end{cases}$$

### 7.7.3 Obligatory Adjuncts

Recall that some elements which really seem to be adjuncts are not optional, for example *He makes a *(good) father*. In MGAs there is a featural difference between nouns that have been modified and nouns that have not. For example, *father* is of category `[N,0,0]` and *good father* has category `[N,0,4]`. **Merge** is defined to ignore everything but the first element, `N`. However, the architecture is available to let **Merge** look at the whole category triple, by way of a positive selectional feature of the form `=[N,__,1]`, which selects anything of category `[N,i,j]` with $\mathtt{j} \geq \mathtt{1}$.

**Definition 7.7.2** (Merge 2)**.** For $\alpha, \beta$ sequences of negative **lic** feature;, $s, t$ strings; $\mathtt{X}\in$ **sel**; $\mathtt{i},\mathtt{j},\mathtt{m} \in \mathbb{N}$; $\mathbf{C} = \mathtt{X}$ or $\mathbf{C} = [\mathtt{X}, \_\_, \mathtt{m}]$ & $\mathtt{j} \geq \mathtt{m}$:

$$\mathbf{Merge}(\langle s, \mathtt{=}\mathbf{C}\alpha\rangle :: \mathrm{mvrs}_s, \langle t, [\mathtt{X},\mathtt{i},\mathtt{j}]\beta\rangle :: \mathrm{mvrs}_t) = \begin{cases} \langle ts, \alpha\rangle :: \mathrm{mvrs}_s \cdot \mathrm{mvrs}_t & \text{if } \beta = \epsilon \\ \langle s, \alpha\rangle :: \langle t, \beta\rangle :: \mathrm{mvrs}_s \cdot \mathrm{mvrs}_t & \text{if } \beta \neq \epsilon \end{cases}$$

However, such an expansion of the definition of Merge is not of immediate help in all cases. In the case of *He makes a good father*, the NP *good father* is selected by `D` before the resulting DP is selected by *makes*, which is the verb that cares about whether the noun is modified. One solution is to cross-list *a* with a new determiner category only for modified NPs, and let *makes* select that category, as in Figure 7.19.

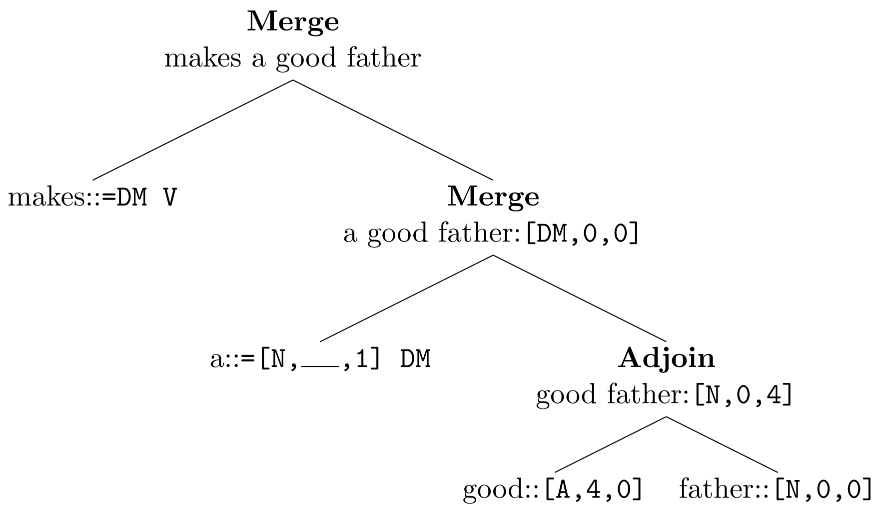



Figure 7.19: Determiners of modified NPs could have their own category DM

Obligatory adjuncts are not the only reason to suspect that the tighter relationship is between the verb and the noun, not the verb and the determiner; i.e. that V should perhaps select N, not D. For one, it is well known that in terms of semantics, verbs select nouns. For example, *The man slept* makes sense, but *The table slept* does not, because men are the kinds of things that sleep and tables are not. Both DPs are headed by *the*, which does not carry the animacy information that the noun does. Another piece of evidence comes from noun incorporation. When a head is incorporated into a verb, normally it is the head that the noun selects that is incorporated, as in (20).

(20) a. He [**stab**bed me [$_{\mathrm{PP}}$ in [$_{\mathrm{DP}}$ the [$_{\mathrm{N}}$ **back**]]]]
b. back-stabbing
c. *back-in, *back-the, *back-the-stabbing, *back-in-the-stabbing

Sportiche (2005) proposes that verbs select NPs, and the NPs move to their Ds, which are functional heads on the spine.

For example we might have something like the partial derivation in Fig 7.20.

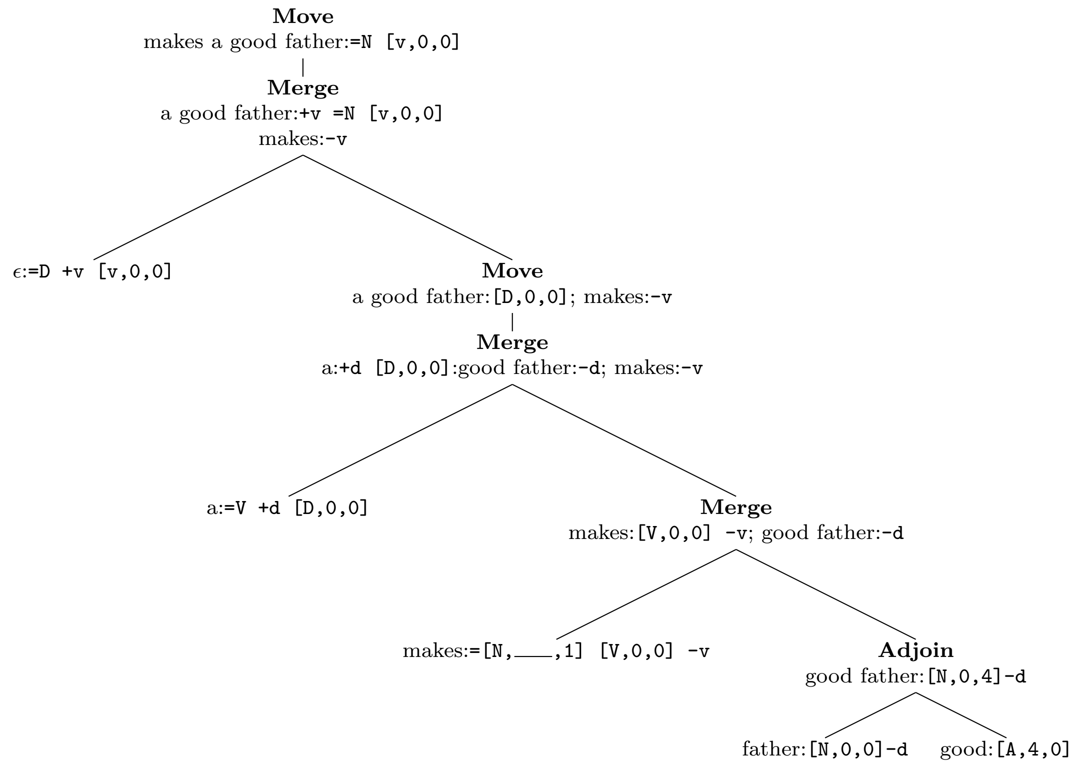



Figure 7.20: Directly selecting N; moving NP up to functional projection D

## 7.8 Formal Properties

MGAs are clearly not strongly equivalent to traditional MGs, if we take *strong equivalence* to mean that the set of derivations trees are the same. This is of course is impossible since MGAs have an extra function, Adjoin. MGAs are, on the other hand, weakly equivalent to MGs, meaning that for every MGA, an MG can be defined that generates the same strings, and vice versa.

For this proof, we need to define the notion of a *suffix* of a lexical item. A suffix of a lexical item $\langle s, \alpha \rangle$ is the features $\alpha$ or any suffix of it. We need to refer to suffixes because as a minimalist grammar acts on an expression, it removes features from the beginning of a feature stack, and that feature stack came originally from the lexicon.

**Definition 7.8.1** (suffix($Lex$)). For a minimalist grammar lexicon $Lex$,

$$\text{suffix}(Lex) = \{\alpha \in F^* | \exists s \in \Sigma^*, \beta \in F^* \ s.t. \ \langle s, \beta\alpha \rangle \in Lex\}$$

To show that traditional MGs and MGAs generate the same string sets (languages), first we show that any language generated by an MG can also be generated by an MGA (Lemma 7.8.2) and then that any language generated by an MGA can also be generated by a traditional MG. The second proof runs intermediately through a Multiple Context Free Grammar, which are already known to be weakly equivalent to MGs (Harkema, 2001).

**Lemma 7.8.2.** *$L(MG) \subseteq L(MGA)$*

*Proof.* MGAs also include Merge and Move, and place no additional restrictions on their action. Any MGA language could have Adjoin stripped away and what remained would be an MG. □

For the next lemma, we a use Multiple Context Free Grammar Seki et al. (1991). MCFGs are like context free grammars, except tuples of strings, not just single strings, can be generated by the application of a rule.

**Definition 7.8.3** (MCFG). An 2,m-MCFG is a 4-tuple $G = \langle N, T, P, S \rangle$ such that:

- $N$ is a finite ranked alphabet of non-terminals of maximal rank $m$
- $T$ is a finite alphabet of terminals (rank 0)
- $P$ is a set of rules of the form

$$A(s_1, ..., s_k) : -B(x_1, ..., x_i) \ \ C(y_1, ...y_j)$$

where:

- $A, B, C \in N$ with ranks $k, i, j$ respectively
- the strings $s_i$ consist only of words from the $T$ and $x_1, ..., x_i, y_1, ..., y_j$.
- Each variable $x_1, ..., x_i, y_1, ..., y_j$ appears at most once in $s_1, ..., s_k$
- $S$ is a non-terminal of rank 1 (the start category)

The language generate by an MCFG is the set of all terminal strings derivable by the rules $P$ from the start category(s) $S$.

**Lemma 7.8.4.** *L(MGA)* $\subseteq$ *L(MG)*

*Proof.* MGs are weakly (and indeed strongly) equivalent to Multiple Context Free Grammars (MCFGs) so it suffices to show that $L(MGA) \subseteq L(MCFG)$.

We translate an MGA into an MCFG is the normal way, following Harkema (2001): the nonterminals of the MCFG are sequences of feature sequences from the MGA. This translation is based on the basic grammar given in Definition 7.6.3, but it is easy to see how it could be expanded to include the extentions suggested in later sections.

Given MGA $G = \langle \Sigma, F = \mathbf{sel} \cup \mathbf{lic}, Lex, M, S, Ad \rangle$, define an MCFG $MCFG(G) = \langle \Sigma, N, P, S \rangle$ defining the language

$N = \{\langle \delta_0, \delta_1, ..., \delta_j \rangle | 0 \leq j \leq |\mathbf{lic}|$, all $\delta_i \in$suffix$(Lex)\}$

Let $h =$Max$(\{i | \exists \mathtt{X} \in \mathbf{sel} : i = |\mathbf{Ad}(\mathtt{X})|\})$

The rules $P$ are defined as follows, $\forall \alpha, \beta, \delta_0, ..., \delta_i, \gamma_0, ..., \gamma_j \in$ suffix(Lex). $s_0, ..., s_i, t_0, ..., t_j$ are variables over strings.

**Lexical rules:** $\alpha(s)$ $\qquad\qquad$ $\forall \langle s, \alpha \rangle \in Lex$

**Merge-and-stay rules:** Here is the first case of the Merge rule for MGAs.

$$\mathbf{Merge}(\langle s, \mathtt{=X}\alpha \rangle :: \text{mvrs}_s, \langle t, [\mathtt{X}, \mathtt{m}, \mathtt{n}] \rangle :: \text{mvrs}_t) = \langle st, \alpha \rangle :: \text{mvrs}_s \cdot \text{mvrs}_t$$

It becomes a set of MCFG rules as follows. In the rule set below, $s = s_0, t = t_0$, the tree parts of mvrs$_s$ and mvrs$_t$ are $s_1, ..., s_i$ and $t_1, ..., t_j$ respectively, and their features become $\delta_1, ..., \delta_i$ and $\gamma_1, ...\gamma_j$. One rule is made for each index less than the maximum possible index $h$ for the grammar. (Any rule indices that fall outside the set of indices for that particular category simply go unused in practice.)

Here is the description of the MCFG rules corresponding to this Merge rule:

$$\langle \alpha, \delta_1, ..., \delta_i, \gamma_1, ...\gamma_j \rangle (s_0 t_0, s_1, ..., s_i, t_1, ..., t_j)$$
$$:\text{-} \ \langle = \mathtt{X}\alpha, \delta_1, ..., \delta_i \rangle (s_0, ..., s_i) \ \langle [\mathtt{X}, \mathtt{m}, \mathtt{n}], \gamma_1, ...\gamma_j \rangle (t_0, ..., t_j)$$
$$\forall \mathtt{X} \in \mathbf{sel}, \forall \mathtt{n}, \mathtt{m} \leq h$$

The rest of the MCFG rules are formed similarly.

**Merge-and-move rules:** $\forall \mathtt{X} \in \mathbf{sel}, \forall \mathtt{n}, \mathtt{m} \leq h$

$$\langle \alpha, \beta, \delta_1, ..., \delta_i, \gamma_1, ...\gamma_j \rangle (s_0, t_0, s_1, ..., s_i, t_1, ..., t_j)$$
$$:\text{-} \ \langle = \mathtt{X}\alpha, \delta_1, ..., \delta_i \rangle (s_0, ..., s_i) \ \langle [\mathtt{X}, \mathtt{m}, \mathtt{n}]\beta, \gamma_1, ...\gamma_j \rangle (t_0, ..., t_j)$$

**Adjoin-and-stay rules:** $\forall \mathtt{X}, \mathtt{Y} \in \mathbf{sel} \ \ s.t. \ \mathtt{Y} \in \mathbf{Ad}(\mathtt{X}), \forall \mathtt{k}, \mathtt{l}, \mathtt{n}, \mathtt{m} \leq h \ \ s.t. \ \mathtt{n} \geq \mathtt{k}$

$$\langle [\mathtt{X}, \mathtt{m}, \mathtt{n}], \delta_1, ..., \delta_i, \gamma_1, ...\gamma_j \rangle (s_0 t_0, s_1, ..., s_i, t_1, ..., t_j)$$
$$:\text{-} \ \langle [\mathtt{X}, \mathtt{m}, \mathtt{k}], \delta_1, ..., \delta_i \rangle (s_0, ..., s_i) \ \langle [\mathtt{Y}, \mathtt{n}, \mathtt{l}], \gamma_1, ...\gamma_j \rangle (t_0, ..., t_j)$$

**Adjoin-and-move rules:** $\forall \mathtt{X}, \mathtt{Y} \in \mathbf{sel} \ \ s.t. \ \mathtt{Y} \in \mathbf{Ad}(\mathtt{X}), \forall \mathtt{k}, \mathtt{l}, \mathtt{n}, \mathtt{m} \leq h \ \ s.t. \ \mathtt{n} \geq \mathtt{k}$

$$\langle [\mathtt{X}, \mathtt{m}, \mathtt{n}], \beta, \delta_1, ..., \delta_i, \gamma_1, ...\gamma_j \rangle (s_0, t_0, s_1, ..., s_i, t_1, ..., t_j)$$
$$:\text{-} \ \langle [\mathtt{X}, \mathtt{m}, \mathtt{k}], \delta_1, ..., \delta_i \rangle (s_0, ..., s_i) \ \langle [\mathtt{Y}, \mathtt{n}, \mathtt{l}]\beta, \gamma_1, ...\gamma_j \rangle (t_0, ..., t_j)$$

**Move-and-stop rules:** $\forall \mathtt{f} \in \mathbf{lic}$

$$\langle \alpha, \delta_1, ..., \delta_{i-1}, \delta_{i+1}, ..., \delta_j \rangle (s_i s_0, s_1, ..., s_{i-1}, s_{i+1}, ..., s_j)$$
$$:\text{-} \ \langle +\mathtt{f}\alpha, \delta_1, ..., \delta_{i-1}, -\mathtt{f}, \delta_{i+1}, ..., \delta_j \rangle (s_0, ..., s_j)$$

**Move-and-keep-moving rules:** $\forall \mathtt{f} \in \textbf{lic}$

$$\langle \alpha, \delta_1, ..., \delta_{i-1}, \beta, \delta_{i+1}, ..., \delta_j \rangle (s_0, ..., s_j)$$
$$:\!\!- \langle +\mathtt{f}\alpha, \delta_1, ..., \delta_{i-1}, -\mathtt{f}\beta, \delta_{i+1}, ..., \delta_j \rangle (s_0, ..., s_j)$$

These rule sets are finite since MGAs never add anything to feature sequences, but only either remove features or change just the indices of `[X,i,j]` features. As such, the suffixes $\alpha, \beta, \delta, \gamma$ are limited in number. Since any given grammar has a maximal hierarchy depth $h$, the indices `k,l,m,n` in the rules are defined to be limited by $h$. The $\delta$'s and $\gamma$'s are limited by the SMC. They are the movers, and there can never be more than one mover in the mover list for a given licensing feature, and **lic** is finite.

The MCFG defined as above derives the same strings as the MGA. The proof is by induction on the depth of the derivation tree. A derivation tree of depth $k$ has a Merge, Move, or Adjoin node dominating two sisters of depth less than $k$, deriving expressions with MCFG equivalents. The construction of the grammar provides corresponding Merge, Move, or Adjoin rule equivalencies to build the equivalent expression.

A sentence $s$ is in an MGA language if there is a derivation deriving an expression $\langle\langle s,$ `[C,i,j]`$\rangle\rangle$ for some final category `C` and any `i,j`$\in \mathbb{N}$. A sentence $s$ is in an MCFG language if it can derive expression $\langle C \rangle (s)$ for some final category C. The final categories of this MCFG are `[C,i,j]` for all MGA final categories `C` and all $\mathtt{i}, \mathtt{j} \leq \mathtt{h}$.

$\square$

**Theorem 7.8.5** (Weak equivalence of MGAs and MGs)**.** *For any MGA* $G = \langle \Sigma, \boldsymbol{sel}, \boldsymbol{lic}, \boldsymbol{Ad}, Lex, \{\boldsymbol{Merge,\ Move,\ Adjoin}\}\rangle$, *there is a weakly equivalent MG* $G' = \langle \Sigma, \boldsymbol{sel}_{MG}, \boldsymbol{lic}, Lex_{MG}, \{\boldsymbol{Merge,\ Move}\}\rangle$.

*Proof.* By lemmas 7.8.2 and 7.8.4 $\square$

## 7.9 Interim Summary

We have seen that to account for both the "looseness" of adjuncts – their optionality and transparency – and their "strictness" – their ordering properties is difficult. Previous models account for one or the other, but not both. MGAs are able to account for both, by the simple expedience of splitting their categories into triples. The first element is the basic category name; its adjunct relation with other categories accounts for the optionality and transparency of adjunction. The second and third elements add a hierarchy level, and Adjoin is defined to forbid adjunction to a phrase that already has a higher adjunct adjoined to it.

The following table summarises the models we've seen so far.

| | Trad<br>7.5.1 | F&G<br>7.5.2 | Syn. glue<br>7.5.3.1 | Homoph<br>7.5.3.2 | Fowlie13<br>7.5.4 | MGA<br>7.6 |
|---|---|---|---|---|---|---|
| Optional | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Eff (opt) | ✓ | ✓ | ✗ | ✗✗ | ✓ | ✓ |
| Transparent | ? | ✗ | ✗ | ✗ | ✓ | ✓ |
| 1-at-a-time | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ |
| Order | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ |
| Select | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Eff (Sel) | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| Adj of adj | ✓ | ? | ✓ | ✓ | ✓ | ✓ |
| Eff (Adj) | ✗ | ✓? | ✗ | ✗ | ✗ | ✓ |
| Unordered | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Oblig | ✗ | ✗ | ✗ | ✗ | ? | ✓ |
| Island | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ |

Table 7.4: Summary of models re: desiderata

For a minimalist analysis of adjunction with feature-driven Merge, in which adjunct ordering is part of the syntax, a mechanism along the lines of MGA indices is required. The adjuncts are optional and they belong to what appears syntactically to be only a few categories (perhaps just Adjectives, Adverbs, Prepositions, and Intensifiers), yet they are ordered. As such, they behave simultaneously as though they belong to the same categories and to distinct categories. The apparent difference in categories accounts for ordering; everything else about them is accounted for by their simply being adjuncts, not arguments.

I therefore claim that a function like Adjoin that operates on two complete phrases (in the features calculus, this means they are displaying their category features) according to what may adjoin to what, and which generates a phrase of the adjoined-to category, most naturally accounts for the non-order-related behaviour of adjuncts. If order is to be accounted for in the syntax, something like these indices is needed, separating the categories (so that the adjuncts may continue to behave as a small number of unified classes) from the hierarchy levels. (Two indices are needed to distinguish the hierarchy level of potential adjuncts-of-adjuncts from the hierarchy level of the adjunct itself.)

## 7.10 What should move?

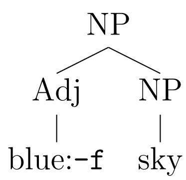
This grammar models adjuncts as part of the phrase projected by the adjoined-to; for example, adjectives are part of the NP. This means that while adjectives can move out of the NP, the noun can't move independently, except possibly in the case of head-movement.

Adjoin could be defined differently, with both adjunct and adjoinee moveable.

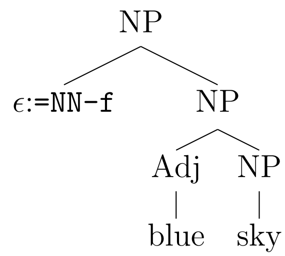
**Definition 7.10.1** (Adjoin$_3$)**.** Let $s, t \in \Sigma$ be strings, $\mathtt{Y,X}\in\mathbf{sel}$ be categories, $i, j, n, m \in \mathbb{N}$, $mvrs \in (\Sigma^* \times F^*)^*$ be a mover list, and $\alpha, \beta \in F^*$.

$$\mathbf{Adjoin}(\langle s, [\mathtt{X},\mathtt{i},\mathtt{j}]\alpha :: mvrs\rangle, \langle t, [\mathtt{Y},\mathtt{n},\mathtt{m}]\beta\rangle)$$
$$= \begin{cases} \langle ts, [\mathtt{X},\mathtt{i},\mathtt{n}]\alpha\rangle :: mvrs & \text{if } \mathtt{n} \geq \mathtt{j} \ \&\ \mathtt{Y} \in \mathbf{Ad}(\mathtt{X}) \ \&\ \alpha = \beta = \epsilon \\ \langle s, [\mathtt{X},\mathtt{i},\mathtt{n}]\alpha\rangle :: \langle t, \beta\rangle :: mvrs & \text{if } \mathtt{n} \geq \mathtt{j} \ \&\ \mathtt{Y} \in \mathbf{Ad}(\mathtt{X}) \ \&\ \alpha = \epsilon \ \&\ \beta \neq \epsilon \\ \langle t, [\mathtt{X},\mathtt{i},\mathtt{n}]\rangle :: \langle s, \alpha\rangle :: mvrs & \text{if } \mathtt{n} \geq \mathtt{j} \ \&\ \mathtt{Y} \in \mathbf{Ad}(\mathtt{X}) \ \&\ \beta = \epsilon \ \&\ \alpha \neq \epsilon \\ \langle \epsilon, [\mathtt{X},\mathtt{i},\mathtt{n}]\rangle :: \langle s, \alpha\rangle :: \langle t, \beta\rangle :: mvrs & \text{if } \mathtt{n} \geq \mathtt{j} \ \&\ \mathtt{Y} \in \mathbf{Ad}(\mathtt{X}) \ \&\ \beta \neq \epsilon \ \&\ \alpha \neq \epsilon \end{cases}$$

Such a grammar would preclude move features from being introduced by the adjoinee and yet moving the whole phrase; to move the whole phrase we would need to Merge a silent element that introduces a move feature.



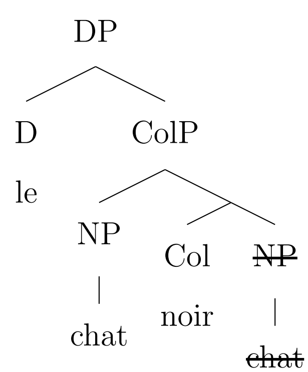

(a) Using $\text{Adjoin}_3$ defined in Def 7.10.1, *blue* introduces the move feature and moves independently of *sky*: the moving phrase here is *blue*.




(b) To make *blue sky* move, a silent head Merges.

Figure 7.21: Adjoinee can move independently using $\text{Adjoin}_3$

Whether this grammar or something like the original in Definition 7.6.2 is correct is an empirical question. Graf (2014) argues against and Koopman (2015) argues in favour of such a grammar, accounting for right adjuncts in French with phrasal movement of the noun phrase, excluding the adjective's projection. Notice that in her grammar, adjuncts are Merged, as in the model in Section 7.5.3.2, but the notion is the same.




## 7.11 Graf 2014

Graf (2014) analyses the proposals given here from a model-theoretic perspective, and describes how the proposals alter some basic mathematical properties of MGs, as well as evaluating them for their ability to account for empirical facts. He finds that of the models analysed, only MGAs can account for all linguistic phenomena under consideration, but they do so at the expense of several nice mathematical properties, including locality of Merge. He argues, though, that this is not a result of the model itself so much as the phenomena to be modelled.

In a traditional MG, Merge is strictly $k$-local.[15] What this means is that if you had an MG that never employed Move, the derivation trees would form a strictly $k$-local ($SL_k$) tree language. An $SL_k$ tree language is one that can be defined by a set of legal k-factors, which are building block subtrees of depth $k$.

A lexical item $l$ controls a *slice*, a subpart of a derivation tree. The slice controlled by $l$ is $l$ itself plus any Merge and Move nodes dominating it that occur because of positive features in $l$'s feature stack. For example, in the derivation tree in Figure 7.22, the slice controlled by *man* is just *man* (blue), that for *the* is *the* and the Merge node above it (red), and *slept* controls itself and the root node (green).

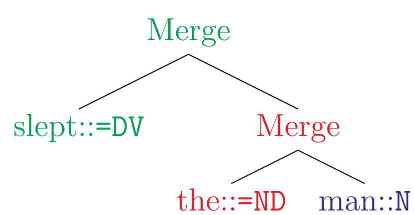



Figure 7.22: Slices by colour

The depth of an LI's slice depends of the number of features preceding the category feature. Since Lex is finite, the largest possible slice is fixed for each grammar. One way of thinking about it is this: suppose you have a Merge node in a derivation tree, and you want to know if it is valid case of Merge. The farthest you have to look to find out if a given Merge node is licit is the maximal number of positive features on any item in Lex, plus one. For example, in Fig. 7.23, the

[15]For more on subregularity of MGs, please see chapter **??**.

farthest we need to look to determine the validity of the root node is 3 nodes, but we can imagine that if the node labelled `=A=BC` had more features before `C`, we would need to look that much farther. `=A=BC` has two features before the category feature `C`, but we also need to include the Merge node that makes it sister to `=CD`.

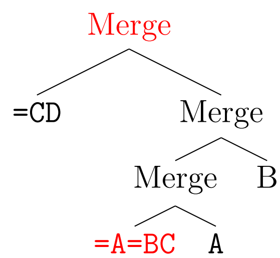



Figure 7.23: To determine if the root node is a valid application of Merge, we need to look 3 nodes away, to `=A=BC`.

Move interrupts this locality, because a mover's first Merge site (which is where it occurs in the derivation tree) can be indefinitely far away from its final landing site.

(21) Who did River think that Simon said that Jayne would shoot __?

We can't know if a given subtree is legal if the mover hasn't landed yet, and there is no way of knowing, locally, when that will be. For example, we can't know whether the (sub)tree in in Fig.7.24, is valid until *who* Moves. Perhaps a lexical item with feature `+wh` will never be Merged.

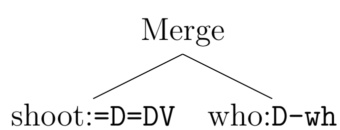



Figure 7.24: The validity of this derivation tree cannot be determined.

The question Graf raises is what happens when Adjoin is added to the grammar. The answer is that even without Move, the derivation tree language loses its $SL_k$ status. Slices are no longer bounded, since Adjoin is defined by a partial order. For instance, in (22), an indefinite number of *big*s can intervene between *the* and *ship*, which translates into an indefine number of Adjoin nodes intervening between the root node and *ship*, which it needs to see in order to determine whether *the* can safely Merge with *big ship*.

(22) the big big … big ship

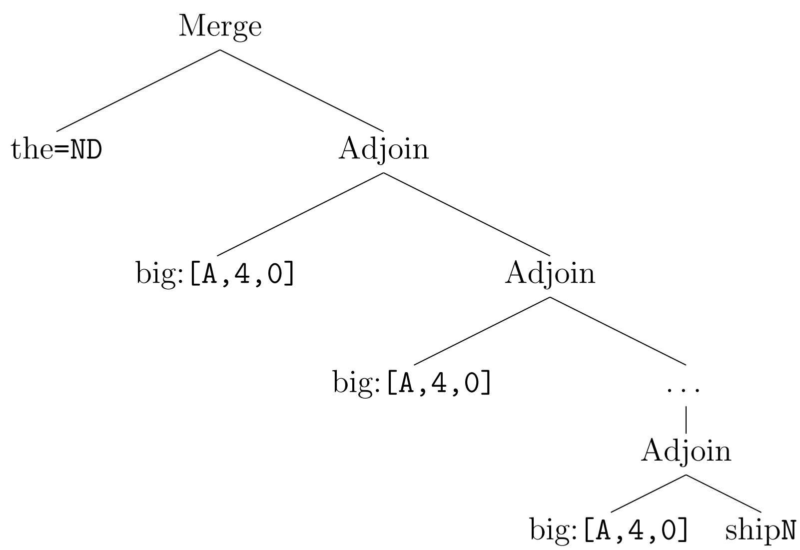



Figure 7.25: Adjoin interrupts the locality of Merge

Thus MGAs are not $\mathrm{SL}_k$ for any $k$.

Other properties of MGs lost in MGAs are local threshold testability (a kind of locality) and recognisability by a type of deterministic top-down tree automaton.

## 7.12 Proposal 2: HoP-Merge

Adger (2008) proposes a related model of functional heads. He proposes a splitting of Merge into two functions: Sel-Merge (selectional Merge), which is the traditional Merge, and HoP-Merge (Hierarchy of Precedence Merge), which is not entirely unlike Adjoin.

An order is defined on the functional heads, and HoP-Merge is only defined if the Merging head is higher in the hierarchy than the most recently merged functional head.

The difference between his HoP-Merge and my Adjoin is that Hop-Merge is otherwise just like regular Merge: The features are symmetrically checked, which begins a new phrase. With Adjoin, the old category is kept; only the hierarchy level changes.

I have transformed his grammar into an MG, with Hop-Merge added. Hop-Merge is valid when the categories of its arguments are in the same hierarchy (defined in the grammar) and the left daughter is higher than the right.

Suppose we have the following grammar:

$Lex = \{\text{the:}\mathtt{=D\ V,0}, \text{three:}\mathtt{Num,4}, \text{kings:}\mathtt{N,0}, \text{sang:}\mathtt{=D\ V,0}\}$

$H_1 = \mathtt{D}, 5 > \mathtt{Num}, 4 > \mathtt{Poss}, 3 > \mathtt{n}, 2 > \mathtt{N}, 1$

$H_2 = \mathtt{C}, 3 > \mathtt{T}, 2 > \mathtt{V}, 1$

We can derive *the three kings sang* as in Figure 7.26.

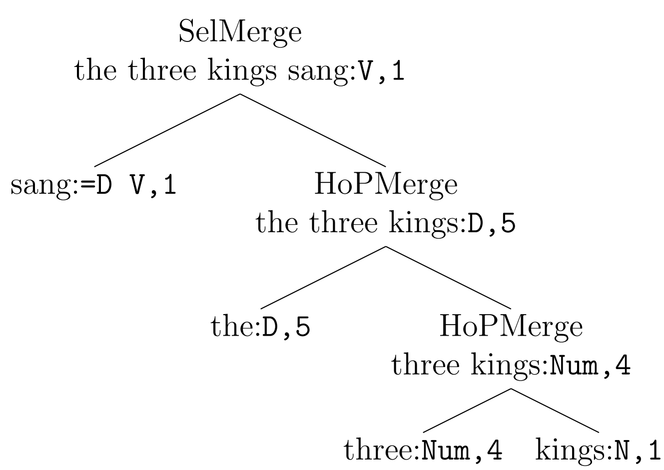



Figure 7.26: Derivation tree of *the three kings sang* using HopMerge

The interaction of the two Merges can force the presence of a functional head by having SelMerge select for it directly. Without SelMerge, the verb must be defined instead to select any nominal projection, by making it part of $H_1$:

$H_1' = \mathtt{V}, 6 > \mathtt{D}, 5 > \mathtt{Num}, 4 > \mathtt{Poss}, 3 > \mathtt{n}, 2 > \mathtt{N}, 1$

This, however, is not only the wrong result (verb are not part of the functional projections of nouns) but makes it impossible for a verb to select a CP under normal assumptions. The reason is that we cannot get a partial order on the functional heads this way, because V selects C (for verbs like *think*) but in hierarchy 2, we have $H_2 = \mathtt{C}, 3 > \mathtt{T}, 2 > \mathtt{V}, 1$. The hierarchies are supposed to form a partial order, but now, between $H_1'$ and $H_2$, we have $V > C > V$, a contradiction.

Therefore, if we are to have HoPMerge, we must also have SelMerge.

### 7.12.1 A combined approach

To solve the problem laid out in section 7.6.3.1, I propose an alternate approach, call it Minimalist Grammars with Hierarchies (MGHs), that incorporates both Adjoin and HoPMerge.

The traditional MG is enriched with two partial functions, *Ad* and *Fn*, that map categories to their adjuncts and functional projections, respectively.[16]

Categories are themselves enriched with two numbers, the first of which indicates the hierarchy level of the item, and the second of which indicates the hierarchy level of the whole phrase. This is just as in MGAs.

Merge, Move, and Adjoin are defined just as in MGAs. HopMerge is defined as follows. Note that we have to dig down into the category feature, rather than always taking features off the top of the stack.

**Definition 7.12.1** (HoPMerge)**.** Let $s, t \in \Sigma$ be strings, $\mathtt{Y},\mathtt{X} \in \mathbf{sel}$ be categories, $i, j, n, m \in \mathbb{N}$, $mvrs_s, mvrs_t \in (\Sigma^* \times F^*)^*$ be mover lists, and $\alpha, \beta, \gamma \in F^*$. Suppose $\exists \mathtt{Z} \in \mathbf{sel}$ such that $\mathtt{X}, \mathtt{Y} \in Fn(\mathtt{Z})$ and suppose $i \geq j$. Then:

[16] We might want to require that the values of $Fn$ be disjoint. This would avoid a projection from "changing its mind" partway through about what kind of projection it is. Whether this is the right result is an empirical question.

$$\mathbf{HopMerge}(\langle s, \gamma[\mathtt{X},\mathtt{i},\mathtt{m}]\alpha :: mvrs_s\rangle, \langle t, [\mathtt{Y},\mathtt{n},\mathtt{j}]\beta\rangle :: mrvs_t)$$
$$= \begin{cases} \langle st, \gamma[\mathtt{X},\mathtt{i},\mathtt{i}]\alpha\rangle :: mvrs_s \cdot mrvs_t & \text{if } \beta = \epsilon \\ \langle s, \gamma[\mathtt{X},\mathtt{i},\mathtt{i}]\alpha\rangle :: \langle t, \beta\rangle :: mvrs_s \cdot mrvs_t & \text{if } \beta \neq \epsilon \end{cases}$$

Figure 7.27 shows a derivation of *the three old kings*, with $\mathtt{A} \in \mathbf{Ad}(\mathtt{N})$ and $\mathtt{D}, \mathtt{Num}, \mathtt{N} \in Fn(\mathtt{N})$.

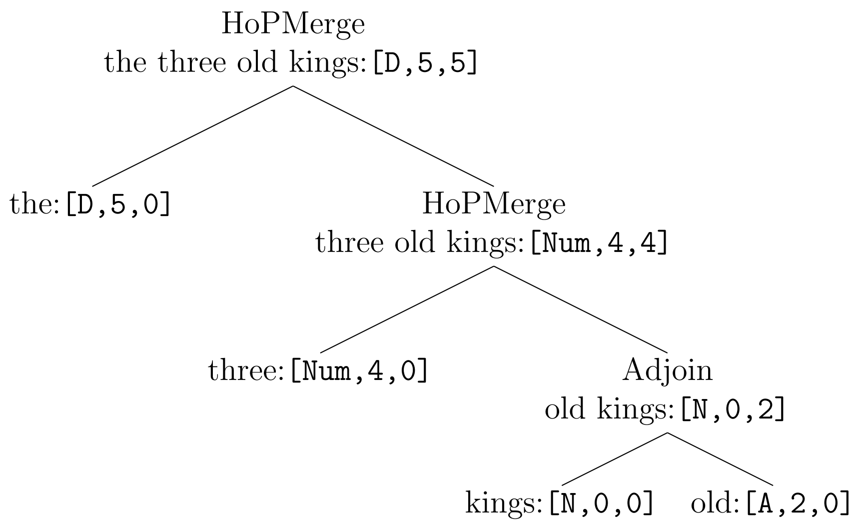



Figure 7.27: The three old kings derived by an MGH

The left argument of HoPMerge must be at a higher hierarchy level than any previously Adjoined or HoPMerged item. The new category feature takes on the category of the functional head. Both numbers are set to the hierarchy level of the functional head. This ensures that even if the functional head has been adjoined to, changing its second number, the result is something that can only be Adjoined to or HoPMerged to by something at a higher level than the functional head.[17]

[17] Here too we may want to constrain the grammar, requiring that the functional head be new to the derivation – truly a functional head, not a complex structure. To do this we need a way to track whether an element is new to the derivation. For a tree-generating grammar this is straightforward: it will be a trivial tree. For a string-generating grammar we take inspiration from Keenan and Stabler (2003) and add a boolean element that tells us whether the item is new. In Keenan and Stabler (2003) this is used to put the head linearly between the specifier and the complement.

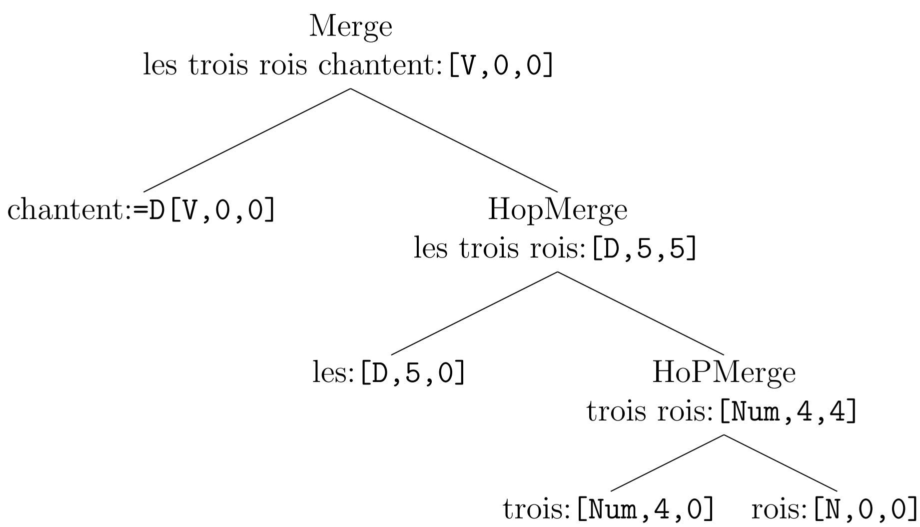
### 7.12.2 Discussion

This approach solves the problem of low functional heads Merging after high adjuncts. We use the same numbering system to track both adjuncts and functional heads, but functional heads (HoP)merge, while adjuncts Adjoin. Both have the possibility of being optional. Adjuncts are optional as they are in MGAs, and functional heads are optional when they have another functional head HopMerged right above them. That higher functional head may not be optional – it may itself be properly selected – but the lower one is. Whether this is the correct result is an empirical question. It seems to work well for French DPs. For example, in *Les trois rois chantent* 'the three kings sing', *trois* 'three' is optional, while *les* 'the' is not. The obligatoriness of the determiner is enforced by Merge, since *chantent* 'sing' requires a `D` (Fig.7.28).

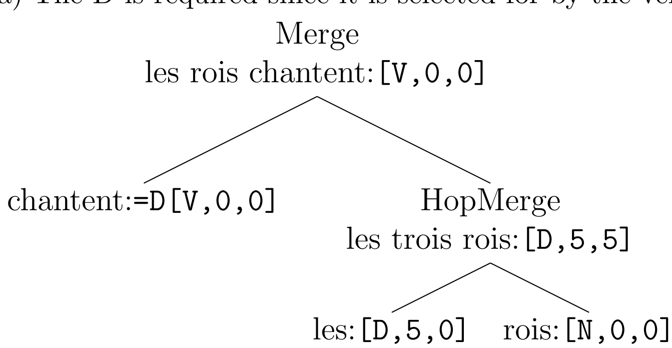
One might ask why the functional heads' order is not just given directly by a partial order in the grammar instead of explicitly by numbers. After all, functional heads do project their own phrases, and they have their own category names. The reason is that Cinque's hierarchy (Fig. 7.29) has a plethora of heads of the same sort: about a dozen Aspect heads, all ordered with respect to one another and interwoven with other categories, and Mod and T heads scattered throughout as well. Given the choice between giving them separate categories and ordering the categories, and giving them the same category and listing the order in the lexical items, I chose the second. The second solution fits much better in the existing MGA model – since the numbers are right there in the category triples – and allows similar heads to have the same category.

We can say that $F(\mathtt{V}) = \{\mathtt{Asp,\ Voice,\ T,\ Mod,\ Mood}\}$ rather than listing all 30, and include such lexical items as will:`=V+nom[T,25,0]` and -ed:`=V+nom[T,26,0]`.

Still, if a partial order on **sel** is preferred to a separate index set, this a different MGH with HoPMerge 2 can be used (Def 7.12.3).




(a) The D is required since it is selected for by the verb




(b) The Num is optional since it's only HoPMerged to by D. HoPMerge only requires that the right argument be lower in the hierarchy than the left.

Figure 7.28: Optionality of internal functional head in French DPs

[Mood$_{\text{speech-act}}$ *frankly*
[Mood$_{\text{evaluative}}$ *fortunately*
[Mood$_{\text{evidential}}$ *allegedly*
[Mod$_{\text{epistemic}}$ *probably*
[T$_{\text{past}}$ *once*
[T$_{\text{future}}$ *then*
[Mod$_{\text{irrealis}}$ *perhaps*
[Mod$_{\text{necessity}}$ *necessarily*
[Mod$_{\text{possibility}}$ *possibly*
[Asp$_{\text{habitual}}$ *usually*
[Asp$_{\text{repetetive}}$ *again*
[Asp$_{\text{frequentative(I)}}$ *often*
[Mod$_{\text{volitional}}$ *intentionally*
[Asp$_{\text{celerative(I)}}$ *quickly*
[T$_{\text{anterior}}$ *already*
[Asp$_{\text{terminative}}$ *no longer*
[Asp$_{\text{continuative}}$ *still*
[Asp$_{\text{perfect(?)}}$ *always*
[Asp$_{\text{retrospective}}$ *just*
[Asp$_{\text{proximative}}$ *soon*
[Asp$_{\text{durative}}$ *briefly*
[Asp$_{\text{generic/progressive}}$ *characteristically*
[Asp$_{\text{prospective}}$ *almost*
[Asp$_{\text{sg.completive(I)}}$ *completely*
[Asp$_{\text{pl.completive}}$ *tutto*
[Voice *well*
[Asp$_{\text{celerative(II)}}$ *fast/early*
[Asp$_{\text{repetetive(II)}}$ *again*
[Asp$_{\text{frequentative(II)}}$ *often*
[Asp$_{\text{sg.completive(II)}}$ *completely* ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]

Figure 7.29: Complete Cinque hierarchy

**Definition 7.12.2.** Let $\geq\ \subseteq \mathbf{sel} \times \mathbf{sel}$ be a partial order on **sel**.

**Definition 7.12.3** (HoPMerge 2)**.** Let $s, t \in \Sigma$ be strings, $\mathtt{Y,X}{\in}\mathbf{sel}$ be categories, $\mathtt{I}, \mathtt{J} \in \mathbf{sel}$, $mvrs \in (\Sigma^* \times F^*)^*$ be a mover list, and $\alpha, \beta, \gamma \in F^*$. Suppose $\exists \mathtt{Z} \in \mathbf{sel}$ such that $\mathtt{X}, \mathtt{Y} \in F(\mathtt{Z})$ and suppose $\mathtt{X} \geq \mathtt{J}$. Then:

$$\begin{aligned}&\mathbf{HopMerge}(\langle s, \gamma[\mathtt{X}, \mathtt{X}, \mathtt{X}]\alpha :: mvrs\rangle, \langle t, [\mathtt{Y}, \mathtt{I}, \mathtt{J}]\beta\rangle)\\ =&\begin{cases}\langle st, \gamma[\mathtt{X}, \mathtt{X}, \mathtt{X}]\alpha\rangle :: mvrs & \text{if } \beta = \epsilon\\ \langle s, \gamma[\mathtt{X}, \mathtt{X}, \mathtt{X}]\alpha\rangle :: \langle t, \beta\rangle :: mvrs & \text{if } \beta \neq \epsilon\end{cases}\end{aligned}$$

In Adjoin, we still need both indices, but we index with the corresponding functional head rather than an arbitrary indexing set like $\mathbb{N}$. Adjuncts are listed in the lexicon with their corresponding functional head in their triple.

**Definition 7.12.4** (Adjoin (for HoPMerge 2 model))**.** Let $s, t \in \Sigma$ be strings, $\mathtt{Y,X}{\in}\mathbf{sel}$ be categories, $I, J, N, M \in \mathbf{sel}$, $mvrs \in (\Sigma^* \times F^*)^*$ be a mover list, and $\alpha, \beta \in F^*$.

$$\begin{aligned}&\mathbf{Adjoin}(\langle s, [\mathtt{X}, \mathtt{I}, \mathtt{J}]\alpha :: mvrs\rangle, \langle t, [\mathtt{Y}, \mathtt{N}, \mathtt{M}]\beta\rangle)\\ =&\begin{cases}\langle ts, [\mathtt{X}, \mathtt{I}, \mathtt{N}]\alpha\rangle :: mvrs & \text{if } \mathtt{N} \geq \mathtt{J}\ \&\ \mathtt{Y} \in \mathbf{Ad}(\mathtt{X})\ \&\ \beta = \epsilon\\ \langle s, [\mathtt{X}, \mathtt{I}, \mathtt{N}]\alpha\rangle :: \langle t, \beta\rangle :: mvrs & \text{if } \mathtt{N} \geq \mathtt{J}\ \&\ \mathtt{Y} \in \mathbf{Ad}(\mathtt{X})\ \&\ \beta \neq \epsilon\end{cases}\end{aligned}$$

However, this model will only work if there are indeed functional heads corresponding to every adjunct level. At least for adjectives, it is not clear that this is really the case. As a technical fix, dummy category names can be added to **sel** to mimic the functional hierarchies that lack functional heads.

A potential inefficiency to be addressed in MGHs as well as MGAs is in how adjuncts interact with functional heads. Since functional heads can Merge, `Adv` needs to be in Ad(`V`), Ad(`Mod`), Ad(`Asp`), Ad(`T`), Ad(`Mod`), and Ad(`Mood`). It is not a coincidence that adjuncts of the verbal spine may also be adjuncts of the verb itself. Do we perhaps want to say that $Ad(X) = Ad(Y)$ if $Y \in F(X)$? In this case we need only specify some adjuncts in the grammar, indeed perhaps only for lexical categories (as opposed to functional). Whether adjuncts of a lexical category are also adjuncts of all its functional projections is an empirical question.

MGHs look like a better model of human language than MGAs, as they have the same properties in terms of our desiderata as MGAs, and they solve the problem of low functional heads Merging above high adjuncts. However, they do so at the expense of an additional operation, bringing the total to 4, double that of traditional MGs.

### 7.12.3 Formal Properties

MGHs are weakly equivalent to MGs. To the MCFG laid out in Lemma 7.8.4 we add two more sets of rules to match the two HoPMerge cases.

$\forall \mathtt{X}, \mathtt{Y} \in \textbf{sel} \;\; s.t.\ \exists \mathtt{Z}\ s.t.\ \mathtt{X}, \mathtt{Y} \in Fn(\mathtt{Z});$

$\forall \mathtt{k}, \mathtt{l}, \mathtt{n}, \mathtt{m} \leq h \;\; s.t.\ \mathtt{k} \geq \mathtt{l};$

$\forall i, j \leq |\textbf{lic}|, \forall \alpha[\mathtt{X}, \mathtt{k}, \mathtt{m}]\alpha', \beta, \delta_1, ...\delta_i, \gamma_1, \gamma_j \in \text{suffix}(Lex),$

Form a rule:

**HoP-and-stay rules:** $\langle \alpha[\mathtt{X}, \mathtt{k}, \mathtt{k}]\alpha', \delta_1, ..., \delta_i, \gamma_1, ...\gamma_j \rangle (s_0 t_0, s_1, ..., s_i, t_1, ..., t_j)$

:- $\langle [\mathtt{X}, \mathtt{k}, \mathtt{m}], \delta_1, ..., \delta_i \rangle (s_0, ..., s_i)\ \langle [\mathtt{Y}, \mathtt{n}, \mathtt{l}], \gamma_1, ...\gamma_j \rangle (t_0, ..., t_j)$

And a rule:

**HoP-and-move rules:** $\langle [\mathtt{X}, \mathtt{k}, \mathtt{k}], \beta, \delta_1, ..., \delta_i, \gamma_1, ...\gamma_j \rangle (s_0, t_0, s_1, ..., s_i, t_1, ..., t_j)$

:- $\langle [\mathtt{X}, \mathtt{k}, \mathtt{m}], \delta_1, ..., \delta_i \rangle (s_0, ..., s_i)\ \langle [\mathtt{Y}, \mathtt{n}, \mathtt{l}]\beta, \gamma_1, ...\gamma_j \rangle (t_0, ..., t_j)$

Because the lexicon is finite and HoPMerge never adds anything to the feature sequences, the number of suffixes $(\alpha' s, \beta' s, \delta' s, \gamma' s)$ is finite. The finite depth $h$ of the hierarchies constrains the indices `k,l,m,n` and the SMC constrains the numbers of $\delta' s$ and $\gamma' s$ to $|\textbf{lic}|$. Thus this grammar is indeed an MCFG, and an extension of the inductive proof on the depth of derivations sketched in 7.8.4 shows that this is the right MCFG to correspond to the minimalist grammar.

## 7.13 Conclusion

We have seen several models of adjuncts and functional heads. Their properties in terms of the desiderata in figure 7.4 are summarised in Table 7.5

| | Trad 7.5.1 | F&G 7.5.2 | Syn. glue 7.5.3.1 | Homoph 7.5.3.2 | Fowlie13 7.5.4 | MGA 7.6 | MGH 7.12 |
|---|---|---|---|---|---|---|---|
| Optional | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Eff (opt) | ✓ | ✓ | ✗ | ✗✗ | ✓ | ✓ | ✓ |
| Transparent | ? | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| 1-at-a-time | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ |
| Order | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Select | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Eff (Sel) | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ | ✓ |
| Adj of adj | ✓ | ? | ✓ | ✓ | ✓ | ✓ | ✓ |
| Eff (Adj) | ✗ | ✓? | ✗ | ✗ | ✗ | ✓ | ✓ |
| Unordered | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Oblig | ✗ | ✗ | ✗ | ✗ | ? | ✓ | ✓ |
| Island | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

Table 7.5: Summary of models re: desiderata

My proposals, MGAs (Minimalist Grammars with Adjunction) and MGHs (Minimalist Grammars with Hierarchies) account for the empirical facts laid out in Cinque (1999) as well as eight more desiderata, unlike the other proposal considered. They do so at the expense of some of the simplicity of MGs. MGAs and MGHs add 1 and 2 operations respectively, require categories to be tuples, and add to the grammar some additional functions (Ad and F). The derivation trees of MGAs and MGHs are not locally threshold testable, recognisable by a (certain type of) deterministic top-down tree automaton, or strictly $k$-local, unlike those of traditional MGs. However, it is the properties of adjunction facts, not merely of the models, that force these losses of simplicity (Graf, 2014).

# Bibliography


Adger, David. 2008. A minimalist theory of feature structure. MSS, May 2008.

Bervoets, Mel, Niki Foster, Meaghan Fowlie, Laura Kalin, Jianjing Kuang, Laura McPherson, Pamela Munro, Kathleen O'Flynn, Denis Paperno, Michael Tseng, Kaeli Ward, David Wemhaner, and Craig Sailor. 2011. Notes on Q'anjob'al as spoken by Alejandra Francisco. Database.

Chomsky, Noam. 1995. *The minimalist program*. Cambridge, MA: MIT Press.

Cinque, Gugliemo. 1999. *Adverbs and functional heads: a cross-linguistic perspective*. Oxford studies in comparative syntax. Oxford: Oxford University Press.

Cinque, Gugliemo. 2010. *The syntax of adjectives: a comparative study*. Linguistic Inquiry monographs. Cambridge, MA: MIT Press.

Fowlie, Meaghan. 2013. Order and optionality: Minimalist grammars with adjunction. In *The 13th Meeting on the Mathematics of Language*, 12.

Fowlie, Meaghan. 2014. Adjuncts and minimalist grammars. In *Proceedings of Formal Grammar 2014*, ed. Glyn Morrill, Reinhard Muskens, Rainer Osswald, and Frank Richter, volume 8612 of *Lecture Notes in Computer Science*.

Frey, Werner, and Hans-Martin Gärtner. 2002. On the treatment of scrambling and adjunction in minimalist grammars. In *Proceedings of the Conference on Formal Grammar (FGTrento)*, 41–52. Trento.

Graf, Thomas. 2013a. The price of freedom: Why adjuncts are islands. *Slides of a talk given at the Deutsche Gesellschaft für Sprachwissenschaft* 12–15.

Graf, Thomas. 2013b. The price of freedom: Why adjuncts are islands. Slides of a talk given at the Deutsche Gesellschaft für Sprachwissenschaft 2013, March 12–15, University of Potsdam, Potsdam, Germany.

Graf, Thomas. 2014. Models of adjunction in minimalist grammars. In *Formal Grammar*, 52–68. Springer.

Grimshaw, Jane, and Sten Vikner. 1993. Obligatory adjuncts and the structure of events. In *Knowledge and language*, 143–155. Springer.

Harkema, Henk. 2001. A characterization of minimalist languages. In *Logical aspects of computational linguistics*, 193–211. Springer.

Huang, CTJ. 1982. Logical relations and the theory of grammar. Doctoral Dissertation, MIT.

Keenan, Edward L., and Edward P. Stabler. 2003. *Bare grammar*. Stanford: CSLI Publications.

Kobele, Gregory M., Christian Retoré, and Sylvain Salvati. 2007. An automata-theoretic approach to minimalism. In *Model Theoretic Syntax at ESSLLI '07*, ed. J. Rogers and S. Kepser. ESSLLI.

Koopman, Hilda. 2015. Violations of the final-over-final constraint (Nawdem). UCLA presentation handout.

Rizzi, Luigi. 2004. Locality and left periphery. In *Structures and beyond: The cartography of syntactic structures*, ed. Adriana Belletti, volume 3, 223–251. Oxford: Oxford University Press.

Seki, Hiroyuki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science* 88:191–229.

Sportiche, Dominique. 2005. Division of labor between merge and move: Strict locality of selection and apparent reconstruction paradoxes. *LingBuzz* .

Stabler, Edward. 1997. Derivational minimalism. *Logical Aspects of Computational Linguistics* 68–95.