Nicholas LaCara · University of Toronto 

# Bare phrase structure

- Minimalist thinking has significantly revised how phrase structure works in modern syntactic theory.
- A major result is the total revision of traditional X̄-theory and its replacement with Bare Phrase Structure (a revision of how labels are conceptualized) and Merge (which builds structure).
- Today, I will start by focusing mostly on the conceptual background behind Bare Phrase Structure (BPS) and Merge before returning to the more substantive differences between BPS and traditional X̄-theory beginning in Sections 4–7.

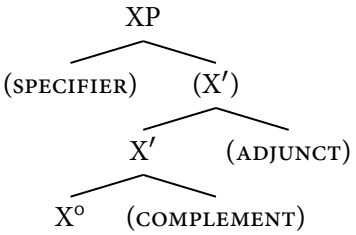
This is one of the harder chapters in the book, particularly because there are a number of difficult, unanswered questions BPS raises about labeling and adjunction. If you are confused, that is totally acceptable.

## 1 X̄-theoretical concepts

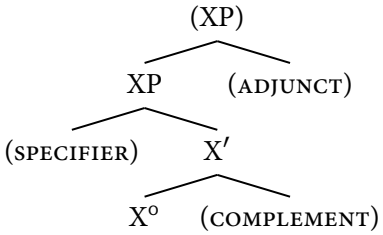
- X̄-theory has been a very successful view of how phrase structure works in UG.
- It has taken several forms over the years, but the following structures are still widely seen:

Older versions distinguished several more bar levels. See, *e.g.*, Jackendoff 1977.

(1) *Traditional X̄ representation*:



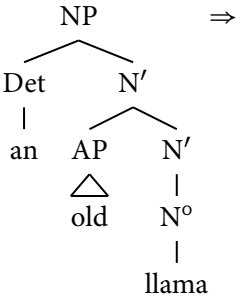

(2) *Assuming phrasal adjunction*:



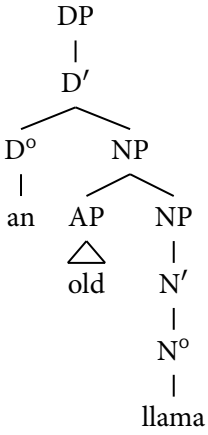

- Over time, though, the assumptions motivated this theory were chipped away.
  - The idea that functional heads in specifiers 'closed off' the phrase to further projection gave way to the idea that functional heads projected phrases.
  - It was later found that it was more natural to assume that arguments were dominated by XP and that modifiers adjoined to XP.

Basically, this is the idea that functional elements were the highest thing in a phrase.

(3)




(4)



- Nonetheless, there are still a number of vacuous projections whose presence seems to have no effect on the syntax (*i.e.*, lots of bar levels).
- Specifier and complement positions are often left empty.
- Still, X̄-structure encodes certain properties that we want any theory of phrase structure to encode.

## 1.1 Endocentricity and Periscope

- One in-built property of X̄ structure is that every phrase has a head and all heads project phrases. This is ENDOCENTRICITY.
- Simply put, it ensures that any phrase will have the same category as the head of that phrase.
- Being the head of the phrase is an important notion *prominence*:
  - Phrases contain lexical items of many categories, but only the head is relevant for syntactic processes.
- Directly related to this notion is the idea of PERISCOPE. For instance, in subject–verb agreement, the verb may agree with the head of a subject, but not some other nominal in the subject:

  (5) a. This picture of those llamas amuse*(-s) me.
  b. These pictures of that llama amuse(*-s) me.

  Notice here we have to assume that number features are shared with $D^{\circ}$. I've used demonstratives here to make this clear, but it must be true for all nominals.
- This may play a role in semantics as well. *Burritos* is a perfectly good object for *eat*, but *music* is not. However, *eat* cannot see *burritos* in (6c), only the head of its complement.

  (6) a. Sally ate the burritos.
  b. #Sally ate the song.
  c. #Sally ate the song about the burritos.
- There are no known syntactic processes that care about anything other than the head of a complement.
  - For instance, no verb seems to care if its complement has a specifier.

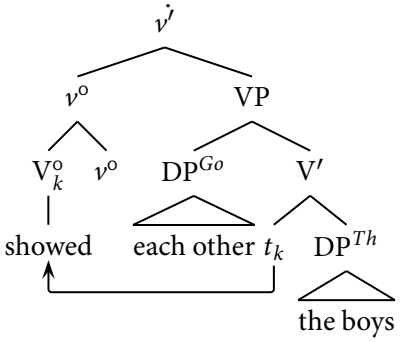
## 1.2 Binary branching

- By the mid to late 1980s, it was becoming accepted that phrase structure was BINARY BRANCHING.
- In other words, no mother has more than two branching nodes.
- We have, in fact, already studied some of the evidence for and results of this idea in the form of Larsonian shells.

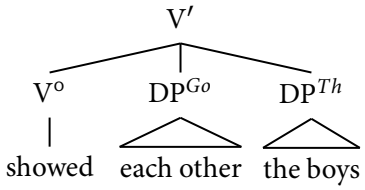
- The shell analysis of double objects was a strong argument against ternary (or any *n*-ary) branching, given that both the theme and goal arguments mutually c-command each other in a ternary branching tree.

(7) *I showed each other the boys.

a. *Th. does not c-command Go.*:




b. *Th. mut'ly c-commands Go!*:

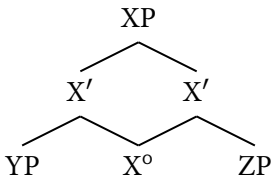



As long as *each other* can be c-commanded by *the boys*, this sentence should be grammatical. Binary branching rules out (7), but ternary branching does not.

### 1.3 Singlemotherhood

- Finally, in X̄ representations, every node has only a single mother. We don't seem to find structures like the following:

(8)




There are theories of phrase structure out there, known as MULTIDOMINANCE that seek to model certain movement dependencies as having multiple mothers. These are alternatives to the copy theory of movement, which I will cover in Section 7.

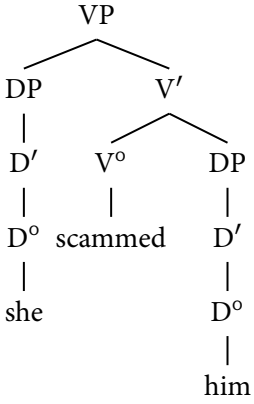
- This is binary branching and endocentric! But it does not seem to occur.

## 2 Phrase structure and endocentricity

- X̄-structures encode a lot of redundant and unnecessary information.
- Most notoriously: the proliferation of apparently vacuous X′ levels and empty specifier and complement positions:

For more detail, see Chomsky 1995, section 4.3.

(9)




The book uses the old VP structure rather than *v*P structure. I have opted not to change this, since its easier to show what is happening here, but everything said here can be straightforwardly converted to the *v*P structure.

## 2.1 Reconceptualizing projections

- How do we actually interpret the claim that a phrase consists of parts with various levels?
  - The appearance of material in specifiers, complements, and adjunct positions is mediated by other components of the grammar: The Theta Criterion, checking features, *etc.*…
  - From the perspective of phrase structure, these positions are all optional. It's not clear that there need to be separate structural levels for them in every projection.
  - This is, *e.g.*, what we've been starting to see by admitting multiple specifiers. We project as many as we need rather than some rigid number thereof.

In other words, the postulation of these levels in every phrase is theory-internal. There is no direct empirical evidence that they are always there.

- We can conceptualize many of the properties of the X̄-structure as relations between levels of a projection:

(10) *Minimal projection*:
A minimal projection is a lexical item selected from the numeration.
(Until now we labeled this X°.)

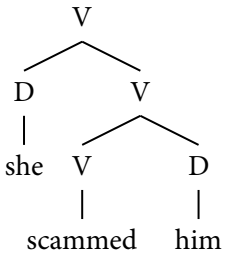
(11) *Maximal projection*:
A maximal projection is a syntactic object that doesn't project.
(Until now we labeled this XP.)

(12) *Intermediate Projection*:
An intermediate projection is a syntactic object that is neither an X° nor an XP. (Until now we labeled this X′.)

A critical point about the definitions in (10) and (11) is that they are not mutually exclusive. An element can be both minimal and maximal.

- Rather than saying that every lexical item projects the same structure regardless of the arguments and modifiers it takes, we instead *omit the intermediate projections* if they are unnecessary:

(13)




Notice here that the definitions in (10)–(12) do not obligate us to use the °, ′ and P symbols familiar from X̄-trees. The information is deduced from the definitions.

  - *She*, *scammed*, and *him* are all minimal projections (lexical items).
  - The labels dominating *him* and *her* (D) are maximal projections, as is the topmost V label (they do not project further).
  - The other projections of V are intermediate projections.

- This has a nice result: By being immediately Contained by a projection of X, a complement, a specifier, or an adjunct of X are necessarily maximal projections because they don't project further.

We need not stipulate that specifiers and complements are phrases, which is common in X̄-theory.

- This lets us encode the following:

(14) *Strong Endocentricity Thesis*:
An expression $E$ will establish a local grammatical relation (either spec–head, modification, or complementation relation) with a given head $H$ only if $E$ is immediately Contained within projections of $H$.

- But this lets us make another, more interesting shift: Being a bar-level under this conception of phrase structure is not an inherent property of a node in a tree, but rather the reflex of the position of that node with respect to others.

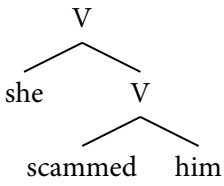
An alternative view is that the symbols °, ′, and P express some sort of non-lexical feature on each node. If we are committed to building structures only with material from the numeration (as per the INCLUSIVENESS CONDITION; see Hornstein et al. 2005: 74), it is unclear where these features could come from.

## 2.2 Labels

- We can continue this reduction to an extreme end: What purpose do category labels above lexical items actually serve?
- Presumably, the lexical items carry their own category information, so the category label does not contribute any additional information.
- We can get rid of them without actually technically losing any information:

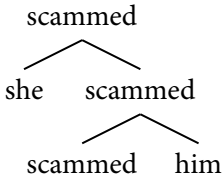
Though we do visually lose some information, we still assume that it is there.

(15)




- Once this jump is made, however, it becomes unclear whether the labeling mechanism actually needs to make *any* reference to categorial features.
- Tree (16), actually encodes the fact that *scammed him* and *she scammed him* are the same lexical category as *scammed*.

(16)




- In fact, there is some reason to want to adopt this, given that lexical items are independently necessary elements. Labels may introduce new information.
- Note, too, that this preserves – in an extreme but hopefully obvious way – the idea of endocentricity discussed above!
- This move away from labels and extraneous levels of representation is essentially BARE PHRASE STRUCTURE.
- Let's turn now to how we make these structures.

I will turn to some of the bigger conceptual problems with labeling in Section 5.

## 3 Merge

- As we've discussed several times now, MERGE is the operation responsible for building structure under minimalist conceptions of phrase structure.
- Merge combines two syntactic elements and produces a new syntactic object.
- The way it does this, though, is a bit more complicated than just combining two elements, as Merge must also determine the label of the new syntactic object.

### 3.1 Notation

- Discussions of Merge often employ a bracket notation similar to set notation.
- When two elements Merge, one of them projects a label.
- The original two elements are grouped together in curly brackets, the label appears external to this grouping:

Chomsky (1995) even remarks trees are just the informal representation of this notation!

(17) $\text{Merge}(\textit{scammed}, \textit{him}) \Rightarrow \overbrace{\{\underbrace{\textit{scammed}}_{\text{Label}}, \underbrace{\{\textit{scammed}, \textit{him}\}}_{\text{Original terms}}\}}^{\text{New syntactic object}}$

- The label is important. It distinguishes the projecting element from the non-projecting element.
- More technically, we do not want the representation of the output of Merge to be symmetrical.
  - An output like {*scammed*, *him*} does not make a distinction between which element projects and which does not.
- Notice that in (18), the information about the head is represented clearly, whereas in (19) it is not.

This is endocentricity, again.

In other words, we don't know what the head of this phrase is.

(18) $\text{Merge}(\textit{she}, \{\textit{scammed}, \{\textit{scammed}, \textit{him}\}\}) \Rightarrow \{\textit{scammed}, \{\textit{she}, \{\textit{scammed}, \{\textit{scammed}, \textit{him}\}\}\}\}$

(19) $\text{Merge}(\textit{she}, \{\textit{scammed}, \textit{him}\}) \Rightarrow \{\textit{she}, \{\textit{scammed}, \textit{him}\}\}$

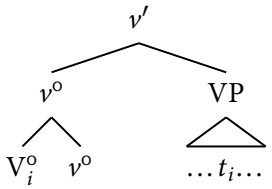
In other words, is {*she*, {*scammed*, *him*}} a VP or a DP?

- The critical thing here is that every instance of Merge must produce a label.

## 3.2 Labeling adjunction

- The right way to represent adjuncts is a controversial topic in modern Syntax, especially since they are not obviously distinguished from specifiers.
- They do have a number of distinct properties, however:
  - Adjuncts do not enter into agreement relations.
  - DP adjuncts have different Case requirements.
  - They are interpreted differently in the semantics.
  - They come in a wide array of categories.
- Furthermore, it is not clear what features – if any – are checked by adjunction.
- It's also not clear what the right syntactic relation is between adjuncts and the elements they modify. Take head-adjunction:

(20)




I've reverted back to old notation just for clarity here.

- Here, it's clear $v^{o}$ and $V^{o}$ form a constituent.
  - $V^{o}$ is not dominated by $v^{o}$, but contained by it. If it weren't, $V^{o}$ would not c-command its trace.
  - We don't want to disrupt the head–complement relation between $v^{o}$ and VP.

$V^{o}$ c-commands its trace here because the first node that dominates it is $v'$, and $v'$ dominates $t_i$.

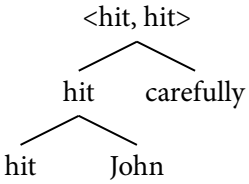
- Adjunction must not affect the label or bar-level of the thing it adjoins to – we want to ensure that adjunction only targets maximal projections. To do this, we adopt another kind of label to notate the distinction:

If it changed the label, we could not distinguish adjuncts from specifiers.

(21) hit John carefully = {<*hit*, *hit*> {{*hit*, {*hit*, *John*}} *carefully*}}

*Or*:

(22)




It is an unfortunate fact that the angle bracket notation introduced here looks a lot like the notation for unpronounced copies we will adopt below. This is something to remember.

- Importantly, though, *this is just notation*. It's not an analysis! Understanding what distinguishes adjunction from normal Merger is a very difficult question.

It's worth noting, too, that the distinction in $\bar{X}$-theory is also notational. We are not really worse off than before.

## 4 Phrase structure properties under BPS

- BPS, as we've seen, basically has endocentricity built in.
  - This may be a key result of how labeling works in this system (see below).
- Let's turn now to how BPS deals with binary branching and singlemotherhood.
  - Binary branching is a result of Merge only taking two arguments at a time. The reason for this limitation is largely to do with parsimony.
  - Given how Merge works, it is not possible for a lexical projection to project more than one mother.

Given some additional assumptions, though, it does not rule out having multiple mothers in all cases; See Section 7.4.

### 4.1 Binary branching

- The assumption is that Merge takes two terms as its elements and produces a new syntactic object.
- One of these elements – the head – projects a label which dominates the original two elements.
- The effect of this is that *all resulting trees will be binary branching*.
- This is motivated on empirical grounds.
  - For $n > 2$, there does not seem to be any evidence for $n$-ary branching trees that cannot be explained with binary branching trees.
- This is fine, but the Minimalist question asks: Why should language be like this?
- To look at this question from a Minimalist perspective, what is the simplest way to instantiate Merge that respects the 'big facts' we know about natural language?

These are the relevant big facts from Hornstein et al. 2005: 7.

$F_3$: Sentences are composed of smaller expressions.

$F_4$: These smaller units are composed into units with hierarchical structure (*i.e.*, phrases) larger than words and smaller than sentences.

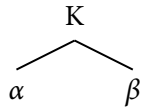
$F_6$: Language is recursive, that is, there's no upper bound on the length of sentences in any given natural language.

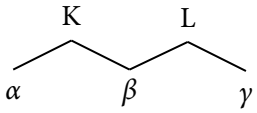
- Merge must be recursive, since it can apply both to bare lexical items and elements it has itself formed. This buys us $F_6$.
- It must be the case that Merge can combine *at least* two elements for recursion to even be possible, let alone buiding phrases, as $F_3$ and $F_4$ require.
- But is it possible that Merge takes *at most* two arguments?
  - *A priori* there is no reason it has to be limited to two arguments.
  - But we should try to make do with only the properties that are conceptually necessary to explain linguistic data.
  - Unless we find evidence for structures that can't be explained with binary branching, there is no reason to propose a more complicated operation.

Notice this is an argument about theoretical parsimony. Here the motivation is about trying to build the simplest theory possible that can explain linguistic data. This is just Occam's Razor.

### 4.2 Singlemotherhood

- Singlemotherhood falls out from the recursive nature of Merge and the Extension Condition.
- Suppose you generate K by merging $\alpha$ and $\beta$, and then try to merge $\gamma$ with $\beta$ to form L:

(23)



(24)



- The Extension Condition requires that Merge target root syntactic objects. Once K is formed, its constituents can no longer be taken as arguments of Merge.
- This, too, may be a reflex of simplicity in the computational system.
  - It reduces the computational complexity greatly if only root syntactic objects can be considered by Merge.

As an exercise, try to represent (24) in the set-like notation. It's not totally clear to me that this is even possible.

## 5 Labeling and Merge

- One issue with the adoption of Merge is that it becomes harder to understand why certain heads project and not others.
- Under $\bar{\text{X}}$ theory, every head projected and arguments were just placed in complement or specifier positions as necessary.
- Part of how we understand this in terms of Merge is based on requirements of the objects that the operation takes as its arguments.
- However, there are difficult and unanswered questions floating around in this area, including whether labels violate the Inclusiveness Condition.

Notice that this does not even let us ask the question about what projects – everything does, so the question does not come up.

### 5.1 How do we determine the label?

- Recall from last time that the operation Merge combines two elements and assigns it a label. Conventionally, this label is often just *K*:

(25) $\text{Merge}(\alpha, \beta) \Rightarrow \{K, \{\alpha, \beta\}\}$

I don't know why K is always the name for an arbitrary label, but it's what people use.

- The question here is how we determine the identity of *K* from terms $\alpha$ and $\beta$. .
- Let's start by assuming the Strong Endocentricty Thesis and the Extension Condition.

Put another way, how do we know whether $\alpha$ or $\beta$ will be the head of *K*?

(26) *Strong Endocentricity Thesis*:
An expression *E* will establish a local grammatical relation (either spec–head, modification, or complementation relation) with a given head *H* only if *E* is immediately contained within projections of *H*.

Repeated from (14).

(27) *Extension Condition*:
Overt applications of the operations Merge and Move can only target root syntactic objects.

- (26) requires local grammatical relations with a head X to be established under a projection of X.
  - This means that any $\theta$-role assignment, feature checking, or selection induced by X must happen within XP.
- The Extension Condition requires Merging new material with projections of X by targeting root syntactic objects.
- Let's also assume the economy constraint Last Resort, which precludes unnecessary or superfluous steps in the derivation.

See Hornstein et al. 2005: 293.

- As I discussed in Section 3.1, simply combining two syntactic elements – say a preposition *to* and a pronoun *him* – without a label is not sufficient.
  - The resulting syntactic object {*to*, *him*} does not distinguish what category the resulting element is.
  - No operation would be able to interact with this label-less object (due to its lack of category) and this would violate Last Resort.

This step would be superfluous because it creates an object with which nothing else could interact.

- So one of the terms must project. But why should it be the preposition?
  - A speculative idea is that this is due to the fact that *to* has the information that it requires a complement.
  - *Him* on the other hand does not take a complement but is the right kind of element to combine with a preposition.
  - Consequently, the preposition gets to be the head.

Can you think of a case where each element could select the other? This might break this idea.

- Since the preposition takes *him* as its argument, it projects the label, not *him*.

(28) Merge(*him*, *to*) ⇒ {*to*, {*to*, *him*}}

This is, in some sense, not dissimilar from what happens in X̄-theory. There, an element gets to be a specifier or complement if it is an argument of the head. That is essentially what is motivating the idea here, except now we apply it to what element gets to project.

- If the core of this idea is correct, then it may be the case that a head projects as many times as it needs to until all its requirements are met.
  - A head will project until all of its argument positions are satiated, all of its features are checked, and all appropriate modifiers have been added.
- We will continue to use the set-theory-like notation, but it is important to remember that this is just notation.
  - We could just as easily mark the head/label with a big ★ or circle it or anything.
  - Though we can write this any way we choose, the issue is still important. It is just not clear how exactly labeling ought to be implemented technically.

## 5.2 Other conceptual issues

- Besides the issues with how a label is determined, there are conceptual questions having to do with what labels even are.
- Depending on how one formulates the Inclusiveness Condition, labels might count as adding features that are not in the numeration.
  - The Inclusiveness Condition requires that LF objects must be built only from the features of the lexical items in the numeration.
  - Adding notational elements such as °, ′, and P to labels may be an issue for the Inclusiveness Condition if those elements are thought of as non-lexical features.
  - But even the idea of an element like a label could be problematic, since the lexical items are not actually labels themselves.

See Hornstein et al. 2005: 74.

And these would have to be non-lexical features, since they are not properties of the lexical items but properties of the relative positions of the labels themselves.

- Additionally, the information encoded by a label is determined by the head of a phrase, and being a head is largely a function of the local grammatical relation being established.
  - In other words, it's not clear that labels are necessary, since the information they encode is derived elsewhere by the derivation.
  - Given the big facts about phrase structure noted above, it's not clear that labels are conceptually necessary.

Why do derived syntactic units need to have heads?

- Even if the content of a label can be independently determined, it might still be argued that labels are required in the system as optimal design features.
- Consider the derivation of the VP in (29) and (30).

(29) Merge(*it*, *show*) ⇒ {*show*, {*show it*}}

(30) Merge(*him*, {*show*, {*show it*}}) ⇒ {*show*, {*him*,{*show*, {*show it*}}}}

  - Merge need only look at the label of the output of (29) to determine whether Merge can combine *him* with {*show*, {*show it*}}.
  - There is no need to look inside {*show it*} to figure out what the head is.
- That is, without labels, things would be less efficient:
  - Without labels, how would the system know that *him* may enter into a local relation with {*show it*} in (30)?
  - The system could backtrack to see whether any of the previously merged elements were of the appropriate type and see if the right element is the head. This seems fairly inefficient.
  - Additionally, what kind of locality might exist between a specifier and a head if not for a label, given that a specifier and a head do not form a constituent if a complement is present.

*Him* is a specifier in (30). Again, this is a question about endocentricity.

- If this is on the right track, labeling not only allows head-to-head relations to be locally stated (via Periscope), but also makes it possible to locally state several grammatical relations to the head.

## 6 Important differences between BPS and X̄-theory

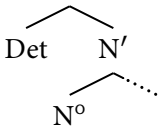
- In this section, we'll summarize some key differences between BPS and X̄-theory.
- The big differences are that BPS allows for multiple specifiers, allows for simultaneously minimal and maximal elements in specifier and complement positions, and eliminates vacuous projections.

### 6.1 Multiple specifiers

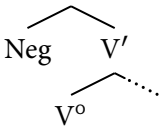
- In X̄-theory, it was largely assumed that phrases only had one specifier.
- In classic X̄-theory, Det occurs in SpecNP, and Neg is sometimes said to be in SpecVP.

Once the internal subject hypothesis was adopted, Neg got sent to its own projection.

(31)




(32)

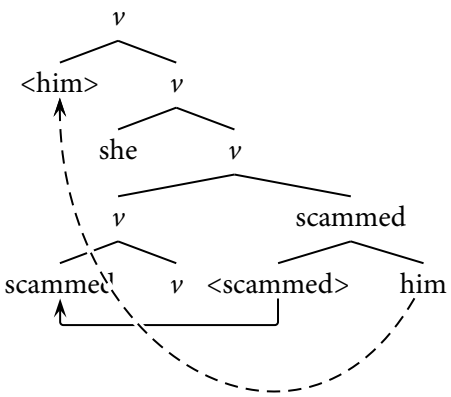



- Once an NP has a determiner, it seems to stop projecting. It looks as if adding a determiner to NP ends the NP, and adding negation ends the VP.
- If this were correct, then it would be impossible for a phrase to have more than one specifier and so it was concluded that phrases could only have one specifier.
  - But the logic underpinning this argument shifted once it was shown that functional heads projected their own phrases.
  - However, the assumed limit on the number of specifiers held on longer than the evidence that supported it.

For DPs, see Abney 1987.

- Nothing about bare phrase structure appears to limit the number of specifiers that occur in a phrase.
- As the discussion on labeling in Section 5.1 suggests, the phrase may simply keep projecting until all of its requirements are met.
  - This requires multiple specifiers to at least be a possibility.
  - We anticipated this for Case checking and minimality conditions in *v*Ps!

(33)




Here, both *him* and *she* are specifiers in *v*. *v* must take a complement, it must introduce the external argument, and it must check the Case feature on *him*.

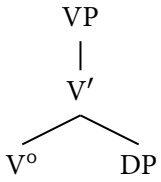
The <brackets> around things here mean that they are unpronounced copies – the equivalent of traces. You'll see in a sec.

- An interesting question to ask yourself at this point: What is the difference between a specifier and a complement in this system?

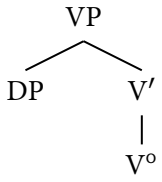
## 6.2 Vacuous projections

- One of the most outwardly noticeable differences between BPS and $\bar{X}$-theory is the elimination of vacuous X′ projections – intermediate projections with only a single daughter.
- It was thought that vacuous projections could explain the difference between unaccusative and unergative verbs in $\bar{X}$-terms.
  - Internal arguments were the complements of the verb, while external arguments were the specifer of the verb.
  - As such, these two kinds of intransitives could be structurally distinguished by the presence of V′:

(34) *Unaccusative*:



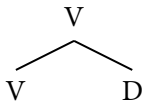

(35) *Unergative*:



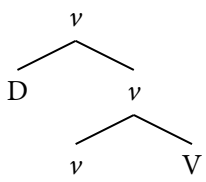

- Now that we have *v*°, though, we don't need vacuous projections to make this distinction:

So again, a shift in our assumptions makes the above proposal obsolete, and the underlying structure no longer receives independent justification.

(36) *Unaccusative*:




(37) *Unergative*:




- As mentioned above, most of the remaining reasoning for proposing vacuous X′ projections in every phrase was theory-internal.
- Because of the way Merge works, vacuous projections cannot be created.
  - A new label will only be projected when two elements merge.
  - Since Merge must combine exactly two elements, there will be no label projected that does not have two daughters.

## 6.3 Phrasehood of arguments and adjuncts

- As I discussed earlier, $\bar{X}$-theory requires that complements, adjuncts, (in later versions) specifiers be phrases.

This follows if all heads project phrases in $\bar{X}$-theory.

- Under BPS, elements in these positions need not be phrasal.
- The assumptions about how projection works will, however, cause these elements to be maximal projections.

  - Merging an element X in the complement, specifier, or adjunct position of Y stops further projection of X. There are two reasons for this.
    i. The Strong Endocentricity Thesis (26) will ensure that all requirements of X are met in a projection of X.
    ii. The Extension Condition (27) will prevent that element from merging with any other material once it merges with Y.
  - As a result, whatever elements merge in these positions will necessarily be maximal projections (roughly equivalent to a phrase in $\bar{X}$-theory).

In other words, phrasehood is not actually relevant here.

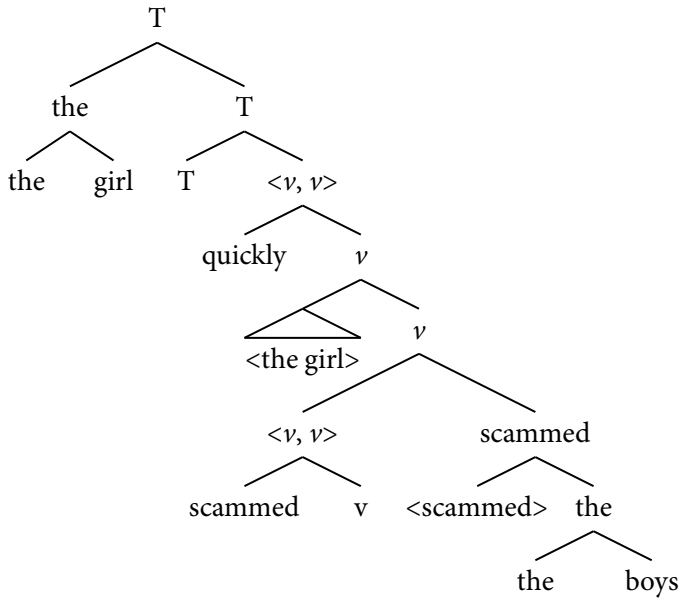
- Because a lexical item that does not project is at once minimal and maximal, a bare lexical item can occupy a specifier, complement or adjunct position.

## 7 The copy theory of movement

- The operation Move, under standard Minimalist assumptions, is broken down into two sub-components: Copy and Merge.
- Merge, we just discussed, but Copy is new. Under this view movement isn't really movement at all.
- To 'move' something under the Copy Theory of Movement, we find the element that we want to move and create a copy of it.
- We then take that copy and Merge it in the position where it needs to go.
- We leave an (unpronounced) copy in the position of the original element, surrounded by <angle brackets>.

(38) The girl quickly scammed the boys.

As mentioned above, be careful not to confuse the adjunction notation and the unpronounced copy notation.



## 7.1 Why would you do this?

- There are several reasons to want to adopt this alternative under the theory of Merge sketched above.
- Most notably, because of the way Merge works, there are no empty specifier positions into which phrases can Move.
  - Under X̄-theory, any phrase in principle had a specifier position, and when empty these could be targeted by movement.
  - Here, though, specifiers do not exist until you merge a phrase with a projection of the target phrase!
  - So we want to link Move directly to Merge somehow.
- A second reason was that the introduction of traces violates the Inclusiveness Condition, since traces are not elements of the numeration.

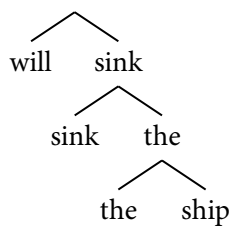
  Traces are not even lexical items.
  - The inclusiveness condition requires that only lexical items in the numeration be used in the derivation.
  - We still want something like a trace to be left behind by movement, though, since there is some evidence that some element stands in for a displaced phrase at LF.

    See Fox 1999.
- To deal with these issues, we propose that a 'moved' element is actually a copy of some other element.
  - Since lexical items are in the numeration, unlike traces, we are not introducing elements that are not in the numeration, satisfying the Inclusiveness Condition.
  - Producing a copy of an element that is already in the tree creates a new element that can be an argument of Merge, meaning we do not need to target an empty position.

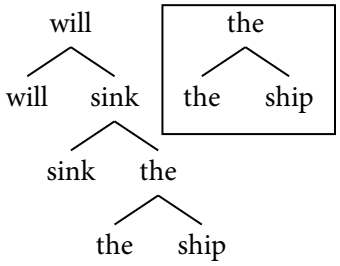
## 7.2 How do you do this?

- First, build the tree from the numeration, as in (39). Copy the item that you want to 'move' (40), then merge the new copy at the root of the tree (41).

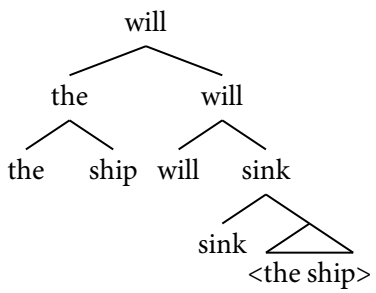
(39)




(40)




(41)



- The new copy here behaves like an element from the numeration. It merges at the root, just like any other element that Merges with the tree.
- Once this view is adopted, additional work has to be done at LF so that the semantics can interpret the unpronounced copies correctly. See Fox (1999).
- Additionally, we need to ensure that the original copy is not (always) pronounced. See, *e.g.*, Nunes 2004.

I put the copy in a box above, but there is no standard way of representing this.

## 7.3 Why should I believe it?

- There is some empirical evidence that the Copy Theory of movement is on the right track.
- One of the most famous cases is *wh*-copying in Afrikaans. In (42) , a copy of the *wh*-element *met wie* 'with whom' is left at every SpecCP, consistent with are view that *wh*-movement is successive cyclic.

See du Plessis 1977.

(42) *Met wie* het jy now weer gesê *met wie* het Sarie gedog *met wie*
*with who* did you now again said *with who* did Sarie thought *with who*
gaan Jan trou?
go Jan marry
'Whom did you say (again) that Sarie thought Jan is going to marry?'

- This makes it look like *wh*-movement doesn't leave traces at CP boundaries, but *copies* that get pronounced in this language.
- Another example that is analyzed as copying involves VP movement in languages with verb movement. In (43) from Brazilian Portuguese,

This happens in many languages, including Hebrew, Korean, Portuguese, Russian, and Spanish.

(43) [lavar o carro] o João lavou.
wash-INF the car the João wash-PST.3SG
'As for washing the car, João washed (it).'

Bastos 2001

- The general view is that the verb root gets copied twice in these sorts of sentences, once when the verb is moved to T$^{o}$, and once when the VP is topicalized to SpecCP:

We verb root usually has to match, which is why we think the verb root is getting copied.

(44) $[_{CP}\ \underbrace{[_{VP}\ V^{o}_{3}\ DP\ ]}\ C^{o}\ [_{TP}\ V^{o}_{2}{+}T^{o}\ \underbrace{[_{VP}\ V^{o}_{1}\ DP\ ]}\ ]\ ]$

- Additionally, it lets us get a handle on how binding is taken care of when things like reflexives move:

(45) Which picture of himself did John say Bill liked?

- At LF, it becomes possible to interpret different copies of the reflexive. This allows *himself* to be bound by *John*:

This is sort of like covert Afrikaans.

(46) $[_{CP}$ [which pic. of himself] did John say $[_{CP}$ [which pic. of himself] Bill liked [which pic. of himself]]]

## 7.4 The alternative? Multidominance/Remerge

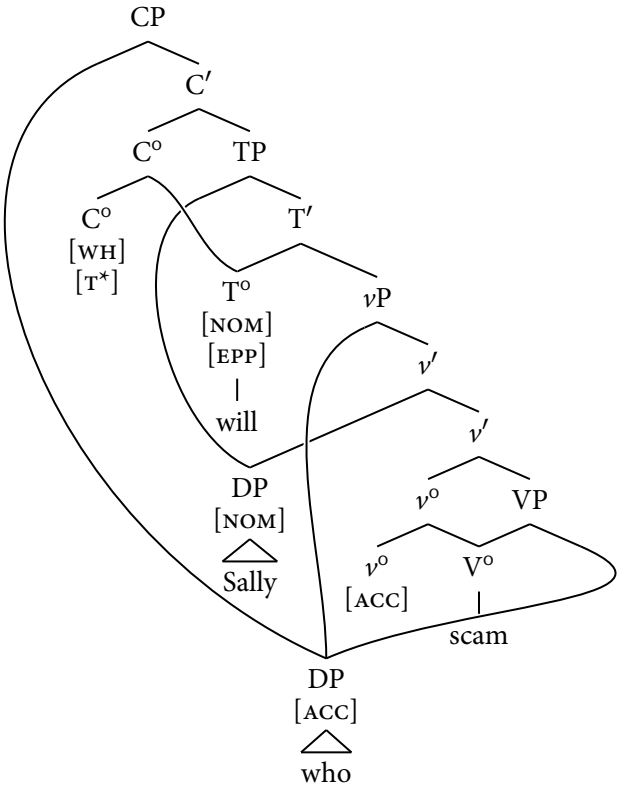
- There is, however, some concern that introducing copies also violates the Inclusiveness Condition, since copies are not strictly elements of the numeration.
- If we want to avoid generating copies as part of our displacement operation, what can we do?
- If we abandon the idea of singlemotherhood, we can allow MULTIDOMINANCE in our trees.
- Rather than moving or copying an element, we say that an element that has already been merged in the tree is *remerged* in another position, causing it to have multiple mothers:

Specifically, we assume that Merge actually can take terms that have been merged previously. It's not totally clear that this can be blocked for any principled reason; compare with the claim in Section 4.2.

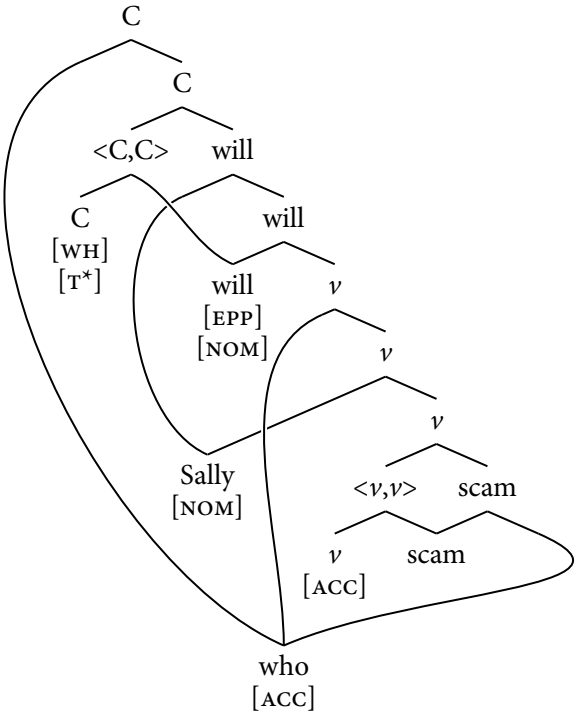
As with the Copy Theory of Movement, we will need to find ways both to interpret these structures in the semantics as well as linearize them at PF. In particular, we need a system that tells us where to pronounced remerged elements.

(47) Who will Sally scam?

a. *With category labels*:



b. *Bare phrase structure!*:




## References


Abney, Steven Paul. 1987. The English Noun Phrase in its Sentential Aspect. Doctoral Dissertation, Massachusetts Institute of Technology.

Bastos, Ana Cláudia Pinto. 2001. Fazer, eu faço! Topicalização de constituintes verbais em português brasileiro. Master's thesis, Universidade Estadual de Campinas.

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

du Plessis, Hans. 1977. *Wh* Movement in Afrikaans. *Linguistic Inquiry* 8:723–726.

Fox, Danny. 1999. Reconstruction, Binding Theory, and the Interpretation of Chains. *Linguistic Inquiry* 30:157–196.

Hornstein, Norbert, Jairo Nunes, and Grohmann. 2005. *Understanding Minimalism*. Cambridge University Press.

Jackendoff, Ray. 1977. *X̄ Syntax: A Study of Phrase Structure*. Linguistic Inquiry Monographs. Cambridge, Mass.: MIT Press.

Larson, Richard K. 1988. On the Double Object Construction. *Linguistic Inquiry* 19:335–391.

Nunes, Jairo. 2004. *Linearization of Chains and Sideward Movement*. Number 43 in Linguistic Inquiry Monographs. Cambridge, Mass: MIT Press.