CAS LX 522
Syntax I

do-support, subjects, agreement, and case
(5.5; 6.1-6.3)

11

French vs. English

- In English, adverbs cannot come between the verb and the object.

  1) *Pat eats often apples.

  2) Pat often eats apples.

- In French it's the other way around.

  3) Jean mange souvent des pommes.
  Jean eats often of.the apples
  'Jean often eats apples.'

  4) *Jean souvent mange des pommes.

- If we suppose that the basic structures are the same, why might that be?

French vs. English

- Similarly, while only auxiliaries in English show up before negation (*not*)…
  - John does **not love** Mary.
  - John **has not** eaten apples.
- …all verbs seem to show up before negation (*pas*) in French:
  - Jean (n')**aime pas** Marie.
  Jean (ne) loves not Marie
  'Jean doesn't love Marie.'
  - Jean (n')**a pas** mangé des pommes.
  Jean (ne)has not eaten of.the apples
  'Jean didn't eat apples.'

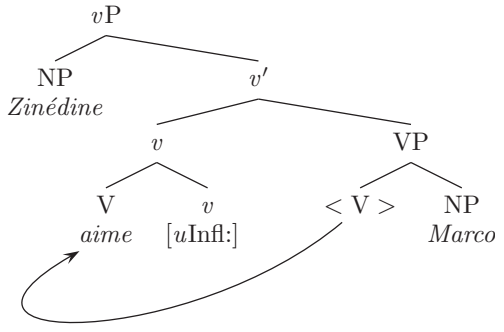
V raises to T in French

- What it looks like is that both V and auxiliaries raise to T in French.
- This is a **parametric difference** between English and French.
- A kid's task is to determine whether V moves to T and whether auxiliaries move to T.

	T values [*u*Infl:] on Aux	T values [*u*Infl:] on *v*
English	Strong	Weak
French	Strong	Strong

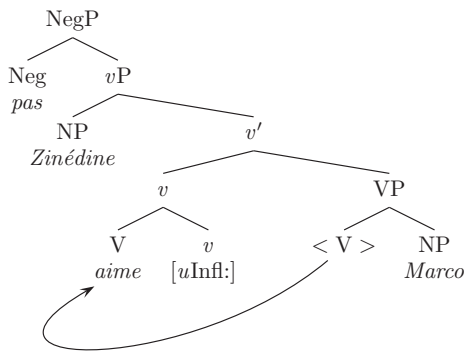
Zinédine (n') aime pas Marco

- First, build the *v*P just as in English.
  - Merge *aime* and *Marco* to form the VP, Merge *v* and VP to satisfy the HoP, move V to Adjoin to *v* to check *v*'s [*u*V*] feature, Merge *Zinédine* and *v'*.



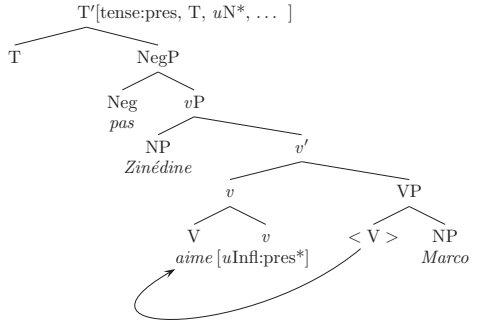
Zinédine (n') aime pas Marco

- Merge Neg with *v*P to form NegP (following the HoP).



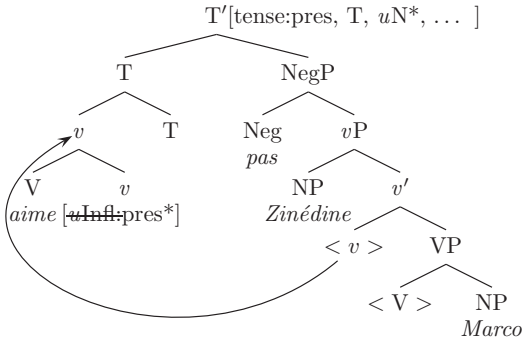
Zinédine (n') aime pas Marco

- Merge T with NegP to form T′ (again, following the HoP).
- Now T with its [tense:pres] feature c-commands *v* and its [*u*Infl:] feature. They Match. **But** in French, when [*u*Infl:] on *v* is valued by T it is **strong**. So…



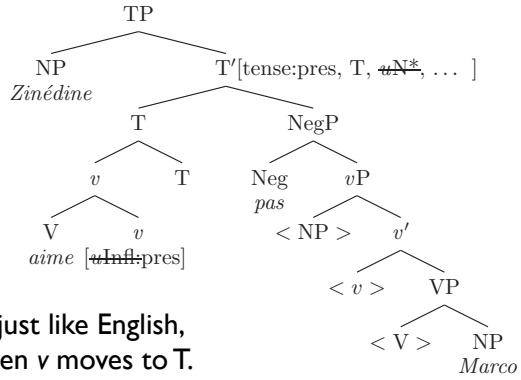
Zinédine (n') aime pas Marco

- *v* has to move to T. Notice that at this point *v* has V adjoined to it. **You can't take them apart**. The whole **complex head** moves to T.



Zinédine (n') aime pas Marco

- And then, we move the subject up to SpecTP to check the final uninterpretable (strong) feature of T, [*u*N*].



So, French is just like English, except that even *v* moves to T.

Swedish

- Looking at Swedish, we can see that not only do languages vary on whether they raise main verbs to T, languages also vary on whether they raise auxiliaries to T:
  - …om hon **inte köpte** boken
    whether she not bought book-the
    '…whether she didn't buy the book.'
  - …om hon **inte har** köpt boken
    whether she not has bought book-the
    '…whether she hasn't bought the book.'
- So both parameters can vary.
  - Remember the light box: By saying these were parameters, we predicted that we would find these languages.

Typology of verb/aux raising

- Interestingly, there don't seem to be languages that raise main verbs but not auxiliaries.
  - This double-binary distinction predicts there would be.
  - It overgenerates a smidge.
- This is a pattern that we would like to explain someday, another mystery about Aux to file away.
  - Sorry, we won't have any satisfying explanation for this gap this semester.

	T values [*u*Infl:] on Aux	T values [*u*Infl:] on *v*
English	Strong	Weak
French	Strong	Strong
Swedish	Weak	Weak
Unattested	Weak	Strong

Irish

- In Irish, the basic word order is VSO (other languages have this property too, e.g., Arabic)

  1) Phóg Máire an lucharachán.
     kissed Mary the leprechaun
     'Mary kissed the leprechaun.'
- We distinguish SVO from SOV by supposing that the head-complement order can vary from language to language (heads precede complements in English, heads follow complements in Japanese).
- We may also be able to distinguish other languages (OVS, VOS) by a parameter of specifier order.
- But *no* combination of these two parameters can give us VSO.

Irish

- But look at auxiliary verbs in Irish:

  1) Tá Máire ag-pógáil an lucharachán.
     is Mary ing-kiss the leprechaun
     'Mary is kissing the leprechaun.'

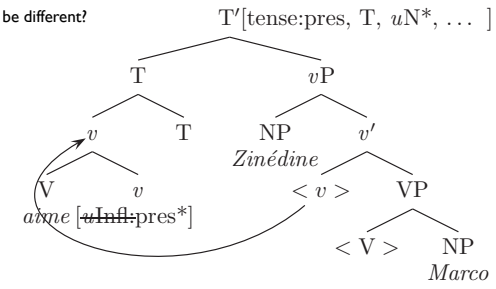
- We find that if an *auxiliary* occupies the verb slot at the beginning of the sentence, the main verb appears between the subject and verb: **Aux S V O**.
- What does this suggest about
  - The head-parameter setting in Irish?
  - How VSO order arises?

SVO to VSO

- Irish appears to be essentially an SVO language, like French.
- Verbs and auxiliaries raise past the subject to yield VSO.
- We can analyze the Irish pattern as being minimally different from our existing analysis of French— just one difference, which we hypothesize is another parametric difference between languages.
- V and Aux both raise to T (when tense values the [*u*Infl:] feature of either one, [*u*Infl:] is strong) in Irish, just as in French.

French vs. Irish

- Remember this step in the French derivation before? (I've omitted negation to make it simpler.)
- **What if we stopped here?**
  - In French it would crash (why?).
  - But what if it didn't crash in Irish?
  - What would have to be different?



Parametric differences

- We could analyze Irish as being just like French except without the strong [*u*N*] feature on T.
  - Without that feature, the subject doesn't need to move to SpecTP. The order would be VSO, or AuxSVO.
- So, languages can vary in, at least:
  - Head-complement order
  - (Head-specifier order)
  - Whether [*u*Infl:] on Aux is strong or weak when valued by T
  - Whether [*u*Infl:] on *v* is strong or weak when valued by T
  - Whether T has a [*u*N*] feature or not. (Later, when we look at German, we'll suggest a different analysis of Irish, but this will work for now.)

do-support

- In French, verbs move to T. In English, they *don't* move to T.
  - That's because in French, when [tense:past] values [*u*Infl:] on *v*, it is strong, and in English, it is weak.
- What this *doesn't* explain is why *do* appears sometimes in English, seemingly doing nothing but carrying the tense (and subject agreement).

  - The environments are complicated:
    1) Tom **did** not **commit** the crime.
    2) Tom did not commit the crime, but someone **did**.
    3) Zoe and Danny vowed to prove Tom innocent, and prove Tom innocent they **did**.
    4) Tom (has) never **committed** that crime.

do-support

- The environments are complicated:
  1) Tom **did** not **commit** the crime.
  2) Tom did not commit the crime, but someone **did**.
  3) Zoe and Danny vowed to prove Tom innocent, and prove Tom innocent they **did**.
  4) Tom (has) never **committed** that crime.
- When *not* separates T and *v*, *do* appears in T to carry the tense morphology.
- When T is stranded due to VP ellipsis or VP fronting, *do* appears in T to carry the tense morphology.
- When *never* (or any adverb) separates T and *v*, tense morphology appears on the verb (*v*).
- So, *do* appears when T is separated from the verb, but adverbs like *never* aren't "visible", they aren't in the way.

Technical difficulties

- How do we generally know to pronounce V+*v* as a past tense verb?
  - T values the [*u*Infl:] feature of *v*. The presumption is that e*at* +*v*[*u*Infl:past] sounds like "ate." And T doesn't sound like anything.
  - But this happens whether or not *v* is right next to T. *v* still has a [*u*Infl:] feature that has to be checked.
  - So, the questions are, how do we:
    - Keep from pronouncing the verb based on *v*'s [*u*Infl:] feature if T isn't right next to it?
    - Keep from pronouncing *do* at T if *v* is right next to it?
  - We need to connect T and *v* somehow.

Technical difficulties

- The connection between T and *v* is that (when there are no auxiliaries), T values the [*u*Infl:] feature of *v*.
- This sets up a relationship between the two heads.
  - Adger calls this relationship a **chain**.
- We want to ensure that tense features are pronounced in exactly one place in this chain.
  - If the ends of the chain are not close enough together, tense is pronounced on T (as *do*). If they are close enough together, tense is pronounced on *v*+V.

Technical difficulties

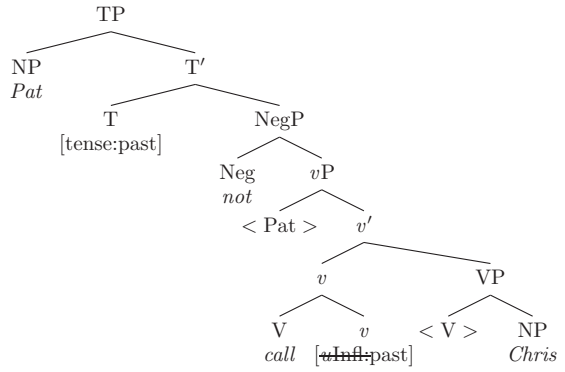
- Let's be creative: Suppose that the tense features on *v* (the value of the [*u*Infl:] feature) "refer back" to the tense features on T.
  - Agree can see relatively far (so T can value the [*u*Infl:] feature of *v*, even if it has to look past negation).
  - But "referring back" is more limited, basically only available to features that are sisters. Negation will get in the way for this.
  - So if you try to pronounce tense on *v* but T is too far away, the back-reference fails, and *v* is pronounced as a bare verb. But the tense features have to be pronounced somewhere, so they're pronounced on T (as *do*).

PTR

- Adger's proposal:
  - **Pronouncing Tense Rule (PTR)**
    In a chain (T[tense], *v*[*u*Infl:tense]), pronounce the tense features on *v* only if *v* is the head of T's sister.
- NegP, if there, will be the sister of T (HoP), but Neg has no [*u*Infl:] feature. *do* will be inserted.
- Adverbs adjoin to *v*P, resulting in a *v*P. *v* has a [*u*Infl:] valued by T and adverbs don't get in the way of *v*P being the sister of T. Tense is pronounced on the verb (*v*).
- If *v*P is gone altogether, *do* is inserted.

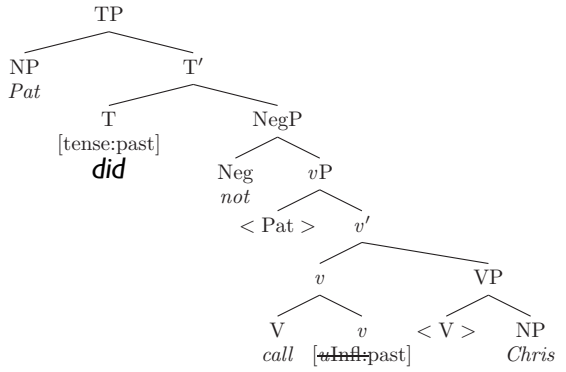
Pat did not call Chris

- So, here, T and *v* form a chain because [tense:past] valued [~~*u*Infl~~:past]. But *v* is not the head of T's sister.



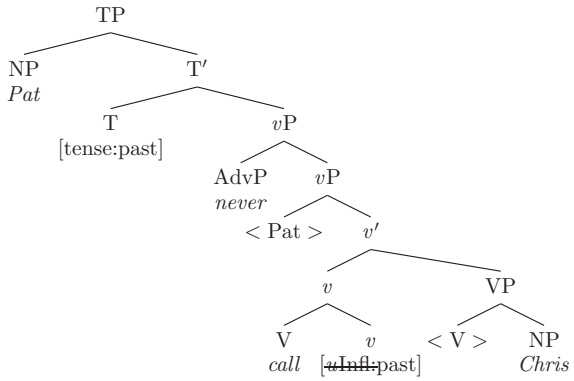
Pat did not call Chris

- *Do*-support comes to the rescue. What this means is just that T is **pronounced** as *do* with the tense specifications on T. According to PTR, we don't pronounce them on *v*. **The tree doesn't change**.



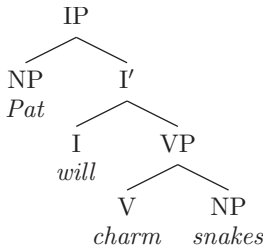
Pat never called Chris

- If there is an adverb like *never*, PTR still allows tense to be pronounced on *v* (so T doesn't have any pronunciation of its own at all).



Historical interlude

- Back in the days of yore, people hypothesized that *Pat will charm snakes* had a structure like this.
  - The subject NP *Pat* was in the specifier of "IP" (what we call "TP"), and the VP contained only the verb *charm* and the object NP *snakes*.
- *Pat* got an Agent θ-role by being in SpecIP, even though the fact that there is an Agent θ-role to be had is determined by the verb down in the VP.

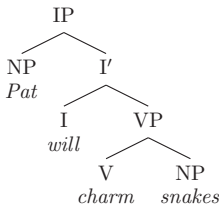


The students will all...

- This predicts the normal word order pretty well, and so it was hypothesized that the verb simply assigned one of its θ-roles directly to SpecIP.
  - No big deal, syntax works in strange and mysterious ways.
- At a certain point, someone started thinking about sentences like these:

  1) All the students will take the exam.
  2) The students will all take the exam.

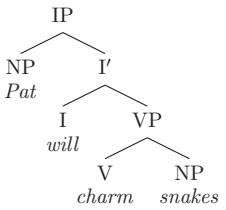
- It's fairly clear here that *all the students* is an NP, that it forms a coherent unit, a coherent concept. *All* really belongs with *the students*.



Floating quantifiers

1) All the students will take the exam.
2) The students will all take the exam.

- Back in the even older days, the hypothesis was that there was a special rule that turned the first sentence into the second.
- The **Quantifier Float** rule would move *all* over to the right, next to the VP.
  - *all* NP … VP → NP … *all* + VP



Only some quantifiers float

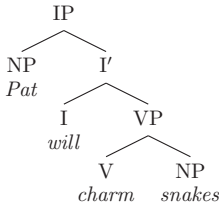
- Quantifiers: *every*, *some*, *all*, *most*, *several*, *many*, *both*, *four*, …

  1) Every student will take the exam.
  2) *Student will every take the exam.
  3) Several students will take the exam.
  4) *Students will several take the exam.

- It works for *both* and *all*:

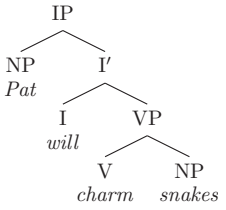
  5) The students will both take the exam.
  6) The students will all take the exam.

- What's a difference between *every*, *some*, *several*, *many*, and *both*, *all*?



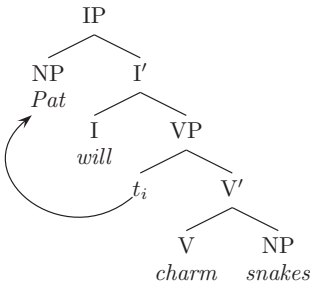
Leaving all behind

- Upon further reflection, some enterprising syntacticians hit upon the idea that rather than floating *all* to its position next to VP, *all* might instead have been "left behind" by a subject that had moved.
  - will [all [the students]] take the exam.
  - [all [the students]]$_i$ will t_i take the exam.
  - [the students]$_i$ will [all t_i] take the exam.
- And why would all the students have been down there? Well, that would simplify assignment of θ-roles.



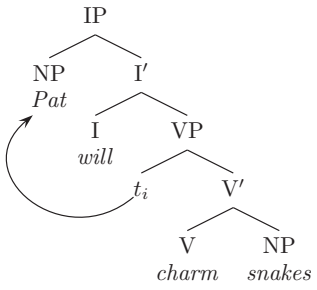
The VP-Internal Subject Hypothesis

- The verb (head of VP) can assign θ-roles to other things within the VP, which is a natural explanation for how the choice of verb controls whether an Agent θ-role is assigned or not.
- This idea became known as the **VP-Internal Subject Hypothesis**.



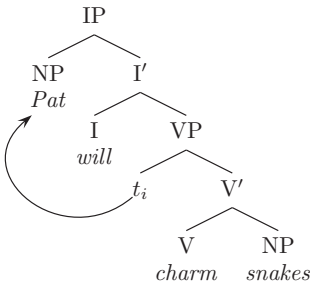
The VP-Internal Subject Hypothesis

- For us, we've supposed from the beginning that assignment of θ-roles is necessarily local. This may not seem like a very surprising hypothesis.
- But it was at the time a rather unintuitive idea, and so various people set out to see if some of the predictions this makes are borne out in the grammatical data.



The VP-Internal Subject Hypothesis

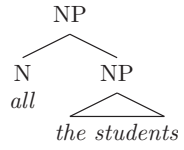
- It turns out that as people looked, there were reasons to believe this.
  - The new analysis of Quantifier Float no longer relies on an idiosyncratic rule of English, but more general principles.
  - The assignment of θ-roles can now be more directly related to the properties of the verb.
  - And we can make sense of *there* constructions in a more straightforward way.



Back to the present

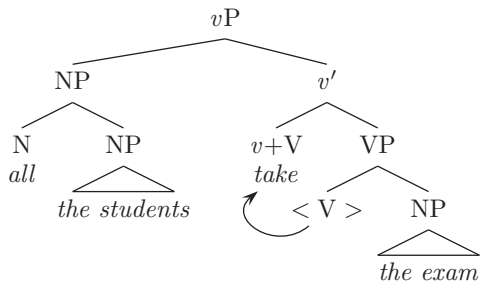
- The basic components of the quantifier "stranding" phenomenon are:
- *All the students* is a constituent. *The students* is an NP inside *all the students*.
  - [all [$_{NP}$ the students]]
- Either *all the students* or just *the students* can move to SpecTP, to satisfy the [*u*N*] feature of T.
- So *all the students* and *the students* are both NPs.
  - [$_{NP}$ all [$_{NP}$ the students]]
- So *all* is essentially a noun, but one that takes an NP complement (*all*: [N, *u*N*, …]).
  - We're assuming here that *all* is not an adjunct, but in fact a head, taking the NP as a complement. Why?

Quantifier stranding is still often referred to as "quantifier float" to this day, even though the name no longer reflects the analysis.



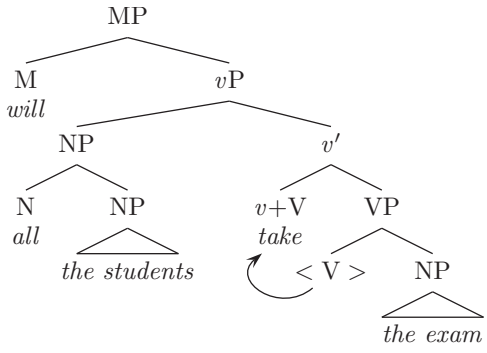
All the students will take...

- We start by building our *v*P.
  - Merge the NP *the exam* and the V *take* (checks [*u*N*] on V)
  - Merge *v* and VP (HoP)
  - Move V to *v* (checks [*u*V*] on *v*)
  - Merge the N *all* and the NP *the students* (checks [*u*N*] on *all*)



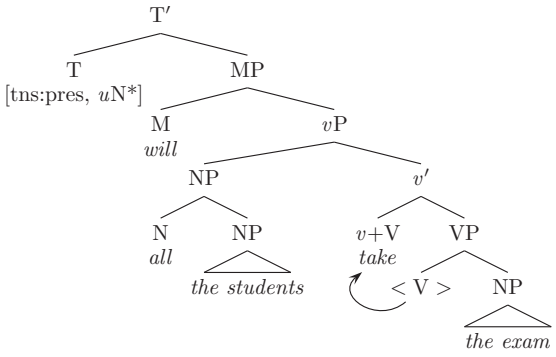
All the students will take...

- We Merge the M *will* with *v*P (HoP)
  - This values [*u*Infl:] on *v* as [~~*u*Infl:~~M].



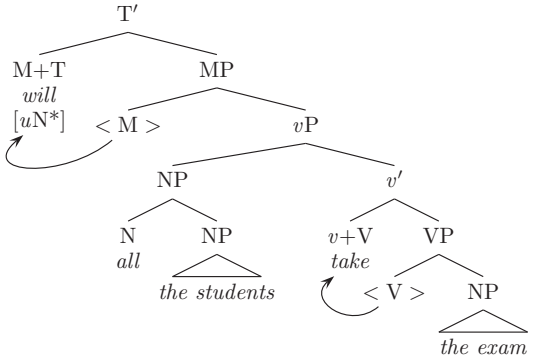
All the students will take...

- We Merge the T with MP (HoP)
  - This values [*u*Infl:] on M as [*u*Infl:pres*] (strong).



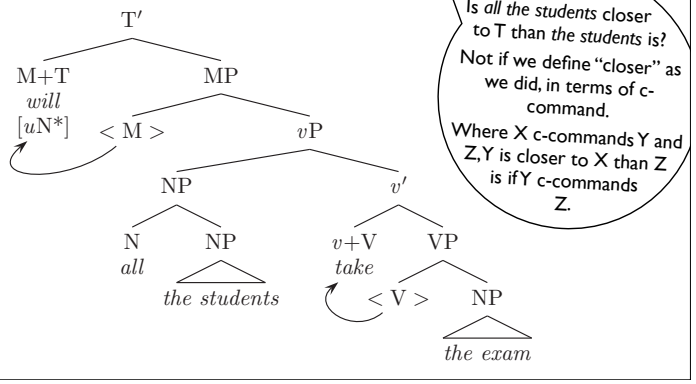
All the students will take...

- We move M up to T
  - This checks the strong [*u*Infl:pres*] on M.



All the students will take...

- Now, there are two possibilities:
  a) Move the NP *all the students*.
  b) Move the NP *the students*.



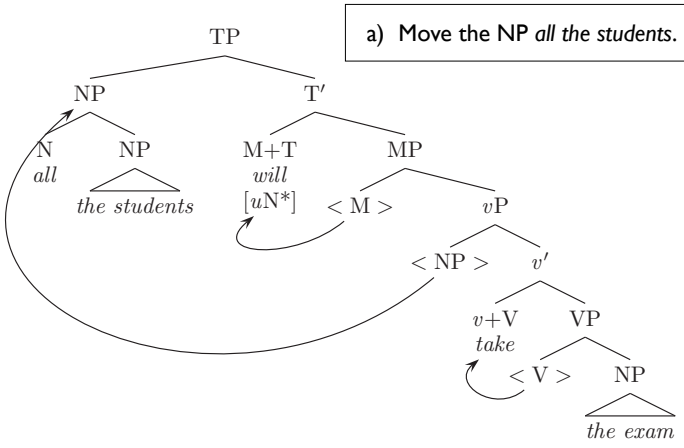
Is *all the students* closer to T than *the students* is?

Not if we define "closer" as we did, in terms of c-command.

Where X c-commands Y and Z, Y is closer to X than Z is if Y c-commands Z.

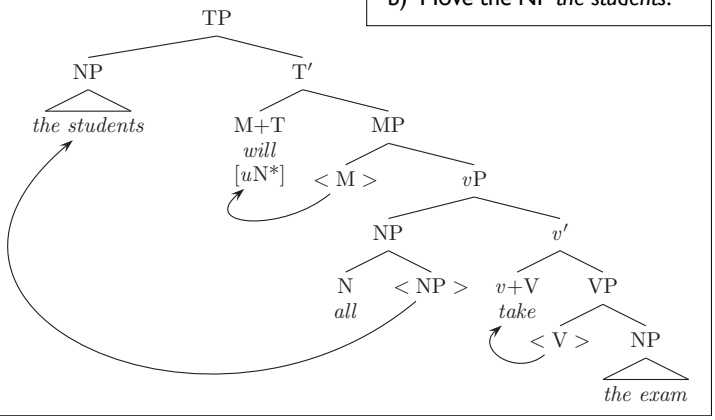
All the students will take...

a) Move the NP *all the students*.



The students will all take...

b) Move the NP *the students*.

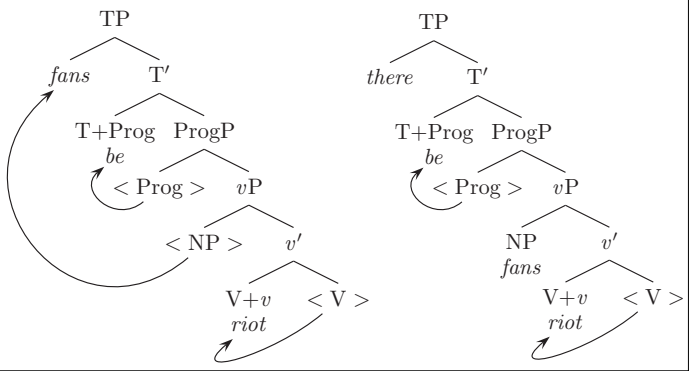


Expletive constructions

- An **expletive** is an element that can be in subject position without having received a θ-role from anywhere.
  - **It** had been raining.
  - **There** were fans rioting on Comm Ave.

Expletive constructions

1) There were fans rioting on Comm Ave.
2) Fans were rioting on Comm Ave.



The Big Picture

- Now that we've gotten some idea of how the system works, let's back up a bit to remind ourselves a bit about why we're doing what we're doing.
- People have (unconscious) knowledge of the grammar of their native language (at least). They can judge whether sentences are good examples of the language or not.
- Two questions:
  - What is it that we know?
  - How is it that we came to know what we know?

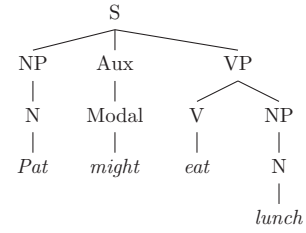
History

Phrase Structure Rules

S → NP (Aux) VP	VP → V (NP) (PP)
NP → (Det) (Adj+) N	PP → P NP
Aux → (Tns) (Modal) (Perf) (Prog)	P → at, in, to, …
N → Pat, lunch, …	Modal → can, should, …
Tns → Past, Present	Prog → be -ing
Perf → have -en	

- In trying to model what we know (since it isn't conscious knowledge) some of the first attempts looked like the phrase structure rules above (Chomsky 1957).
  - An S can be rewritten as an NP, optionally an Aux, and a VP. An NP can be rewritten as, optionally a determiner, optionally one or more adjectives, and a noun. …
  - What we know is that an S has an NP, a VP, and sometimes an Aux between them, and that NPs can have a determiner, some number of adjectives, and a noun.

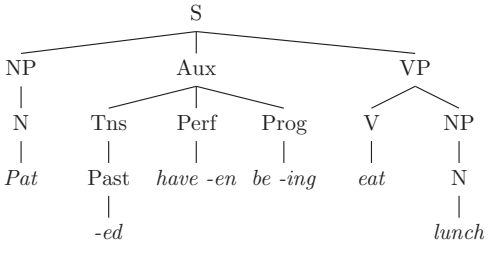
Phrase Structure Rules
S → NP (Aux) VP
NP → (Det) (Adj+) N
Aux → (Tns) (Modal) (Perf) (Prog)
N → Pat, lunch, …
Tns → Past, Present
Perf → have -en
VP → V (NP) (PP)
PP → P NP
P → at, in, to, …
Modal → can, should, …
Prog → be -ing



History

- In this way, many sentences can be derived, starting from S.
- The tree-style structure is a way to record the history of the derivation from S to the words in the sentence.
- We model our knowledge of English as a machine that (ideally, when it's finished) will generate all of the sentences of English and no others.

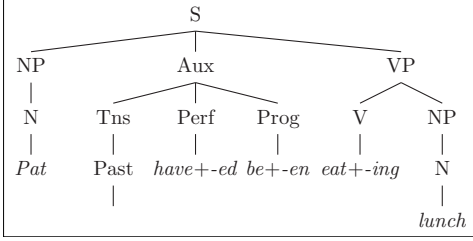
Phrase Structure Rules
S → NP (Aux) VP
NP → (Det) (Adj+) N
Aux → (Tns) (Modal) (Perf) (Prog)
N → Pat, lunch, …
Tns → Past, Present
Perf → have -en
VP → V (NP) (PP)
PP → P NP
P → at, in, to, …
Modal → can, should, …
Prog → be -ing



Affix Hopping

- If you build a sentence this way, things aren't in the right order, but there's a simple *transformation* that can be done to the structure to get it right.
- Empirically, tense, perfect *have*, and progressive *be* each control the form of the verbal element to their **right**.

Phrase Structure Rules
S → NP (Aux) VP
NP → (Det) (Adj+) N
Aux → (Tns) (Modal) (Perf) (Prog)
N → Pat, lunch, …
Tns → Past, Present
Perf → have -en
VP → V (NP) (PP)
PP → P NP
P → at, in, to, …
Modal → can, should, …
Prog → be -ing



Affix Hopping

- **Affix Hopping**
  SD: afx verb
  SC: verb+afx
- The affixes all "hop to the right" and attach to the following word.
- An ancestor to the kinds of movement rules and of course the Agree operation we've been talking about.

History continues

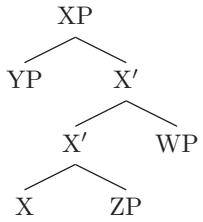
- Through the 60s there were good people working hard, figuring out what kinds of phrase structure rules and transformations are needed for a comprehensive description on English.
- As things developed, two things became clear:
  - A lot of the PSRs look pretty similar.
  - There's no way a kid acquiring language can be learning these rules.

- Chomsky (1970) proposed that there actually is only a limited set of phrase structure rule types.
- For any categories X, Y, Z, W, there are only rules like:
  XP → YP X′
  X′ → X′ WP
  X′ → X ZP

X-bar theory

- If drawn out as a tree, you may recognize the kind of structures this proposal entails. These are structures based on the "X-bar schema".
  - XP → YP X′
    X′ → X′ WP
    X′ → X ZP
  - YP being the "specifier", WP being an "adjunct", ZP being the "complement". Adjuncts were considered to have a slightly different configuration then.

Why is this better? The types of rules are much more constrained. AND it also makes predictions about structure and constituency that turn out to be more accurate.



GB

- Around 1981, the view shifted from thinking of the system as constructing all and only structures with PSRs and transformations to a view in which structures and transformations could apply freely, but the grammatical structures were those that satisfied constraints on (various stages of) the representation.

- First, a "deep structure" (DS) tree is built, however you like *but*
  - Selectional restrictions must be satisfied
  - θ-roles must be assigned
  - Etc.
- Then, adjustments are made to get the "surface structure" (SS)
  - Things more or less like Affix Hopping, or moving V to *v*, or moving the subject to SpecTP.
  - Further constraints are verified here: Is there a subject in SpecTP? Etc.
- Finally, the result is assigned a pronunciation (PF), and, possibly after some further adjustments, an interpretation (LF).

Why is this better? Most of the construction-specific rules were made to follow from more general principles interacting. AND again, it caused us to look for predictions, which were better met.

Which brings us to 1993

- The most recent change in viewpoint was to the system we're working with now (arising from the Minimalist Program for Linguistic Theory).
- The constraints that applied to the structures in GB were getting to be rather esoteric and numerous, to the extent that it seemed we were missing generalizations.

- The goal of MPLT was to "start over" in a sense, to try to make the constraints follow from some more natural assumptions that we would need to make anyway.
- This new view has the computational system working at a very basic level, forcing structures to obey the constraints of GB by enforcing them locally as we assemble the structure from the bottom up.

Why is this better? It's a further reduction to even more general principles. The idea is that you need a few things to construct a language-like system—and there's nothing else.

Features and technology

- The use of features to drive the system (uninterpretable features force Merge, because if they are not checked, the resulting structure will be itself uninterpretable) is a way to encode the notion that lexical items need other lexical items.
- What the system is designed to do is assemble grammatical structures where possible, given a set of lexical items to start with.

- A comment about the technology here:
- The operations of Merge, Adjoin, Agree, and feature checking, the idea that features can be interpretable or not (or, strong or weak) are all **formalizations** of an underlying system, used so that we can **describe the system precisely enough to understand its predictions** about our language knowledge.

Features and the moon

- We can think of this initially as the same kind of model as this:

$$f = G\frac{m_1 m_2}{r^2}$$

- The Earth and the Moon don't compute this. But if we write it this way, we can predict where the Moon will be.

- Saying lexical items have uninterpretable features that need to be checked, and hypothesizing mechanisms (matching, valuing) by which they might be checked is similarly a way to formalize the behavior of the computational system underlying language in a way that allows us deeper understanding of the system and what it predicts about language.

The "Minimalist Program"

- The analogy with the gravitational force equation isn't quite accurate, given the underlying philosophy of the MP.
- The Minimalist Program in fact is trying to do this:
  - Suppose that we have a cognitive system for language, which has to interact with at least two other cognitive systems, the **conceptual-intensional** and the **articulatory-perceptual**.
  - Whatever it produces needs to be interpretable (in the vernacular of) each of these cognitive systems for the representation to be of any use.
  - Suppose that the properties of these external systems are your boundary conditions, your specifications.
  - The hypothesis of the MPLT is that the computational system underlying language is an optimal solution to those design specifications. So everything is thought of in terms of the creation of interpretable representations.