CAS LX 321 / GRS LX 621 *Syntax: Introduction to Sentential Structure*

November 20ish, 2018

1 Summary

While I have you here, I'll also kind of try to summarize where we are and how we got here, not just introduce new things. In particular, one thing that I want to try to get out of this is a solidification of what we did leading up to homework 8, which has caused some confusion due to a lot of things being very new and densely packed (and not entirely well specified).

So, I'll start by filling in some things from last time's handout, and take it from there.

2 Auxiliaries

We have quite recently added "auxiliaries" to our grammar. We kind of had these before, but they were limited to the modals like *should*, *may*, etc. I had intentionally tried to dodge any of the others until we got to this point.

2.1 Modals

"Modals" are *must*, *may*, *might*, *shall*, *should*, *can*, *could*, *will*, *would*. And it is possible to put one of these between the subject and the verb.

(1)   Pat should sleep.

What we had done <u>before</u> was to say that these were realizations of category T (for "tense"). The justification for this (which in fact I'm about to co-opt for a different conclusion) is that they "feel" kind of like they have a tense distinction in their morphology. This is an argument that relies a lot on an intuitive feeling and not a lot on much else. But it goes like this:

- Modals (except *must*) seem to come in pairs. *May / might*; *can / could*; *shall / should*, *will / would*.

- This seems to be a regular binary distinction, and the morphological alternation is one that seems like one we might see between non-past and past.

- By analogy to *do~did*, *bring~brought*, *buy~bought*, . . . , we say modals can be "past" or "non-past."

HOWEVER, this is at least as a first approximation only a <u>morphological</u> thing. The "past" modals (*should*, *could*, *would*, *might*) do not have any simple interpretation as being semantically in the past.

That is to say: Neither of the examples in (2) are interpreted about talking about a time before now. If there is to be yodeling by Pat, it is to be in the future. That is because the modal *itself* is intrinsically essentially forward-looking.

(2)   a. Pat may yodel.
     b. Pat might yodel.

I do not really know what the difference in meaning is between (2a) and (2b). It is super subtle to be sure.

Part of what complicates judgments about modal meanings is that there are really at least two things going on semantically. Generally, a modal says "The sentence is true in some/all of a given set of possible ways the world could be." The "given set of possible ways the world could be" can vary, common ones being (i) all of the physically possible options, (ii) all of the options where the rules are followed, (iii) all of the options consistent with what I know. The choice of subset is usually referred to as the "modal base" (root, deontic, episitemic, respectively). And then modals vary on whether they are true for some (existential) or all (universal). So, *will* is a universal, which can at least be root (X happens in all possible worlds) or epistemic (In all possible worlds consistent with what I know, X happens). And, *may* is existential, and can be deontic (In at least one of the possible worlds where the rules are followed, X happens), or epistemic (In at least one of the possible worlds consistent with what I know, X happens), or root (In at least of one of the possible worlds, X happens). Individual modals are biased toward particular modal bases to varying extents, and it is not necessarily consistent for both members of the pair.

Given this complexity, it is tough to zoom in on what distinguishes (2a) from (2b), and thus to figure out if there really is a contribution of "pastness" in the semantics of the past modal forms. Perhaps something like: *shall* and *should* are universal and can be interpreted deontically, and *Guests shall remove their shoes* is interpreted as being somewhat more confident that the rules will actually be followed than *Guests should remove their shoes* does. Perhaps some change in circumstance might change the compliance requirements? Such that what might have fallen outside the rules in the past might currently be allowed. Thus, *shall* is based on worlds in which the rules as of now are followed, while *should* is based on worlds in which the rules as they stood at a past time are followed? Maybe. Hard to say, but the main thing is that there is no straightfoward *intuition* that either of the sentences in (2) is talking about the past.

Here's one other argument that *should*, etc., are actually in some abstract sense "past" though. It's from "sequence of tense." Observe:

(3) a. I think [she is injured].
     b. I think [she was injured].
     c. I thought [she was injured].
     d. * I thought [she is injured].

The generalization here seems to be:

- Under non-past, either past or non-past is allowed.

- Under past, only past is allowed.

If that's true, then we have a way to test whether a given modal is "grammatically past" or not. We just put it under a past tense verb. If it sounds bad under a past tense verb, it is non-past. And the modals do split into two types based on this (and just where we'd expect based on the morphological intuition):

(4) a. I think [she may yodel].
     b. I thought [she might yodel].
     c. * I thought [she may yodel].
(5) a. I think [she can yodel].
     b. I thought [she could yodel].
     c. * I thought [she can yodel].

(6) a.   I think [she will yodel].

     b.   I thought [she would yodel].

     c.   * I thought [she will yodel].

(7) a.   I think [she shall yodel].

     b.   I thought [she should yodel].

     c.   * I thought [she shall yodel].

(8) a.   I think [she must yodel].

     b.   * I thought [she must yodel].

If the judgments are right, then:

- Modals *may*, *will*, *shall*, *can*, *must* behave like non-past.

- Modals *might*, *would*, *should*, *could* behave like past.

So, it's not *entirely* about intuitions concerning morphology, there is *something* "past-like" about *might*, *would*, *could*, and *should*.

2.2   Past vs. non-past vs. present

Verbs in English can certainly be in the past tense. *Wrote*, or *yodeled*, for example. But what about the verbs that are not in the past tense? *Write(s)* or *yodel(s)*, for example?

I've been trying to avoid calling these non-past verbs "present," mainly because English really seems to have only a binary distinction. A verb is either past or it is not. Perhaps I could instead say a verb is either present or it is not. But the main thing is that one is +F for some feature F, and the other is –F for that feature F.

This is really pretty obscure or pedantic, but I still try to use "non-past" instead of "present." My rationale for this is that if a verb can be "past" and a verb can be "present" then there isn't anything in the system that prevents it from being both. Whereas it is obvious that [+past] and [–past] are incompatible.

So, I'd prefer we all use "non-past" (possibly written as "–past"), but as long as "present" is understood as "–past" it's probably ok. I'm sure that I slip up occasionally and refer to the non-past form as "present" or even write "PRES" as a value of T.

2.3   Aspect, *have*, *be*

Up until just recently, we placed modals directly in T. So, we had several things that could be "T" in our trees. One is a silent/abstract "PAST" morpheme (like the one in *Pat yodeled*), another is a silent/abstract "–PAST" morpheme (like the one in *Pat yodels*), and the others are the modals *might*, *should*, and the rest.

That was a shortcut, however, and it becomes more complicated when we start thinking about the "helping verbs" (auxiliary verbs) *have* and *be* that show up in more complex sentences. These auxiliary verbs correspond to the addition of semantic aspect (perfective, progressive) and (later) voice (passive).

The kinds of sentences we have in mind are:

(9) a.   Pat must write poetry.

     b.   Pat is writing poetry.

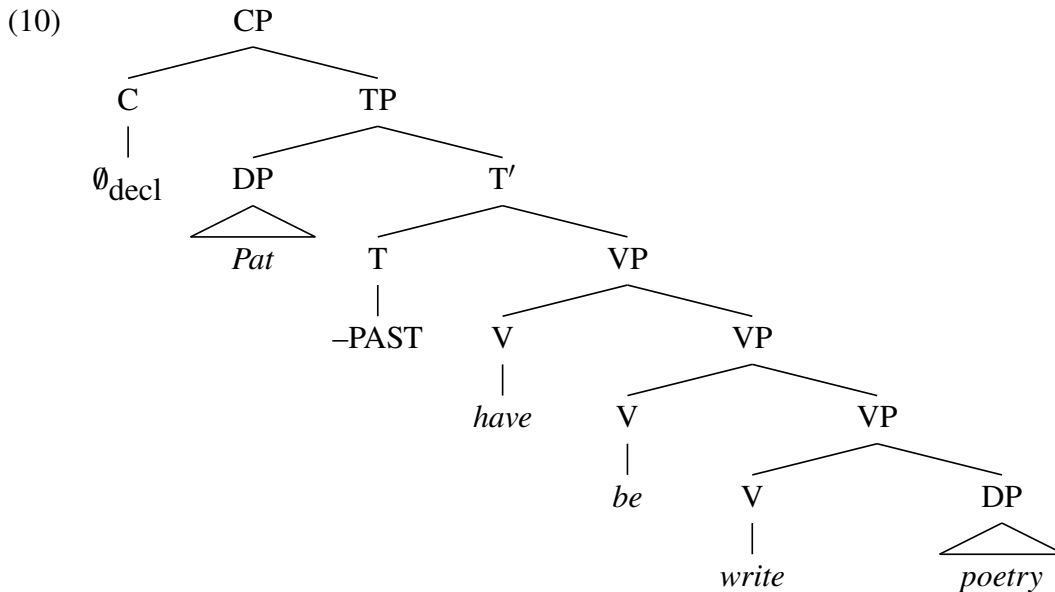   c.   Pat has written poetry.

     d.   Pat must be writing poetry.

e. Pat must have written poetry.

f. Pat has been writing poetry.

g. Pat must have been writing poetry.

It is clear that there are now a lot of places between the subject and the verb, in that spot we were calling "T" and putting modals. So, the question arises: how to we draw these structures?

Without giving a lot of motivation for it in class, the move I suggested is the following:
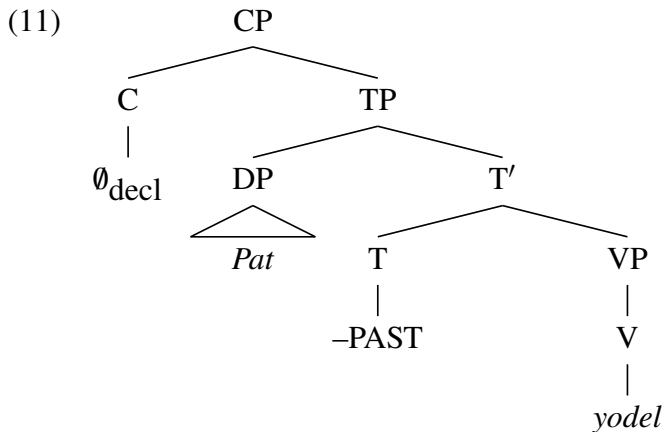
- Progressive *be* is a V subcategorizing for VP.

- Perfective *have* is a V subcategorizing for VP.

What that means is that we have a tree for (9f) that looks something like this (to be revised):

(10)

```
                        CP
               _____/  _____
              C                    TP
              |              _____/  _____
           Ø_decl           DP              T'
                           /\        _____/  _____
                          Pat       T                VP
                                    |          _____/  _____
                                  –PAST       V                VP
                                              |          _____/  _____
                                            have        V                VP
                                                        |          _____/  _____
                                                       be          V              DP
                                                                   |              /\
                                                                 write          poetry
```

But how do we get from that tree to the actual pronunciation of (9f)?

The pronunciation of each V in the tree above is determined by the element in the tree just above it. So, while T is –PAST, the top V is pronounced *has*. This is the same general case as in *Pat yodels*:
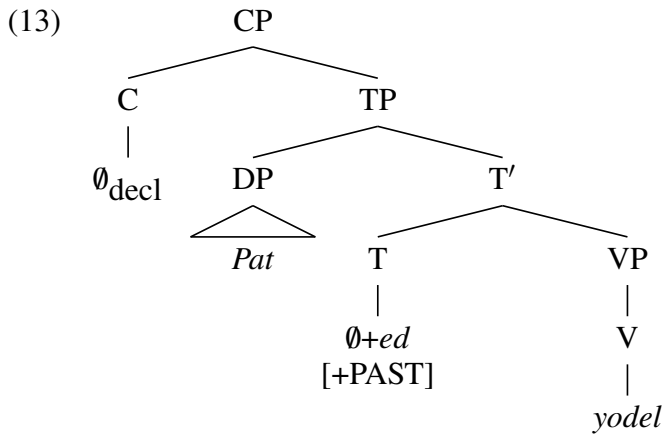
(11)

```
                        CP
               _____/  _____
              C                    TP
              |              _____/  _____
           Ø_decl           DP              T'
                           /\        _____/  _____
                          Pat       T                VP
                                    |                |
                                  –PAST              V
                                                     |
                                                   yodel
```

If T is –PAST, V is pronounced *yodels*, and if T is +PAST, V is pronounced *yodeled*. So, similarly in (10), if T is –PAST, the top V is pronounced *has*, and if T were +PAST, then the top V would be pronounced *had*.

One way to envision this is to say that the –PAST T is itself silent, but has a non-past suffix (like *-s* for third-person-singular subjects) to contribute. Or the +PAST T, which has a past suffix (like *-ed*) to contribute. If we think of it this way, then the lexical entries of T would look like:

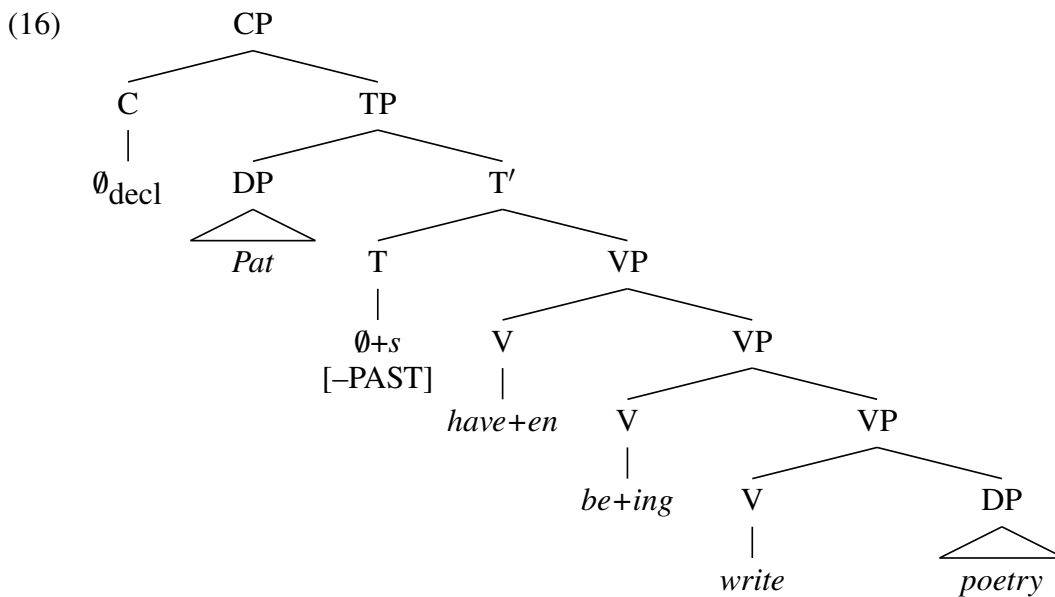(12) a. $\emptyset$+*s*, T, [–PAST]

b. $\emptyset$+*ed*, T, [+PAST]

And then we could draw the tree for *Pat yodeled* like this:

(13)

```
                CP
          _____|_____
         C             TP
         |         _____|_____
       Ødecl      DP          T′
                  △      _____|_____
                 Pat    T          VP
                        |           |
                      Ø+ed          V
                      [+PAST]       |
                                  yodel
```

The idea is that the suffix *ed* (indicated as "+*ed*" under T) will "hop" over to the V when we pronounce this structure.

Given this mechanism of "affix hopping" we can complete the description of the auxiliary verbs as well: when you have a perfective *have*, the next verb gets an *en* ending; when you have a progressive *be*, the next verb gets an *ing* ending.

(14)   have+*en*, V, [AUX, PERF], [+ _ VP]

(15)   be+*ing*, V, [AUX, PROG], [+ _ VP]

(16)

```
                CP
          _____|_____
         C             TP
         |         _____|_____
       Ødecl      DP          T′
                  △      _____|_____
                 Pat    T          VP
                        |       ___|___
                      Ø+s      V       VP
                     [–PAST]   |     ___|___
                            have+en  V      VP
                                     |    ___|___
                                  be+ing  V      DP
                                          |      △
                                        write  poetry
```

Getting back to modals for a second, observe that they too can be described in terms of the effect they have on the form of the following verb:

(17)   Pat wrote poetry.

(18)   Pat should write poetry.

(19)   Pat should be writing poetry.

(20)   Pat should have written poetry.

(21)   Pat should have been writing poetry.

In particular, the suffix that a modal contributes is -∅. The verb after a modal is in the bare form. To write this analogously to *have* and *be*, it might be best to write modals as being something like *could+∅*, though I have tended just to leave the "+∅" out.

But one other thing: Since modals can be past or non-past, just like all of these other verbs, the simplest thing to do is to say that the choice between a past and non-past modal is done in the same way as the choice between a past and non-past form of any other verb. That is to say, the +PAST and –PAST part is in T, but the modal itself is a V in the next phrase down. So, for *Pat should have been writing poetry* we have:

(22)
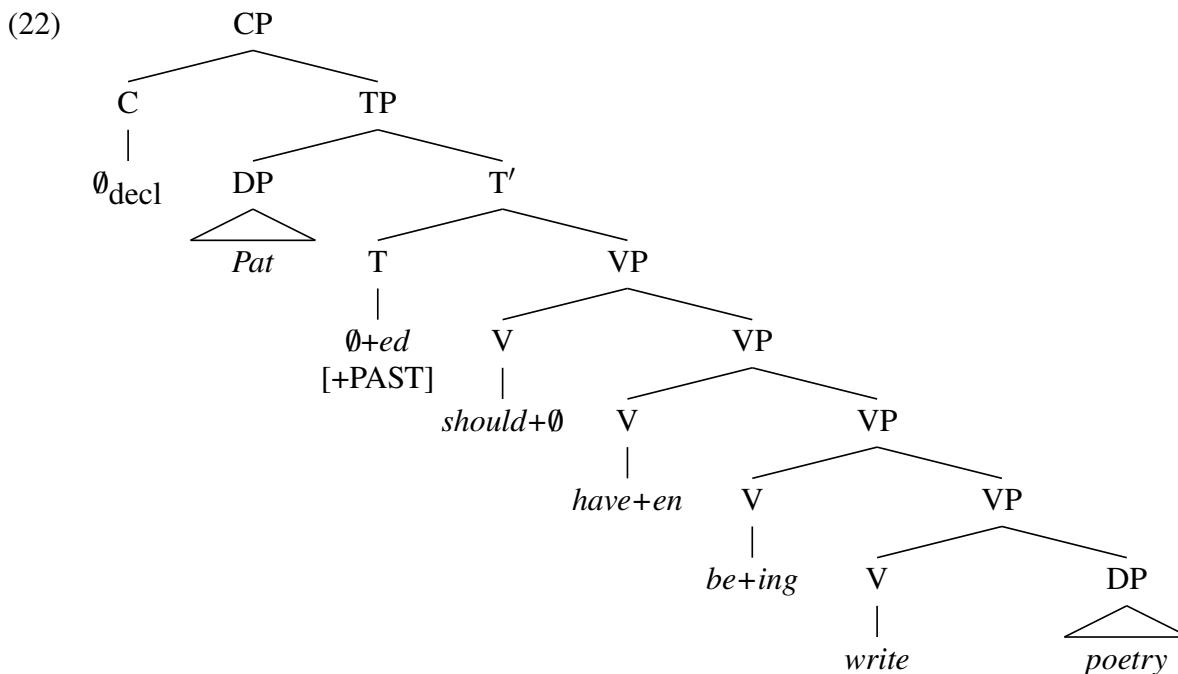
Notice too that each head in this tree is making a semantic contribution. T is contribing "past," *should* is contributing a universal modal, *have* is contributing perfective aspect (completed), *be* is contributing progressive aspect (ongoing at the temporal reference point), *write* is contributing the main event description. But the realization of progressive is in two parts: one is the *be* verb (which is in the *en* form due to the higher perfective *have*), and the other is the *ing* ending on *write*.

2.4   External ordering

It turns out that if you have both perfective and progressive in a clause, they need to come in that order. And if you have both a modal and progressive, they need to come in that order. And so forth.

(23)   a.   Pat has been yodeling.

        b.   * Pat is having yodeled.

(24)   a.   Pat should be yodeling.

b.   * Pat is shoulding yodel.

The generalization is that the order is always this:

(25)   modal/to > perfective > progressive > verb

Not everything needs to be there, but anything that *is* there needs to be in the order specified above.

I consider this to be an "external ordering" because it does not arise from the subcategorization frames in the lexical items. That is, we can't really say (26) is the lexical entry for perfective *have* (that it requires a VP complement that is the progressive *be*), because sometimes you have sentences that have perfective *have* but no progressive *be*.

(26)   have+*en*, V, [AUX, PERF], [+ _ VP$_{[PROG]}$]

So, instead, we'll suppose that there's something outside that says that, given that perfective *have* takes a VP, then it will take a progressive VP if we have one, otherwise a VP. That is, we have the order in (25), which allows only some of the possible trees our rules could generate.

I tried to make this seem a bit less weird by pointing out that we already kind of have to assume there is something like this for adjectives. That is, adjectives do seem to need to come in a particular order (you can say *big red balloon* but not *red big balloon*), but the syntactic rules we have just allow any adjectives to be adjoined to an NP in whatever order. So, here too, the syntax allows for any kind of possibilty, but this external ordering rule filters out some of the possibilities such that only the trees where the adjectives are in the right order are actually grammatical.

3   Negation

Negation throws a couple of extra wrinkles in, so I've kind of avoided it where possible up to now.

Looking at the data, it seems fairly clear that we can't just blindly put Neg (*not*) into the ordering in (25), because there's pretty much nowhere we can put it without running into a paradox (until we make one change).

(27)   a. I am not yodeling.
        b. I have not been yodeling.
        c. I should not have yodeled.

We can't put Neg between modal and perfective, because although that reflects the order in (27c) (*not* comes before *have*), it is the opposite of the order in (27b) (where *have* comes before *not*). We can't put Neg before *be*, because although *not* comes before *be* in (27b), *not* comes after *be* in (27a).

The basic generalization is that *not* comes after the highest of the auxiliaries (setting aside the case where there are no auxiliaries, for the moment).

This makes perfect sense if we suppose that what is actually going on is that the topmost auxiliary V is moving up to where T is, while Neg is just below it. That is, the ordering is actually:

(28)   CP > TP > negation > modal/to > perfective > progressive > passive > verb
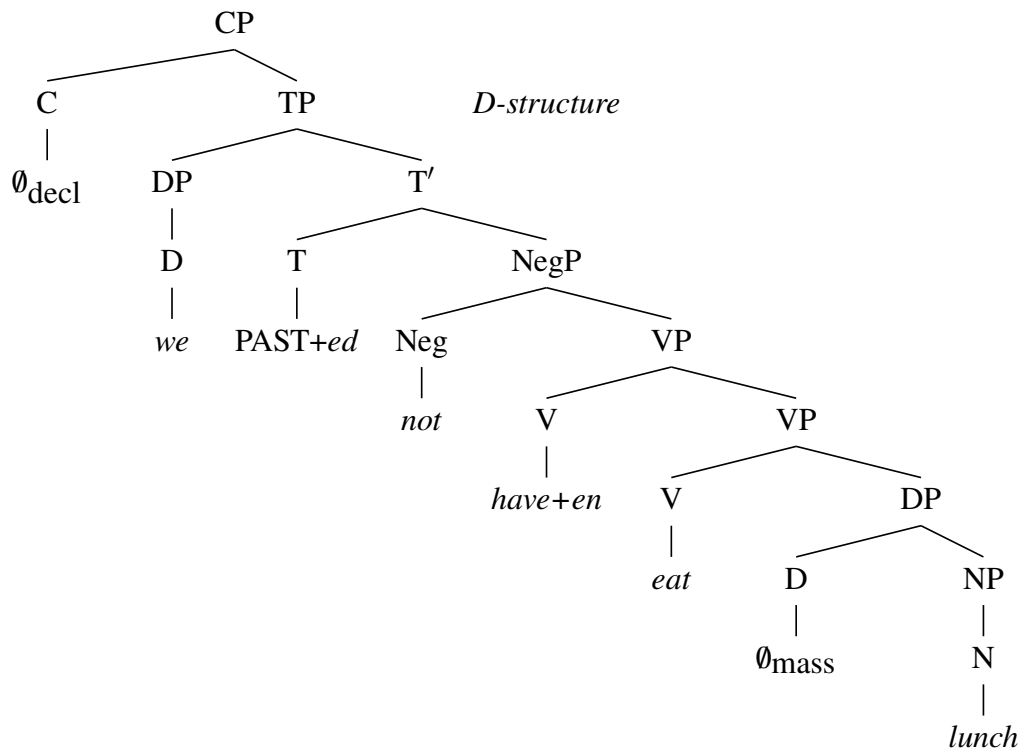
So, *not* is actually before everything except T, and the reason that the highest auxiliary appears before *not* is that it moves from below *not* to T (above *not*).
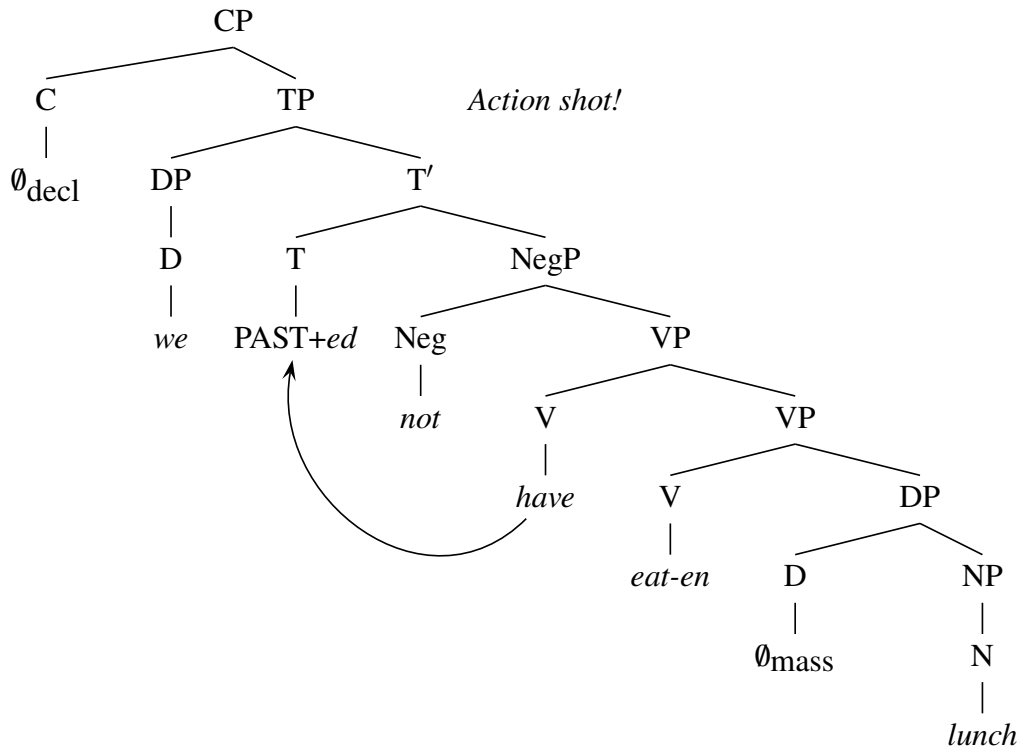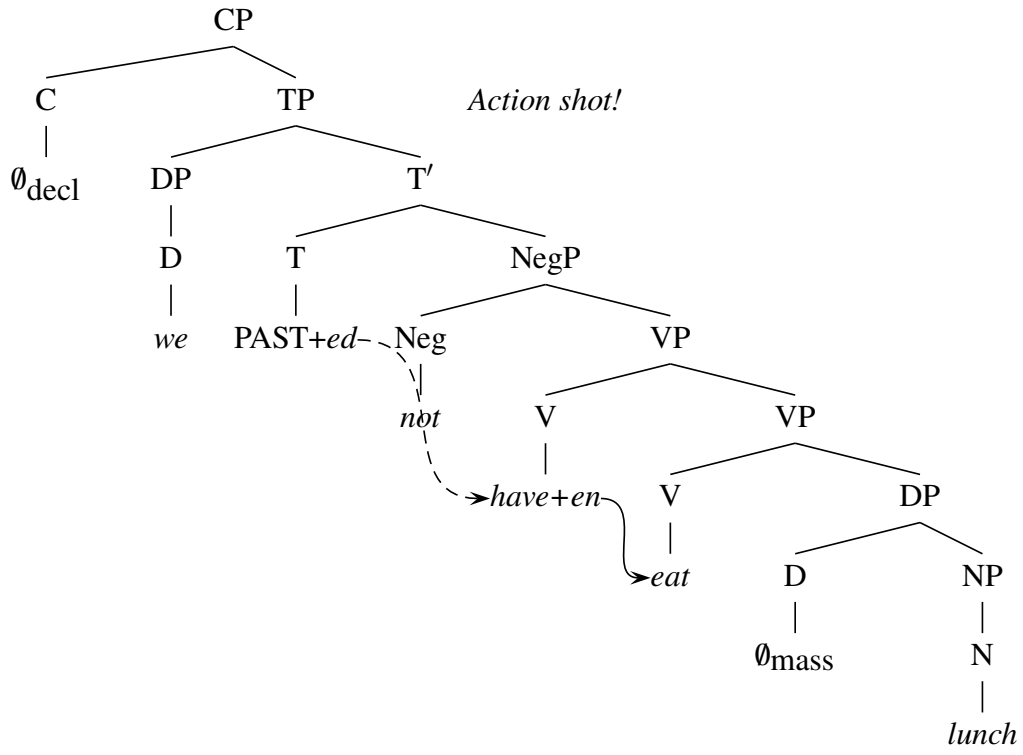
I demonstrate this with a little "movie" of a sort, in four parts, showing how we get *we have not eaten lunch*. In the first frame ("D-structure"), we have the tree as generated by the rules, before anything moves.
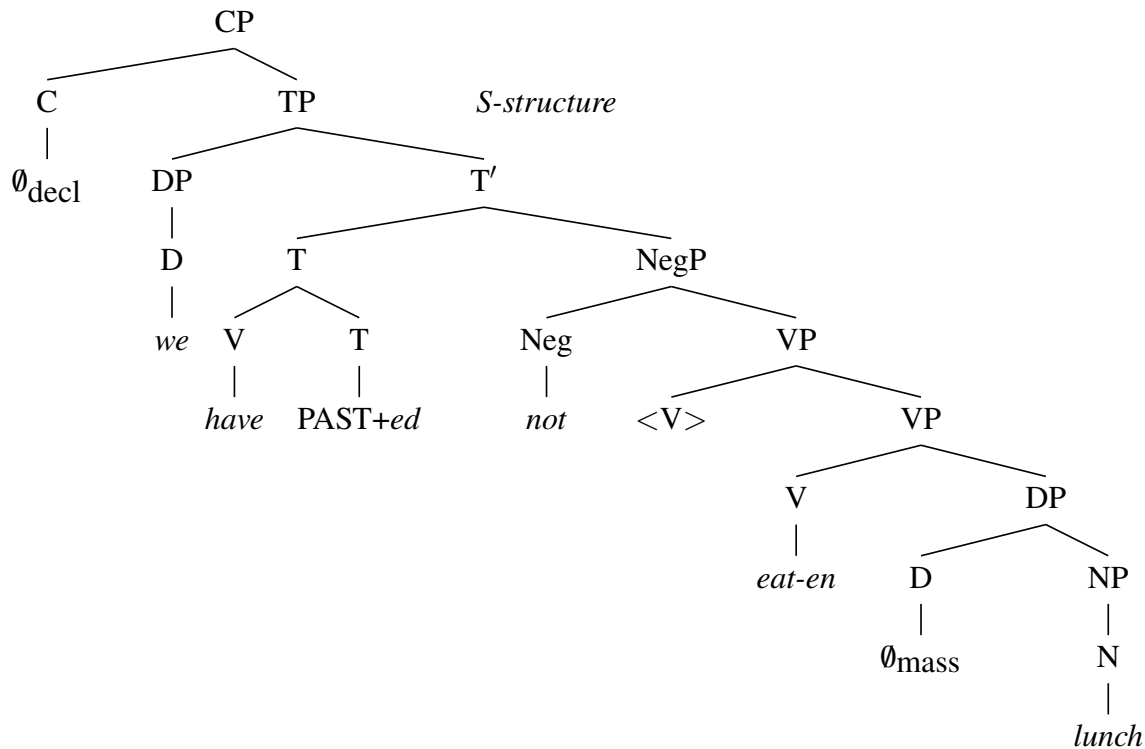
In the second frame ("Action shot!"), we have the affixes hopping off to the right. I don't usually draw this step, and this is not really the movement I normally want you to think about. However, it is relevant that the *en* from *have* becomes the suffix for *eat* while *have* is still right next to *eat*. We're about to move *have* away, to a position where affix hopping should not be possible. (The reason will be that affix hopping can't get into NegP.) So in this second frame, the past tense *ed* suffix can't associate to *have* at this point because NegP is in the way (which I indicated with the dashed arrow, symbolizing the dashed hopes of this suffix to attach to a host verb).

In the third frame, *have* is indicated as moving up to T by an arrow. I call this an "Action shot!" as well because it is not the final form. In this frame I have also put the suffixes on their hosts (in particular, *en* is now a suffix on *eat*).

In the final frame ("S-structure"), the structure after *have* moves to T is shown. I'll comment below the tree about the T and the $<$V$>$.

CP
├── C
│   └── $\emptyset_{\text{decl}}$
└── TP   *D-structure*
    ├── DP
    │   └── D
    │   └── *we*
    └── T′
        ├── T
        │   └── PAST+*ed*
        └── NegP
            ├── Neg
            │   └── *not*
            └── VP
                ├── V
                │   └── *have+en*
                └── VP
                    ├── V
                    │   └── *eat*
                    └── DP
                        ├── D
                        │   └── $\emptyset_{\text{mass}}$
                        └── NP
                            └── N
                                └── *lunch*

First tree

```
                    CP
          ┌─────────┴─────────┐
          C                   TP
          │          ┌────────┴────────┐
        ∅_decl       DP               T'          Action shot!
                     │        ┌────────┴────────┐
                     D        T               NegP
                     │     ┌──┴──┐       ┌──────┴──────┐
                    we   PAST+ed  Neg    Neg           VP
                                  │            ┌────────┴────────┐
                                 not           V               VP
                                               │        ┌───────┴───────┐
                                          have+en        V              DP
                                               │         │       ┌──────┴──────┐
                                              eat        eat      D            NP
                                                                  │            │
                                                               ∅_mass          N
                                                                               │
                                                                             lunch
```

Second tree

```
                    CP
          ┌─────────┴─────────┐
          C                   TP
          │          ┌────────┴────────┐
        ∅_decl       DP               T'          Action shot!
                     │        ┌────────┴────────┐
                     D        T               NegP
                     │     ┌──┴──┐       ┌──────┴──────┐
                    we   PAST+ed  Neg    Neg           VP
                                  │            ┌────────┴────────┐
                                 not           V               VP
                                               │        ┌───────┴───────┐
                                             have        V              DP
                                                         │       ┌──────┴──────┐
                                                      eat-en      D            NP
                                                                  │            │
                                                               ∅_mass          N
                                                                               │
                                                                             lunch
```

S-structure tree:

CP
— C: Ø_decl
— TP
  — DP → D → *we*
  — T′
    — T: V *have*, T PAST+*ed*
    — NegP
      — Neg: *not*
      — VP: <V>
        — VP
          — V: *eat-en*
          — DP
            — D: Ø_mass
            — NP → N → *lunch*

Now, about the tree above. Two things. What we've said we want to do here is to "move the highest [AUX] V to T" but we need to have some kind of interpretation of what that actually means in the structure.

Without getting too deep into it, the basic consideration reflected above is this: Even after movement, X-bar structures must be maintained. Specifically, a VP needs a V head, and TP needs a T head. What that rules out is the possibility of actually just replacing T with V and leaving a hole of some kind where V used to be. We need the information in T anyway, T is what tells us that the sentence is semantically past.

So, what we do is *adjoin* V to T. This is essentially the same kind of thing as adjoining an adverb to VP; we have a T that, when combined with a V, is (still) a T. When you adjoin an adverb, you have a VP that, when combined with an AdvP, is (still) a VP. So, T is still the head of TP, but now it is a T with a V "hanging off of it."

As for where V came from, what I've done is just mark the V as being a "trace" or remnant. The <V> means that this is where V was. The V-hood needs to still be there in some abstract sense, because it is (still) serving as the head of the VP. But, nodes surrounded by < > marks are not pronounced.

3.1 *Do*-support

In cases where there are no auxiliary verbs (just a regular "main verb"), introducing negation has the curious effect of adding a "dummy verb" *do* in the position of T.

With the rest of this machinery in place, we have a fairly good way to understand this. The "affix hopping" procedure (that allows a head to affect the pronunciation of the next verbal head down) is unable to "see through" NegP. The idea is that: T has a suffix it needs to have realized (e.g., the *ed* suffix on [+PAST] T), but if it can't "reach" the thing it would attach to, we need to insert a V close enough to the T that it can attach.
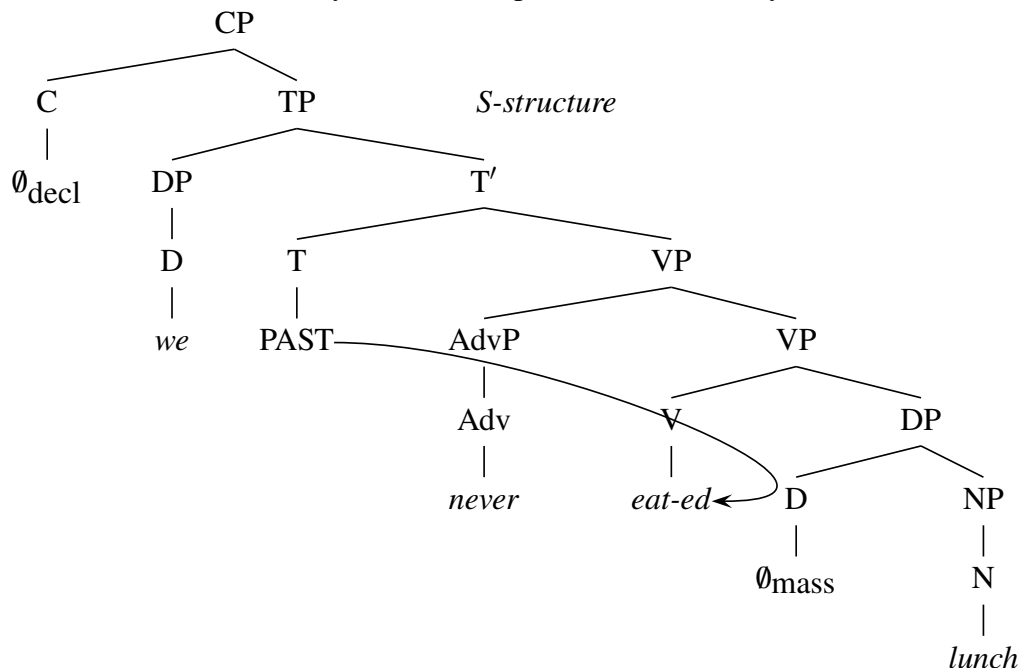
So, if T is "stranded" by being separated from the VP (e.g., by NegP), we adjoin a dummy verb *do* to T, and then the suffix from T is realized on that.

The first tree diagram (S-structure) shows:

CP
- C → ∅_decl
- TP — *S-structure*
  - DP → D → *we*
  - T′
    - T
      - V → *DO*
      - T → PAST+*ed*
    - NegP
      - Neg → *not*
      - VP
        - V → *eat*
        - DP
          - D → ∅_mass
          - NP → N → *lunch*
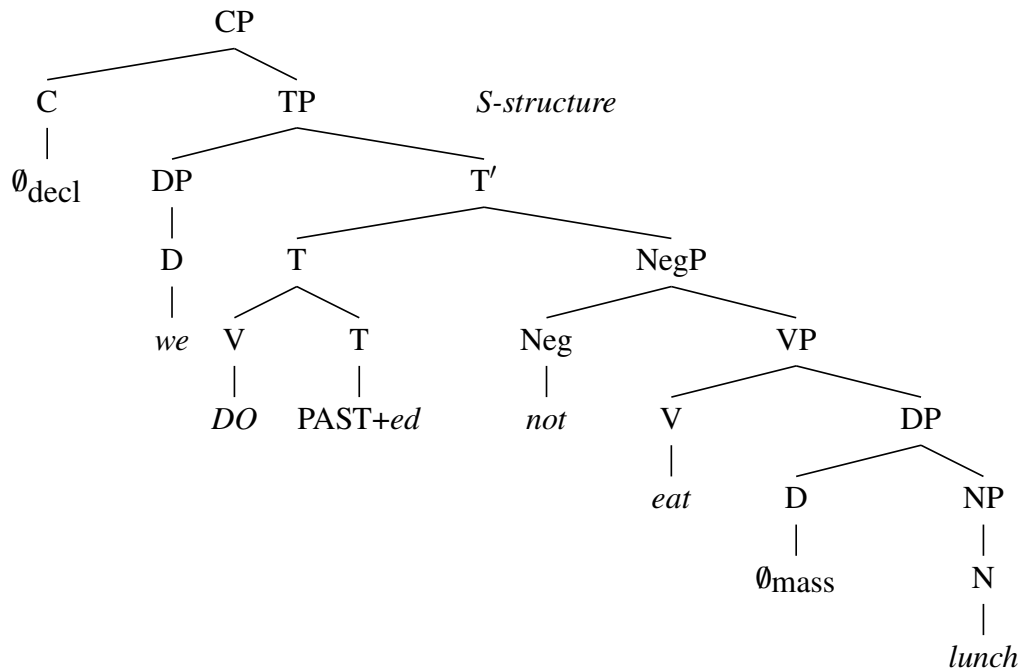
We can be even more specific about the conditions under which T is "stranded" by noticing that we get *do*-support with *not* (the head of NegP), but not with adverbs like *always* or *never*.

(29)   I do not eat lunch.

(30)   I always eat lunch.

(31)   I never eat lunch.

So, an adverb between T and the V is not enough to break "line of sight" but NegP is. One difference betweeh the two is that if we attach an adverb to VP, the result is still a VP. Even with an adverb between T and V, it's still the case that the sister of T is VP. That is, T can put the suffix on the head of the phrase that is its sister. It sounds maybe more complicated than it is if you look at it.

The second tree diagram (S-structure) shows:

CP
- C → ∅_decl
- TP — *S-structure*
  - DP → D → *we*
  - T′
    - T → PAST
    - VP
      - AdvP → Adv → *never*
      - VP
        - V → *eat-ed*
        - DP
          - D → ∅_mass
          - NP → N → *lunch*

11

4 Yes-no questions

Having motivated movement to T for auxiliaries, we now have a straightfoward way to understand inversion in yes-no questions. The way you form a yes-no question from a statement is by moving the auxiliary to the left of the subject (something that has long been referred to as "subject-aux inversion").

(32)   We had eaten lunch.

(33)   Had we eaten lunch?

Assuming that *we* is in the same place both in (32) and in (33) (namely, in the specifier of TP), then the location of *had* in (33) must be even higher. What's higher than Spec,TP? Well, C. Plus, C is basically the main thing that differs between (32) and (33), since C is the place where the [+decl] feature (indicating a declarative) or the [+q] feature (indicating a question) is.

So, we just say that when C is [+q], then T moves up to C. (Where "moves up to" means the same thing as it meant when we said that the highest [AUX] V moves up to T.)

In fact, we assume that even in (32), *had* has moved up to T from its original position, so the tree for that would be:
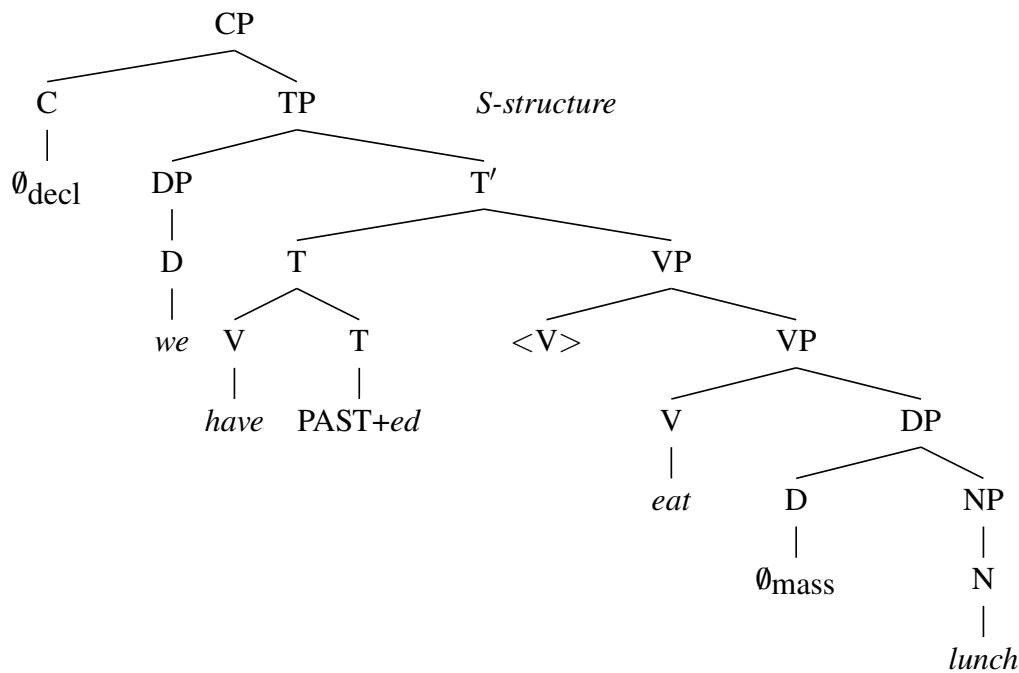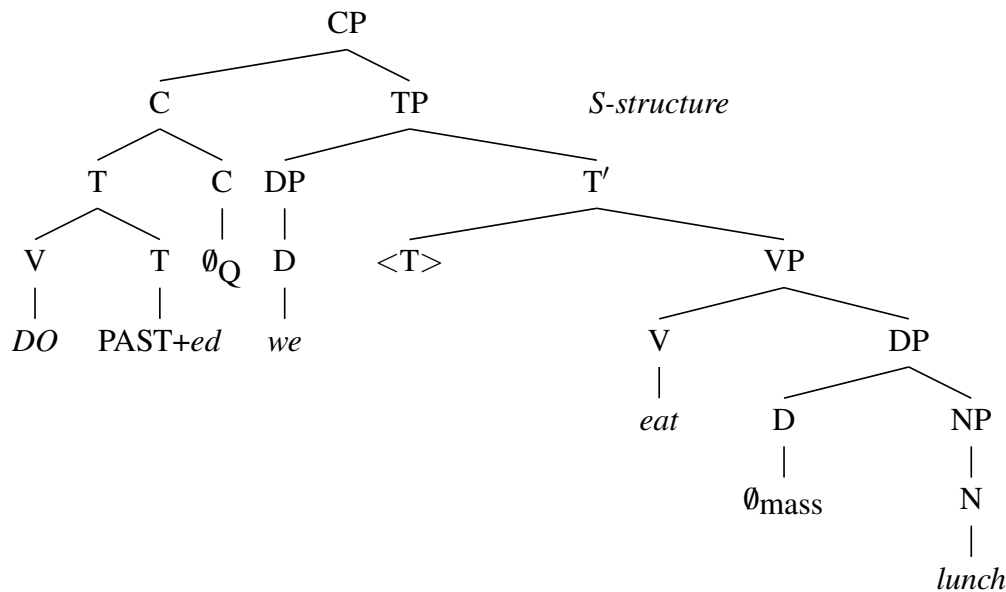

For (33), then, we have a tree that is the same as the one above except that C is [+q] this time, and so we move the T to C.

Note that T is already a "complex head"—it is a T with a V hanging off of it. So, when we move T up to C, we do the same thing: we hang T off of C, so that we have a C with a T hanging off of it. But that T had a V hanging off of it. So, what we really have is a C with a T with a V hanging off of it hanging off of it. It's simpler to just look at the tree:

The first tree (S-structure):

```
                        CP
               _____|_____
              C                 TP          S-structure
           ___|___          ____|____
          T       C        DP        T'
        __|__     |        |      ___|___
       V     T   ∅Q        D   <T>       VP
       |     |              |        _____|_____
     have  PAST+ed         we      <V>         VP
                                           _____|_____
                                          V           DP
                                          |        ____|____
                                       eat-en      D        NP
                                                   |         |
                                                 ∅mass       N
                                                             |
                                                           lunch
```
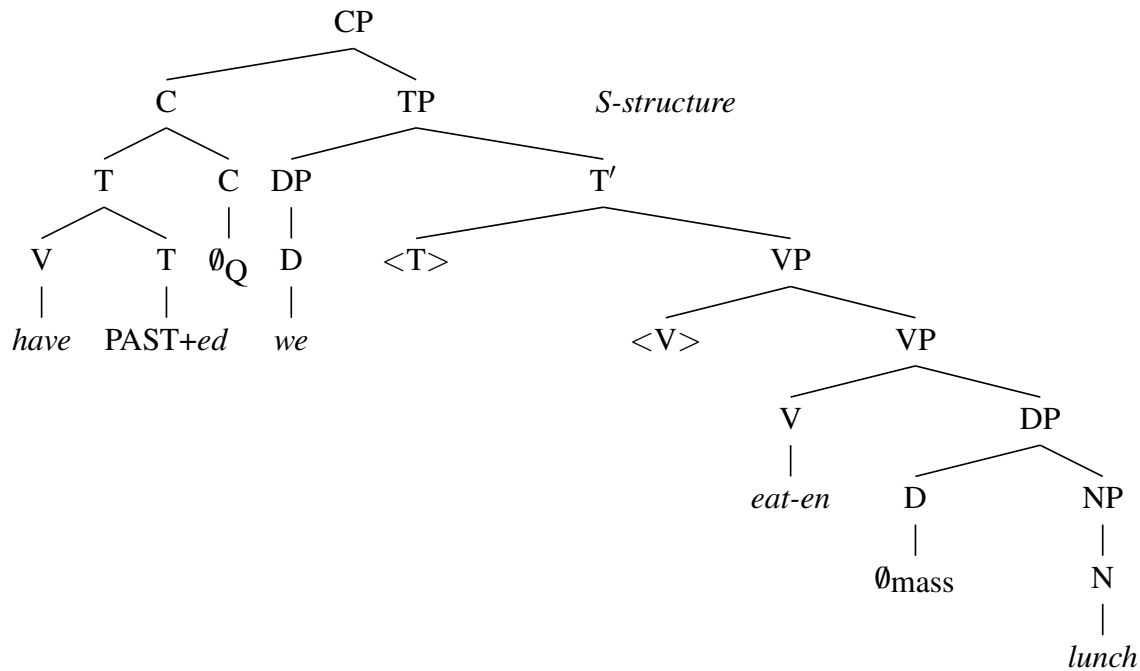
What if there was no auxiliary, but we just form a question from *We ate lunch*? We still need to move T up to C, but once we do that, then the sister of T is no longer VP. So, this means we need *do*-support, because we have stranded T (by moving it away). That's illustrated in the tree below:

```
                        CP
               _____|_____
              C                 TP          S-structure
           ___|___          ____|____
          T       C        DP        T'
        __|__     |        |      ___|___
       V     T   ∅Q        D   <T>       VP
       |     |              |        _____|_____
      DO   PAST+ed         we       V           DP
                                    |        ____|____
                                   eat       D        NP
                                             |         |
                                           ∅mass       N
                                                       |
                                                     lunch
```

If you're paying super-careful attention, you'll see that this is a little bit at odds with what I said about *We had not eaten lunch* in the little four-frame movie given a couple of pages ago. There, in the first "Action shot!" frame, I had the affixes hopping off just before the movement occurred. The reason for that was that moving *have* over negation does not prevent *en* from attaching to *eat*. So, I said that *en* hopped off before *have* moved away. In the tree I just gave, we moved T to C, crucially prior to its suffix hopping off, so that once the suffix is ready to hop it is no longer next to the V it would need to hop onto.

I don't really have a good way to characterize that difference, I think the best we can do for now is just suppose that the suffix hops off T *after* movements happen, but the suffixes hop off auxiliary verbs *before* movements happen. Maybe we can fine tune that later. This will almost never matter, really, but if you're

trying to follow along with the micro-steps of the derivations, there is a discrepancy between when affix hopping happens off of T relative to movement and when it happens off of V relative to movement.

5   *Wh*-questions

Now that we have handled auxiliaries and yes-no questions, we can turn our attention to *wh*-questions. Generally, a *wh*-question is a question that has a *wh*-word in it (like *who* or *what*).

5.1   *Wh*-questions are questions

First, some evidence that *wh*-questions and yes-no questions are both the same kind of thing (questions). For this, observe that there are two classes of verbs, represented by *think* and by *wonder*, which impose restrictions on what their argument can be. Both verbs can take sentential complements, but *think* requires that the sentential complement be a declarative, and *wonder* requires that the sentential complement be a question. This is shown in the examples in (34). The examples in (35) show that *hug* is a transitive verb that requires an object.

(34)   a.   Lisa thinks that Bart will hug Maggie.
         b.   Lisa wonders whether Bart will hug Maggie.
         c.   * Lisa wonders that Bart will hug Maggie.
         d.   * Lisa thinks whether Bart will hug Maggie.

(35)   a.   Lisa thinks that Bart will hug Maggie.
         b.   * Lisa thinks that Bart will hug.

We can encode these restrictions in the lexical items for *think*, *wonder*, and *hug*:

(36)   *think*, V, $[+\_\_ CP_{[+Decl]}]$

(37)   *wonder*, V, $[+\_\_ CP_{[+Q]}]$

(38)   *hug*, V, $[+\_\_ DP]$

And we can define lexical items for the two different C nodes that embed sentences. *Whether* (and *if* will be the same) is a C for questions (yes-no questions, specifically), while *that* is a C for declaratives. Also noted here is the fact that *that* is a C that is only compatible with a tensed clause (so, not with infinitive clauses).

(39)   *whether*, C, [+Q]

(40)   *that*, C, [+Decl], $[+ \_\_ TP_{[+Tense]}]$

Now, turning to *wh*-questions like *Who will Bart hug?*, we can embed such questions as objects of *wonder* but not of *think*. The generalization from before was that *wonder* requires questions and *think* forbids questions. So the facts below constitute evidence that *wh*-questions are questions just like yes-no questions are. *Wonder* needs a question, but it doesn't care whether that question is a yes-no question or a *wh*-question.

(41)   Lisa wonders who Bart will hug.

(42)   * Lisa thinks who Bart will hug.

5.2 *Wh*-phrases

There are a few different *wh*-words. *Who* and *what* are relatively easily understood as being more or less like DP pronouns. But there are also *wh*-words like *when* and *where*. Looking at the examples below, it should be fairly evident that the *wh*-word that appears right after *ask* (in the position at the beginning of the embedded clause) takes the place of some part of the embedded sentence. So, *who* in (43a) takes the place of the object of *hug*—and remember from above that *hug* <u>requires</u> an object, so *who* really must be playing that role, but it is doing so from the beginning of the clause rather than in its normal post-verbal position. In (43b), *who* is taking the place of the subject. In (43c), *when* is taking the place of the PP *at noon*, and in (43d), *where* is taking the place of the PP *at school*.

(43) a. Lisa asked who Bart will hug at noon at school.
      b. Lisa asked who will hug Maggie at noon at school.
      c. Lisa asked when Bart will hug Maggie at school.
      d. Lisa asked where Bart will hug Maggie at noon.

From this, we conclude that *wh*-words are not all the same category. Some are DPs (like *who* and *what*), but some we will call AdvPs (like *where*, *when*, *how*, and *why*). The category of a *wh*-phrase corresponds (roughly) to the category the phrase it takes the place of in the analogous answer:

(44) a. Bart will hug <u>Maggie</u> at noon at school.
      b. <u>Bart</u> will hug Maggie at noon at school.
      c. Bart will hug Maggie <u>at noon</u> at school.
      d. Bart will hug Maggie at noon <u>at school</u>.

I say "roughly" because the normal category we assign to *where*, *when*, *how*, and *why* is "AdvP" even though in the examples above there are PPs in those positions. In general, the AdvP is the most generic adjunct, and all of the PPs we're looking at are adjuncts also. So, in that sense, PP and AdvP are interchangeable.

One other type of *wh*-question involves *which*-phrases. The category of *which* seems to be the same as the category of *the* (D, that takes an NP complement).

(45) Lisa wonders which child Bart will hug.

(46) Lisa wonders which child will hug Maggie.

There are actually parallel non-*wh* pronouns for at least some of the *wh*-words, and we'll assume that at least when there is a correspondent, both have the same category.

where	there
when	then
what	that
why	hmm
who	ok
how	well

Consider again the fact that *hug* requires an object. And *who* seems to be taking on that role in (43a). But it also serving to mark the sentence as a question as well. So, the *wh*-word *who* is doing two jobs—one job for the verb, and one job for C.

Perhaps abstractly, this is a little bit like the two jobs that "Bart" seems to be doing in (48)—there, "Bart" is the *wanter* but also the *hugger*. The conclusion we reached in the case in (48) was that this isn't possible—that if there are two jobs to do, then there must be two things in the tree. And so, one of them must be silent (PRO).

(47)   Bart wants Lisa to hug Maggie.

(48)   Bart wants to hug Maggie.

(49)   Bart persuaded Lisa to hug Maggie.

The conclusion we'll reach in the *wh*-question case is that the *wh*-word is in both places—because it has moved from one place to the other. That is to say, the way that the "one thing doing two jobs" is handled here is that is in both places, at "different times." (And there are people who have argued that this is how we should even interpret PRO—that it's really just the lower position of a subject that moves from a lower clause to a higher clause. We'll continue to consider the two solutions to the problem of "needing to be in two places at once" as different, though: PRO in cases of two many roles for a single argument, and movement in cases where one of the places it needs to be is up in the CP.)

So, the situation is as below, where, e.g., *who* in (50a) starts out in the regular object position, and then moves up to the beginning of the embedded CP.

(50)   a.   Lisa asked [$_{CP}$ who Bart will hug <who> at noon at school].

         b.   Lisa asked [$_{CP}$ who <who> will hug Maggie at noon at school].

         c.   Lisa asked [$_{CP}$ when Bart will hug Maggie <when> at school].

         d.   Lisa asked [$_{CP}$ where Bart will hug Maggie at noon <where>].

Where does the *wh*-phrase move? Well, it moves to the leftmost edge of the embedded CP in the examples in (50) above. And if we have a main clause (non-embedded) *wh*-question, it's clear that it is to the left of C. There are pretty much exactly two places it could be going given our X-bar assumptions.

What are they? The specifier of CP is one of them. What's the other one?[1] The reason that we're going to go with the specifier of CP as the target of *wh*-movement is that it makes some sense of why only one *wh*-word moves (in English, at least). There's one specifier, and all we need to do in a *wh*-question is get a *wh*-word into it. So, e.g., (51b) is ok.
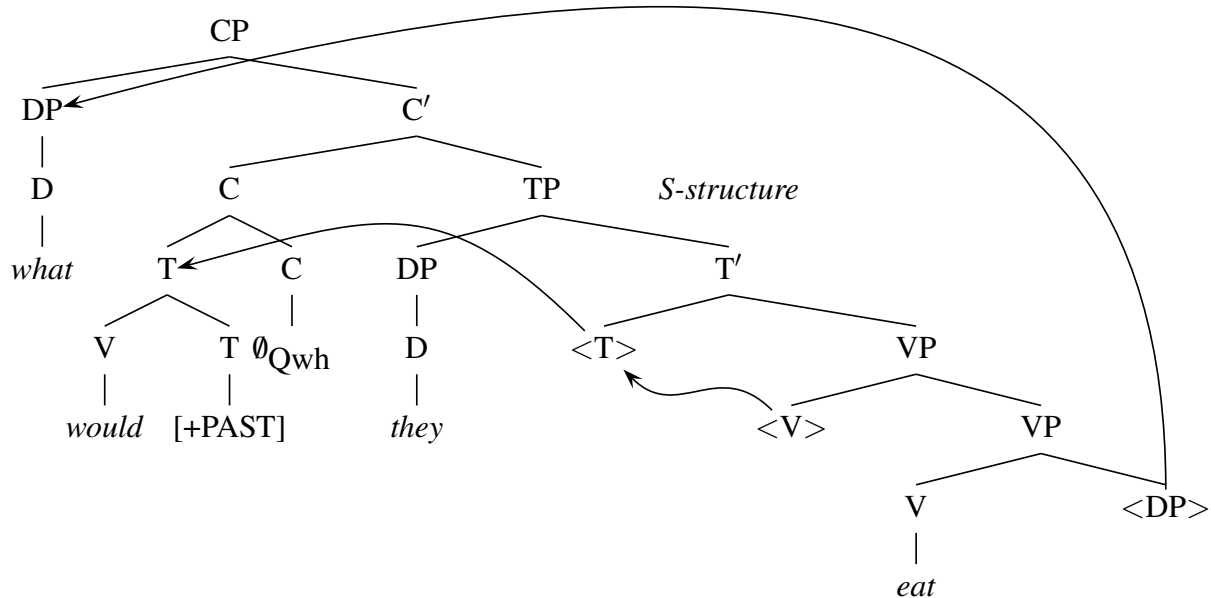
(51)   a.   What does Pat think [that Chris gave <what> to Tracy]?

         b.   What does Pat think [that Chris gave <what> to whom]?

         c.   Who(m) does Pat think [that Chris gave pizza to <who(m)>]?

         d.   * What who(m) does Pat think [that Chris gave <what> to <whom>]?

By the way, I alternate between saying "*wh*-phrase" and "*wh*-word" sometimes—the term *wh*-phrase is more general and more correct, really, because it includes phrases that act as interrogative phrases but contain more than just the *wh*-word, like those below. Also, the movement of a *wh*-phrase to Spec,CP is generally called "*wh*-movement."

(52)   a.   Which pizza does Pat like <which pizza>?

         b.   Whose book will Pat steal <whose book>?

         c.   To whom should Pat give that book <to whom>?

---

[1]Why, the other potential position at the left edge of CP but not the specifier of CP would be an adjunct to CP. But if we moved *wh*-phrases up to be adjuncts of CP it is not clear why you can't just move all of the *wh*-phrases you have up to there (see (51d)).

Ok, by now I'm sure you're wanting to see an example. You probably should have been able to guess what this would look like, but a S-structure tree for *What would they eat?* would look like this:

```
                        CP
        DP                      C'
        |
        D              C              TP        S-structure
        |
      what       T       C      DP              T'
              |         |     |
              V    T Ø_Qwh    D      <T>              VP
              |     |         |
           would  [+PAST]    they         <V>              VP
                                                      V          <DP>
                                                      |
                                                     eat
```

So: V moves up, adjoins to T, then T (with V adjoined to it) moves up and adjoins to C. The *wh*-phrase moves up to the specifier of CP.

- The topmost (closest) [AUX] V (if there is one) moves to (adjoin to) T.

- The T moves to (adjoin to) C if C is [+Q] and in a main clause.

- The topmost (closest) *wh*-phrase moves to Spec,CP if C is [+wh].

I've snuck a couple of new things in there. One isn't really new, but a reminder: In both *wh*-questions and yes-no questions, T only moves to C <u>in main clauses</u>. If you move T to C in an embedded question, the result is ungrammatical.

(53) a.   Should we buy pizza?
      b.   I asked whether we should we buy pizza.
      c. * I asked whether should we buy pizza.
      d. * I asked should whether we buy pizza.

(54) a.   What should we buy?
      b.   I asked what we should we buy.
      c. * I asked what should we buy.

Also a reminder, it is only the <u>topmost</u> [AUX] V that moves to T. We can think of this as something that T "needs" to have happen—and once we have moved an [AUX] V up to T, T is satisfied. But T is both needy and lazy, so it is not going to move an [AUX] V that is further away than the one that is closest.

(55) a.   Pat has not <has> been eating pizza.
      b. * Pat been not has <been> eating pizza.

The fact that only the topmost *wh*-phrase can move is based on the distinction below. Moving the higher of the *wh*-phrases to Spec,CP is fine, but moving the lower one is ungrammatical.

(56)   a.   What does Pat think [that Chris gave <what> to whom]?

      b.   * Who(m) does Pat think [that Chris gave what to <who(m)>]?

The last thing that I think I snuck in is that C has a [+wh] feature if its clause is a *wh*-question. That is to distinguish between yes-no questions and *wh*-questions. Both are [+q] (as we saw from the fact that both kinds of question can be embedded under *ask*), but only *wh*-questions have *wh*-movement, so to distinguish them we say that *wh*-question C has [+wh] as well as [+q].

Now, there is one complexity in the creation of the tree above that I have skipped over. I gave the S-structure for the tree, but what came before that? What was the D-structure?

There's a kind of technical wrinkle here, having to do with whether there is a C′ node in the D-structure. Generally, we don't include X′ nodes unless there is a specifier. So, does CP have a specifier before anything moves there? Or does CP gain a specifier (and therefore a C′ node) only when the *wh*-phrase moves?
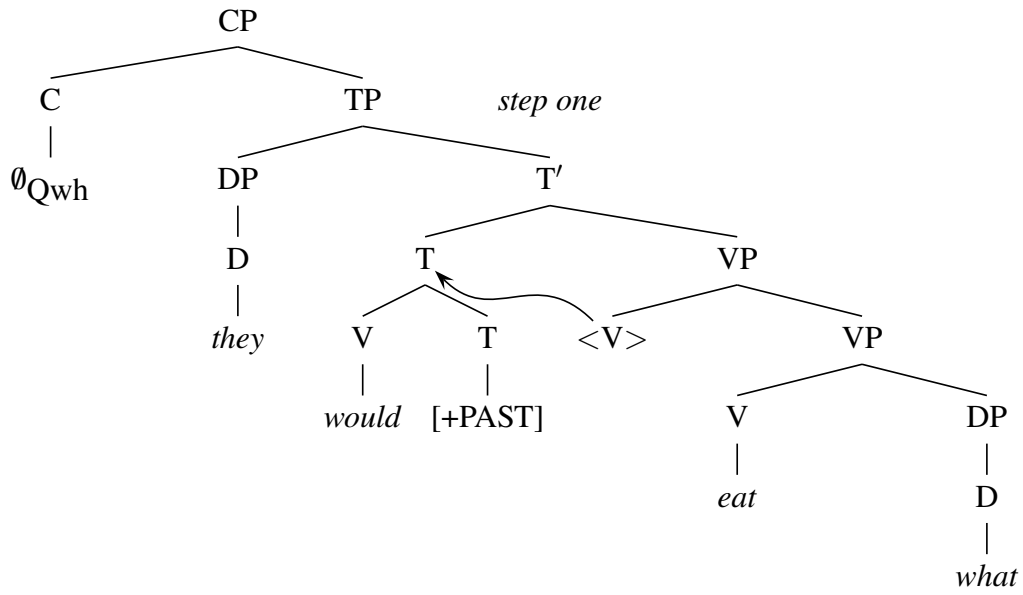
This is something that there really can't be any evidence for or against, it's at best a decision about what constitutes the simplest theory. But it's probably something we should make a call on so that we can draw D-structure trees of *wh*-questions. The call I will make is:

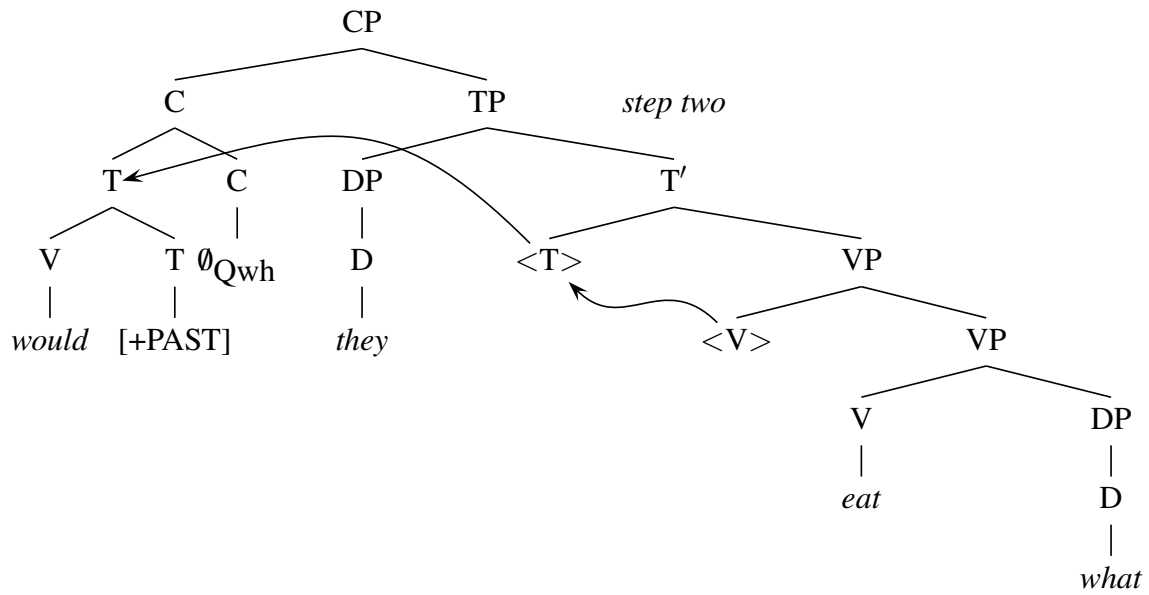- Movement to a specifier position creates the specifier (introducing an X′ node).

So, the D-structure of *What would they eat?* and first two steps would be as below (with the last step being the tree given before).
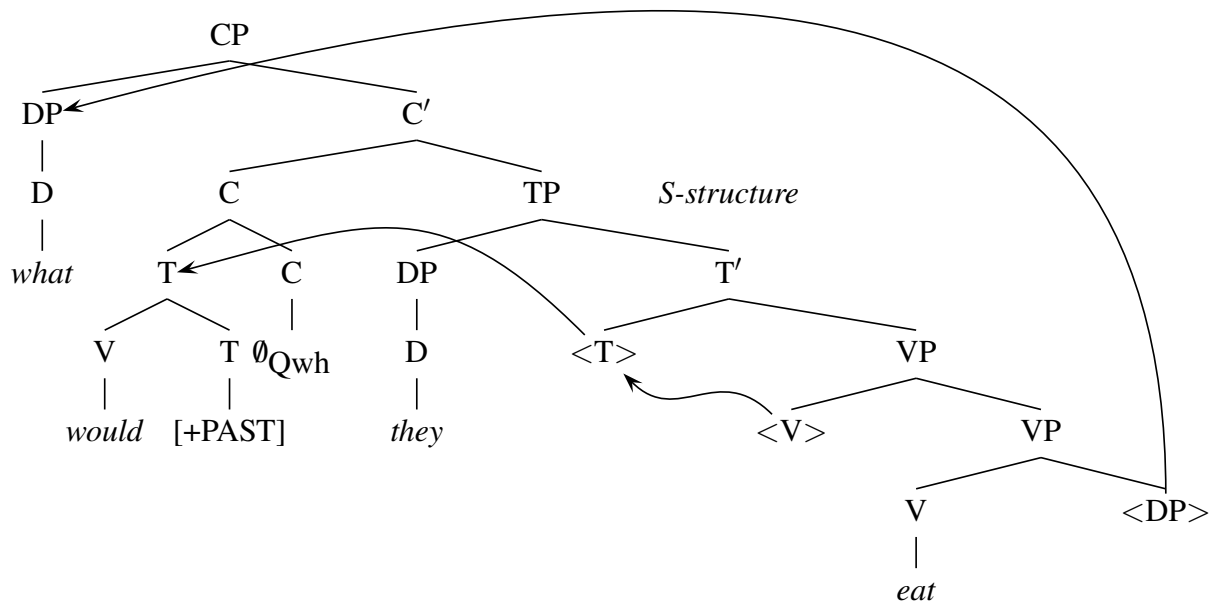
Move highest [AUX] V to T:

CP
C
Ø_{Qwh}
TP *step one*
DP
D
they
T'
T
V *would*
T [+PAST]
<V>
VP
VP
V *eat*
DP
D *what*

Move T to main clause [+q] C:

CP
C
T
V *would*
T [+PAST]
C Ø_{Qwh}
TP *step two*
DP
D *they*
T'
<T>
<V>
VP
VP
V *eat*
DP
D *what*

19
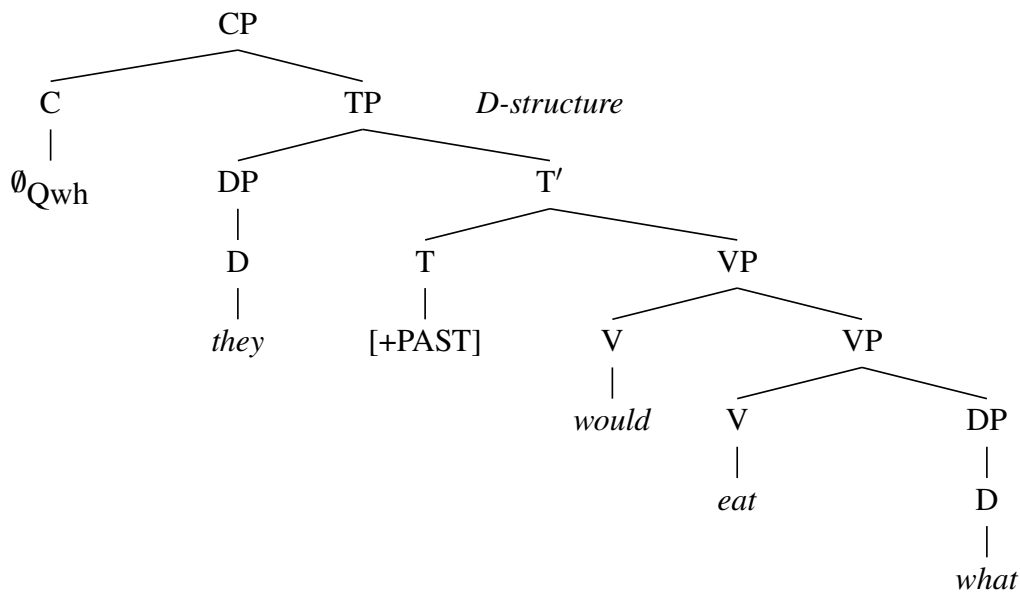
Move closest *wh*-phrase to (create) Spec,CP (what was CP is re-labeled C′):

CP
DP — C′
D
what
C — TP — *S-structure*
T — C — DP — T′
V — T ∅$_{Qwh}$ — D
would [+PAST] *they*
⟨T⟩ — VP
⟨V⟩ — VP
V — ⟨DP⟩
eat

5.3 Misc

A couple of final notes from the *wh*-movement section of the previous handout:

If you needed further evidence that the *wh*-phrase that moves is still also doing a job down in its origianl position, consider:

(57)   She expects [Lisa to embarrass herself].

(58)   Who does she expect [[who] to embarrass herself]?

In the interpretation (57), *she* and *Lisa* have to be different people. The reason for this is not something we will dwell on, but it goes essentially like this:

(59)   A name (like *Lisa*) cannot be c-commanded by a co-referential pronoun (like *she*).

So, the reason that *Lisa* and *she* have to be different people in (57) is that we have this constraint (59) (known as "Principle C") that prevents them from being the same (that is, being "co-referential").

Now consider (58) in this light. We have replaced the subject with *who*, and moved it to the main clause Spec,CP. In its new position in Spec,CP, *who* is not c-commanded by *she*. So, (59) shouldn't forbid *who* and *she* from being co-referential. Yes, they still cannot be: (58) cannot mean 'who is the *x* such that *x* expects *x* to embarrass *x*' (or, maybe better, 'Who expects to embarrass herself?'). Why? Well, (59) *does* constrain *who* in its original position: where *who* starts, *she* c-commands it. And this is enough to prevent *who* and *she* from being co-referential.

There was one other miscellaneous fact at the end of the handout before that I'll mention here, for "fun."

In (60), *himself* can refer to *Bart* but not to *Homer*. Again without spending too much time on this we believe that this is because *Homer* is too far away from *himself*. It is the subject of a higher clause, not in the same clause as *himself*. So the only meaning (60) can have is that (Homer thinks that) Bart will prefer this picture of Bart.

(60)   [CP Homer thinks [CP that Bart will prefer this picture of himself]].

If we replace *this picture of himself* with *which picture of himself*, it is now a *wh*-phrase and moves to Spec,CP.

(61) [_{CP} [which picture of himself] does Homer think [_{CP} that Bart will prefer <wh px of self>]].

Note: There are two CPs here. The outermost (main clause) one is a *wh*-question. The innermost one is still a declarative (headed by *that*).

Weirdly, (61) is ambiguous now. In this question *himself* can still refer to Bart, but it could alternatively refer to Homer.

You can think about that on your own time, but the reason that it can refer to Bart is that in the original position, *which picture of himself* is c-commanded from nearby by *Bart*. In its final position, *which picture of himself* isn't c-commanded by anything. For *himself* to take the reference of another DP, it needs to be c-commanded by the DP nearby. We know from (60) that *Homer* is too far away from *himself* to provide a referent when *which picture of himself* is in its original position. What to make of that? Hmm.