Starting over

- Let’s take a tour of the system from the beginning, to help get a better “wide-angle” view of how everything fits together and to try to tie up the loose ends.

- This is the final statement of where we are, what you should take as the end result.

The lexicon

- The lexicon is where it all begins, where the component parts of a sentence come from.
- A sentence is a number of lexical items, arranged.
- Lexical items have certain properties, or features. Some are nouns, for example. Some are wh-words, some are quantifiers, some are tense.

- Every head we see in our trees came from the lexicon. So, AgrS, AgrO, C, T, v, these are also in the lexicon, components from which we build sentences.

The lexicon

- Since phonological realization and even aspects of meaning can be considered to be properties of lexical items, really what a lexical item is is a bunch of features, bundled together. A thing, with properties.
- Some of the properties lexical items have are in the form of requirements, which need to be satisfied by the time the syntactic structure is finished.

DS

- The first step in constructing a sentence is arranging the lexical items into a DS (etymologically “Deep Structure”) tree.
- Lexical items have certain requirements that need to be satisfied in the initial arrangement at DS.

- The two most important DS concerns are role assignment and categorial selection.

DS: [-theory

- Lexical items can be classified in terms of being predicates or arguments.
- Predicates require something else for the computation of their meaning. They might be considered to be relations between the facts of the world (“truth”) and some other entity.
- Arguments are those other entities, that are placed in relations. These are often DPs, like John or the sandwich.
**DS: θ-theory**

- The number of participants that predicates require are at the heart of θ-theory.
- The θ-criterion says that:
  - Every θ-role required by a predicate must be assigned to some argument.
  - No argument can play more than one role.
  - No argument can be inserted superfluously; every argument must get a θ-role.

**DS: θ-theory**

- The number (and type) of θ-roles assigned by the predicates are recorded in the lexicon.
- “Weather verbs”: assign no θ-roles, there are no participants (e.g., rain, snow).
- Transitive verbs: assign two θ-roles, often Agent and Theme.
- Intransitive verbs: assign one θ-role, can be Agent (unergative verbs) or Theme (unaccusative verbs).
- Ditransitive verbs: assign three θ-roles, often Agent, Theme, Goal.

**DS: Categorial selection**

- Another requirement on DS is categorial selection.
- This refers to the concept that, e.g., C requires a TP (or, perhaps, AgrSP) sister.
- In fact, this can be considered to be an extension of θ-theory, and should probably just be considered to be a more general (more abstract, less intuitive) form or θ-role assignment.
- C assigns an abstract role to something of the type TP represents, like V or P assigns an abstract role to its object.
- We will not name these or write them down anywhere, but this is behind the idea that the tree looks like CP-(AgrSP)-TP-(AgrOP-vP-)VP.

**DS**

- So, DS is assembled from lexical items in accordance with their (sometimes abstract) θ-role assignment requirements.
- The assembly of this structure must also satisfy structural requirements on how trees are put together: X-bar theory.

**X-bar theory**

- X-bar theory is a statement of the strict requirements on the kinds of structures that syntactic trees are and are composed of.
- Every lexical item is a head.
- Every combination in the tree is a binary one, no node has more than two daughters.
- There is only one head (except contained in a different XP phrase) in any XP structure.

**X-bar theory**

- The sister to the head (the complement) is a unique position.
- The daughter of XP unrelated to the head (the specifier) is a unique position.
- The level of combination above the sister is a “bar-level” label (X').
- Every X-bar structure has an intermediate level (X').
- Every mother node between the head (X) and the phrase (XP) has one member which is related to (“projected from”) the head X.
**X-bar theory and [-] theory**

- The two unique positions in an X-bar structure (complement and specifier) are the positions available—and in fact are the only two positions available—for assignment of a [-] role required by the head.
- If X has a [-] role to assign, it must assign it to either ZP or WP. There are no other alternatives (“locality of [-] role assignment”).

**X-bar theory and [-] theory**

- There is also a recursive position in the X-bar template, the adjuncts, which are sister to X’ and daughter of X’.
- Any number of adjuncts (on either side) may be in an XP.
- These are generally “modifiers” (AdvPs, AdjPs, or PPs).
- Adjuncts are not eligible for [-] roles, and hence are never required for a given head.

**The X-bar template: adjunction**

- One last extension of the X-bar template is the possibility of adjunction at the XP and X level.
- We encounter with these have been always been as a result of movement (so they do not exist at DS, but they do constitute part of the requirements).
- We think of these as taking a head H and “hanging it off of” the head X, or a phrase UP and “hanging it off” of the phrase XP.
- Although there are two XPs drawn and two Xs drawn, they are in a sense a single node, stretched out.

**The X-bar template: adjunction**

- The main thing this concept of a “stretched” out node affects is what c-commands what in this structure.
- Dominance: A node [] dominates a node {} if {} is contained within all of []
  - Under this definition XP does not dominate UP, because part of XP does not contain UP.
- C-command: A node [] c-commands a node {} if:
  - {} is not contained in [], and
  - every node [] that dominates {} also dominates {}.
- By contained in, we mean either dominated by or “hanging off of”.

**The X-bar template: adjunction**

- C-command: A node [] c-commands a node {} if:
  - {} is not contained in [], and
  - every node [] that dominates {} also dominates {}.
- Does H c-command WP?
  - Is WP contained in H? No.
  - Does every node that dominates H dominate WP?
  - X does not dominate H.
  - X? X dominates H and it dominates WP.
  - The rest? They dominate H and dominate WP.
  - So, H c-commands WP.
The X-bar template: adjunction

- C-command: A node [] c-commands a node [ ] if:
  - [] is not contained in [], and
  - every node [ ] that dominates [] also dominates []
- Does H c-command X?
  - Is X contained in H? No.
  - Does every node that dominates H dominate X?
  - X? X dominates H and it dominates X.
  - The rest? They dominate H and dominate X.
  - So, H c-commands X.

The X-bar template: adjunction

- C-command: A node [] c-commands a node [ ] if:
  - [] is not contained in [], and
  - every node [ ] that dominates [] also dominates []
- Does ZP c-command UP?
  - Is UP dominated by ZP? No.
  - Does every node that dominates ZP dominate UP?
  - No—XP dominates ZP but not UP.
  - So, ZP does not c-command UP.
- Does XP c-command UP?
  - No.
- Does X c-command H?
  - No.

The X-bar template: adjunction

- C-command: A node [] c-commands a node [ ] if:
  - [] is not contained in [], and
  - every node [ ] that dominates [] also dominates []
- Does H c-command X?
  - Is X contained in H? No.
  - Does every node that dominates H dominate X?
  - X? X dominates H and it dominates X.
  - The rest? They dominate H and dominate X.
  - So, H c-commands X.

The X-bar template: adjunction

- C-command: A node [] c-commands a node [ ] if:
  - [] is not contained in [], and
  - every node [ ] that dominates [] also dominates []
- Does ZP c-command UP?
  - Is UP dominated by ZP? No.
  - Does every node that dominates ZP dominate UP?
  - No—XP dominates ZP but not UP.
  - So, ZP does not c-command UP.
- Does XP c-command UP?
  - No.
- Does X c-command H?
  - No.

The X-bar template: adjunction

- Nothing in X-bar theory prevents multiple adjunction, indefinitely.
- So any number of things can adjoin to XP (e.g., quantifier phrases, adjoin to TP).
- GP c-commands UP, XP, etc.
- UP is-commands XP, ZP, etc.
- Extra stipulation: In multiple adjunction to XP higher adjuncts c-command lower adjuncts. CP c-commands UP, UP does not c-command GP.
- Side note: these adjunctions to X and XP can be considered to be, in fact, basically the same as X adjunctions when you come right down to it, they haven’t been drawing double-branches.

Back to DS

- So, the number and type of []-roles assigned are properties of predicates.
- DS must be built as an X-bar compliant structure where the []-criterion is satisfied.
- X-bar structures allow only to positions to which []-roles can be assigned, complement and specifier.
- Ditransitive verbs, in a sense, cannot exist.
Ditransitive verbs

- In order to assign three \( \ell \)-roles, we need two XPs, which we’ve drawn like this.
- The labor of assigning \( \ell \)-roles is divided between \( v \), the light verb that assigns the Agent \( \ell \)-role, and \( V \), the main verb that assigns the Theme and Goal \( \ell \)-roles.
- We’ll come back to speculate about how \textit{give} can require a \( v \).

Nonverbal predicates

- Verbs are not the only predicates.
- Prepositions also have \( \ell \)-roles to assign, which they assign to their complement:
  - \textit{On the ship}.
- Sometimes PPs can themselves get a Goal type \( \ell \)-role, which they in a sense “transmit” to their object.
  - \textit{Give the book to Mary}.

Nonverbal predicates

- \textit{Be} does not assign \( \ell \)-roles.
- Possible exceptions:
  - \textit{Be} meaning “equation”, which assigns some kind of \( \ell \)-role to each of the two things being equated.
    - The answer is “four”.
  - \textit{Be} meaning “exists”, which assigns some kind of \( \ell \)-role to the thing that exists.
    - There is a solution.

Nonverbal predicates

- In purely auxiliary uses, since \textit{be} doesn’t assign \( \ell \)-roles, we assume that adjectives and nouns can also in some circumstances assign a \( \ell \)-role.
  - \textit{John is tall}.
  - \textit{John is the president}.
- Note that \( \ell \)-roles assigned by \textit{tall} and \textit{president} must be assigned to either the complement or specifier of \textit{tall} or \textit{president}, respectively.

Nonverbal predicates

- So, we end up with DS representations like this.
- By convention, we assume that the “subject” of such predicates is in the specifier, though we have no evidence that it isn’t in the complement.
- \textit{President} is a property that’s true of \textit{John}. \textit{Tall} is a property that’s true of \textit{John}.

Bill tried PRO to leave

- There is a class of verbs which embed nonfinite clauses that seem to be “missing an argument”: \textit{try}, \textit{want}, ...
- Think about the \( \ell \)-roles: \textit{leave} has to assign a \( \ell \)-role, to the leaver, and \textit{try} has to assign two \( \ell \)-roles, one to the proposition (TP) wanted, and one to the trier (\textit{Bill}).
- But we only see two of those arguments: the TP and \textit{Bill}.
Bill tried PRO to leave

- Starting with try, we know that its two [-] roles have to be assigned within the VP that want heads. So, Bill must start in the specifier and the TP must start in its complement.
- (Since Bill is an Agent, we’d actually assume that it starts in Spec vP, and the TP is the complement of VP)

Bill tried PRO to leave

- Since our DS is only legitimate if all of the [-] roles have been assigned, and no argument can receive two [-] roles, it can’t be that Bill is getting the [-] role from leave.
- There must be something there, but we can’t see anything there.
- Therefore, there must be something we can’t see there, PRO.
- PRO is a DP that starts in Spec vP of leave.
- PRO is the only DP that does not need Case.

Other notes about DS

- A well formed DS has a C, indicating what the force of the clause is.
- [-Q] statement, assertion.
- [+Q] question.
- [+wh] information-seeking question.
- [-wh] yes-no question.
- Possibly others, e.g., [+Imp] for imperatives, [+Exc] for exclamation.

Other notes about DS

- Embedded clauses have more freedom; it is possible to embed nonverbal predicates without a TP (I want John off the boat) or with a TP (I want John to be off the boat), and in some cases with a CP (I know that John is on the boat), or without (I want John to be off the boat).
- Policy: Finite clauses always have a CP. Nonfinite clauses are simply TP unless there is evidence to indicate a CP (e.g., I know what to buy).

Other notes about DS

- Main clauses always have a CP and a TP.

Other notes about DS

- Once DS has been arranged to satisfy the [-] criterion, so that the [-] role requirements of the lexical items are satisfied in a structure that conforms to the X-bar template, further requirements imposed by the lexical items must be taken into account.
- DPs need Case to be assigned/checked.
- [+Q] C needs T to move up to it.
- etc.
- These are requirements that force a single lexical item to be “in two places at once”, hence it has to move from its DS position to the place where it for other reasons needs to be. Note: Movement happens for a reason.
Further requirements

- T needs to have something in its specifier (EPP).
- A [+Q] C needs to have T move up to it.
- A [+WH, +Q] C needs to have a [+wh] XP in its specifier.
- A DP must have Case checked.
- A wh-word must be in SpecCP.
- A quantifier must bind its trace (hence must adjoin to AgrSP/TP).
- A v needs to have V move up to it.
- A T needs to have a V/v move up to it. (French)
- A C needs to have a T move up to it (German)
- C needs to have something in its specifier (German)

Timing

- It turns out that not all languages appear to meet all of these requirements. Japanese has wh-words that remain in situ, as do English objects, quantifiers, and wh-words other than the first one.
- In French, T seems to “need a V”, but in English, it doesn’t seem to.

Timing

- Given that we need to suppose that there is a final point in the derivation suitable for logical interpretation (LF), the assumption has been that languages can vary in which of those requirements their lexical items must satisfy by SS.
- SS is the “pronunciation focus” of the trip between DS and LF, the part of the derivation that we pronounce.

Timing

- On this view, we might be able to make a stronger statement than we’ve made previously: All languages look alike at both DS and LF.
- This is actually a research program—whether it will work out in the end remains to be seen, but things look like this might be right, and where there are places that it doesn’t look like it works, this motivates research questions to see what might be going on to make it appear that way.

Timing

- What this would mean is that what differs among languages is primarily which requirements of lexical items must be satisfied by the time SS, the pronunciation focus, is reached.
- Those which don’t have to be satisfied by SS are delayed until after SS. (“Procrastinate”)
- In English: EPP, T-to-C[+Q], V-to-T (for auxiliaries only), single wh-movement (requirement of C[+Q, +WH]).

SS: EPP

- Lexical items of the T category have a special requirement, that they head a TP with something in its specifier.
- As far as we can tell, this seems to be a requirement that has to be satisfied by SS in all languages. A universal.
- To solve this problem, the subject DP is moved up out of vP/VP and into SpecTP.
SS: [+WH] C
- In many languages, English included, the C for wh-questions (with features [+Q] and [+WH]) requires a wh-phrase (that is, an XP with the property that question words have, a [+wh] feature) in its specifier.
- To solve this problem, a [+wh] XP (for example, a [+wh] DP what) moves into SpecCP.

XP movement
- Both the EPP and the [+WH] C requirement have in common that they result in the movement of a (satisfactory) phrase from lower in the tree up into the specifier of the element (T or C) that has the requirement.
- Both also seem to require that it is the closest satisfactory XP that moves.
- A principle of “least effort,” assuming it’s “harder” to move things longer distances. If you can satisfy the requirement with a short move, you must do that, you can’t use a longer move.

XP movement: Superiority
- For wh-movement (satisfaction of the [+WH] C requirement), this goes by the name of Superiority.
- Who, did John say t, bought what?
- ??What, did John say who bought t?

XP movement: EPP
- For the EPP, there isn’t really a comparable name other than “Shortest Move”.
- John, T[PAST] t, eat the sandwich.
  (“The sandwich did John eat”)

EPP: Raising
- Even non-finite T needs to have a specifier. In some cases, this results in a situation where a DP moves into “subject position” (SpecTP) of the nonfinite T, but then moves up to a higher SpecTP to satisfy its EPP requirement. This is “(subject-to-subject) raising”. Verbs that embed such nonfinite TPs are “raising verbs” (seem).
- John, T[PRES] seem [TP t, to eat constantly].

EPP: It as an alternative
- For raising predicates (seem, likely), there are two ways to satisfy the EPP. One way is to insert (expletive) it in SpecTP. This also happens with “weather verbs” (rain).
- John, seems [t, to eat constantly].
  It seems [that John eats constantly].
- Raising is preferred, but is not an option when the embedded clause is finite.
EPP: *It* as an alternative

- In general, when a DP gets case, it cannot be recruited for further movement to satisfy the EPP.
  - "John, seems [that t₁, eats constantly]."

- *John* would get case twice if this happened, once in the lower subject position and once in the higher one.

Expletive *it*

- When *it* is inserted to satisfy the EPP, notice that it *cannot have been there at DS*. Intuitively it is clear that it has no role to play, it is semantically empty, it can’t be receiving a [-]-role.

- Since you can’t have any non-predicates at DS that don’t receive [-]-roles, *it* has to be inserted between DS and SS.

Expletive *it* is really there

- We can see that *it* is really inserted into SpecTP (not just the way you pronounce an empty SpecTP, for example) because *it* can move too.

  - John, seems [t₁’ to be likely [t₁ to t₁ leave]]

  - It seems [that John, is likely [t₁’ to t₁ leave]]

  - It₁ seems [t₁ to be likely [that John₁ will t₁ leave]]

SS: Head movement

- A couple of other requirements English lexical items place on SS are that:
  - [+Q] C must have T move to it. ("Inversion")
    - Only for a main clause [+Q] C in English.
    - V must have V move to it.
    - T must have auxiliary (have or be) move to it, if one is there.

- These are satisfied by head movement, where the relevant head moves to adjoin to the head with the requirement.

SS: Inversion

- A [+Q] C (in a matrix clause) needs to have T move up to it.
  - (Matrix) [+Q] C "needs a T."

- When T adjoins to C, they are close enough that C is satisfied.
  - Perhaps "(Matrix) [+Q] C must include a T"

- Does T₁ c-command t₁?

PF: Pronouncing SS

- SS is the “pronunciation focus point” in the derivation. We pick that point specially to pronounce the tree.

- Note that basically we do this as soon as we’ve satisfied all the requirements that need to be satisfied before SS. That’s another way to say that if you don’t have to move before SS, wait ("Procrastinate")
PF: Pronouncing SS

- There are a couple of things that happen as you go to pronounce an SS tree.
- First, you have to say some things before other things.
  - This is the first time there’s really a concept of order—everything before was about hierarchy (c-command, dominance, inclusion).
  - The parameter about whether the head or the complement is pronounced first (SOV vs. SVO) might well be a parameter of pronunciation.
  - Policy: We’ll continue to draw SS trees as if the order reflects the pronunciation order (i.e., heads on the right for Japanese, heads on the left for English).

LF: Remaining requirements

- Once we’ve reached SS and satisfied all of the requirements that have to be satisfied by SS, there are still some further requirements that lexical items have.
  - These are requirements that were not specially designated as having to be satisfied before SS, but they still have to be satisfied before the derivation is done (LF).

Case

- All DPs must receive/check Case before LF.
- Earlier, we had taken this to be one of the requirements that had to be satisfied before SS.
- However, ever since we began to suppose that objects get accusative Case in SpecAgrOP, we must assume that at least objects usually get their Case (checked) after SS (in order to get the word order right).
- We don’t have much evidence with respect to whether the subject must have Case (checked) before SS or not, so for uniformity, let’s suppose it’s under the same restrictions as the object. (Policy)
  - I can think of exactly one argument to the contrary, that subjects must check Case overtly, an indirect argument from acquisition made in Wexler (1998). We will disregard this.

Structural case

- AgrSP, Subject agreement phrase, is where the subject receives nominative case.
- AgrOP, Object agreement phrase, is where the object receives accusative case.
- These are the structural cases, the cases assigned in specific places in the structure.
**Inherent case**
- There is a second way that a DP can receive Case, which is from something that assigns inherent case.
- These are generally Ps.
- In *on the hill, the hill* is a DP and needs case, but gets its (oblique) case from the P by virtue of being its sister. No movement required.
- One might say this requirement happens to be satisfied already at DS in such structures.

**Case**
- Only DPs receive/need Case.
- A DP can only get Case once.

**QR**
- Another requirement that needs to be satisfied before LF (but not necessarily by SS) is that quantifiers (*every N, some N, most Ns*) need to move out of the clause.
- This needs to happen because they need a trace (with Case), to bind as a variable.
- For every student x, John met x.

**QR**
- QR adjoins the quantifier to the clause (AgrSP if there is one, or TP if there isn’t).
- QR must happen for every quantifier.
- A quantifier is interpreted with its c-command domain in its scope.
- For multiple adjunction structures, we need the extra stipulation from earlier: QP, c-commands QP, but QP does not c-command QP.

**wh-scope**
- Just like quantifiers, all wh-words must move before LF.
- Unlike quantifiers, wh-words have a target, they need to all move to SpecCP.
- We treat this as adjunction to the existing (overly moved) wh-phrase in SpecCP (since they all have to fit in SpecCP and there’s only one SpecCP).

**What about this “all languages look alike at LF” deal?**
- There is another thing we need to consider if we want to suppose that all languages look alike at DS and at LF, and that what’s different among languages are which requirements they must meet by SS.
- Japanese wh-words do not move on the surface; they are in situ, they appear where their non-wh counterparts would.
- But we assume that all wh-words move to SpecCP at LF in Japanese as well.
- So, we say they do but their requirements need not be met by SS.
We have reason to believe that verbs in French move to (T then) AgrS. By the same logic, if the LF in French has V adjoined to (T adjoined to) AgrS, and the LF in English looks like the LF in French... Well, then (between SS and LF), V must move up that high even in English. What differentiates French from English is that the requirement on T in French is designated as having to be satisfied by SS. (That’s what “the verb moves in French and not English” means).

So how about the fact that V moves all the way up into C in German? Does that force us to say the same thing about French and English (but covert in the latter two cases)? Yes, presumably. This kind of logic can get very complicated very fast, particularly because we don’t yet know what all of the attested phenomena are. Policy: For the purposes of this class we will not consider covert head movement. Leave the V where it ends up at SS in English.

Another set of requirements that must be satisfied by LF are the principles of Binding Theory.
- Principle A: An anaphor must be bound in its binding domain.
- Principle B: A pronoun must be free in its binding domain.
- Principle C: An r-expression must be free.

Binding domain = smallest AgrSP containing the relevant element. These are generally not requirements that force movement. They are simply either met or not met at LF (resulting in a grammatical or ungrammatical sentence, respectively).

Somewhat similar to binding theory is the issue of how PRO comes to get its reference. Interpretively, PRO seems to be one of three types (a property of the higher clause verb):
- PROobj: Forced to co-refer to the higher clause object (John persuaded Bill PRO to leave).
- PROsub: Forced to co-refer with the higher clause subject (John tried PRO to leave).
- PROarb: Has an arbitrary someone/anyone meaning (PRO to leave now would be crazy).

Problems in the syntax are solved by movement; you arrange your DS according to the dictates of t-theory and then movement allows you to satisfy all of the other requirements on the lexical items in your sentence. But only some moves are possible—you can’t save a sentence from being ungrammatical if the only way to satisfy the requirements is with an impossible move.

One primary fact about movement is that it must be “upward” in the tree. Where X is a moved element and t is the trace of X (sitting where X moved from), the move is legitimate iff X c-commands t. Sometimes this is referred to as the “Proper Binding Condition”.

*What did you know who bought it?*
Head Movement Constraint

- When moving a head, you cannot move it far. You can’t “skip over” a closer head when moving a head—again, essentially, you have to make the shortest move you can.
- This boils down to saying: Only this kind of movement is possible (where YP is the complement of X; Y can move to X).

Subjacency

- The way Subjacency violations are avoided is through the use of successive-cyclic movement. A moving wh-phrase will stop off in each SpecCP on the way from its original case position to its scope position.
- If a SpecCP is full along the way, the wh-phrase would have to skip past that SpecCP, which would entail a movement that is too long (wh-island violations).

Wh-islands

- Subjacency. A’-movement cannot cross more than one bounding node.
- TP is a bounding node in English.

Subjacency

- Not only do movements of wh-words need to be as short as they can be (cf. Superiority), they also have an upper bound on how long they can be even if there isn’t a shorter competitor.
- Subjacency: A single movement cannot cross more than one bounding node.
- Bounding nodes (English): TP (if sister to C) and DP.
- Bounding nodes (Italian): CP and DP.

Specific constructions

- Now that we’ve got the basics of the theory, let’s look at some other more specific ideas we have about various constructions.

CNP violation
Recall that in order to properly analyze ditransitive verbs, we needed to suppose that the VP is made of two “shells”, the vP and the VP. The vP is where the Agent $\ominus$-role is assigned. John will give the book to Mary. And given that we needed $v$ to assign the Agent $\ominus$-role in these constructions, we might as well assume that there is only one way that the Agent $\ominus$-role gets assigned: The Agent $\ominus$-role is only ever assigned to the specifier of vP.

Whenever there is an Agent $\ominus$-role (transitives, unergatives, ditransitives), there is also a $V$ to assign it.

There seems to be a correlation between a verb being able to assign accusative Case to its object and there being an external argument (Agent). (Burzio’s Generalization).

Translated into our terms, it seems that AgrOP (which is responsible for assigning accusative Case) can only be present if there is a vP assigning the Agent $\ominus$-role.

In cases where an embedded subject seems to get accusative case from the higher verb (I want Bill off the boat, I consider Bill to be annoying), this is due to raising the embedded subject into the higher clause’s AgrOP, as here.

Recall that one kind of verb that embeds a clause with PRO is the object control verb (I persuaded John PRO to leave).

These are like (well, they are) ditransitives, we need the vP structure to even be able to draw them.

You would draw them like this at DS, where DO later raises to SpecAgrOP (above vP) to get case (John persuaded me PRO to leave).

As mentioned earlier, the policy on embedded non-finite clauses is that they are just TPs unless there is evidence of a CP.

Consider: I know what PRO to buy.

We have evidence of a CP here, since what must be occupying SpecCP in the lower clause.
Embedded non-finite clauses

- The subject of a finite clause can get nominative case in its clause.
- Subject moves to SpecAgrSP in a finite clause, gets case.
- In a non-finite clause, nominative case is not available to the subject.
- Policy: Nonfinite clauses do not have AgrSP.
- Note: Nothing prevents a nonfinite verb from assigning accusative case, so AgrOP can be in a nonfinite clause (plus, the evidence from French in favor of AgrOP in the first place was about nonfinite clauses).

Object wh-phrases and Case

- Movement must always be upwards.
- Wh-objects like what (in What should I buy?) are DPs, and need to get Case like any other DP.
- Wh-movement to SpecCP happens before SS (in English). Objects don’t need to get Case (move to SpecAgrOP) until after SS.
- But if the wh-word is already in SpecCP, it can’t move back down to SpecAgrOP.
- The only option is for the object to stop off in SpecAgrOP on its way up to SpecCP.

Passives

- The effect of passivizing a verb like eat is that it loses the external \(+\text{Q}\)-role (vP) and the ability to assign accusative Case (AgrOP).
- So, a passive form a verb is drawn (at DS) without vP and, thus, without the associated AgrOP.
- Remember: AgrOP goes with vP—you don’t have AgrOP without vP.

Auxiliaries, tense, & aspect

- -ing is an Asp (the progressive), selected by be.
- Others would include -en (the perfect), selected by have, and -en (the passive), selected by be.
- Auxiliaries (be, have) head their own VP, but don’t assign \(+\text{Q}\)-roles to arguments, so nothing starts out in their specifier.
- This tree does not show the vP for write, but the “official structure” should have they starting in SpecvP, getting the Agent \(+\text{Q}\)-role.

Relative clauses

- The structure of a relative clause is like this.
- A \([+\text{Q}, +\text{WH}]\ CP is adjoined inside the NP, like an adjective, or a PP modifier.

Op

- Relative clauses can also make use of Op, the silent wh-word.
- That is, the book \(\text{which}\ Mary\ read\) and the book \(\text{Mary}\ read\) are really exactly the same except that in one case you pronounce the wh-word, and in the other, you don’t.
- the book \([\text{CP} \text{which}\ Mary\ read\ t_i]\)
- the book \([\text{CP} Op_t (that)\ Mary\ read\ t_i]\)
Op, DFC, & Recoverability

- The Doubly-Filled COMP filter is the traditional “explanation” for why “the book which that Mary read is bad.”
- Doubly-Filled COMP filter:
  \[\text{iff word if/that/for...}\]
- Recoverability condition: The content of a null category must be recoverable.
  - the place \(\text{[Op]}\) (that) Mary bought that book \(t, i\)
  - the day \(\text{[Op]}\) (that) Mary bought that book \(t, i\)
  - the reason \(\text{[Op]}\) (that) Mary bought that book \(t, i\)
  - the way \(\text{[Op]}\) (that) Mary bought that book \(t, i\)
- This is why you can’t just ask a regular \textit{wh}-question with Op.

Summarizing some: DS

- Lexical items must be arranged in conformance with the \textit{[j]} criterion and X-bar theory.
  - Agent \textit{j} role is assigned by \(\text{i}\).
  - AgrOP is only there if there is a \textit{j}P as well.
  - Auxiliaries head their own VP and take AspP as a complement.
- Finite clauses and main clauses always have a C and a T.
- Embedded nonfinite clauses only have a C if there is overt evidence for one.
- Nonfinite clauses do not have AgrSP.

Summarizing some: SS

- Universally (by SS in all languages):
  - SpecTP must be filled (EPP).
  - Move the closest eligible DP.
  - \(v\) moves to \(V\).
- Special head movements (by SS in some languages):
  - Main clause \(\text{[+Q]}\) C: T moves to C. (English)
  - Finite T: V moves to T (French, not English)

Summarizing some: SS/LF

- Languages can choose whether other things happen overtly (by SS) or just by LF.
  - SpecCP must be filled with an \textit{wh}-phrase \([+Q, +\text{WH}] C\).
  - All \textit{wh}-phrases must be in SpecCP for \([+Q, +\text{WH}] C\)
  - All quantifiers must bind a (case-marked) trace (moved to adjoin to AgrSP).
  - Object to SpecAgrOP for Case
  - Subject to SpecAgrSP for Case

Variation we’ve seen:

- English:
  - Subject moves to SpecTP overtly.
  - DPs move for case covertly.
  - (Topmost) auxiliary verb V raises to finite T overtly.
  - Main verb V does not raise higher than \(\text{SpecAgrOP}\).
  - First \textit{wh}-phrase moves to SpecCP for \([+Q, +\text{WH}] C\) overtly.
  - All other \textit{wh}-phrases move to SpecCP covertly.
  - All quantifiers move to adjoin to top of the clause (AgrSP or TP) covertly.
  - T moves to \([+Q] C\).
  - SVO (head-first) word order.

Variation we’ve seen:

- French:
  - Subject moves to SpecTP overtly.
  - DP’s move for case covertly.
  - Any kind of V (topmost aux or main V) raises to finite T overtly.
  - (Topmost) auxiliary verb V may raise to nonfinite T overtly.
  - Main verb V may raise to AgrO overtly.
  - First \textit{wh}-phrase moves to SpecCP for \([+Q, +\text{WH}] C\) overtly.
  - All other \textit{wh}-phrases move to SpecCP covertly.
  - All quantifiers move to adjoin to top of the clause (AgrSP or TP) covertly.
  - T moves to \([+Q] C\).
  - SVO (head-first) word order.
Variation we’ve seen:

- Irish, Arabic (VSO):
  - Subject moves to SpecTP overtly.
  - DPs move for case covertly.
  - (possibly overt of object over visible AgrO in one special case)
  - Any kind of V (topmost auxiliary or main V) raises to AgrS.
  - Main verb V may raise to AgrO overtly.
  - SVO (head-first) word order.

- German (SOV V2):
  - Any kind of V (topmost auxiliary or main V) raises to C in a finite clause.
  - SpecCP must be filled (V2).
  - SOV (head-final) word order.

Variation we’ve seen:

- Japanese:
  - All wh-movement to SpecCP covert
  - SOV (head-final) word order.
  - Possible to (optionally) scramble a DP to adjoin to AgrSP (like QR).