Some mid-term policy decisions and clarifications

- Proper names in English as DPs with Ø D.
- Pronouns are DPs with no NP inside.
- Main clauses are CPs
- Predicate-internal subjects, auxiliaries, nonfinite clauses.
- EPP holds in nonfinite clauses
- Expletives don’t get [-]roles.
  - Expletives are not there at DS.
  - [-]roles can only be assigned within the XP headed by the [-]assigner. So, within VP.
- ECM, embedded TPs.

Proper names

- Henceforth, we will consider proper names in English to be DPs with a Ø D head, in order to capture the crosslinguistically common form of proper names the Bill, as well as to allow for the Bill I know, etc.

Matrix clauses are CPs...

- We will also consider all matrix clauses to be full CPs.
- In questions, we need a CP headed by a [+Q] morpheme in C.
- In declaratives, we will assume that we have a CP headed by a (null) [-Q] morpheme.

Predicate-internal subjects

- VP-Internal Subject Hypothesis
  The subject of a verb originates in the specifier of VP at DS.
- This goes for other subjects of other predicates, e.g., small clauses like I find Bill intolerable.
- All [-]roles are assigned within the predicate’s own XP.
Internal subjects and auxiliaries

- Note that this means that the subject has to be in the specifier of the main verb in cases where there are auxiliaries. Not in the specifier of the auxiliary verb—it's the main verb which assigns the [ ]-roles.

- Also note: This has nothing to do with whether the clause is finite or not—this has to do with VP (or AP, etc.), not with TP. The subject is always in the specifier of the predicate.

EPP: Clarification

- The EPP is a constraint on TP, it says that SpecTP must be filled.

- It is not a property of finite T alone, it is a property of T in general. In particular, the SpecTP position of a nonfinite clause must be filled as well. This will be relevant later today.

Expletives and [ ]-roles

- Let me reiterate, the reason we have expletives at all is because we have a conflict between the [ ]-criterion and the EPP.
  - The EPP requires something in SpecTP.
  - The [ ]-criterion says we can only have as many arguments as there are [ ]-roles.

- In it rains, it is not present at DS—it cannot be, because it cannot get a [ ]-role (since there is none around for it to get), but is inserted between DS and SS in order to satisfy the EPP.

Government

- These three environments
  - Sister
  - Specifier
  - Specifier of sister
  - ...are together the positions which are governed by the head X.

- A Case-assigning head X can assign Case to a DP which is any of these positions.

- Case-assignment can only take place between a Case-assigner and a DP within the radius of government.

- Take this to be The Truth.
  - Bill wants me to leave.
  - Here the verb want assigns an Experiencer [ ]-role and a Proposition [ ]-role, the proposition assigned to the embedded clause.

- Me is getting Case from want, apparently, since it is accusative.
Case

Given what we’ve got so far, we might expect this structure.

But can this be right?

Can want provide Case for me?

Answer: No.

Want and me are too far apart.

Me is not in the government radius of want.

Instead, it must look like this, where there is no CP containing the embedded clause, just a bare TP.

Now, everything is fine.

So when do we have CP and when don’t we?

Finite clauses always have a CP (this includes matrix clauses now too.).

Nonfinite clauses generally don’t have a CP unless you can see it (unless there is a complementizer or some other evidence of CP).

I want for Bill to leave.

I want Bill to leave.

I don’t know what to buy.

Note! The textbook provides an altogether different analysis of how me gets Case in this sentence, under the name “object raising”.

Problem is, doing it the way the textbook does right now breaks X-bar theory and we don’t want to do that. So, for now, this is the official way to analyze these sentences.
Abstract thoughts
- Pick a bunch of things from the lexicon.
  - The lexicon is where we store all of our language-particular information—not only words like student, but also words like the and that and -ed.
  - Assemble them logically into predicates and arguments in a DS tree, using the X-bar schema.
  - Does every \( q \)-role of every predicate get assigned to exactly one argument? Does every argument get assigned exactly one \( q \)-role?
  - Clauses have information about force (question, statement, exclamation—\( C \)), tense and modality (past, present, certain, conditional—\( T \)), and predicate-argument combinations (VP).

Problems at DS
- The arrangement of things at DS is not good enough.
  - DP's need to be in one of the privileged positions in the structure (near a Case-assigner) — “DPs need Case” (Case Filter).
  - The specifier of TP cannot be left empty (EPP).
  - And some other things...
  - We think of these as requirements that need to be met, and often they are requirements imposed by a particular head in the tree.
  - \( T \) requires that its XP have a non-empty spec.
  - \( D \) requires that its XP be near a Case assigner.

Problems at DS
- More requirements of this sort...
  - Question-type C (that is, \([+Q]\)) needs to be near \( T \).
  - Hence in questions \( T \) will have to move up to \( C \).
  - \( T \) (when the type that gets realized as a suffix—e.g., -ed, -s, but not \textit{will} or \textit{might}) needs to be near an auxiliary verb if there is one.
  - Hence auxiliaries will have to move up to \( T \).
  - It’s a requirement of \( T \) not of the auxiliary.
    - John wasn’t under the table.
    - John will be under the table.
    - John hasn’t been under the table for some time now.

Crash?
- If you get to the end of the moves you can make and there is still some requirement left unmet, sometimes we say that the \textit{derivation crashes}. That is, the sentence you were trying to make is ungrammatical.

Solving problems through movement
- The problems with DS are generally that things which need to be next to each other aren’t.
  - Note that we couldn’t have put things next to each other initially at DS, though, because \( q \)-role assignment has to be local, among other things.
  - So, we \textit{move} things from one place in the tree to another, to satisfy the requirements. Move heads to adjoin to heads (but not over other heads), move XPs into specifiers (e.g., moving to SpecTP).

Syntax vs. phonology
- When something is pronounced differently from how you’d expect based on the DS, this could be either
  - Movement in the syntax
  - Alteration in the morphology
  - Hard to tell the difference.
  - Clues: In syntax \textit{movement is only upward} (moved element must c-command its trace).
Syntax vs. phonology

- Given that, do-support must be phonological.
- Some kinds of T have the morphological property that they are verbal suffixes—they are pronounced at the ends of verbs.
- If you try to pronounce T without a verb to hook onto, there’s no crash—the morphology just deals with it as best it can, by inserting the most meaningless verb, do, and pronouncing T on that.

Labeling

- When labeling things in a tree, there’s a certain amount of flexibility in what you write.
- [past], -ed, ...
- However, if a feature indicates a requirement that’s affecting the syntax, you need to put it in. Hence: C [+Q].
- If a pronunciation is seriously ambiguous as to what lexical item you have, this is also no good. For example: T -Ø. What tense is that?

Ø vs.

- We often distinguish the syntactic structure of something from its pronunciation.
- C is there in the tree, whether it’s pronounced that or not pronounced at all.
- We write the “silent C” as Ø.
- But if there’s no C at all, you don’t write Ø, since that means “silent C”.