Recall...

- Mary is likely to leave.
- Mary starts in SpecVP, gets a [q-role from leave.
- Mary moves up to the embedded SpecTP to satisfy the EPP.
- Mary still doesn’t have Case.

Recall...

- Mary is likely to leave.
- Mary starts in SpecVP, gets a [q-role from leave.
- Mary moves up to the embedded SpecTP to satisfy the EPP.
- Mary still doesn’t have Case.
- Mary moves up to main clause SpecTP, satisfying the EPP and getting Case.

Recall...

- This happens because likely assigns only one [q-role, an internal [q-role.
- Likely does not assign Case, and so Mary must keep moving, both to satisfy the EPP and to get Case.

Reluctance to leave

- Now, consider:
  - Mary is reluctant to leave.
- This looks very similar to Mary is likely to leave.
- Can we draw the same kind of tree for it?
- How many [q-roles does reluctant assign?
Reluctance to leave

- Reluctant has two [-] roles to assign.
  - One to the one feeling the reluctance (Experiencer)
  - One to the proposition about which the reluctance holds (Proposition)

- Leave has one [-] role to assign.
  - To the one doing the leaving (Agent).

- In Mary is reluctant to leave, what [-] role does Mary get?

Reluctance...

- Mary is reluctant to leave.
- Reluctant assigns its [-] roles within AdjP as required, Mary moves up to SpecTP in the main clause by SS.

But what gets the [-] role from leave, and what satisfies the EPP for the embedded clause?

Reluctance...

- Mary is reluctant to leave.
- There must be something there, getting the [-] role and satisfying the EPP.

But we can’t see it.

It’s a phonologically empty (Ø) DP. We will call it PRO.

Reluctance...

- Mary is reluctant to leave.
- PRO does not get Case.
  - “Mary is reluctant Bill to leave.
  - In fact, PRO cannot get Case.
  - “Mary is reluctant for to leave
  - Mary is reluctant for Bill to leave
  - PRO refers (like a pronoun or an anaphor) to Mary.

Reluctance to leave

- In Mary is reluctant to leave,
  - Mary is doing the leaving, gets Agent from leave.
  - Mary is showing the reluctance, gets Experiencer from reluctant.

And we have a problem:
- Mary appears to be getting two [-] roles, in violation of the [-] criterion.
If there’s a PRO, how do we know?
- Mary is reluctant [PRO_m to leave]
- Mary, is likely [ t_i to leave].

These two sentences look very much alike—when faced with a sentence that looks like this, how do we know which kind it is?

Best method for finding PRO: Count the [-] roles. If there appear to be fewer arguments than [-] roles (in a grammatical sentence), there must be a PRO.

Another way is to try with idioms like The cat is out of the bag or The cat’s got your tongue or The jig is up.

Idioms
- For something to have an idiomatic interpretation (an interpretation not literally derivable from its component words), the pieces need to be very close together at DS.
  - It is likely that the jig is up.
  - It is likely that the cat is out of the bag.
  - It is likely that the cat has your tongue.

It is ok if the pieces of the idiom move away after DS, we can still get the idiomatic interpretation:
- [The cat], is likely ti to have your tongue.
- [The cat], is likely ti to be out of the bag.
- [The jig], is likely ti to be up.

The important thing is that they are together at DS (the [-] role needs to be assigned by the predicate to the noun)

Idioms
- If we break up the pieces, then we lose the idiomatic interpretation and can only get the literal meaning.
  - The cat thinks that it is out of the bag.
  - The cat thinks that it has your tongue.

With PRO sentences ("control sentences"), we also lose the idiomatic reading.
- #The cat is reluctant to be out of the bag.
- #The cat attempted to have your tongue.
- #The jig tried to be up.

Unlike with raising verbs:
- [The jig], is likely [ t_i to be up]
Control

- PRO is similar to a silent pronoun; it gets its referent from somewhere outside its sentence. In many situations, however, PRO is forced to co-refer to a preceding DP, unlike a pronoun.
  - Bill, thinks that he is a genius.
  - Bill is reluctant PRO to leave.
- We say that PRO is controlled (here by the matrix subject).

Subject and object control

- There are actually two different kinds of “control verbs”, those whose subject controls an embedded PRO and those whose object does.
  - Bill, is reluctant [PRO to leave]
    - reluctant is a subject control predicate
  - John, persuaded Bill, [PRO to leave]
    - persuade is an object control predicate

PROarb

- Finally, there is a third use of PRO, in which it gets arbitrary reference and means something like “someone/anyone”.
  - [PROarb to leave] would be a mistake.
- The conditions on which interpretation PRO can/must get are referred to as Control Theory, although to this day the underlying explanation for Control remains elusive.

“Control theory”

- For now, what control theory consists of is just marking the theta grids of specific predicates (persuade, reluctant) with an extra notation that indicates when an argument is a controller.

<table>
<thead>
<tr>
<th>reluctant</th>
<th>Experiencer controller</th>
<th>Proposition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>i</td>
<td>j</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>persuade</th>
<th>Agent</th>
<th>Theme controller</th>
<th>Proposition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>i</td>
<td>j</td>
<td>k</td>
</tr>
</tbody>
</table>

“Control theory”

- Predicates that have a controller marked are control predicates. When the controller is the external argument, it is a subject control predicate, otherwise it is an object control predicate.

<table>
<thead>
<tr>
<th>reluctant</th>
<th>Experiencer controller</th>
<th>Proposition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>i</td>
<td>j</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>persuade</th>
<th>Agent</th>
<th>Theme controller</th>
<th>Proposition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>i</td>
<td>j</td>
<td>k</td>
</tr>
</tbody>
</table>

The PRO conundrum

- Back when we talked about Binding Theory, we said that DPs come in one of three types, pronouns, anaphors, and R-expressions.
  - PRO is a DP, so which kind is it?
    - It gets its reference from elsewhere, so it can’t be an R-expression.
    - It is sometimes forced to get its referent from an antecedent, like an anaphor and unlike a pronoun.
    - But that referent is outside its clause, meaning it can’t be an anaphor (the antecedent would be too far away for Principle A). Plus, it’s not always forced (PROarb), like a pronoun.
The PRO conundrum

- Back when we talked about Binding Theory, we said that DPs come in one of three types, pronouns, anaphors, and R-expressions.
- PRO is a DP, so which kind is it?
- Conclusion: It doesn’t seem to be any one of the three. It doesn’t seem to fall neatly under Binding Theory.
- ...hence, we need “Control Theory” to deal with the distribution and interpretation of PRO.

PRO: One possible piece of support

- Bill is reluctant [PRO, to buy himself, a gift]
- Bill, promised Mary [PRO, to buy himself, a gift]
- *Bill, promised Mary, [PRO, to buy herself, a gift]
- *Bill, promised Mary, [PRO, to buy him, a gift]
- Bill, promised Mary, [PRO, to buy her, a gift]
- *Bill, is reluctant [PRO, to buy him, a gift]
- While it’s true that Bill is outside of the binding domain of himself, and hence Bill cannot be the antecedent for himself, PRO is in the binding domain and its reference is controlled.

PRO: recap

- Although we can’t see that PRO is there, all of our theoretical mechanisms point to its being there.
- EPP says that clauses need a subject.
- The -criterion says that there must be exactly as many arguments as -roles.
- Binding Theory indicates something is present inside embedded clauses.
- If the rest of our theory is right, it seems that PRO must be there.

Control Theory

- Despite the fact that PRO does not submit to Binding Theory, there are some binding-theory-like requirements on control of PRO.
- PRO is only obligatorily controlled by a c-commanding controller.
- [Bill’s mother], is reluctant [PRO, / to leave]
So far, we’ve only talked about *is likely*, but there are a couple of other raising verbs as well.
- [The cat], seems [TP *t* to be out of the bag].
- [The cat], appears [TP *t* to have his tongue].
- [The jig], proved [TP *t* to be up].
- [The cat], began [TP *t* to get his tongue].

What these verbs have in common is that they have no external [-]role and an internal Proposition [-]role.

In fact, nothing keeps us from piling raising verbs one atop the other:
- [The cat] it seems [TP *t* to get his tongue].
- [The cat] it appears [TP *t* to have his tongue].
- [The jig] it proved [TP *t* to be up].
- [The cat] it began [TP *t* to get his tongue].

In these cases, the subject moves from SpecTP to SpecTP, only receiving Case at the last stop, satisfying the EPP at each TP.

Raising verbs will cause anything in a complement TP that isn’t getting Case to move up to their SpecTP.

Passive arguments:
- [The sandwich] it seems [TP *t* to have been [eaten *t*]].

Even expletive *it*:
- *It* began [TP *t* to rain].
- *It* began [TP *t* to seem [TP *t* likely *t* to be up]].

Here, *it* was inserted to satisfy the EPP in the most embedded TP, but then raised from SpecTP to SpecTP to satisfy the rest of their EPP conditions.

Some time ago we saw the term **chain** applied to the concept of positions occupied by a (moving) constituent in a structure.

Here, the **chain** for *The jig* is:
- (*The jig*), (TP *t* to *t*), (TP *t* to *t*). …referring to all the places its been in the tree.

Every (argument) chain gets exactly one [-]role.
Every (argument) chain receives Case. (except PRO’s)

Doing this allows us to avoid saying every argument gets case at some point, and a [-]role at some different point.

Many languages have the property that when the subject is understood (often in the cases where in English we would use a pronoun subject), it can be just left out entirely. For example, Italian:

*Parlo.*  
It ‘I speak’

*Parli.*  
*Parli.*  
It ‘You speak’

### Italian subjects

**Back to raising**

**Side note: Chains**

**Italian subjects**
Italian subjects

- So what about the EPP and the $\Box$-criterion? Clearly ‘speak’ assigns a $\Box$-role, and presumably the Italian SpecTP needs to be filled as well.

- This sounds like a familiar question… should we hypothesize that the subject in these sentences is PRO?

Little pro

- There is one important difference between the Italian null subject and PRO, namely the null subject in Italian appears in a position that gets Case.
  - Io parlo. ‘I speak’
  - Since PRO cannot appear in a Case-marked position, we have to take this to be something similar but different: Little pro.

Little pro

- Little pro is really just a regular pronoun, only null. It doesn’t have the fancy control properties exhibited by PRO, it appears in Case-marked positions.

- Languages seem to be divided into those which have little pro and those which don’t, often correlating with the amount of agreement on the verb (rich agreement makes it more likely that a language will have pro). Languages with pro are often called “pro-drop languages” or “null subject languages”.

Features and checking

- An elaboration…

- We assume that we have a lexicon full of items (“words”) that get inserted into terminal nodes of the tree. These items can be considered to be little collections of properties, or “features.”

Features and checking

- What do we know about she in English?
  - It’s a D
  - It’s pronounced “she”
  - It has nominative Case
  - It is 3rd person
  - It is singular
  - It is feminine

- These things we know are all properties, or features, of the lexical item she. (These are the grammatically relevant properties anyway…)

Features of T

- Now, let’s think about T.

- English T has features like [past] or [pres], and sometimes we’ve written [past] as -ed to indicate its pronunciation.

- But what determines the (regular) pronunciation of the affix in T?
  - I walk. You walk. He walks. They walk.
  - I walked. You walked. He walked. They walked.
Features of T

- It seems that both the tense feature and the person specification of the subject affects how T is pronounced.
- Why?
- The modern approach to this phenomenon (which often goes by the name of Spec-Head Agreement) is to suppose that there are features both on T and on the subject (for person, number) and that when they are in a Spec-Head relationship, the features are close to each other.

Spec-head agreement

- The reason it is important for the features to be close to each other is that the syntax needs to be able to check to make sure the features match. Spec-head counts as “close”.
- If the subject has different person features from the tense/agreement suffix in T, then the sentence is ungrammatical.

SpecTP

- Another thing SpecTP is famous for its ability to host nominative case-marked subjects.
- This is implemented in the same way, by analogy to agreement.
- To say that finite T is a nominative case assigner is to say that it has a feature [(Assign) Nom], and DPs like I and he have a feature [Nom].
- A subject “getting Case” in SpecTP is then not exactly getting Case so much as it is checking to be sure that the Case it has is the right one.
- Case has to be checked (guilty until proven innocent).

SpecTP

- This is really just another way to state the Case Filter (“DPs need (to check their) Case”) but it’s now in terms of a more specific understanding of what it means to “assign Case”.
- This also means that the “government radius” is a way to characterize the positions which are close enough for feature checking to occur.

Features and checking

- There is a distinction between features that need to be checked and features that do not.
- Case features like [nominative] need to be checked. These are the kinds of features which often motivate movement.
- Category features like [D] on a determiner are fine as they are, they don’t need to be checked against anything else.

Features and checking

- Another point worth observing about checking features like [Nom] on a DP is that it only happens once. Once you’ve checked to be sure that the Case is right, you’re fine—in fact, you can’t check it a second time.
- For this reason, sometimes people think of the features as being removed when checked (like on a checklist). Either way, you only check them once.
Case checking seems symmetrical
- Recall that we said T has a feature [Assign Nom], and this is checked against the [Nom] feature of a subject like we in order to validate the Case on the subject.
- There is actually reason to think that both the [Assign Nom] feature on T and the [Nom] feature on the DP need to be checked—and that each can happen only once.
  - Finite T needs to check Nom on a DP.
  - DPs need to check Case.

Moving to Case positions
- Consider:
  - It is likely that we will leave.
  - *We, are likely that t_i will leave.
- What’s the problem with the second one?

Moving to Case positions
- *We, are likely that t_i will leave.
- We moved up to the finite SpecTP, and checked off its [Nom] feature with [Assign Nom] feature of T. Both are now inactivated.
- But then we is moved up to the matrix SpecTP. Yet we no longer has an active [Nom] feature (it’s been checked already), so the matrix T can’t get rid of its [Assign Nom] feature.

Moving to Case positions
- It is possible to move solely for the EPP if there is no Case to check (i.e. in a nonfinite TP).
  - [The sandwich], is likely t| have been eaten t_i.
- So, we could have moved we to the matrix SpecTP—something else went wrong.
  - *We, are likely that t_i will leave.
- And what went wrong is that this leaves the matrix SpecTP without a DP to check its [Assign Nom] feature against.

So where are we?
- The generalizations here are:
- The Case Filter:
  - DPs are inserted into the structure with a Case feature which must be checked.
  - Case assigners are inserted into the structure with a Case-assignment feature which must be checked.
  - A DP cannot move from a Case-checking position to another Case-checking position.
  - Case checking only happens once (de-activating the Case feature on both the Case-assigner and the DP)

Wh-questions
- This was kind of complicated, but it was worth going through in order to set up wh-questions for next time.
- To get us started:
  - Wh-questions are information-seeking (not yes/no) questions, in English involving one or more of the ”wh-words” (who, where, what, when, why, how, …)
**Wh-questions**

- In a *wh*-question, we find that we do the same inversion that happens with yes-no questions… (moving T to C).
- Will, Bill *i* ate lunch?
- …plus, we move the *wh*-word into SpecCP:
  - What, will, Bill *i* eat *i*?
- This movement of *wh*-words is similar, but different, from the DP movement we’ve seen so far with passives and raising verbs.

**Wh-questions**

- With yes-no questions, we posited a [+Q] C at the head of CP, which caused the movement of T to C.
- For *wh*-questions, we can think of a different kind of C, a [+Q, +WH] C, which prompts both the movement of T to C and the movement of the *wh*-word into SpecCP.
  - (So, yes-no questions would have a [+Q, –WH] C)

**Wh-questions**

- What causes the movement of the *wh*-word to SpecCP is considered to also be a case of feature checking.
- In this case, the C has a [+WH] feature to check, and the *wh*-words have [+WH] features that can be checked against it.
- So, the *wh*-word is brought up into SpecCP to bring the features close enough for checking, and then presto! everybody wins.

**Wh-questions**

- Interestingly, looking at English, [+WH] feature checking appears not as symmetrical as Case checking. In particular, moving just one *wh*-word to SpecCP seems to be sufficient.
  - Who gave what to whom?
  - That is, all of the other *wh*-words can remain, seemingly “unchecked”.

**Wh-questions**

- *Wh*-words may check their [+WH] feature.

- In a sense, English *wh*-movement provides a pretty good motivation for a “feature” view of these phenomena. It appears that [+WH] C has a “need” which a *wh*-feature can satisfy, and once satisfied (even with other *wh*-words around), everything is fine.