CAS LX 522
Syntax I

Week 11a.
Possessors again, then CP & PRO (8.1-8.2.5)

Passive nouns

• Last week, we looked at the passive construction:
  1) The sandwich was eaten

• Here, the Theme the sandwich becomes the subject because the strong feature of T forces it to move to SpecTP. The v does not project an Agent.

Passive nouns

Very similar to the passive, if an n doesn’t introduce an Agent, the Theme can move to SpecDP and surface as genitive.

Passive nouns

• If the DP has a head D like the that does not check genitive case, then there can be no Agent (nothing could check its case), and the Theme stays unmoved (its of-case checked by n).

Case and θ-roles

• We now predict the observation Adger makes: Either an Agent or a Theme can show up in the genitive, but only a Theme can show up with of-case.
  1) Adger’s analysis of the DP is simple.
  2) The DP’s analysis is simple.
  3) *The analysis of Adger is simple.

• This is essentially the same as the generalization that, in a clause, either an Agent or a Theme can show up with nominative case, but only a Theme can show up with accusative case.
  1) I called her.
  2) She tripped.
  3) *Her tripped.
  4) *Tripped her.
Back to possession

• Prior to today, the genitive case was associated with the possessor. So far today we’ve been looking at deverbal nouns, where genitive case goes to the subject.
• Our new improved UTAH says, among other things:
  • DP daughter of NP: Theme
  • DP daughter of nP: Agent
• Possessors are neither of these, so possessors need to be initially Merged into a distinct place in the structure.

Possessors

[Diagram showing Possessors]

• Adger proposes that Possessors are introduced by a new head, Poss.
• HoP: D > (Poss) > n > N

Hungarian possessors

1) Az en kalapom
   the 1 hat
   ‘my hat’
2) A Mari kalapja
   the Mary hat
   ‘Mary’s hat’
3) A te kalapod
   the you hat
   ‘your hat’
4) Marinak a kalapja
   Mary the hat
   ‘Mary’s hat’

• Assuming that the DP in Hungarian has the basic structure we’ve been discussing, what is the structure of this kind of possessive construction?
• How about that (person?) agreement on ‘hat’?

Adjectives

• Adjectives are to nouns as adverbs are to verbs. So what would the structure be for Pat’s complete destruction of the sidewalk? Or the silly idea? Or the pencil on the desk?
• In Pat completely destroyed the sidewalk, we adjoin completely to vP. The subject moves to SpecTP.
• In the same way, we adjoin complete to nP, and Pat moves to SpecDP.

Adjuncts

[Diagram showing Adjuncts]

The Italian DP

• In Italian, in many cases, there is simply an option (stylistically governed) as to whether you say The Gianni or just Gianni:
  1) Gianni mi ha telefonato.
     Gianni me has telephoned
     ‘Gianni called me up.’
  2) Il Gianni mi ha telefonato.
     the Gianni me has telephoned
     ‘Gianni called me up.’
The Italian DP

However, there is a difference with respect to the order of adjectives and the noun depending on which one you use.

1) L’antica Roma
   the ancient Rome
   ‘Ancient Rome’
2) *Antica Roma
   ancient Rome
3) Roma antica
   Rome ancient

Generalization: If there’s a determiner, the noun follows the adjective. If there isn’t the noun precedes the adjective.

The Italian DP

- We can apply the same analysis to the order nouns and adjectives as we did to the order of adverbs and verbs.
- Recall that in French, verbs precede adverbs, but in English, verbs follow adverbs. We conclude that in French, \( v \) moves to \( T \).
- In Italian, when the noun precedes the adjective it has moved over it, to \( D \). The generalization is that this happens except if \( D \) is already filled.
   - L’antica Roma
   the ancient Rome
   - Roma antica
   Rome ancient
   *Antica Roma
   ancient Rome

Parameters

- Languages differ on whether \( n \) moves to \( D \), yielding some languages where nouns precede adjectives, and some languages where nouns follow adjectives.
  - Likewise, languages differ on whether \( v \) moves to \( T \), yielding some languages (e.g., French) where verbs precede adverbs, and some languages (e.g., English) where verbs follow adverbs.
- What governs whether \( n \) moves to \( D \) is the strength of an uninterpretable feature checked on \( D \) or \( n \) by the other. One such feature is \([\text{unum}]:\).
  - Italian: \([\text{unum}]:\) is strong on null determiners.
  - English: \([\text{unum}]:\) is weak, even on null determiners.
  - \([\text{Ø indef}]:\) Happy students] poured forth from the classroom.

More Italian, same point

- \([\text{DP}] \) Il mio Gianni] ha finalmente telefonato.
  the my G. has finally called
  ‘My Gianni has finally called.’
- *[\([\text{DP}] \) Mio Gianni] ha finalmente telefonato.
  *[\([\text{DP}] \) Gianni mio] ha finalmente telefonato.

Some Hebrew

- harisat ha-oyev ‘et ha-‘ir
  destruction the-enemy OM the-city
  ‘The enemy’s destruction of the city’
- tipul ha-Siltonot ba-ba’aya
  treatment the-authorities in-the-problem
  ‘The authorities’ treatment of the problem’

Construct state. What seems to be happening here? Again, parametric variation.

- \([\text{gen}]\): feature of \( D \) is weak in Hebrew, strong (when there) in English. But \([\text{unum}]\): feature is strong in Hebrew.
  Rather like VSO languages, where \( v \) moves to \( T \) (like in French, unlike in English), but the subject doesn’t move to SpecTP (the \([\text{uD}]\): feature of \( T \) is weak).

Types of sentences

- Sentences come in several types. We’ve mainly seen declarative clauses.
  1) Horton heard a Who.
- But there are also questions (interrogative clauses)…
  2) Did Horton hear a Who?
  3) Who did Horton hear?
- …exclamatives…
  4) What a crazy elephant!
- …imperatives…
  5) Pass me the salt.
Declaratives & interrogatives

• Our syntactic theory should allow us to distinguish between clause types.
• The basic content of Phil will bake a cake and Will Phil bake a cake? is the same.
• Two DPs (Phil, nominative, and a cake, accusative), a modal (will), a transitive verb (bake) that assigns an Agent $\theta$-role and a Theme $\theta$-role. They are minimally different: one's an interrogative, and one's a declarative. One asserts that something is true, one requests a response about whether it is true.

Clause type

• Given this motivation, we seem to need one more category of lexical items, the clause type category.
• We'll call this category $C$, which traditionally stands for complementizer.
• The hypothesis is that a declarative sentence has a declarative $C$ in its structure, while an interrogative sentence (a question) has an interrogative $C$.

Embedding clauses

• The reason for calling this element a complementizer stems from viewing the problem from a different starting point.
• It is possible to embed a sentence within another sentence:
  1) I heard [Lenny retired].
  2) I heard that [Lenny retired].
• So what is that that?

What’s that?

• We can show that that “belongs” to the embedded sentence with constituency tests.
  1) What I heard is that Lenny retired.
  2) *What I heard that is Lenny retired.
• There’s a demonstrative that, but that’s not what that is.
  3) *I heard this Lenny retired.
• So, that is its own kind of thing. It’s an introducer of embedded clauses, a complementizer.

Complementizers

• There are a couple of different kinds of complementizer. That is for embedding declarative sentences.
  1) I understand that Alton dislikes unitaskers.
• It’s also possible to embed an interrogative sentence, like so:
  2) I wonder if Alton dislikes unitaskers.
  3) I wonder whether Alton dislikes unitaskers.
• Here, if and whether serve as complementizers, introducing the embedded interrogative.
• I wonder about the answer to Does Alton dislike unitaskers?

Selection

• Just like the verb dislikes takes the DP unitaskers as its object, some verbs take clauses as their object.
• Some verbs specify what kind of clause they take:
  1) I claimed that Alton dislikes unitaskers.
  2) *I claimed if Alton dislikes unitaskers.
  3) *I wondered that Alton dislikes unitaskers.
  4) I wondered if Alton dislikes unitaskers.
• This is a matter of selection. Some verbs select for declaratives, some verbs select for interrogatives.
  Some verbs can take either, some neither.
  5) I know that Alton dislikes unitaskers.
  6) I know if Alton dislikes unitaskers.
  7) *I washed that Alton dislikes unitaskers.
  8) *I washed if Alton dislikes unitaskers.
So, we have lexical items like *that* and *if*, which are complementizers (category: C), and have a value for clause type.

- *that*  
  \[C, \text{clause-type:decl}, \ldots\]
- *if*  
  \[C, \text{clause-type:Q}, \ldots\]

Where is it structurally? We know it forms a constituent with the clause it introduces. We know that verbs can select for different kinds of C. The natural conclusion is that it is a sister to TP, at the top of the tree, which projects.

\[
\text{CP}
\]
\[
\begin{array}{c}
\text{CP} \\
\downarrow \\
C \\
\downarrow \\
\text{TP} \\
\downarrow \\
\text{DP} \\
\downarrow \\
\text{Subject} \\
\downarrow \\
T \\
\downarrow \\
vP
\end{array}
\]

- C is the head of CP.
- Saying this also provides a natural explanation of why in SOV languages, complementizers are generally on the right.

1) Hanako-ga [Taroo-ga naita to] itta.  
H.- nom T. -nom cried that said  
‘Hanako said that Taro cried.’

that or not that

- C specifies the clause type; *that* indicates a declarative clause. Why then are both of these good?

1) Jack claimed that Jill fell.
2) Jack claimed Jill fell.
   - In French, Spanish, probably most other languages you don’t have the option to leave out the C.
3) J’ai dit *qu’elle* était malade  
   I’ve said *that* she was ill  
   ‘I said that she was ill’
4) *J’ai dit elle était malade  
   *Claim doesn’t embed interrogatives.
5) *Jack claimed if Jill fell.
   - So *Jill fell* is declarative in *Jack claimed Jill fell*.

Jill fell is a declarative

- But hold on a minute. *Jill fell*, just as its own sentence (not embedded) is also declarative.
- *Cf. Did Jill fall?*
- So, we’ll suppose that since the function of C is to mark clause type, there’s a C in simple sentences as well.
- The C that heads the whole structure has somewhat special properties. Declarative C in that position is never pronounced. Interrogative C is not pronounced as a word, but makes its presence known by causing movement.

∅

- Where does that leave us?

1) Jack claimed Jill fell
   - *Claim* only takes declarative complements.
   - *Jill fell* is declarative.
   - Clause type is a feature of C.
   - Thus: There is a declarative C.
     You just can’t hear it.
   - English has two declarative complementizers. One is *that*, one is ∅. In most cases, either one works equally well.

SAI in YNQs

- In yes-no questions, the subject and auxiliary “invert” (Subject-Auxiliary Inversion):

1) Scully will perform the autopsy.
2) Will Scully perform the autopsy?
- Assuming everything we’ve got so far:
   - T has a [uD∗] (EPP) feature to check, so Scully is in SpecTP.
   - The question is an interrogative.
   - (Unpronounced) C is to the left of TP.
- So what must be happening in yes-no questions?
A natural way to look at this: **T is moving to C**.

- Just like V moves to v, or like Aux (Perf, Prog, or Pass) moves to T, or like N moves to n.
- In (main clause) questions, T moves to C.

Specifically:
- Suppose T has an uninterpretable feature that matches a feature of C: \[\text{[clause-type:]}\].
- Suppose that when C values \[\text{[uclause-type:]}\] as Q, the uninterpretable feature is strong.
- Cf. When T values \[\text{[uInf]}\] on Aux (Prog, Perf, Pass), the feature is strong, and Aux moves to T.

In (main clause) questions, T moves to C.

**A simple declarative clause**

\[\text{[clause-type:Decl]}\]

- \[\text{[Q]}\] = \[\text{[uclause-type:]}\]
- \[\text{[Q*]}\] = \[\text{[uclause-type:Q*]}\]

**YNQ**

- In a YNQ, the \[\text{[Q]}\] feature of C matches and values the \[\text{[uclause-type:]}\] feature of T as strong (\[\text{[Q*]}\]).
- T moves up to adjoin to C, checking the feature.

**Embedding questions**

- So, you can embed declaratives and you can embed questions
  1) I heard (that) Jill fell.
  2) I asked if Jill fell.

- Notice that the main clause is different:
  - If the topmost C is interrogative, we get SAI. If the topmost C is declarative, it is pronounced \(\emptyset\).
  - If an embedded C is declarative, it can be pronounced either as \(\emptyset\) or as \(\text{that}\). If an embedded C is interrogative, C is audible (if) and no SAI.
  - So, T moves to C only in main clause interrogatives. \[\text{[uclause-type:]}\] is strong only when valued as Q by a main clause C.
Nonfinite clauses

- Some verbs embed finite declaratives, as we have seen: I heard (that) Jill fell.
- There are other verbs that embed nonfinite clauses. These come in a few types, but we'll start with the try type.
  1) Scully tried to perform the autopsy.
- This is two clauses: Scully tried something, and what it was was to perform the autopsy.

θ-roles

1) Scully performed the autopsy.
2) Scully tried to perform the autopsy.
- The verb perform has an Agent and a Theme, here Scully and the autopsy, respectively.
- The verb try also has two θ-roles, an Agent (the one trying) and a Theme (the thing attempted). Suppose that the Theme of try is [to perform the autopsy] here.

θ-roles

1) Scully performed the autopsy.
2) Scully tried to perform the autopsy.
- In the second sentence, Scully is both the one trying and, if you think about it, the one performing the autopsy. The same individual is the Agent of both.
- Agent θ-roles are assigned to the DP that is Merged into SpecvP.
- However: You are not allowed to assign two different θ-roles to the same DP. Otherwise, it should be possible for Scully admires to mean Scully admires herself.

PRO

1) Scully tried to perform the autopsy.
- So, we have something of a problem here. We need an Agent DP in the vP for perform, and an Agent DP in the vP for try. But it appears as if there is only one DP around, Scully.
- What to do? Once again gritting our teeth, we resolve ourselves to the fact that we need two DPs and can only see one— therefore, there must be a DP we can't see.
- The DP we can't see, we call PRO.

Control

1) Scully tried [PRO to perform the autopsy].
- PRO is a DP that is the Agent of perform, Scully is a DP that is the Agent of try.
- It is impossible to actually pronounce an Agent for perform.
2) *Scully tried [Mulder to perform the autopsy].
- The PRO Agent of perform must be interpreted as being the same person as the Agent of try.
- PRO is a little bit like an anaphor in this respect; this fact is similar to the fact that herself in Scully admires herself must refer to Scully.
- This obligatory co-reference goes by the name control. Scully controls PRO. Sentences with PRO in them are often called control clauses.

PRO

- So why is it impossible to say this?
- *Scully tried [Mulder to perform the autopsy].
- The answer we'll give is that nonfinite T (to) does not have a case feature.
- Finite T has a [nom] feature which matches, values, and checks the [case] feature of the subject, checking itself in the process.
- Nonfinite T has no case feature at all, so Mulder would be left with its case unchecked.
Null case

- As for PRO, it is a DP so it has a [case] feature. If Mulder can't get its case checked by the nonfinite T, how does PRO get its case checked?
- A standard (and perhaps less than completely elegant) way to look at this:
  - **PRO is special**, it can only "show up" with "null case" ([u\text{case}:null]).
  - **Null case is special**, it is only allowed on PRO.
  - **Control clauses are special**, they are introduced by a null C that has a [null] case feature, which can check the [case] feature on PRO.

Try

- So, try embeds a nonfinite CP, headed by the special null C with the [null] case feature.
- In turn, the subject must be PRO, in order to successfully check that feature of C.
  - If the [case] feature of any other DP is valued and checked as [null], the derivation crashes: only PRO can have null case.
  - The embedded clause must be nonfinite (T can't itself have a [nom] feature).
  - If the [nom] feature of T checks the [case] feature of the subject, nothing is left to check C's [null] feature.

ECM

- The idea is that believe (actually the v that combines with the V believe) has an [acc] feature that can check the case of him in I believe him.
- Suppose that believe can either have a DP or a TP as its complement.
- What do we expect?

Believe

- Another place where nonfinite clauses can be embedded is under the verb believe.
  1) I believe [him to be innocent].
- Here, we have an accusative subject, and a nonfinite T that is not capable of checking case.
- How is the (accusative) case of him checked?
- This relates to the fact that believe can also simply take a DP object:
  2) I believe him.
- So, how is the accusative case of him checked here?
Arranging to leave

- A somewhat similar phenomenon occurs with verbs like arrange.

  1) Harry arranged for Tom to leave MI-5.

- Here, we have:
  - Nonfinite T, which cannot check case.
  - An overt subject (Tom) in the accusative.
  - The word for, which we classify as C.
  - For, as a P, checks accusative case (He baked a cake for her). If the C for also has an [acc] feature, it could check the [case] feature on Tom.

---

Arranging to leave

- Arrange-type verbs can take a CP complement.

  1) Harry arranged for Tom to leave MI-5.
  - Notice that it is also possible to say
  2) Tom arranged PRO to leave MI-5.

- But this is expected.
  - Nonfinite T, cannot check case.
  - The null C with [null] case can check the case of PRO.
  - An overt subject can’t get null case:
    *Harry arranged Tom to leave MI-5.
  - PRO cannot get anything but null case:
    *Tom arranged for to leave MI-5.

---

Summary

- Complementizers indicate clause type (that/Ø for declaratives, if/whether for interrogatives).
- Some verbs embed clauses. Finite clauses are always CPs.
- Some verbs can embed nonfinite clauses, some embedding TP and others embedding CP.
  - Believe (expect, …) embed TP and check accusative case (ECM verbs).
  - Try (want, …) embed CP. This can either be:
    - C[null], checking null case on PRO.
    - for[acc], checking acc case on an overt subject. Not all verbs allow this option (want does, try doesn’t).