Movement

Last time, we talked about subcategorization.

(1) a. I can solve this problem.
    b. This problem, I can solve.

(2) solve: [— NP]

Two problems:
   • solve has no object.
   • This problem has no place in the structure
     (can’t be generated)

We could complicate the rules, or we could derive (1b) from (1a).

(3) Topicalization
Move XP and attach it as the leftmost constituent of S.

(4) S
   NP Aux VP
   I can solve this problem

(5) S
   NP NP Aux VP
   I can solve this problem

Trace convention
Movement transformations leave a trace behind.

(9) a. I want to read this novel.
    b. I wanna read this novel.

(10) wanna contraction: want to \(\rightarrow\) wanna.

(11) a. This novel, I want to read.
    b. This novel, I want to be considered for a prize.

(12) a. This novel, I wanna read.
    b. * This novel, I wanna be considered for a prize.

(13) a. [NP this novel], I want to read \(t_{[NP]}\).
    b. * [NP this novel], I want \(t_{[NP]}\) to be considered for a prize.

Idea: The rule in (10) can see the trace; hence want is not next to to. wanna.
The trace isn’t pronounced, but it is nevertheless there.
Some history…

Traces weren’t always a part of syntactic theory. This novel is the thing read: *This novel I want to read*—It is still interpreted as the object.

- Meaning could be read off *deep structure* (from PS rules, prior to movement).

One problem: Two sentences with the same deep structure can have different meanings.

(14) Some student read every chapter.
   ‘There is a student \( x \) such that for each chapter \( x \), \( x \) read \( y \).’
   ‘For every chapter \( y \), there is a student \( x \) such that \( x \) read \( y \).’

Preview: These meaning differences seem to come about by *movement*.

(15) a. Some student read every chapter.
    b. \([NP \text{ Every chapter}], \text{ some student read } t_{[NP]}\].

But if something “moves” why can’t we hear it in a different place?

The structure of the grammar:

- DS \( \leftarrow \) phrase structure rules
- SS \( \leftarrow \) movement rules
- PF \( \leftarrow \) more movement rules
- LF \( \leftarrow \) “logical form” (meaning)

surface structure (abstract) \( \rightarrow \) SS

The idea: The movement of *every chapter* in (14) on the meaning of (15b) happens after surface structure (no access to pronunciation).

You probably shouldn’t be convinced yet, but we’ll add evidence for this view later…

On traces: If interpretation (meaning) is based on the representation at LF, the trace in the object position also helps explain how *this problem* is understood as the object of *solve* even after *this problem* has been moved away.

**Note:** Syntacticians often talk about things happening “before” and “after”—but at this point, we aren’t really proposing that people do this in real-time (construct a DS, apply movement rules to construct a SS, pronounce it, apply more movement rules to construct a LF, interpret the resulting representation). We are still in the business of characterizing knowledge, and this system (with its before–after structure) predicts native-speaker intuitions to a certain degree of success. It remains an open question how much (if any) of the procedure we use to arrive at our prediction reflects how people actually produce a sentence.

### Wh-movement

(16) Q. Which problem did you solve?
    A. I solved number 18.

(17) *solve* \( \rightarrow \) NP.

(18) [which problem], did you solve \( t_i \)?

Dodging various complications for a moment…

(19) I wonder [which problem will solve \( t_i \)].

(20) *I wonder…*

*Comp = complementizer. E.g., that.*

Wait—didn’t topicalization attach to S? Why move wh-phrases to Comp instead of attaching them to S too?

We have real evidence that topicalization attaches to S (in particular, after Comp):

(21) I know [\( S \) \( \text{that } [S \text{ this problem}, I can solve } t_i. \)]

We don’t have such evidence here; you can’t use *that* with questions:

(22) a. *I wonder [which problem], that Mary solved \( t_i \).
    b. *I wonder that [which problem], Mary solved \( t_i \).

**Assertion:** There is good evidence for moving wh-phrases higher, but we need to talk about some more things first.

So, for now: Assume *that* and the target of wh-phrase movement is the same, Comp:
Wh-movement
Move wh-XP to Comp.

XP because…

(23) a. I wonder \[ S \' [AP how difficult], [S the problem was \( t_i \)] \].
    b. I wonder \[ S \' [AdvP when], [S Mary solved the problem \( t_i \)] \].
    c. I wonder \[ S \' [PP to whom], [S Mary gave the book \( t_i \)] \].
    d. I wonder \[ S \' [NP who(m)], [S Mary gave the book \( t_i \)] \].

“Pied-piping”

Notice that wh-phrases are regular phrases (NP, AP, PP, AdvP) but with the property that they are question words. “[+wh]”, or a wh-XP.

With respect to (23) and wh-XPs: who(m) is an NP, like which problem:

<table>
<thead>
<tr>
<th>(24) a.</th>
<th>b.</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>NP</td>
</tr>
<tr>
<td>Det</td>
<td></td>
</tr>
<tr>
<td>which</td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>problem</td>
<td>who</td>
</tr>
</tbody>
</table>

Conjunctions on wh-movement

Not all S-bars are created equal:

(25) a. I wonder which problem Mary solved.
    b. I wonder if Mary solved this problem.
    c. * I believe which problem Mary solved.
    d. * I believe if Mary solved this problem.

(26) a. * I wonder that Mary solved this problem.
    b. I believe that Mary solved this problem.

(27) \( \text{wonder} \) \[ \rightarrow S':[+Q] \]
    \( \text{believe} \) \[ \rightarrow S'[:–Q] \]

(28) a. I know that Mary solved the problem.
    b. I know if Mary solved the problem.
    c. I know which problem Mary solved.

(29) \( \text{know} \) \[ \rightarrow S' \]

So, sentences are either [+Q] or [–Q], and wh-movement only works for [+Q] sentences.

Wh-movement (statement two)
Move wh-XP to Comp when Comp is [+Q].

Echo questions: actually [–Q]:
(30) a. Mary solved which problem?
    b. You believe Mary solved which problem?
    c. * You wonder Mary solved which problem?

Questioning out of questions: The wh-island condition

(31) a. How, do you think \[ S \'_(that) [S Mary solved the problem \( t_i \)] \]?
    b. * How, do you wonder \[ S \'_(whether) [S Mary solved the problem \( t_i \)] \]?
    (Ans.: I wonder whether Mary solved the problem \textit{with her Palm Pilot}.)

b'. ? How, do you wonder \( t_i \) \[ S \'_(whether) [S Mary solved the problem \( t_i \)] \]?
    (Ans.: I wonder \textit{desperately} whether Mary solved the problem.)

(32) a. [Which way], do you think \[ S \'_(that) [S John went \( t_i \) (to fix his car)] \]?
    b. * [Which way], do you wonder \[ S \'_(why) [S John went \( t_i \)] \]?
    (Ans: I wonder why John went \textit{north}.)

(33) a. [Which way], do you think John went \( t_i \) ?
    b. * [Which way], do you wonder \[ S \'_(if) [S John went \( t_i \)] \]?
    (Ans: I wonder if John went \textit{north}.)

A wh-island is a [+Q] S; an island to wh-movement (things inside are “stranded”).

Wh-movement (statement three)
Move wh-XP to Comp when:
    (i) Comp is [+Q]
    (ii) the wh-XP is not inside a [+Q] S except the one containing the Comp.

Complex Noun Phrase Condition (CNPC)…

Complex noun phrase: the claim that John went north, the hat I gave to John.

(34) a. You believed \[ \text{NP the claim that John went north} \].
    b. * [Which way], did you believe \[ \text{NP the claim that John went \( t_i \)} \]?

(35) a. Bill stole \[ \text{NP the hat I gave to John} \].
    b. * [To who(m)], did Bill steal \[ \text{NP the hat I gave \( t_i \)} \]?
**Wh-movement** (statement four)
Move \(wh\)-XP to Comp when:
(i) Comp is \([+Q]\)
(ii) the \(wh\)-XP is not inside a \([+Q]\) \(S'\) (except the one containing the Comp).
(iii) the \(wh\)-XP is not inside a complex NP.

**Successive-cyclicity**

(36) a. \([\text{which problem}]_s \text{ did } [_{s'} \text{ Mary heard } [_{s'} \text{ Robin solved } t_i ]]?)
   b. Which problem did John say Mary believed Robin heard Susan solved \(t_i\)?

*Wh-movement* looks like it can go unboundedly far.

But there are good reasons to believe (we’ll hear about them later) that this is an illusion.

(37) a. \([_{s'} [_{s'} \text{ John knows } [_{s'} \text{ which problem}, [_{s'} \text{ Mary solved } t_i ] ] ] ])
   b. \([_{s'} [\text{ which problem}], \text{ does } [_{s'} [_{s'} \text{ John know } [_{s'} [_{s'} \text{ Mary solved } t_i ] ] ] ] ])

We see that there’s a “place to stop” below \textit{know}:

(38) \([_{s'} [\text{which problem}, _{t'}_s \text{ does } [_{s'} \text{ John know } [_{s'} [_{s'} \text{ Mary solved } t_i ] ] ] ]])?

The **Transformational Cycle** — each \(S'\) is self-contained.
You apply all the rules inside \(S'\) before moving on to the higher \(S'\).

Another way to think about this: *wh*-movement can only go to the closest Comp.
(A locality condition…)

**Wh-movement** (statement five)
Move \(wh\)-XP to Comp when:
(i) Comp is \([+Q]\)
(ii) the \(wh\)-XP is not inside a \([+Q]\) \(S'\) (except the one containing the Comp).
(iii) the \(wh\)-XP is not inside a complex NP.
(iv) there is no nearer Comp.

This doesn’t \textit{quite} work. The first movement should be ruled out by condition (i).

(39) \([_{s'} [\text{which problem}], \text{ does } [_{s'} [_{s'} \text{ John know } [_{s'} [_{s'} \text{ Mary solved } t_i ] ] ] ] ])

**Relative clauses**

(40) Bill heard \([_{s'} \text{ the speech, } [_{s'} \text{ which }, [_{s'} \text{ Mary made } t_i ] ]])？

Relative clauses have a structure very similar to *wh*-questions, attached inside an NP.

**Restrictive relatives** restrict the reference of the head noun.
(sort of like an adjective: *The moving speech, The speech which Mary made*)

The head noun and the *wh*-phrase are co-indexed (meaning they share the same referent).

**Appositive relatives** don’t restrict the reference, but provide additional information

(41) a. Mary, who you met yesterday, just bought a house.
   b. Mary, \([_{s'} \text{ who }, [_{s'} \text{ you met } t_i \text{ yesterday } ] ])

**Free relatives** involve -*ever* and don’t modify a head noun

(42) a. I will buy \([\text{whatever you sell } t_i ]\).
   b. Whoever just arrived unplugged my lamp.

(43) a. Bill heard \([\text{the speech, which Mary made } t_i ]\).
   b. Bill heard the speech \([\text{that Mary made }]\).
   c. Bill heard the speech \([\text{Mary made }]\).

(44) \textit{made}: \([-NP]\) (in the sense of \textit{made a speech}).

The structure of (43a–c) is the same except for the relative clause.

But \textit{made} needs an NP argument—where is it in (43b–c)?

It \textit{must} be there. So it must be silent, but basically means the same thing as \textit{which}.

A null *wh*-phrase; we’ll call it \textit{Op} (for \textit{Operator}) for reasons that will become clear later.

(45) Bill heard \([_{s'} \text{ the speech, } [_{s'} \text{ Op } [_{s'} \text{ Mary made } t_i ] ]])\). = (43c)
The analysis of (43b) is that Op moves to attach to Comp, which already has that.

(46) a. ...the speech that Mary made.
   b. ...[NP the speech, [x' Op, that [S Mary made t₁]]].
   c. 
      \[\text{Comp} \quad \text{Op} \quad \text{that}\]

Some support for this analysis: If Op (or indeed which) is really wh-movement in relative clauses, it should be sensitive to the conditions we discussed before.

(47) a. * I know the way Op, (that) John wonders [wh-island why Bill went t₁].
   b. * I know the way Op, that John made [CNP the claim that Bill went t₁].
(48) a. * I know the way which, John wonders [wh-island why Bill went t₁].
   b. * I know the way which, John made [CNP the claim that Bill went t₁].

So:
- If relative clauses can have a null wh-phrase, why can’t wh-questions?
- Why can you have that with Op but not with which?
  *I heard the speech, which, that Mary made t₁.

Recoverability Condition
The content of a null category must be recoverable (from a co-indexed overt category in the sentence).

(49) a. When did Mary buy the book?
   b. Where did Mary buy the book?
   c. How did Mary buy the book?
   d. * Op, did Mary buy the book?
(50) a. * Op, did Mary buy t₁?
   b. * Op, did Mary give a book t₁?
(51) Bill heard the speech, [S Op, that [S Mary made t₁]].

Doubly Filled Comp Filter
*[Comp wh-XP that], if wh-XP is overt (non-null).

Filters. Exclude sentences which the PS rules and transformations otherwise generate.
(Instead of encoding the restriction in the transformations).

NP-movement

Passive.

(52) a. Mary solved the problem.
   b. The problem was solved (by Mary).

The problem, still the thing solved (the logical object), is in subject position [NP,S].

(53) solve [— NP]

We could introduce a new lexical item, solve\text{passive}, but we’d have to do it for every verb. Instead, let’s derive (52b) from (52a).

NP-movement
Move NP to an empty subject position

(Where we have to allow our PS rules to generate an empty subject position.)

Raising constructions.

(55) a. It seems (that) Mary has solved the problem.
   b. Mary seems to have solved the problem.

These are synonymous; Mary is the logical subject of solve, but it’s the structural subject of seems, the matrix (top level) clause.

Like with passive, we will derive (55b) from a sentence with Mary as subject.

(56) [S [NP e] seems [S [NP Mary ] to have solved the problem]].
Only certain verbs do this: *seems, be likely, appear, …—raising verbs.* These verbs also can take *it* as a subject, an *expletive.*

(57) a. It appears (that) Mary has solved the problem.  
    b. Mary appears to have solved the problem.  
    c. It is likely that Mary will solve the problem.  
    d. Mary is likely to solve the problem.

(58) a. * It is confident (that) Mary will win.  
    b. * Mary is confident to win.  
    c. Mary is confident (that) she will win.

**Tensed S condition**

The embedded sentences from which subject-to-subject raising applies are not, and cannot be, tensed:

(59) a. Mary seems [to have solved the problem].  
    b. * Mary seems [has solved the problem].  
    c. * Mary seems [will solve the problem].

**Tensed clauses** have [+Tense] in their Aux element,  
**Untensed clauses** have [–Tense] in their Aux element,  
pronounced as *to* (infinitive marker)

(60)  

**Specified Subject Condition**

(61) a. Mary, appears [t_i to be likely [t_i to win]].  
    b. * Mary, appears [it is likely [t_i to win]].

(62) a. John is believed [t_i to have beaten Bill].  
    b. * Bill is believed [John to have beaten t_i].

(61b) is bad by TSC, but (62) can’t be explained that way.

(63) a. [S [np e ] is believed [S [np John ] to have beaten Bill] ].  
      b. * [S [np e ] is believed [S John to have beaten [np Bill ] ]].

There are many options here (e.g., you can only raise a subject, or you can’t skip a subject position), but the explanation people adopted was the **Specified Subject Condition**.

**NP-movement** (third statement)  
Move NP to an empty subject position, as long as  
(i) NP is not contained in a Tensed S (other than the target S).  
(ii) NP is not separated from the targeted position by a specified subject.

(64) a. [Details of a secret plan to finance the rebels] have emerged.  
    b. Details have emerged of a secret plan to finance the rebels.

(65) [np Details [np t_i ] ] [vp have emerged [pp of a secret plan to finance the rebels]].

**Extraposition and Heavy NP shift**

(66) a. A man [t_i has come forward [s who claims to be the culprit ]].

**Extraposition** (a *rightward* movement rule—unlike the rest we’ve seen)  
Move XP and attach it as the rightmost constituent of VP.

XP because it isn’t just PP…
The extraposition rule wildly overgenerates—there are conditions on extraposition too, but we’ll worry about that later.

**Heavy NP shift** (a subcase of **Extraposition**)
Move a ‘heavy’ NP and attach it as the rightmost constituent of VP.

(67) a. Mary read [NP all the books she had borrowed].
    b. ? Mary returned [NP all the books she had borrowed] to the library.
    c. Mary returned \( t_i \) to the library [NP all the books she had borrowed].

(68) a. Mary read *Aspects*.
    b. Mary returned *Aspects* to the library.
    c. * Mary returned \( t_i \) to the library [NP *Aspects*].

### Movement of terminal categories

**Affix hopping.**

(69) \( \text{Aux} \rightarrow \text{Tense (Modal) (Neg)} \)

(70) Mary solved the problem.

(71) \( \text{S} \)

\( \text{NP} \)

\( \text{Aux} \)

\( \text{VP} \)

\( \text{N} \) Mary

Tense solve

NP

\( \text{V} \) solve

\( \text{-ed} \)

\( \text{the problem} \)

We have to get from (71) to (70). [+Past] ends up on V (solved). Well, **movement**.

Two options: Move solve to attach to Aux.
Move Tense to attach to V.

It *matters* which one we choose…

Consider:

(72)

\( \text{S} \)

\( \text{NP} \) Mary

\( \text{Aux} \)

\( \text{VP} \)

\( \text{V} \) solve

\( \text{Tense} \)

\( \text{-ed} \)

\( \text{the problem} \)

We can decide between them on the basis of evidence from *adverbs*. Adverbs are a constituent of VP, on the left or right

(74) \( \text{VP} \rightarrow (\text{Adv}) \text{ V ... (Adv)} \)

(75) a. John \( [\text{vp} \text{ cleverly avoided Bill}] \).
    b. John \( [\text{vp} \text{ rarely visited Bill}] \).
    c. * John avoided \( [\text{vp} \text{ cleverly Bill}] \).
    d. * John visited \( [\text{vp} \text{ rarely Bill}] \).

So the verb is *inside* the VP; (72) is the victor.

**Affix Hopping**

Move Tense (from Aux) to V provided Aux does not dominate a modal.

(76) John might cleverly avoid Bill.
John did not avoid Bill.

For some reason, avoid does not get the past tense, which is instead realized on did. Affix Hopping was blocked:

John -ed not avoid Bill.

not intervenes between past tense -ed, and so Affix Hopping cannot merge them.

Affix Hopping (second statement)
Move Tense (from Aux) to V provided Aux does not dominate a Modal or Neg.

Do-support
Insert do to support stranded Tense.

Both Affix Hopping and Do-support provide “morphological support” for Tense. Tense is an affix, it can’t stand alone.

Neither is needed if there is a modal in Aux:

a. John should not avoid Bill.
   b. John cannot eat bacon.

Auxiliary verbs have and be:

a. John is not happy.
   b. John has not visited Bill.
   c. * John did not be happy.
   d. * John did not have visited Bill.

Have and be are unique among English verbs, using strategy (73) instead; the verb raises to Aux to merge with Tense.

To distinguish have and be from other verbs, we will say they are [+AUX], whereas other verbs are [–AUX].

If Tense needs support, a [+AUX] verb can raise—but won’t if not needed.

a. John must be happy.
   b. John might have visited Bill.

V-raising
Raise V to Aux, provided:
   i) V has the feature specification [+AUX]
   ii) Aux does not dominate a modal.

Affix Hopping
Move Tense (from Aux to V), provided:
   i) V has the feature specification [–AUX]
   ii) Aux does not dominate a modal or neg.

Ouhalla adds to Affix Hopping
   iii) VP does not dominate a V with the feature specification [+AUX].

This is getting complicated. V-raising and Affix Hopping have almost complementary conditions. We can simplify this a little bit:

Goal: Support Tense.

Preference: Prefer V-raising to Affix Hopping, Affix Hopping to Do-support.

• If Tense is already supported (if there is a modal in Aux), goal is met.
• If Tense is still unsupported and V-raising can be applied (there is a [+AUX] V), apply it; goal is met.
• If Tense is still unsupported and Affix Hopping can be applied (Aux doesn’t dominate Neg), apply it; goal is met.
• If Tense is still unsupported, apply Do-support (as a “last resort”).

V-Raising
Raise V to Aux provided V has the feature specification [+AUX].

Affix Hopping
Move Tense (from Aux to V), provided Aux does not dominate Neg.

(83) a. * John did avoid Bill.
     b. John did avoid Bill.
a. Mary will solve the problem.
b. Will Mary solve the problem?

a. You can drive my car.
b. Can you drive my car?

To make a question, the Aux element is moved over the subject—to Comp.

\[ S' \]
\[ \text{Comp} \]
\[ [+Q] \]
\[ \text{NP} \]
\[ \text{Aux} \]
\[ \text{VP} \]

**Aux-raising**
Move Aux to Comp, when Comp has the feature specification [+Q].

This happens in *wh*-questions too… (we ignored this part earlier)

a. Which problem can Bill solve?

b. DS: \[ [S' [+Q] [S Bill [Aux can] [VP solve which problem]]] \]

c. SS: \[ [S' [NP which problem], [Aux can], [S Bill t; [VP solve t]]] \]

But, Aux-raising doesn’t apply in embedded questions:

a. I wonder which problem Bill can solve.
b. I wonder if Bill can solve this problem.

**Aux-raising (second statement)**
Move Aux to Comp, provided

i) Comp has the feature specification [+Q]

ii) Comp is situated in a root clause.

We could even conflate these two if Neg counts as making Aux non-adjacent to VP. More soon…

---

(89) a. Was John at home?
b. DS: \[ [S' [+Q] [S John [Aux Tense] [VP be at home]]] \]
c. VR: \[ [S' [+Q] [S John [Aux v be], Tense] [VP t; at home]] \]
d. AR: \[ [S' [Aux v be], Tense] [S John t; [VP t; at home]] \]

(90) a. Did John solve the problem?
b. DS: \[ [S' [+Q] [S John [Aux Tense] [VP solve the problem]]] \]
c. AR: \[ [S' [Aux Tense], [S John t; [VP solve the problem]]] \]
d. *do*: \[ [S' [Aux Tense [do]], [S John t; [VP solve the problem]]] \]

Note: Affix Hopping didn’t apply here, but there was no Neg.

**Affix Hopping**
Move Tense (from Aux) to V provided

i) Aux does not dominate a Modal or Neg

ii) V has the feature specification [–AUX]

iii) VP does not dominate a V with the feature specification [+AUX]

iv) Aux is adjacent to VP which dominates the targeted V

or

**Affix Hopping**
Move Tense (from Aux to V), provided

i) Aux does not dominate Neg.

ii) Aux is adjacent to VP which dominates the targeted V