1 Sideward Movement

Sideward movement is expected to be available if (Hornstein 2001):

- Move is Copy/ReMerge
- There are multiple derivational "workspaces."

Multiple workspaces are needed independently to derive mixed left/right-branching structures:

(1) TP
   Det the Nidan T
   VP saw the Nidan woman

2 What's it good for?

Hornstein (2001) accounts for adjunct control using sideward movement:

(2) a. John₁ took the cheese without t₁ asking.

DERIVATION:

| [without John asking] | Workspace 1 |
| [John took the cheese] | Workspace 2 |

b. [without t₁ asking] [John took the cheese]

⇒ [John₁ took the cheese] [without t asking]

John moves out of the adjunct-to-be before it is adjoined, thus obviating the CED.

3 Merge over Move

(3) This can be made more precise... Merge over Move: If at a stage S in a derivation it is possible to perform a Merge operation, and if one of the available Merge operations does not block the derivation from eventually converging, then a Merge operation must be chosen at S.

Important: There is no requirement that the competing derivations evaluated by Merge over Move have the same interpretation (though they must all be convergent). Hornstein shows that Merge over Move explains the absence of object control into adjunctions.

(4) John₁ kissed Mary₂ without t₁₂ asking.

In (4), it is not possible for one of the DPs to move out of the adjunct to fill the matrix object position, because at this point in the derivation another DP remains in the numeration which could be merged as the object.

4 Blocking overgeneration in adjunct control

The derivation (5b) of (5a) is ruled out by Minimality; the derivation (5c) is ruled out by Merge over Move.

(5) a. John₁ kissed Mary without Jane wanting t₁ to leave.

DERIVATION:

| [without Jane wanting John to leave] | Workspace 1 |
| [John kissed Mary] | Workspace 2 |

b. [without Jane wanting t₁ to leave]

⇒ Minimality violated: "John" moves over "Jane"

(6) b. [without Jane wanting t₁ to leave]

⇒ [John₁ kissed Mary] [without Jane wanting t₁ to leave]]

DERIVATION:

| [without Jane wanting t₁ to leave] | Workspace 1 |
| [John kissed Mary] | Workspace 2 |

⇒ No Minimality violation, but MOM is violated because "Jane" could have merged.

That is, (5c) is bad because (6) is good:

(6) Jane₁ kissed Mary without t₁ wanting John to leave.

5 Features must be valued, not checked

Otherwise, the landing site of an A-moved DP would be predetermined by its unchecked nominative/accusative feature.

6 Selectional restrictions not enforced in the syntax

Otherwise, we would expect to derive (7):

(7) *John dispersed [the swarm of bees₁] without t₁ stinging.

If the selectron restriction on dispense were enforced syntactically then it would be permissible to violate Merge over Move, since the alternative derivation in (8) would be ungrammatical.

(8) # [The swarm of bees₁] dispersed John without t₁ stinging.

7 “Almost c-command”

Sideward movement allows for a kind of “late adjunction.” After initial merger of a DP, an adjunct can be adjoined to the DP in a separate workspace. When we consider adjunct DPs, we find that DP movement is predicted to be limited to an “almost c-command” configuration:

(9) a. People₁ want t₁ to win.
   b. People₁’s friends want t₁ to win.

(But in English, but on the face of it predicted to be OK; see Boeckx & Hornstein (2004, 2007))

(10) a. [John₁, [of a friend]] wants t₁ to win.

DERIVATION:

| [John, [of a friend]] | Workspace 1 |
| [John wants t₁ to win] | Workspace 2 |

⇒ [John, [of a friend]] wants t₁ to win | Workspace 1

⇒ # [John, [of a friend]] wants t₁ to win | Workspace 2

Movement of John in (9a) is illicit because at the point in the derivation when it occurs, there is also the option of merging of with a friend.

Deriving almost-c-command in a principled fashion is an interesting result. The relation is known to restrict certain phenomena (e.g. variable binding), but it has previously seemed a rather ad hoc structural relation.

There is some evidence that the interpretative interface cares about almost-c-command.

(11) a. Everyone₁ loves his₁ mother.
   b. [Everyone’s mother] loves him₁.
   c. [The mother of everyone₁] loves him;

(12) a. An occasional sailor walked by.
   b. An occasional sailor’s arms went up.
   c. An arm of an occasional sailor went up.

(Not acceptable under the weird scope reading available for (a) and (b))

Since binding in (11) is probably not derived via movement, and since the weird scope reading in (12a/b) certainly is not, it seems that almost-c-command may ultimately derive from an interface requirement of some sort. Merge over Move restricts the syntax to movement dependencies that accord with this interface requirement.

References


This poster summarizes part of my UMD generals paper (Drummond 2009).