Find the gap

On structure building and the formation of filler–gap dependencies in syntax

Marcel den Dikken

Linguistics Program — CUNY Graduate Center
Research Institute for Linguistics — Hungarian Academy of Sciences

---

1 The direction of structure building

1.1 The traditional generative approach: Bottom-up with 'movement'

- generative syntactic theories have generally placed the root of the tree at the top of the diagram, with the branches and leaves below them

→ the leaves of the tree, i.e., the nodes immediately dominating the individual lexical items or terminals, provide the lexical information (features and argument-structural details) from which trees can be grown

(1) he kissed her

(2) $[_{VP}$ V OB]]

(3) $[_{vP}$ SU $[_{vP}$ *v* $[_{VP}$ V OB]]]

(4) $[_{TP}$ T $[_{vP}$ SU $[_{vP}$ *v* $[_{VP}$ V OB]]]]

(5) he will kiss her

(6) $[_{TP}$ SU $[_{TP}$ T $[_{vP}$ ~~SU~~ $[_{vP}$ *v* $[_{VP}$ V OB]]]]]

- in GB, the element in Spec*v*P marked in (6) as '~~SU~~' was called a *trace*
- the older approach had been to assume that there are multiple copies of the displaced constituent present in the syntactic representation: '~~SU~~' as a 'silent copy' of the subject

→ (6) is neutral with respect to the question of how placement of the subject in SpecTP comes about: all (6) says is that there are two tokens (or 'copies') of SU

→ the 'copy theory' is not itself a theory of 'movement'

→ the idea that one of the copies arrives in a different position from the one it started out life in as a result of movement through the tree is extrinsic to the idea as such that there are two copies of the subject

→ there are at least three logical possibilities:

(a) the copy can be made after the subject has been merged into the tree, in Spec*v*P, and then be transported up the tree into SpecTP, via 'movement' or 'Internal Merge'

(b) the copy can be made before the subject is even merged in Spec*v*P, so that the two tokens of the subject can be Externally Merged in the syntactic structure independently of one another, in two different positions — one the θ-position for the subject, the other its spell-out position

(c) the subject can be Externally Merged prior to copying, not in Spec*v*P but in SpecTP; the copy of the subject can then be taken to be stored in a push-down stack and to be associated to the θ-position as soon as this position presents itself in the course of the structure-building process, which, on this conception, proceeds from the root node down to the lower branches and leaves

1.2 A top-down alternative

- *(c)* leads us to think again about the question of whether syntactic structures must be built 'from the bottom up' — from the predicate head (and its lexical properties) up to the clause

Q could we build the same tree in exactly the oppsite direction, 'from the top down'?

- it seems impossible to grow a syntactic tree *entirely* in that direction, i.e., from the root to the leaves

→ there is no way to predict *a priori* whether the root of the tree (the top node) will be a TP or a CP, and if it is a CP, whether it has interrogative force or declarative force; it could even be something smaller than TP, as in '*Mad Magazine* sentences'

(7) Barack Obama a great president?! (you must be kidding!)

- but we can often predict once the initial constituent (which is often a specifier) is in place what the root node and a large portion of the rest of the structure will look like
  - → initial constituent = a *wh*-expression ⇒ root node = interrogative CP
  - → initial constituent = explicitly nominative noun phrase (in nominative–accusative languages like English) ⇒ root node = finite TP

- with the nature of the root node determined, we can rely on important regularities about the internal structure of syntactic projections to predict a significant amount of the rest of the structure
  - – endocentricity/projection
  - – extended projection (Grimshaw 1993)

(8) an *extended projection* is the maximal projection of a lexical category plus all the functional projections belonging to that lexical category

→ CP or TP *qua* root node will, in languages that lack verbless clauses, always be traceable back to a verbal predicate head

→ if the internal constitution of extended projections is fixed and predictable, we can make our way down from CP to V in a 'top-down' approach just as easily as we can make our way up from V to CP on the 'bottom-up' approach

- (1) on a 'top-down' approach:
  - *(i)* we encounter *he*
  - *(ii)* we register that it is
    - necessarily argumental (animate personal pronouns are never predicative)
    - explicitly nominative
  - *(iii)* we conclude that the root node dominating *he* as the subject is TP
  - *(iv)* endocentricity and extended projection direct us down the tree
    - from TP to its head (T)
    - from T to its complement (*v*P)
    - from *v*P to the its head (*v*)
    - from *v* to its complement (VP)
    - from VP to its head (V)
  - *(v)* Spec*v*P provides a θ-position in which a silent copy of the subject can be placed
  - *(vi)* once we are down to the verbal head, its argument-structural properties allow us to accommodate an object for it in the same way they do in the bottom-up approach

(9) a. *he* = nominative, argument
→ the subject of a finite clause
→ finite clauses are minimally TPs
b. $[_{TP}$ *he*
→ TP is a projection of a head (*endocentricity*, *projection*)
→ this projection accommodates not just a specifier (*he*) but also a complement
→ in languages like English, the complement must be verbal: *v*P (*extended projection*)
c. $[_{TP}$ *he* $[_{T'}$ T $[_{vP}$
→ *v*P is a projection of a head (*endocentricity*, *projection*)
→ this projection accommodates a specifier and a complement
→ the specifier provides a θ-position for a silent copy of the subject
→ the complement of *v* must be VP (*extended projection*)
d. $[_{TP}$ *he* $[_{T'}$ T $[_{vP}$ ~~*he*~~ $[_{v'}$ *v* $[_{VP}$
→ VP is a projection of a head (*endocentricity*, *projection*)
→ the V-head is active *kissed*, whose argument structure includes an object
→ the complement of V accommodates the object
e. $[_{TP}$ *he* $[_{T'}$ T $[_{vP}$ ~~*he*~~ $[_{v'}$ *v* $[_{VP}$ V=*kissed her*]]]]]

Q1 what if our lexical verb had been *saw* instead of *kissed*?
→ *saw* is transitive (like *kissed*), hence takes a complement; but for *saw* we cannot predict the nature of its complement: while *kissed* will usually take a nominal complement (you cannot kiss propositions; cf. Williams 1980),[1] for *saw* we can get either nominal or (small-)clausal complements (*he saw her* vs. *he saw her out* vs. *he saw her cross(ing) the street* vs. *he saw that she was crossing the street*)

1 This is actually already a simplification. Resultative constructions such as *he kissed her awake* involve something other than *her* as the complement of the verb *kissed*: what serves as the verb's complement here is the small clause [*her awake*], on an analysis of resultatives along the lines of Hoekstra (1988) and much subsequent work.

→ it is impossible to guess this right in a top-down approach

BUT it is equally impossible to guess everything right on a bottom-up approach

→ imagine that at some point in the bottom-up structure-building process we had arrived at the structure for *her cross the street*

→ this structure could serve as the complement of a verb like *saw* — and if it does, there will be a licenser for the structural accusative case feature of *her*

→ but *her cross the street* could also serve as the root, as a '*Mad Magazine* sentence' (recall (7)): *her cross the street?! no way!*

→ a bottom-up structure-building approach could take the guessing out of the game by working with a lexical array or numeration, so that the syntactic derivation will be aware of whether or not the small clause *her cross the street* could serve as a complement to some higher verb or not

→ but the top-down approach could be enhanced with a lexical array or numeration as well: if the lexical array includes, besides *saw*, also the verb *cross*, then we will not be tempted to map *her* into the object position of the verb *saw*

SO the top-down approach is not *intrinsically* worse off than the bottom-up approach in cases of indeterminacy

Q2 how could we predict an object if we have not come across a verb yet in the linear string (as in OV languages)?

→ bear in mind that a top-down approach is not exactly the same as a left-to-right approach

→ on a strictly left-to-right approach, with little or no superordinate phrase structure, it is indeed very difficult to handle OV languages

→ but on a top-down approach, which works with hierarchical structures, the presence of a nominative subject automatically predicts, via extended projection, an entire clausal skeleton leading down to V

→ so a verb is securely predicted; encountering an object before the verb comes in leads to no particular accommodation problem because the core structure of the VP has already been put in place by the time the object comes along

- in what follows, I present one particular case study that bears in fundamental ways on the choice between bottom-up and top-down approaches: the syntax of **filler–gap dependencies**

2 A top-down approach to filler–gap dependencies

- I will present an outlook on the locality of filler–gap dependencies predicated on the premise that local domains for syntactic computation (so-called phases) are wholly impenetrable for the construction of paths, and defined in terms of predication and Agree relations

- I will show that the top-down approach to the locality of syntactic dependencies presents an explanatory outlook on classic 'Subjacency' and 'ECP' effects, and establishes a close connection between syntactic derivation and sentence processing

2.1 The locality of syntactic dependencies

- dependencies between a filler and a gap can often span great distances, but are most certainly not generally unbounded

(10) a. how quickly did Bob say (that Bill thought (that Bud claimed (that Butch believed (that ...)))) that Bubba had replaced the lightbulb?
b. Bob said (that ...) that it had taken Bubba only a minute to replace the lightbulb
c. it took Bob only five seconds to say (that ...) that Bubba had replaced the lightbulb

(11) a. how quickly did Bob sigh/whisper that Bubba had replaced the lightbulb?
b. *Bob sighed/whispered that it had taken Bubba only a minute to replace the lightbulb
c. it took Bob only five seconds to sigh/whisper that Bubba had replaced the lightbulb

(12) a. how quickly did Bob regret/realise that Bubba had replaced the lightbulb?
b. *Bob regretted/realised that it had taken Bubba only a minute to replace the lightbulb
c. Bob regretted/realised right away that Bubba had replaced the lightbulb

- 'islandhood' is the norm rather than the exception

→ as a working hypothesis it makes good sense to take a conservative approach and assume that clause boundaries are impenetrable unless they are 'opened up' — or the 'island' is 'bridged'

(13) *Phase Impenetrability Condition — Chomsky (2001)*
the domain of H (a phase head) is not accessible to operations outside HP; only H and its edge are accessible to such operations

(14) *Phase Impenetrability Condition — this work*
a phase Φ is impenetrable

- I define phases with an appeal to three notions: *predication*, *extended projection*, and *Agree*

(15) *phase*
a constituent α that is not an Agree-goal is a phase Φ iff *(a)* or *(b)*
(a) α is a minimal predication (RP)
(b) α is the maximal functional extended projection of the predicate head

→ (15) does not say that RPs and the maximal extended projections above them are *automatically* phasal in nature: they acquire phase status only if they are not Agree-goals

(16) *Agree*
a feature-sharing relation between a probe π and a goal γ

- verbs selecting clausal complements can engage in a feature-sharing Agree relation with those clauses

(17) a. szeretem [hogy szerelmes vagyok] (Hungarian)
love.1SG.DEF that in.love am
'I love that I am in love'
b. szeretek [szerelmes lenni]
love.1SG.INDEF in.love be
'I love being in love'

(18) a. szeretem őt/ ezt a szép nőt
love.1SG.DEF (s)he.ACC/this.ACC the beautiful woman.ACC
'I love her/this beautiful woman'
b. szeretek valakit/ egy szép nőt
love.1SG.INDEF someone.ACC/a beautiful woman.ACC
'I love someone/a beautiful woman'

→ agreement between the matrix verb and a subordinate clause is, in languages for which this can be demonstrated, often a requirement imposed on the establishment of a long-distance filler–gap dependency

→ Chamorro long-distance *wh*-dependencies are grammatical if the higher verbs are 'inflected ... for the Case of the intermediate CP out of which Wh-movement has most immediately occurred' (Chung 1998:250)

(19) a. hayi si Manuel h*in*assóso-nña ch*um*uli' i salappi'?
who Manuel WHOBJ.think.PROG-AGR WHNOM.take the money
'who does Manuel think has taken the money?' (Chamorro)
b. guiya esti na boi i mu-na'mäguf gui' na un-li'i
he this LINK boy the WHNOM-make.happy her C WHOBJ.AGR-see
'this boy is the one who it made her happy that you had seen'
(lit.) '... that [that you had seen] made her happy'

– in (19a), the *wh*-constituent *hayi* is nominative, as is reflected on the verb of the clause in which it originates; but the verb of the matrix clause (which *wh*-agrees with the *wh*-phrase) case-agrees with the clause from which *wh*-extraction has taken place: OBJ
– in (19b), the *wh*-constituent (a null operator) has objective case, as reflected in the form of *wh*-agreement on the verb that selects it (*un-li'i*); but since the clause from which long-distance *wh*-movement takes place is the nominative subject of the matrix verb, this higher verb shows *wh*-agreement for nominative case

→ case-agreement between the matrix verb and the clause containing the gap is a necessary tool towards 'opening up' subordinate clauses for the establishment of filler–gap dependencies across their borders, or 'bridging the island'

→ a subordinate CP is the maximal extended projection of the verbal predicate head of its clause, and therefore threatens to be a phase; but if the CP in question is an Agree-goal γ in the sense of (16), it will not be declared a phase

- the idea that Agree serves to 'open up' or 'pre-empt' potential phases is difficult to understand from the perspective of a **bottom-up** approach to structure building

→ if we build syntactic structures from the leaves up to the root ('from the bottom up'), we cannot, on this approach to phasehood, declare CP a phase at the point of its completion: we will have to wait until a potentially quite large portion of the superordinate syntactic structure has been merged, in order to verify whether or not CP is an Agree-goal

→ the answer to the question as to whether or not CP is a phase must hence be postponed, which erodes the determinism of the system

- a **top-down** approach allows for immediate and definitive decisions about the phasehood of a constituent α to be taken right at the point at which α first emerges in the structure

→ at the point at which we merge a CP as a complement to the matrix V, we know that it must be the maximal functional extended projection of a predicate head, and we also have in place the *v* with which that clause may or may not be engaged in an object-agreement relationship

→ verification of the (non-)match between agreement form of the matrix *v*/V and the subrdinate CP can proceed immediately upon Merge of the complement of V, so right upon Merge we will know whether or not this complement is a phase or not

- in general, in a top-down model, when we encounter some constituent α that is a potential phase, we can determine its phasehood right away by verifying whether or not α is an Agree-goal for some probe π *in the portion of the structure that has already been built*

  – if α is an Agree-goal, it is immediately declared a non-phase
  – if α is not an Agree-goal, it is immediately identified as a phase

→ nothing happening inside α could ever affect the Agree relation between α and the higher probe π

→ so the decision to declare α a phase will not be tampered with in the ensuing derivation

→ there is no delay in the decision-making process, and the decision is irrevocable

- once α has been declared a phase, it will be entirely impossible for a filler outside α to build a path leading down to the gap that it binds across α's boundary

→ whether this means that no filler–gap dependency can be established across α's bounds can be established *at all* is a question I will turn to in section 2.2

2.2 The construction of filler–gap dependencies in a top-down system

Q what exactly does it mean for a phase to be impenetrable?

(14) *Phase Impenetrability Condition — this work*
a phase Φ is impenetrable

- in a top-down analysis, we start at the root of the tree and build structures based on *(a)* linguistic material that is presented incrementally ('from left to right') and *(b)* predictions about the structure that can be made on the basis of endocentricity and (extended) projection

- often, in the top-down structure-building process, a piece of material will come into the tree in a position in which it cannot be directly related to what it belongs to
  - an argument is in a position in which it cannot be linked directly to the predicate that assigns it its thematic role, or
  - a predicate or modifier shows up in a position from which it cannot establish a syntactic and semantic relation with its subject/modifiee[2]

→ whenever this happens, the constituent that is 'out of place' is put on a so-caled **push-down or last-in-first-out (LIFO) stack**, waiting to be associated eventually with a gap in a position in which an argument-of or predicate/modifier-of relation can be established

(20) fillers in positions in which they cannot establish a predicational (argument-of or predicate/modifier-of) relation with their sister are placed on a push-down/LIFO stack

- phrasal displacement is restricted to arguments and predicates — the two types of subconstituents of RELATOR phrases (Den Dikken 2006)

→ a displaced phrasal filler (being either an argument or a predicate) must always be linked to a predicate or argument position inside some RELATOR phrase

→ RELATOR phrases thus become the fixed points in the structure-building process at which a push-down/LIFO stack is consulted to see if any elements on the stack can be linked to a gap

→ RPs thus form intermediate checkpoints in the course of the construction of filler–gap dependencies

→ if we call these fixed stack consultation points 'phases' (as in Chomsky's 2001 work), we arrive at the conclusion that RPs are phases (recall (15.*a*) and Den Dikken 2006)

- procedurally, the way the 'consultation' of push-down stacks at RP boundaries happens in a top-down structure-building model is as follows
  - a push-down stack is emptied out, in a last-in-first-out way, onto the edge of the RP
  - the element on the push-down stack that entered it most recently is attached to the outermost edge of RP, as an adjunct; subsequently, the element that entered the stack just prior to the one already attached to the edge of RP is attached as an adjunct, right below the one already there; and so forth
  - once the push-down stack has been emptied in this way, the grammar tries to find gaps in positions inside the RP for the various elements attached to the edge of RP, in accordance with locality theory — a path from the RP-attached filler to a gap can only be successfully constructed if the two are within the same phase

- when there are *multiple* fillers offloaded onto the edge of a single RP, the grammar tries to build multiple paths between the RP-adjoined material from the push-down stack and local gaps inside the RP

→ if they overlap, these paths must be nested

2 For me, predication and modification involve the same kinds of structural relations (see Den Dikken 2006). I will henceforth generalise over them with the cover term 'predication'.

(21) a. *Nested Dependency Constraint* (Fodor 1978)
if there are two or more filler–gap dependencies in the same sentence, their scopes may not intersect if either disjoint or nested dependencies are compatible with the well-formedness conditions of the language

b. *Path Containment Condition* (Pesetsky 1982)
if two paths overlap, one must contain the other

(22)

a. filler_1 ... filler_2 ...
STACK: STACK:
[filler_1] [filler_2 [filler_1]]

b. filler_1 ... filler_2 ... $[_{RP}$ filler_2 $[_{RP}$ filler_1 $[_{RP}$

c. filler_1 ... filler_2 ... $[_{RP}$ filler_2 $[_{RP}$ filler_1 $[_{RP}$ ... t_1 ... t_2 ...]]]]

d. *filler_1 ... filler_2 ... $[_{RP}$ filler_2 $[_{RP}$ filler_1 $[_{RP}$ ... t_2 ... t_1 ...]]]]

→ this derives, from a top-down grammar with push-down stacks plus locality and the nested dependencies requirement, that the relative order of multiple moved elements anchored in a particular RP must be preserved under displacement

– outside RP, filler_1 precedes filler_2, and inside RP, the traces of these fillers (t_1 and t_2) are in the same precedence relation
– offloading the stacked fillers onto the phase edge inverts the order in which these fillers were first encountered: on the edge of RP, filler_2 is higher, hence further to the left, than filler_1
– the requirement that within RP the paths leading from the offloaded fillers to their associated gaps be nested rather than intersecting ensures that the gaps are lined up the same way the fillers are lined up in their spell-out positions

(23) a. koj kogo vižda? (Bulgarian)
who whom sees
'who sees whom?'

b. *kogo koj vižda?
whom who sees

(24) a. koj kogo na kogo e pokazal?
who whom to whom is pointed.out
'who pointed out whom to whom?'

b. *koj na kogo kogo e pokazal?
who to whom whom is pointed.out

→ the derivations in (25)–(26) illustrate how the ordering restriction on multiple *wh*-fronting in Bulgarian-type languages falls out from the top-down approach using push-down stacks and path containment

(25)

a. $[_{CP}\ koj_1 \quad [_{CP}\ kogo_2 \quad ... \quad [_{TP}$
STACK: STACK:
$[koj_1]$ $[kogo_2\ [koj_1]]$

b. $[_{CP}\ koj_1 \quad [_{CP}\ kogo_2 \quad ... \quad [_{TP}\ \text{kogo}_2\ [_{TP}\ \text{koj}_1 \quad [_{TP}$

c. $[_{CP}\ koj_1 \quad [_{CP}\ kogo_2 \quad ... \quad [_{TP}\ \text{kogo}_2\ [_{TP}\ \text{koj}_1 \quad [_{TP}\ \text{T} \ldots t_2 \ldots]]]]$

d. $*[_{CP}\ kogo_1 \quad [_{CP}\ koj_2 \quad ... \quad [_{TP}\ \text{koj}_2\ [_{TP}\ \text{kogo}_1 \quad [_{TP}\ t_2 \ldots t_1 \ldots]]]]$

(26)

a. $[_{CP}\ koj_1 \quad [_{CP}\ kogo_2 \quad [_{CP}\ na\ kogo_3 \quad ... \quad [_{TP}$
STACK: STACK: STACK:
$[koj_1]$ $[kogo_2\ [koj_1]]$ $[na\ kogo_3\ [kogo_2\ [koj_1]]]$

b. $[_{CP}\ koj_1 \quad [_{CP}\ kogo_2 \quad [_{CP}\ na\ kogo_3 \quad ... \quad [_{TP}\ \text{na kogo}_3\ [_{TP}\ \text{kogo}_2\ [_{TP}\ \text{koj}_1\ [_{TP}$

c. $[_{CP}\ koj_1 \quad [_{CP}\ kogo_2 \quad [_{CP}\ na\ kogo_3 \quad ... \quad [_{TP}\ \text{na kogo}_3\ [_{TP}\ \text{kogo}_2\ [_{TP}\ \text{koj}_1\ [_{TP}\ \text{T}\ t_2\ t_3]]]]]$

d. $*[_{CP}\ koj_1 \quad [_{CP}\ na\ kogo_2 \quad [_{CP}\ kogo_3 \quad ... \quad [_{TP}\ \text{kogo}_3\ [_{TP}\ \text{na kogo}_2\ [_{TP}\ \text{koj}_1\ [_{TP}\ \text{T}\ t_3\ t_2]]]]]$

- the theory also needs to provide for languages like Serbo-Croatian (see (27); Bošković 1995 and much subsequent work) or Hungarian (28), where we find multiple *wh*-fronting constructions with 'freedom' of ordering in the left periphery

(27) a. ko je koga vidjeo? (Serbo-Croatian)
who is whom seen
b. koga je ko vidjeo?
whom is who seen
both: 'who saw whom?'

(28) a. ki MIT vett? (Hungarian)
who what-ACC bought
'for every person in the discourse, tell me what it was that (s)he bought'
b. mit KI vett?
what-ACC who bought
'for every item in the discourse, tell me who it was that bought it'

→ not all positions in the tree are created equal; the creation of push-down stacks is sensitive to the nature of the positions that fillers occupy

→ sometimes multiple fillers are placed on the same push-down stack and at other times each filler is placed on a stack of its own — with the choice between the two options being based on the nature of the positions occupied by the fillers
  – a single multi-member stack is created for multiple fillers whose positions are of the same type
  – separate stacks are created for each filler whenever the positions they occupy are of different types

• so far we have looked in detail only at **monoclausal** cases, with a single push-down stack with multiple members (as in the Bulgarian multiple *wh*-fronting cases) or with multiple push-down stacks with single members (as in Serbo-Croatian (27) or Hungarian (28))
→ let us now return to **multiclausal** cases featuring non-argumental fillers, such as (10)–(12)

(10) a. how quickly did Bob say (that Bill thought (that Bud claimed (that Butch believed (that ...)))) that Bubba had replaced the lightbulb?
  b. Bob said (that ...) that it had taken Bubba only a minute to replace the lightbulb
  c. it took Bob only five seconds to say (that ...) that Bubba had replaced the lightbulb

(11) a. how quickly did Bob sigh/whisper that Bubba had replaced the lightbulb?
  b. *Bob sighed/whispered that it had taken Bubba only a minute to replace the lightbulb
  c. it took Bob only five seconds to sigh/whisper that Bubba had replaced the lightbulb

(12) a. how quickly did Bob regret/realise that Bubba had replaced the lightbulb?
  b. *Bob regretted/realised that it had taken Bubba only a minute to replace the lightbulb
  c. Bob regretted/realised right away that Bubba had replaced the lightbulb

→ *how quickly*, from its offload position on the edge of the matrix TP (the first predication structure), needs to be associated to a gap in a position where the manner adverb can be locally related to its modifiee
→ this is straightforwardly possible within the clause of *say*, in all three examples in (10)–(12): the maximal extended projection of the verbal predicate head *say* forms a single local domain ('phase'), and within this phase a gap can certainly be postulated for *how quickly* such that it can be construed with a projection of *say* as a manner modifier
→ so the c–readings are available for all three cases

Q what about the b–readings?
  (10) all the subordinate CPs in (10a) are transparent because they are in Agree relations with the *v*'s of the immediately superordinate verbs ('bridge verbs')
  → since none of the CPs in (10a) are phasal, paths can successfully be constructed from the offload position of *how quickly* on the edge of the matrix TP all the way down into the most deeply embedded clause
  → the path from *how quickly* in the matrix TP-adjoined position down to its gap can be very long thanks to the fact that not a single node along the way constitutes a phase

(11/12) the CP below the manner-of-speaking or factive verb is not an Agree-goal for the matrix *v* (manner-of-speaking verb constructions are 'light verb' constructions: *give a sigh*; for concreteness I follow the Kiparsky & Kiparsky approach to factives, with a silent noun FACT)

→ the CP in (11/12a) is a phase, and hence opaque

→ it is impossible for there to be a path leading down from the offload position of *how quickly* on the edge of the matrix TP down to a gap inside CP

- 'ECP effects' occur in *wh*-island contexts as well: a non-argument *wh*-element cannot build a filler–gap dependency across a *wh*-operator in a lower clause

(29) a. how quickly did Bob ask which lightbulb Bubba had replaced?
b. *Bob asked which lightbulb it took Bubba only a minute to replace
c. it took Bob only five seconds to ask which lightbulb Bubba had replaced

→ from its offload position on the edge of the matrix TP (the first predication structure), *how quickly* in (29a) must try to build a path to a gap in a position from which the manner adverb can be associated to its modifiee

→ regardless of the question of whether the CP-complement of *ask* is an Agree-goal or not, no path can be constructed from *how quickly* into the subordinate clause because of the intervention of *which lightbulb*, which sets up a minimality effect

- an important **argument/non-argument distinction**, ultimately key to the classic 'ECP'

(30) a. argumental fillers can be re-uploaded onto the stack
b. non-argumental fillers usually cannot be re-uploaded onto the stack

→ for non-argumental fillers, it is usually possible to syntactically and semantically associate them with the projection of the matrix verb in multi-clausal constructions: in the a–sentences in (10)–(12) and (29), offloading *how quickly* onto the edge of the matrix TP and establishing a semantic connection with the matrix VP is perfectly interpretable in principle

→ for argumental *wh*'s 'accidentally' linked to the object position of the matrix verb, on the other hand, re-uploading is motivated if no syntactic and/or semantic association between the argument and the matrix verb can be forged

- re-upload is not without cost

→ while <u>non-argumental</u> *wh*'s usually cannot establish a connection with a subordinate clause across a *wh*-element at the edge of that clause, for <u>argumental</u> *wh*-expressions such a connection often cannot but sometimes *can* be built (which is precisely why Ross 1967 stopped short of postulating a general '*wh*-island constraint')

→ (31) is modelled on Ross's (1967) original examples
– (31e) is strongly degraded
– (31c,d) are a little bit better than (31e)
– (31a–b) are slightly marginal but clearly within the realm of grammaticality

(31) a. ?what can't you figure out why she read to him?
b. ?what can't you figure out whether she read to him?
c. ??what can't you figure out when she read to him?
d. ?*what can't you figure out how she read to him?
e. *what can't you figure out who she read to?

- once a *wh*-expression has been offloaded onto the edge of the matrix TP, it cannot build a path to a trace inside an embedded *wh*-question: the intervention of the *wh*-element on the edge of the subordinate clause leads to a minimality effect
- → with *what* offloaded onto the matrix TP, we would seem to condemn all the examples in (31) uniformly to ungrammaticality
- → but for argumental *wh*'s, there is the option of re-uploading the *wh*-expression onto the *wh*-stack and proceeding into the subordinate clause
- → carrying the *wh*-stack over into the embedded clause is itself costless: push-down stacks are simply carried along in the course of the structure-building process, and not sensitive to phases

(32) stacks can be carried into subordinate clauses regardless of whether these are phases

- → the status of (31) has nothing to do *directly* with the traversal of the *wh*-island; the penalty attributed to them has a different source

- when *what* is first offloaded onto the edge of the *matrix* TP, the grammar tries to postulate a trace inside this TP to which *what* can build a path in accordance with locality constraints
- → *figure out* is perfectly capable of taking *what* as its complement
- → on the first pass, the grammar postulates a trace in the complement of *figure out* for *what* to bind
- → if the sentence had come to a full stop after this, we would have been done: we would have constructed a grammatical structure for *what can't you figure out?*
- → but then we discover upon the encounter of the *wh*-element *when* or *who* that the verb *figure out* is taking an interrogative CP as its complement
- → our initial hypothesis to link *what* to *figure out* then has to be revised ('backtracking'): we re-upload *what* onto the *wh*-stack, and **we get rid of the trace that had been postulated**
- → re-uploading *what* is itself costless, just like uploading fillers onto a push-down stack in general comes at no cost; the penalty comes from the **erasure of the *wh*-trace** that had initially been postulated in the complement of *figure out*

- when we get to the *wh*-element introducing the embedded question, we must revise our initial hypothesis that *what* binds a gap in the matrix VP
- → we must re-upload *what* onto the *wh*-stack, but the *wh*-element introducing the lower clause must also go onto that stack

Q what will be the relative placement of the two *wh*'s on the stack?

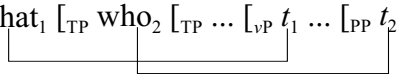
- we reach our decision to reassess the initial structure assigned to the matrix VP only after encountering the *wh*-element introducing the embedded clause
- → the incoming *wh*-element is immediately and automatically placed on the *wh*-stack, *before* we 'backtrack' and revise the structure of the matrix VP
- → once we conclude that *what* must be re-uploaded onto the *wh*-stack, we have by that time already placed on the *wh*-stack the *wh*-element introducing the embedded question
- → since the *wh*-stack is a push-down stack, this means that *what* ends up **on top of** the *wh* introducing the lower clause (wh_2): after re-uploading of *what*, our *wh*-stack looks as in (33a)
- → when we reach the TP of the lower clause, we offload this *wh*-stack, in LIFO fashion, attaching the offloaded *wh*'s as adjuncts to the TP
- → *what* pops off the stack first, and ends up attached to TP above wh_2, as in (33b)
- → with the internal structure of the TP subsequently completed to include traces for the *wh*'s, paths are constructed from the TP-adjoined *wh*-copies to their traces inside TP
- → for (31e), the paths clearly cross, and the sentence is ruled ungrammatical: (33c)

(33)

a. $what_1$ ... [$_{CP}$ wh_2
STACK:
[$what_1$ [wh_2]]

b. $what_1$ ... [$_{CP}$ wh_2 [$_{TP}$ what$_1$ [$_{TP}$ wh$_2$

c. *$what_1$ ... [$_{CP}$ who_2 [$_{TP}$ what$_1$ [$_{TP}$ who$_2$ [$_{TP}$ ... [$_{vP}$ t_1 ... [$_{PP}$ t_2 ...]]]]]]



- for (31c,d), the paths from *what*'s re-offload position down to t_1 and from wh_2's offload position down to t_2 do not strictly speaking cross but they are not properly nested either
- → assuming that in English object-*what* binds a trace on the edge of *v*P (the 'object shift' position) and that *how* and *when* do so likewise, the two *wh*-paths in (31c,d) lead down to the same node in the tree (*v*P), which makes it impossible for the paths to be in a properly nested dependency
- → the fact that there is no proper nesting in (31c,d) is responsible for the degradation of these sentences — though the fact that there are at least no crossing paths, unlike in the case of (31e), makes (31c,d) somewhat less bad
- → the relative contrast between (31c) and (31d) is probably relatable to the fact that *when*, in its offload position on TP's edge, finds a *semantic* associate without the postulation of a trace
- → though temporal adverbial modifiers are arguably attached quite low in the syntax (despite their semantic link to the projection of T), *when* can forge a semantic link with TP
- → for proper syntactic construal, a trace for *when* inside TP is necessary, so without this trace, the syntax of (31c) is not syntactically well-formed — but at least the semantic interpretation of (31c) does not go awry if no trace (and concomitant path) is postulated by the grammar
- → (31d,e) are both worse because not only are the paths not nested, there is also no chance of *not* postulating traces for *how* and *who*: manner modifiers are interpreted as modifiers of the predicate, not of the predication as a whole; and *who* is an argument of the preposition *to*

- for (31a,b), only one *wh*-path needs to be built in the lower clause: *why* and *whether* do not bind traces, so no path containment issues ever arise there

(34)

a. *what*$_1$ ... [$_{CP}$ *why/whether*$_2$
STACK:
[*what*$_1$ [*why/whether*$_2$]]

b. *what*$_1$ ... [$_{CP}$ *why/whether*$_2$ [$_{TP}$ what$_1$ [$_{TP}$ why/whether$_2$

c. *what*$_1$ ... [$_{CP}$ *why/whether*$_2$ [$_{TP}$ what$_1$ [$_{TP}$ why/whether$_2$ [$_{TP}$ ... t_1 ...]]]]

- the path-theoretic account given for (31e) raises the expectation that we should see an improvement with *what* and *who* changing places — and we do (Pesetsky 1982, Richards 1997)

(35) a. *which sonata did you ask which violin to play on?
b. ?which violin did you ask which sonata to play on?

(36)

a. *which violin*$_1$ ... [$_{CP}$ *which sonata*$_2$
STACK:
[*which violin*$_1$ [*which sonata*$_2$]]

b. *which violin*$_1$ ... [$_{CP}$ *which sonata*$_2$ [$_{TP}$ which violin$_1$ [$_{TP}$ which sonata$_2$

c. *which violin*$_1$ ... [$_{CP}$ *which sonata*$_2$ [$_{TP}$ which violin$_1$ [$_{TP}$ which sonata$_2$ [$_{TP}$ ... t_2 ... t_1 ...]]]]

- we also have an account of the classic Chomsky (1977) contrast in *tough*-movement constructions

(37) a. *which sonata is this violin easy to play on?
b. which violin is this sonata easy to play on?

→ the account is the same as the one presented for the contrast in (35), with two small differences
(a) the second *wh*-operator is a null operator in the *tough*-movement constructions, and
(b) trace erasure does not need to take place in the course of the derivation of (37)
– *ask*, the matrix predicate head in (35), could in principle accommodate the trace of the matrix *wh*-operator (it is awkward to ask a violin something, but it is not syntactically impossible)
– *tough*, the matrix predicate head in (37), cannot accommodate a trace for *which violin* at all: *tough* is not a transitive predicate — the grammar, working its way down (37b), is never 'trapped' into postulating a trace for *which violin* in the matrix clause; *which violin*, after having been offloaded from the *wh*-stack on the edge of the matrix TP, simply gets re-uploaded once we get to the infinitival clause, without a trace needing to be erased

→ *(b)* is responsible for the fact that (37b) is essentially perfect while (35b) is slightly marginal
→ while the grammar is 'garden-pathed' in (35b), it is not in (37b)
→ the lack of a 'Subjacency effect' in (37b) vindicates our earlier conclusion that such an effect is the result of trace erasure

Q why do subjects not manage to build filler-gap dependencies across a local *wh*-island at all?

(38) a. *who can't you figure out why read this poem to him?
b. *who can't you figure out whether read this poem to him?
c. *who can't you figure out when read this poem to him?
d. *who can't you figure out what read to him?
e. *who can't you figure out who read this poem to?

(39)
a. who_1 ... [$_{CP}$ wh_2
STACK:
[who_1 [wh_2]]
b. who_1 ... [$_{CP}$ wh_2 [$_{TP}$ who$_1$ [$_{TP}$ wh$_2$
c. *who_1 ... [$_{CP}$ wh_2 [$_{TP}$ who$_1$ [$_{TP}$ wh$_2$ [$_{TP}$ t_1 ... (t_2)]]]]

→ the problem with (39c) is entirely independent of any path-theoretic constraints, and therefore manifests itself regardless of the nature of wh_2
→ because of the fact that who_1 is at the top of the push-down stack of the embedded CP, it gets offloaded onto the edge of TP first, in keeping with the LIFO procedure
→ the other *wh*-element on the stack is subsequently placed in an adjunction position to TP as well, after which the rest of the TP is built
→ if who_1 had been offloaded last (or all by itself) onto the edge of TP, it could have served perfectly well as the subject of predication right in its offload position
→ but in (39b) a direct predication relation between the *v*/VP and the subject on the edge of TP, with T as the RELATOR, is impossible
→ postulating a trace of who_1 in SpecTP, as in (39c), is the only way to ensure a proper predication relation between *v*/VP and the subject (i.e., who_1)
→ but who_1 cannot legitimately bind a trace in SpecTP because the link between the TP-adjoined copy of who_1 and t_1 is too short: it violates 'anti-locality' (Grohmann, Abels, etc.): representationally speaking, no element is allowed to be on the edge of the same phrase twice

• so in the derivation of (39), we either get a predication failure (if we do not postulate a trace for wh_1 in the subordinate SpecTP) or (if we do, as in (39c)) an 'anti-locality' violation
→ no matter whether we do or do not postulate a trace for wh_1 in the lower clause, the sentences in (38) *all* will be ruled ungrammatical — entirely regardless of whether the *wh*-element offloaded onto TP after who_1 binds a trace lower down (as in the case of (38c–e)) or not (as with *why* and *whether* in (38a,b))