Glossa a journal of general linguistics

Ginsburg, Jason. 2016. Modeling of problems of projection: A non-countercyclic approach. *Glossa: a journal of general linguistics* 1(1): 7. 1–46, DOI: http://dx.doi.org/10.5334/gjgl.22

RESEARCH

Modeling of problems of projection: A non-countercyclic approach

Jason Ginsburg
Osaka Kyoiku University, JP
jginsbur@cc.osaka-kyoiku.ac.jp

This paper describes a computational implementation of the recent Problems of Projection (POP) approach to the study of language (Chomsky 2013; 2015). While adopting the basic proposals of POP, notably with respect to how labeling occurs, we a) attempt to formalize the basic proposals of POP, and b) develop new proposals that overcome some problems with POP that arise with respect to cyclicity, labeling, and wh-movement operations. We show how this approach accounts for simple declarative sentences, ECM constructions, and constructions that involve long-distance movement of a wh-phrase (including the *that*-trace effect). We implemented these proposals with a computer model that automatically constructs step-by-step derivations of target sentences, thus making it possible to verify that these proposals work.

Keywords: cyclicity; labeling; feature inheritance; wh-movement; computational modeling

1 Introduction

In the recent Problems of Projection (POP) approach to the study of language (Chomsky 2013; 2015), Chomsky proposes that some of the core notions (notably, endocentricity) of recent work in the Minimalist Program (Chomsky 1995, etc.) be done away with; i.e., these notions are not necessary to account for language. This paper describes a computational implementation of POP that accounts for the main examples from Chomsky (2015). While adopting the basic proposals of POP, notably with respect to how labeling occurs, we a) attempt to formalize the basic proposals of POP, and b) develop new proposals that overcome some problems/issues with POP with respect to cyclicity, labeling, and wh-movement operations. We implemented these proposals with a computer model that automatically constructs step-by-step derivations of target sentences, thus making it possible to verify that these proposals work. We explain how this POP-based computational implementation works.

2 POP: Core algorithms

This section presents the core operations and principles that are necessary to construct derivations in the POP model. These are a) taken directly from Chomsky (2013; 2015) (when possible), b) modifications of proposals from Chomsky (2013; 2015), and c) new proposals that overcome problems that arise in the POP system.

2.1 Merge and labeling

As is typical in recent work in the Minimalist Program (Chomsky 1995, etc.), the operation of Merge is a core component of POP, whereby two syntactic objects (SOs) are combined to form another SO. This operation is described below (adapted from Chomsky 2001: 3).

(1) **Merge**
Merge syntactic objects X and Y to form the syntactic object Z = {X, Y}.

The need for an SO to be labeled is also a core component of POP. According to Chomsky (2015: 6), the labeling algorithm is "a special case of minimal search (like Agree)." The labeling operation of POP is summarized by Chomsky (2013: 43) as follows:

> "Suppose SO = {H, XP}, H a head and XP not a head. Then LA [labeling algorithm] will select H as the label, and the usual procedures of interpretation at the interfaces can proceed.
>
> The interesting case is SO = {XP, YP}, neither a head… Here minimal search is ambiguous, locating the heads X, Y of XP, YP, respectively. There are, then, two ways in which SO can be labeled: (A) modify SO so that there is only one visible head, or (B) X and Y are identical in a relevant respect, providing the same label, which can be taken as the label of the SO."

When a head that is strong enough to label merges with an XP, the labeling algorithm LA finds the head and the head becomes the label. When an XP merges with a YP, the labeling algorithm "finds {X, Y}, the respective heads of XP, YP, and there is no label unless they agree (Chomsky 2015: 7)." This latter proposal does away with the notion of endocentricity, since there is no head of this {XP, YP} projection. Rather, there are two primary ways that an SO of the form {XP, YP} can be labeled. One way labeling can occur is if the SO is modified via movement of the XP or YP out of the SO.[1] For example, if XP moves out of an SO of the form {XP, YP} and merges in a higher position, then the label of the SO becomes the label of the YP, since not all instances of XP are contained within this SO. Note that movement here refers to merge of another instance of the XP in a higher position in the structure. Epstein, Kitahara, and Seely (2014) basically describe this approach as follows: given SO_k = $\{_A$ NP, {T, $\{_B$ NP, {v, VP}}}}, the highest copy/instance of the NP is in the domain of the entire SO that is labeled A, since every instance of the NP is within this domain. The lower NP instance, however, is not in the domain of B since not every instance of this NP is within this domain. Thus, the label of B is the label of {v, VP}. Another way that an SO of the form {XP, YP} can be labeled is if XP and YP share, via agreement, prominent features; prominent features discussed by Chomsky are phi-features and a Q feature, although the possibility of other features that are capable of labeling remains. So if XP and YP in an {XP, YP} structure have identical phi-features (resulting from an agreement relation), the labeling algorithm identifies these shared features as the label. The labeling operation is summarized below. A "strong head" is a head that is strong enough to label.[2]

(2) **Labeling**
a. When a strong head X is merged, the label is X.
b. If {XP, YP} share prominent features that are capable of labeling, the shared features label.
c. If YP moves out of {XP, YP}, then XP labels. If XP moves out of {XP, YP}, YP labels.

[1] This proposal of Chomsky is based on work by Moro (2000).

[2] An anonymous reviewer points out that labeling, as described in (2) is the output of the Labeling Algorithm (LA). LA is a search for possible labels in a syntactic object. Labeling in (2) describes the cases in which LA successfully is able to determine a label.

Feature inheritance also plays an important role in POP. Chomsky (2013) follows Richards' (2007a) proposal that T inherits its features from C, which accounts for the difference between finite and non-finite T. Finite T inherits its features from C, whereas non-finite T occurs without C, and thereby lacks core tense and agreement features. Chomsky (2015) suggests that feature inheritance also occurs within a v*P, so that the features of v* are inherited by a verbal root. The feature inheritance operation is summarized as follows.[3]

(3) **Feature inheritance (version 1)**
A phase head passes its features onto its complement.

One question arises with respect to shared features: when an {XP, YP} structure is labeled via shared features, if YP is unlabeled, how does YP obtain a label? According to Chomsky (2015), lexical roots, as well as T, are too weak to label. In Figure 1a below, there is a verbal root *read* that merges with an object *a book*. Assume that when v* is merged, the uPhi (uninterpretable phi-features) of v* are inherited by *read*, and the uPhi agree with *a book*, which remerges with *read*. The result is that shared phi-features label the structure {a book, read a book}. But it is not clear what the label is of the structure {read, a book}. Similarly, when a subject remerges with T (present tense Tpres), uPhi of T (which are inherited from C) agree with the phi-features of the subject, so that shared phi-features label (Figure 1c). But assuming that T is too weak to label, then it isn't clear what the label of the lower {T, XP} structure is.

Chomsky (2015) relies on strengthening to account for labeling of projections that are initially too weak. Chomsky (2015: 10) writes, "[j]ust as English T can label TP after strengthening by SPEC-T, so R [Root] can label RP after object-raising."[4] In Figure 1c, shared phi-features (indicated as Phi2) label the previously unlabeled SO {Tom, Tpres read a book}, thereby strengthening Tpres, so that Tpres can label the intermediary projection (see Figure 1d). Similarly, in Figure 1a, shared phi-features (Phi1) label the structure {a book, read a book}, and this results in strengthening of the root *read*, so that *read* can label the intermediary projection (see Figure 1b).

Chomsky refers to "strengthening" in this labeling process, raising the question of what exactly strengthening is. Strengthening refers to the process whereby a projection that is "unlabelable", due to the lack of a prominent feature, becomes "labeable". In Figure 1a and 1c, the previously unlabelable {XP, YP} projections obtain shared phi-features, thus becoming labelable (i.e., they are strengthened). In Figure 1a–b, the intermediary projection has the verbal root *read* as a root node, where *read* initially contains no prominent features that are capable of labeling. After *read* inherits uPhi from v* and these uPhi are checked, then *read* contains checked phi-features, which are capable of labeling because they are visible to the labeling algorithm; thus *read* is strengthened. Similarly, in Figure 1c–d, T, which is initially too weak to label, is strengthened after its inherited uPhi features are checked, so that it can label. We formulate strengthening as follows.

[3] Note that this will be revised below. See (19).
[4] We thank an anonymous reviewer for helping to clarify this point regarding strengthening.

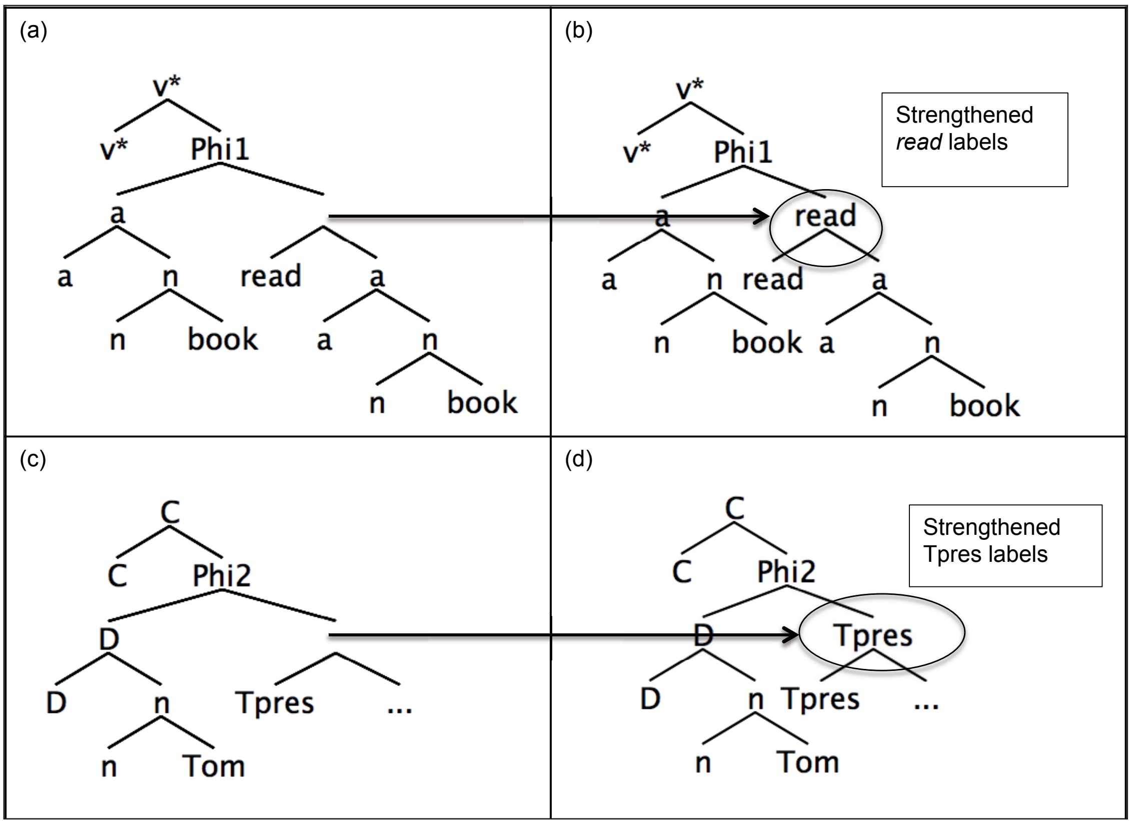


Figure 1: Labeling of intermediary projections via strengthening[5] ("Tom reads a book.").

(4) **Strengthening**
The process in which an unlabeled SO obtains prominent features that are capable of labeling.

Note that Figure 1 could also be represented as shown below in Figure 2, with the strengthened intermediary nodes labeled by the prominent features. When an intermediary projection is strengthened, the labeling algorithm finds the prominent features in the root node of that projection, and in the cases in Figures 1–2, the prominent features are phi-features. Thus, it is reasonable to assume that the labels of the intermediary projections in these cases are phi-features. However, we will use the type of representation shown in Figure 1 because it is easier to read.

2.2 Cyclicity

In the original version of POP from Chomsky (2013), derivations are counter-cyclic. Consider the derivation of (5).

(5) Tom read a book. (Chomsky 2015: 10)

As shown in Figure 3a, v* is merged, and then features are inherited by the verbal root *read*. Next, shown in Figure 3b, there is an agree relation formed between the inherited uPhi of v* and *a book*, followed by remerge of *a book*. This remerge operation is necessary to enable labeling via shared phi-features. Note that remerge of the object does not occur until after v* is merged, so this operation is counter-cyclic. This violates the Extension Condition (Chomsky 1995), since it requires altering, via movement, an already constructed SO. Shared phi-features label the SO, with an {XP, YP} structure, formed

[5] The numbers 1 and 2 in Phi1 and Phi2 signify that these are different sets of phi-features, with Phi1 originating from the object and Phi2 originating from the subject.

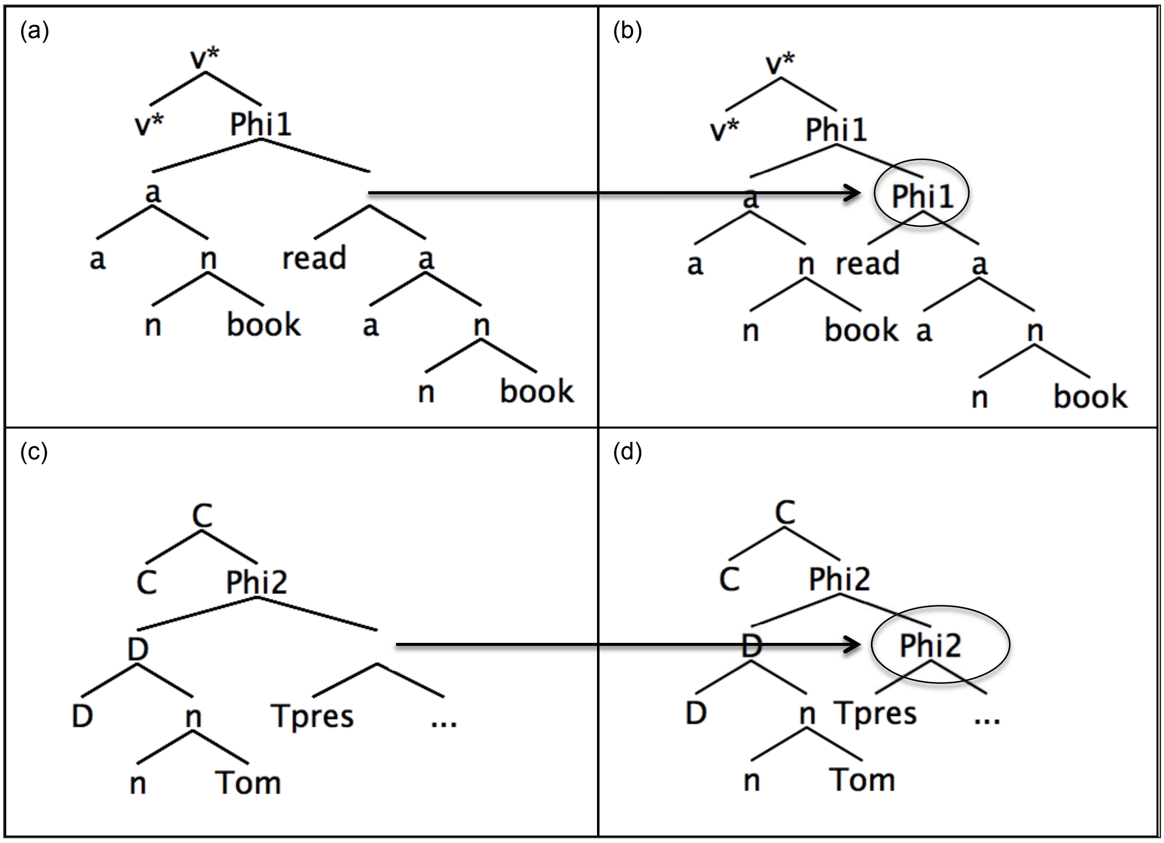


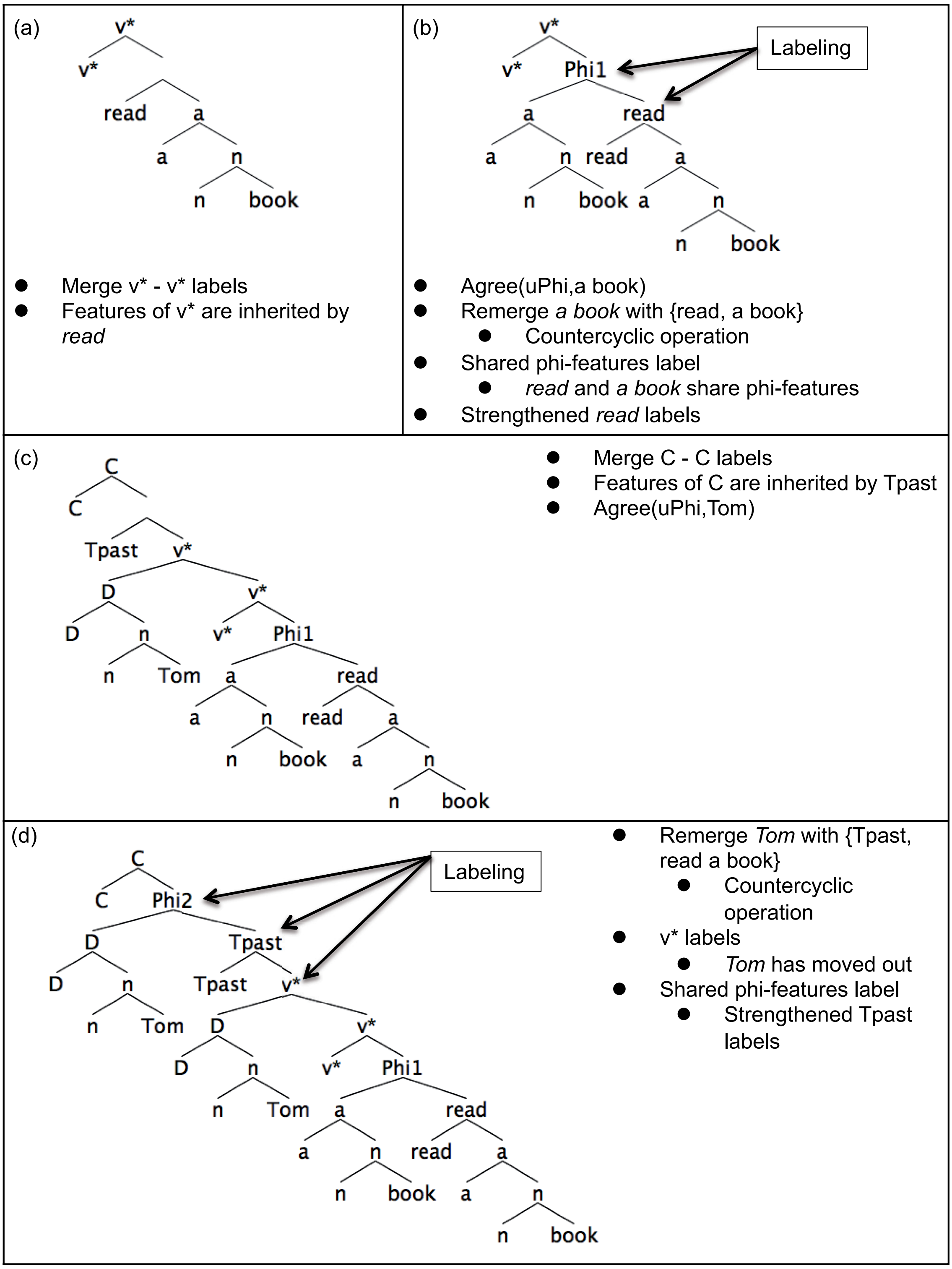
Figure 2: Labeling of intermediary projections with phi-feature label ("Tom reads a book.").

from the remerged object and the verbal root, and the lower structure {read, a book} is labeled by the strengthened *read*, in accord with Strengthening, defined in (4) above. Figure 3c shows merge of C. At this point, the uPhi of C are inherited by Tpast, and an agree relation is established between the inherited uPhi of Tpast and the subject. As shown in Figure 3d, the subject remerges counter-cyclicly. Remerge of the subject enables v* to label – movement of *Tom* leaves v* as the most prominent XP in the relevant structure so that v* can label. Then due to the relation Agree(Tpast,Tom), the shared phi-features of the subject and Tpast label, and the strengthened Tpast labels.

Chomsky (2015) proposes a non-countercyclic approach. He writes that the counter-cyclic operation of movement of a subject to T after merge of C "is problematic, as pointed out by Epstein, Kitahara, and Seely (2012), because it involves a substitution operation that is ternary, even though only narrowly so (Chomsky 2015: 13)."[6] A cyclic derivation avoids this undesirable substitution operation.

Chomsky (2015) suggests a way in which a non-counter-cyclic derivation is possible; this proposal requires that movement precede feature inheritance. The relevant portions of the bottom-up derivation of (5), with no counter-cyclic movement operations, proceed as shown in Figure 4. After the verbal root *read* is merged, the object *a book* remerges with the unlabeled structure {read, a book}, producing another unlabeled structure (Figure 4a).

[6] According to Epstein, Kitahara, and Seely (2012: 256), a counter-cyclic substitution operation creates what they refer to as a "two-peaked object" that consists of two "intersecting set-theoretic SOs [that] do not have a single root." For example, assume that a subject remerges counter-cyclicly with {T, vP} after features are inherited from C. In this case, a substitution operation initially results in two sets consisting of {Subject, T^1} and {C, T^1} where T^1 corresponds to {T, vP}. These sets both contain T^1, and thus intersect with respect to T^1, but they are not dominated by an identical root node, until they converge as a structure in which the subject is in [Spec, T].



(a)

- Merge v* - v* labels
- Features of v* are inherited by *read*

(b)

- Agree(uPhi,a book)
- Remerge *a book* with {read, a book}
  - Countercyclic operation
- Shared phi-features label
  - *read* and *a book* share phi-features
- Strengthened *read* labels

(c)

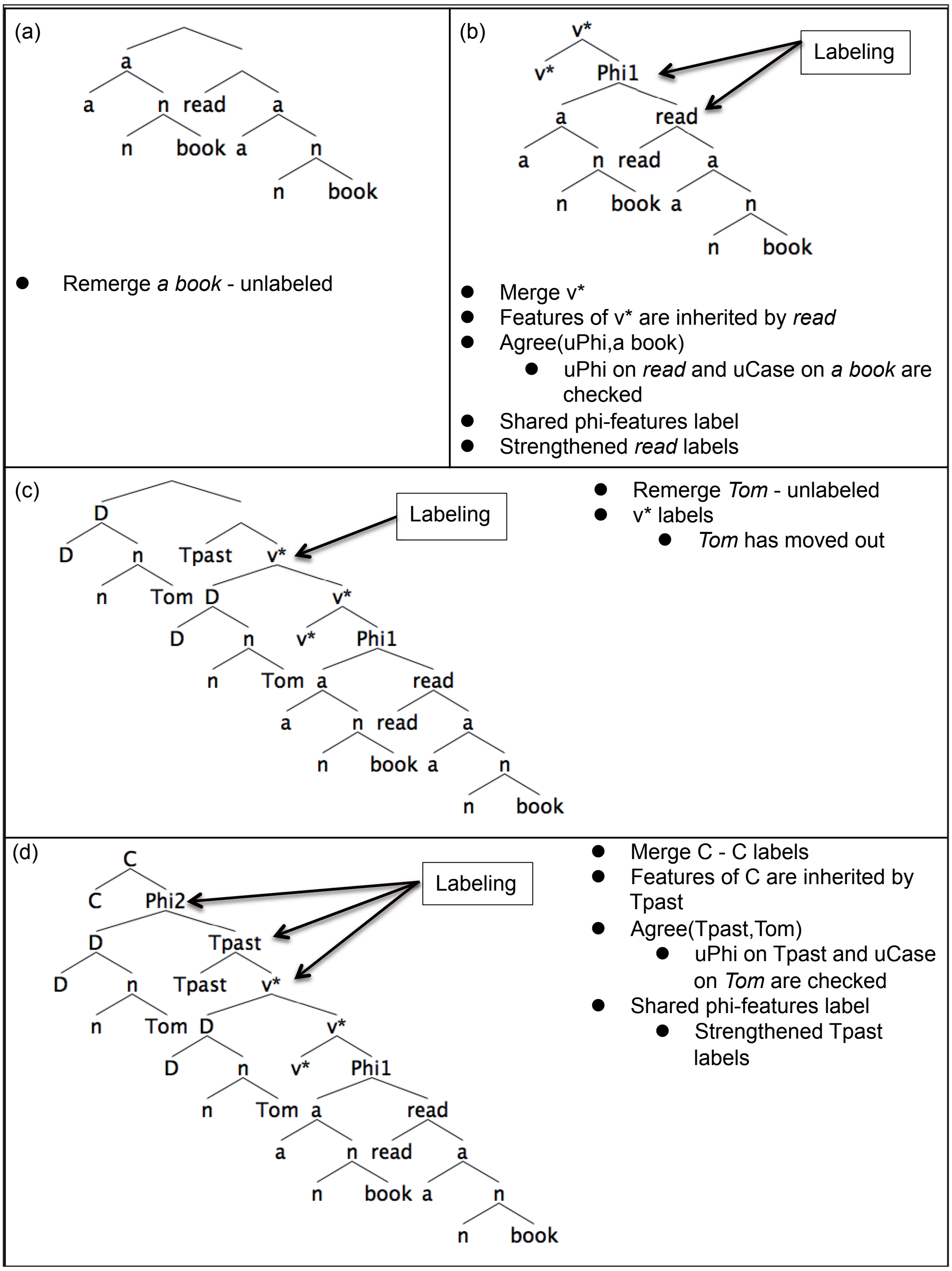
- Merge C - C labels
- Features of C are inherited by Tpast
- Agree(uPhi,Tom)

(d)

- Remerge *Tom* with {Tpast, read a book}
  - Countercyclic operation
- v* labels
  - *Tom* has moved out
- Shared phi-features label
  - Strengthened Tpast labels

Figure 3: "Tom read a book." with countercyclic steps.

After v* is merged (Figure 4b), the features of v* are inherited by *read*. Then the shared phi-features of *read* and the object label. Later in the derivation, shown in Figure 4c, the subject remerges with the Tpast structure to form an unlabeled structure. Note that movement of the subject enables v* to label, since not all copies of the subject are contained within this structure. At the last stage, shown in Figure 4d, C is merged, followed by inheritance of the uPhi of C by Tpast. Tpast agrees with the subject, and uPhi and uCase are checked. The phi-features shared between the Tpast structure and the subject label.



(a)

- Remerge *a book* - unlabeled

(b)

- Merge v*
- Features of v* are inherited by *read*
- Agree(uPhi,a book)
  - uPhi on *read* and uCase on *a book* are checked
- Shared phi-features label
- Strengthened *read* labels

(c)

- Remerge *Tom* - unlabeled
- v* labels
  - *Tom* has moved out

(d)

- Merge C - C labels
- Features of C are inherited by Tpast
- Agree(Tpast,Tom)
  - uPhi on Tpast and uCase on *Tom* are checked
- Shared phi-features label
  - Strengthened Tpast labels

Figure 4: "Tom read a book." with no countercyclic steps.

This derivation shown in Figure 4 has the advantage over the derivation shown in Figure 3 in that it is not counter-cyclic, as there is no need to alter (at least with respect to movement) an already formed SO. We adopt this non-countercyclic view.

2.3 *Probing from the root node and feature unification*

POP faces problems with respect to feature checking and cyclicity. As pointed out by Richards (2007b), probing in Phase Theory (Chomsky 2000; 2001; 2008, etc.) is technically only a property of root nodes (e.g., the root node T probes for and agrees with a goal subject in the specifier of the projection of v*). Probing from a non-root node is counter-cyclic, since it involves altering an already constructed syntactic object. Furthermore, limiting probing to only the root node (the top of the current structure) simplifies the derivational process, since only the root node is involved in probing. But feature inheritance implies that there can be probing from a non-root node. Consider what happens at the v* phase level, as shown in Figure 5. The uPhi of v* are inherited by the verbal root *read*, and *read* eventually agrees with the object *a book*. If the verbal root *read* probes, via its inherited uPhi features, and agrees with the object (either before or after the object has moved), then there is probing from a non-root node, since the root node is v*, not *read*.

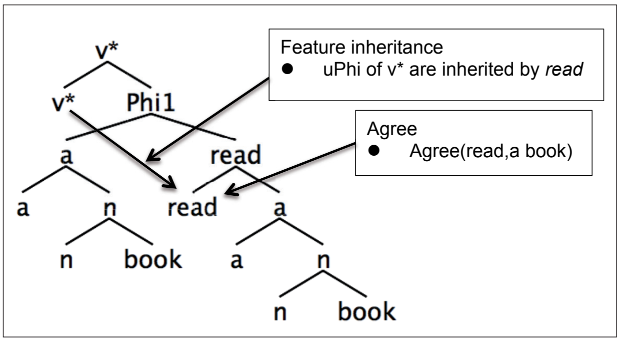


Figure 5: Probing from non-root node *read*.

We take the position that probing is limited to root nodes, relying on the idea that feature inheritance leads to multiple unified (i.e., identical) instances of a feature in a structure, following Fong (2014) who takes a similar approach to account for multiple agreement in Chomsky (2001).[7] When any instance of this feature is checked, then all instances are checked. This view is summarized in (6).

(6) **Probing from the root node, feature unification, and feature checking**
  a. Only a root node can probe.
  b. Feature inheritance leads to unified instances of a feature on multiple SOs.
  c. An uninterpretable feature is checked when it is valued by a matching interpretable feature.

For example, in Figure 6, uPhi of v* are inherited by the verbal root *read*. Thus v* and *read* have unified uPhi features. The uPhi on v* also probe and agree with *a book*. This relation Agree(v*,a book) checks the uPhi on v*, as well as the unified uPhi on *read*. As a result of this feature checking relation, *read* now has phi-features that are shared with *a book*, thereby enabling labeling via shared phi-features. Crucially, only the uPhi on the root node v* probe. Thus, in a single probing operation, the uPhi on v* probe and find *read* (resulting in feature inheritance) and *a book* (resulting in feature checking).[8]

An anonymous reviewer points out that feature inheritance, whereby features pass from a root node to a non-root node in a previously constructed syntactic object, alters an already formed syntactic object, and thus, is counter-cyclic. For example in Figure 6, uFs inherited from v* are passed to *read*, thereby altering an already constructed SO. While this may be an imperfection of the current system, it is one that is inherited from POP, since POP requires feature inheritance. However, we think that our system is an

[7] Frampton & Gutmann (2000) and Pesetsky & Torrego (2007) develop similar analyses in which features that agree are shared. In both of these approaches it is possible for unvalued features to unify and then later be checked by a valued feature. For example, Pesetsky & Torrego (2007) propose that an unvalued interpretable feature and a matching unvalued uninterpretable feature can be shared via agreement. Then, if one of these shared features is valued, all shared instances of this feature are valued.

[8] An anonymous reviewer points out that feature unification is the "null assumption" about how instances/copies work in syntax, since multiple copies/instances of an SO should be identical. The idea that this is the null assumption certainly fits in with our analysis; the idea that all instances of elements are identical is nothing special.

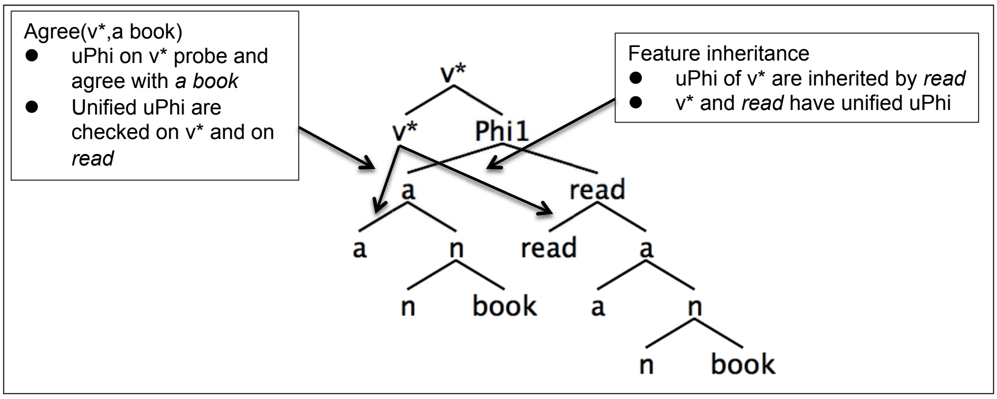

Agree(v*,a book)
uPhi on v* probe and agree with *a book*
Unified uPhi are checked on v* and on *read*
Feature inheritance
uPhi of v* are inherited by *read*
v* and *read* have unified uPhi
v*
v*
Phi1
a
read
a
n
read
a
n
book
a
n
n
book

Figure 6: Probing from the root node and feature unification.

improvement over the manner in which feature checking occurs in POP, because in our system probing is limited to a root node, whereas in POP, it appears as though probing is not limited to root nodes.

2.4 *Movement*

In POP, movement operations are closely connected with labeling, due to the fact that movement is required in order to produce configurations that can be labeled via shared features. The configuration in Figure 7a shows the structure necessary for shared feature labeling. Starting with a label-less structure, formed from a lexical root or head that is too weak to label and an XP complement, as shown in Figure 7b, if the XP remerges with the label-less structure (Figure 7c) and features are shared between the two structures, then labeling occurs, as shown in Figure 7d (assuming that shared phi-features label). In this case, the previously unlabeled intermediary projection is labeled as a result of strengthening, as defined in (4), after its inherited uPhi are checked. This is the type of structure that is found in object shift constructions. Another method of obtaining a label is shown in Figure 7e–f. In this case, a phrasal structure ZP, which is contained within the unlabeled SO complement of Y, remerges with the root node (i.e., undergoes internal merge). If ZP and the unlabeled SO share phi-features, then the shared features label, as shown in Figure 7f. This is the type of movement that occurs when a subject moves from the vP level to the TP level. Chomsky (2015: 9) proposes that tense in languages such as English "is too "weak" to serve as a label." Therefore, movement of a DP to remerge with T enables labeling via shared phi-features, as in Figure 7f.[9]

Assuming that derivations are cyclic, given an unlabeled structure, the closest SO to the unlabeled root node that has phi-features remerges with the root node. In Figure 7c, the closest SO with phi-features is XP, whereas in Figure 7e, the closest SO with phi-features is ZP.

[9] This proposal accounts for the fact that a subject usually must be overt in English. Chomsky (2015) suggests that in languages such as Italian, which don't require an overt subject, T is able to label without sharing features with a remerged subject.

An anonymous reviewer asks why phi-feature agreement is even necessary for labeling, given our assumptions about labeling. With Strengthening (4), it is possible for T and a verbal root to obtain checked phi-features (which are strong enough to label) in the absence of a remerged argument. For example, T and a verbal root inherit uPhi from C and v*, respectively. When the inherited uPhi are checked, the now strengthened T and verbal root should be able to label. We assume, following Chomsky (2015), that phi-feature sharing is necessary for labeling a verbal root and T in English. In other words, there is a particular property of T and certain roots that requires feature sharing for labeling – pure strengthening in the absence of feature sharing is not enough. This does not necessarily hold in all languages. For example, in languages such as Italian, strengthening of T, in the absence of a remerged argument with which it shares phi-features, may be sufficient for labeling.

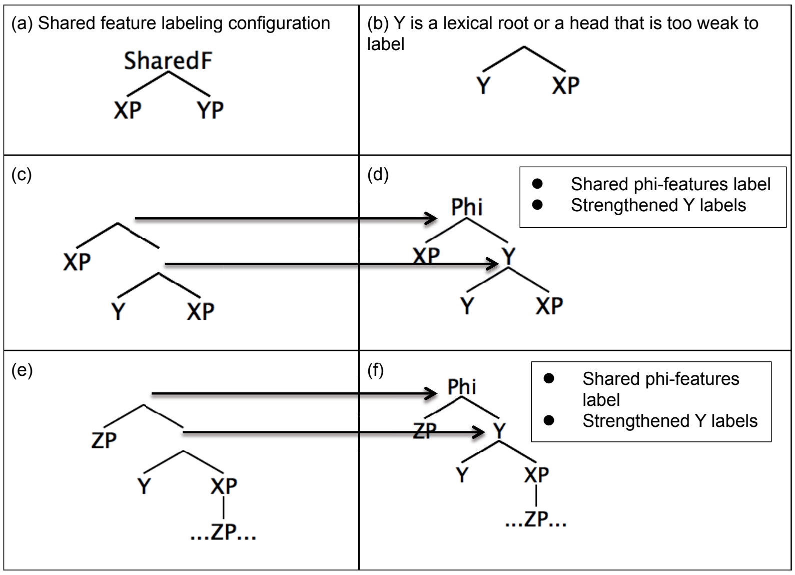

(a) Shared feature labeling configuration
SharedF
XP
YP
(b) Y is a lexical root or a head that is too weak to label
Y
XP
(c)
XP
Y
XP
(d)
● Shared phi-features label
● Strengthened Y labels
Phi
XP
Y
Y
XP
(e)
ZP
Y
XP
...ZP...
(f)
● Shared phi-features label
● Strengthened Y labels
Phi
ZP
Y
Y
XP
...ZP...

Figure 7: Labeling and movement.

Figure 7 shows instances in which movement turns a label-less structure into a labeled structure, but movement does not always result in labeling, as can be seen with respect to wh-movement. Examples (7b–c) show relevant portions of the structure of (7a).

(7) a. What do you think that John read?
b. [_ what [$_C$ that] John read ~~what~~]?
c. [$_Q$ What [$_{C\_Q}$ C_Q do you think [$_C$ ~~what~~ [$_C$ that] John read ~~what~~]]]?

In POP, under normal circumstances, the complement of a phase head is transferred (see section 2.5 below). In (7b), *what* must remerge with the embedded C projection; otherwise, when the complement of C is transferred, *what* will be transferred. Note that (7b) is an unlabeled {XP, YP} structure. (7c) shows the matrix clause, with an interrogative C, indicated as C_Q. If we understand correctly, according to Chomsky (2013; 2015), movement of a wh-phrase in this type of construction is required to create a structure that is labeled by a Q feature that is shared between the wh-phrase and the interrogative C. This shared Q label is needed for semantic reasons, to produce a well-formed wh-question.[10]

The core movement operations are summarized in (8).

(8) **Movement operations**
a. When an unlabeled SO is formed, remerge the closest available SO with phi-features.
b. Movement of an SO occurs to create a structure that can be labeled for semantic reasons.
c. An unlicensed SO with an uninterpretable feature that is about to be transferred remerges with the root node (if possible).

If there is an unlabeled structure, movement occurs whereby the closest available SO with phi-features remerges with the root node. An SO is available if it is not within a transferred domain.[11,12] Movement also occurs in order to create a configuration in which

[10] Note that in (7c) when the wh-phrase moves to the matrix clause, the previously unlabeled embedded clause is labeled by C, in accord with Labeling (2c) above, since the wh-phrase has moved out.

[11] It may also be the case that for an SO to be available (at least in some cases), it must have uninterpretable features. For example in Chomsky (2001), if we understand correctly, a goal must have an uninterpretable feature in order to be visible to a probe.

[12] Note that (8a) can be blocked if there is an SO available for external merge (see the discussion of Figure 27

labeling is possible for semantic reasons (8b), or as a last resort mechanism in order to prevent an unlicensed element from being transferred (8c).[13] Note that there are instances in which other constraints may block remerge/movement of an unlicensed SO (see discussion of (14) below).[14]

2.5 Transfer and dephasing

Chomsky (2015) takes the position that the complement of a phase head is transferred.[15] After transfer, this complement essentially becomes frozen, possibly inaccessible to higher operations. Also, when a derivation is completed, the entire structure must be transferred.[16] The transfer operation is summarized in (9).

(9) **Transfer**
  a. The complement of a phase head is transferred.
  b. A derivation is transferred when all operations are completed.

The transfer operation lightens the computational workload by setting aside already constructed SOs.

Chomsky (2015) proposes that movement of a verbal root results in dephasing of the v* structure. The Exceptional Case Marking (ECM) construction with wh-movement in (10) shows why this is necessary.

(10) Who do you expect to win? (Chomsky 2015: 10)

Consider (10) at the point that the matrix verbal root *expect* and v* have merged, as shown in (11a). The embedded subject *who* has raised to the matrix clause and remerged. The phase head v* merges with this structure, and after feature inheritance and feature checking, shared phi-features of the verbal root *expect* and the subject *who* label. If the complement of a phase head must be transferred, in accord with (9), then the structure labeled by shared phi-features will be transferred. However, if this structure is transferred, then *who* will no longer be accessible to the matrix C and the derivation cannot converge. Chomsky's proposed solution to this problem is that the verbal root *expect* raises and remerges with v*, so that v* affixes onto *expect,* shown in (11b). Since the phase head v* is now an affix, it is invisible – invisibility results in dephasing, so that the complement of v* is not transferred. Rather, phasehood is transferred to the verbal root *expect* and the complement of *expect* (in base position) is transferred. When the matrix C is merged, it can then access *who* and the derivation converges successfully.

(11) a. [v* [$_{\text{Phi}}$ who expect ~~who~~ to ~~who~~ win]]
  b. [expect + v* [$_{\text{expect}}$ who ~~expect~~ ~~who~~ to ~~who~~ win]] → Transfer [$_{\text{to}}$ ~~who~~ to ~~who~~ win]

below). (8a) also might not apply in the same way in languages (e.g., Italian) that do not require feature sharing for labeling of certain projections.

[13] This last resort view that movement occurs due to the need to check an unchecked/unvalued feature goes back at least to Chomsky (1995), and also appears in other work such as Chomsky (2000).

[14] An anonymous reviewer points out that the Movement operations in (8a–c) are contrary to the idea in Chomsky (2015: 14) that "Merge applies freely, including IM [Internal Merge]." If we understand correctly, completely free merge would require merging every SO with every other SO in the course of a derivation. Ill-formed merge operations could then be ruled out due to labeling failures. This type of free merge would create a huge computational burden because our model would have to compute an enormous number of unsuccessful derivations in order to arrive at a successful derivation. While free merge may have its merits (such as eliminating the need for movement operations to be motivated by feature checking, etc.), at this point, it is not clear to us how to implement free merge in a computationally efficient manner.

[15] This view of transfer essentially follows that of Chomsky (2001).

[16] This would be the stage in which a numeration (if there is a numeration) is emptied, all operations are complete, and nothing has caused the derivation to crash.

Chomsky (2015) also proposes an analysis of *that*-trace effects that involves dephasing, in which case T inherits phasehood, and the complement of T, rather than the complement of the phase head C, is transferred. For more on *that*-trace effects, see the discussions in the following sections. The dephasing operation is summarized as follows:

(12) **Dephasing**
a. A verbal root remerges with the local v* (when present) and phasehood is transferred to the verbal root.
b. Deletion of C transfers phasehood to T.

2.6 Wh-movement

We next turn to how the timing of labeling and transfer relates to constructions that involve extraction of a wh-phrase out of an embedded clause and the well-known *that*-trace effect, as in (13a–c).

(13) a. What do you think (that) John read?
b. *Who do you think that read the book? (Chomsky 2015: 10)
c. Who do you think read the book? (Chomsky 2015: 10)

First, we discuss how Chomsky (2015) accounts for examples (13a–c),[17] and then we explain how we account for these constructions.

Note that we assume that a wh-phrase is headed by a Q morpheme that checks an uninterpretable Q feature on an interrogative C, following work by Cable (2010) and Hagstrom (1998), among others.[18]

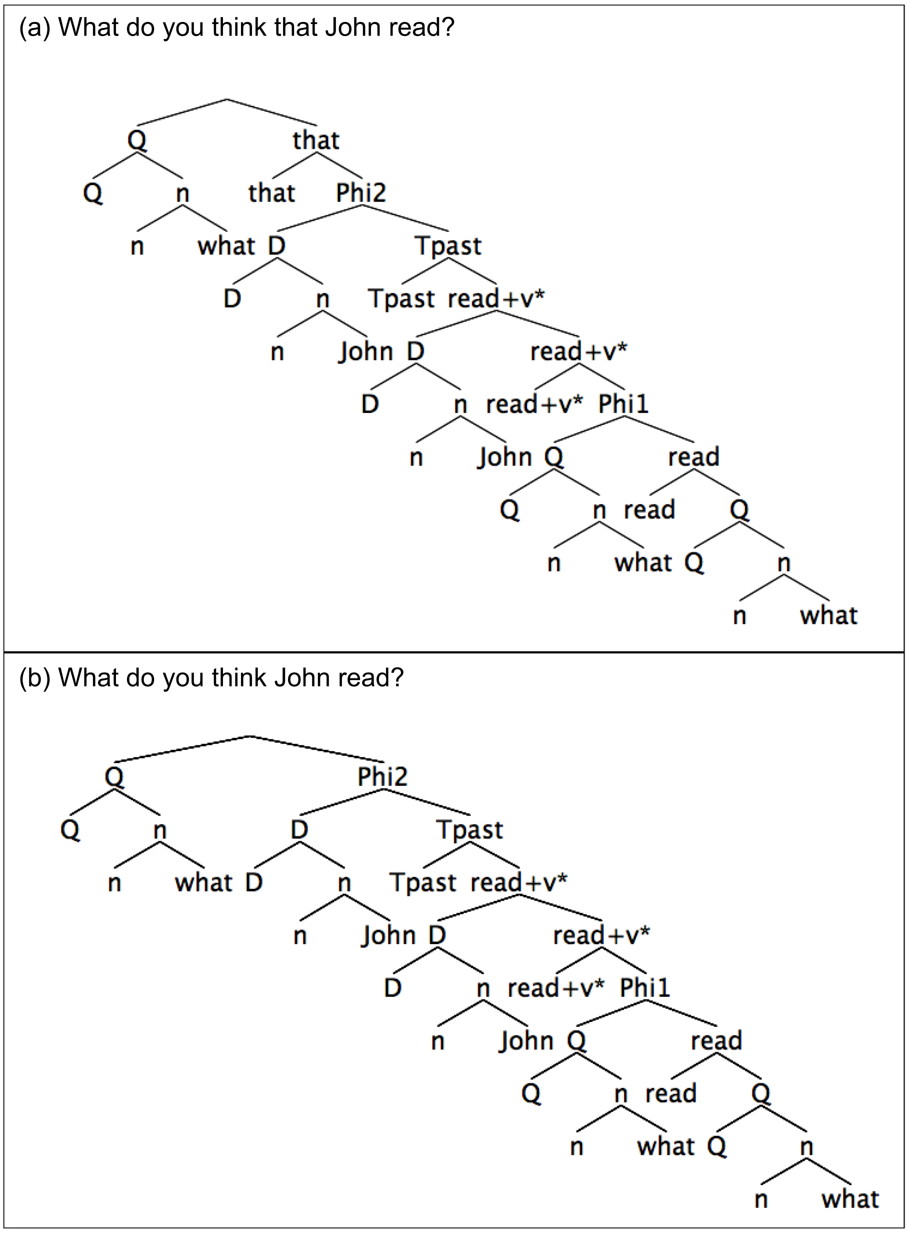
The relevant portions of (13a) are shown in Figure 8. In (13a), after merge of C *that*, the features of *that* are inherited by Tpast, followed by feature checking and remerge of *what*. Then labeling (see (2)) applies. There are no labeling failures within the embedded clause; the object *what* and its SO complement *read a book* are labeled via shared phi-features

[17] We thank two anonymous reviewers for helping clarify how these constructions are accounted for in POP.

[18] With respect to wh-questions, Chomsky (2015: 13) writes that "Agreement holds for a pair of features <valued, unvalued>. The Q feature of C is valued, so the corresponding feature of a wh-phrase must be unvalued, its interpretation as relative, interrogative, exclamative determined by structural position." While we follow that view that an interrogative C in a wh-construction must agree with a wh-phrase, we do not follow the view that the wh-phrase has an unvalued Q feature.

We follow the view of Hagstrom (1998) and Cable (2007) that a wh-phrase has an associated Q-morpheme. One possibility would be that a Q morpheme has an unvalued but interpretable Q-feature that is checked by an interrogative C which has a valued but uninterpretable Q feature (cf. Pesetsky & Torrego 2001; 2007; Cable 2010) for analyses that separate feature interpretability from feature valuation. While this approach seems viable, our model only uses interpretable and uninterpretable features, due to our desire to keep the model as simple as possible; allowing features to differ with respect to interpretability is simpler than allowing features to differ with respect to both interpretability and valuation. If features can only be interpretable or uninterpretable, then it seems unreasonable for a Q morpheme, which takes a wh-phrase complement, to have an uninterpretable Q-feature, since by nature, a Q-morpheme should have a Q property. Therefore, we take the approach that Q has an interpretable Q-feature and interrogative C, our C_Q, has an uninterpretable Q-feature. A Q-phrase (i.e., what is traditionally referred to as a wh-phrase) clearly must undergo some type of agreement relation with an interrogative C to be licensed. If a Q-phrase already has an interpretable Q-feature, then after the Q-phrase obtains case and a theta-role, the question arises of what on the Q-phrase, if anything, needs to be licensed. Rather than create a special rule forcing a wh-phrase to remain unlicensed until it agrees with C_Q, it is simpler to give the wh-phrase an uninterpretable feature that must be checked via agreement with an interrogative C_Q. For this task, we use a uScp feature (responsible for scope), which is checked via agreement with an interpretable Scope feature, iScp, on C_Q. Agreement between C_Q and a wh-phrase checks the uQ on C_Q and the uScp on the wh-phrase. The Q feature gives a clause an interrogative interpretation and the Scp feature is responsible for giving a wh-phrase scope. When C has licensed Q and Scp features, then the C projection is interpreted as a wh-construction. This uScp feature thus plays an important role in keeping a wh-phrase visible to movement operations.

Note that if C_Q comes with a uQ feature, the question arises of what happens in an English yes/no construction, in which there is no wh-phrase available with a Q feature. In this case, a reasonable assumption is that C_Q has a Q feature that is base generated in the interrogative C. See Hagstrom (1998); Cable (2007); Ginsburg (2009), among others.



(a) What do you think that John read?

(b) What do you think John read?

Figure 8: Chomsky's (2015) account of "What do you think (that) John read?"

within the v* phase. The root node in Figure 8a is an unlabeled {XP, YP} structure, but it will be successfully labeled after *what* moves to remerge with the matrix C_Q. In (13a) with a null C, Chomsky takes the position that there is deletion of *that* resulting in dephasing (see (12)). Specifically, phasehood is inherited by T and the complement of T is transferred. In this case, after C is merged, there is feature inheritance, feature checking and remerge of *what* with C. Then C is deleted, resulting in the structure in Figure 8b after labeling has occurred. Again, there are no labeling failures.

Note that in Figure 8a–b, it is necessary for *what* to remerge with C before transfer occurs. In Figure 8a, if *what* does not remerge with C before transfer of the complement of *that*, then *what* will be transferred. In Figure 8b, if *what* does not remerge with C[19] before

[19] If we understand correctly, Chomsky (2015) assumes that *what* remerges with C before C is deleted. Note that if an SO can move to the edge of a phase to avoid transfer, in accord with the Movement operation (8c), then another possibility is that after deletion of C, *what* remerges with the projection of T, which is now the phase head. This is the approach that we will take, but it does not appear to be the approach taken by Chomsky. See the discussion of Figure 13 below.

the complement of T (the v* projection) is transferred, then *what* will be transferred with the v* projection.

The *that*-trace effect in the ill-formed (13b) results from a labeling failure. After *that* is merged, there is feature inheritance, feature checking and remerge of *who* with the *that* projection. But then when the labeling algorithm applies, there is a labeling failure at the point shown in Figure 9 because not all instances of *who* are contained within the phase that is headed by *that*.

In order to account for (13b), as shown in Figure 9, it is necessary to assume that labeling does not occur until after *who* remerges with *that*; i.e., labeling occurs at the C phase level. If labeling were to occur immediately after C *that* is merged, but before *who* remerges in a higher position, then the relevant {XP, YP} structure, formed from *who* and the embedded T clause "Tpast read the book", would be successfully labeled via shared phi-features. This is because at the point at which *that* is merged, all instances of *who* are in the relevant domain, so that *who* should be visible to labeling. Then, after labeling, *who* would move out (assuming that transfer has not yet occurred), thus predicting that this example should be well-formed.

While Chomsky (2015) successfully accounts for the contrast in (13a–b), the situation regarding the well-formed (13c) is somewhat murky. In (13c) deletion of C results in dephasing, thus avoiding a labeling failure involving the subject wh-phrase. There are at least two possibilities for how this works.

The relevant portions of one possible derivation are shown in Figure 10. C is merged, followed by remerge of *who*, shown in Figure 10a. Then C deletes, phasehood is transferred to Tpast and the complement of Tpast is transferred, shown in Figure 10b. Note that after C deletion, a "strange" configuration arises with two instances of *who* at the edge of the embedded clause. In this case, the instance of *who* that is merged with "Tpast read the book" is not the highest instance of *who*. When this structure is eventually labeled, presumably at the next phase level, the question arises as to whether or not this lower *who* is visible to the labeling algorithm. If the lower *who* is not visible, then there should be a labeling failure.

There is also another possible way of accounting for (13c). Simply assume that C deletes before *who* can remerge with C. After C is merged, there is feature inheritance, followed by deletion of C, shown in Figure 11a–b. If labeling then occurs, the {XP, YP} structure consisting of *who* and "read the book" should be successfully labeled.

The facts regarding (13a–c) are summarized in Table 1.

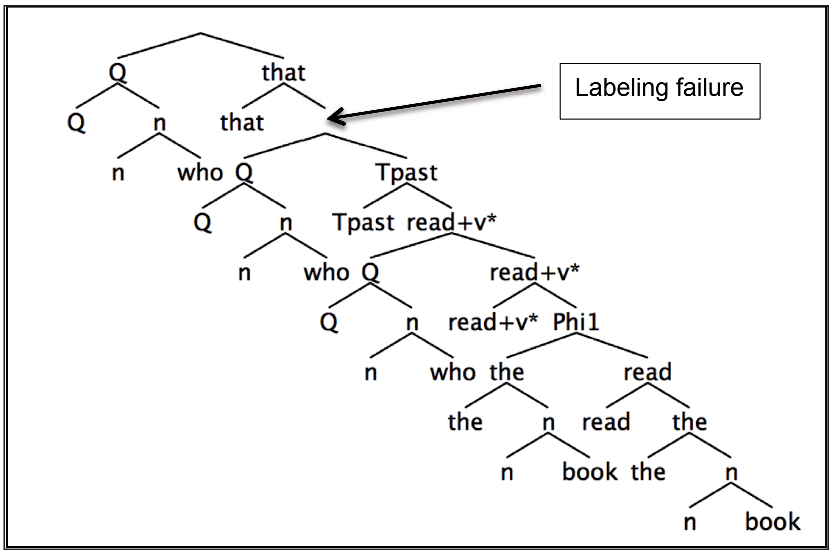


Figure 9: Chomsky's (2015) account of "*Who do you think that read the book?"

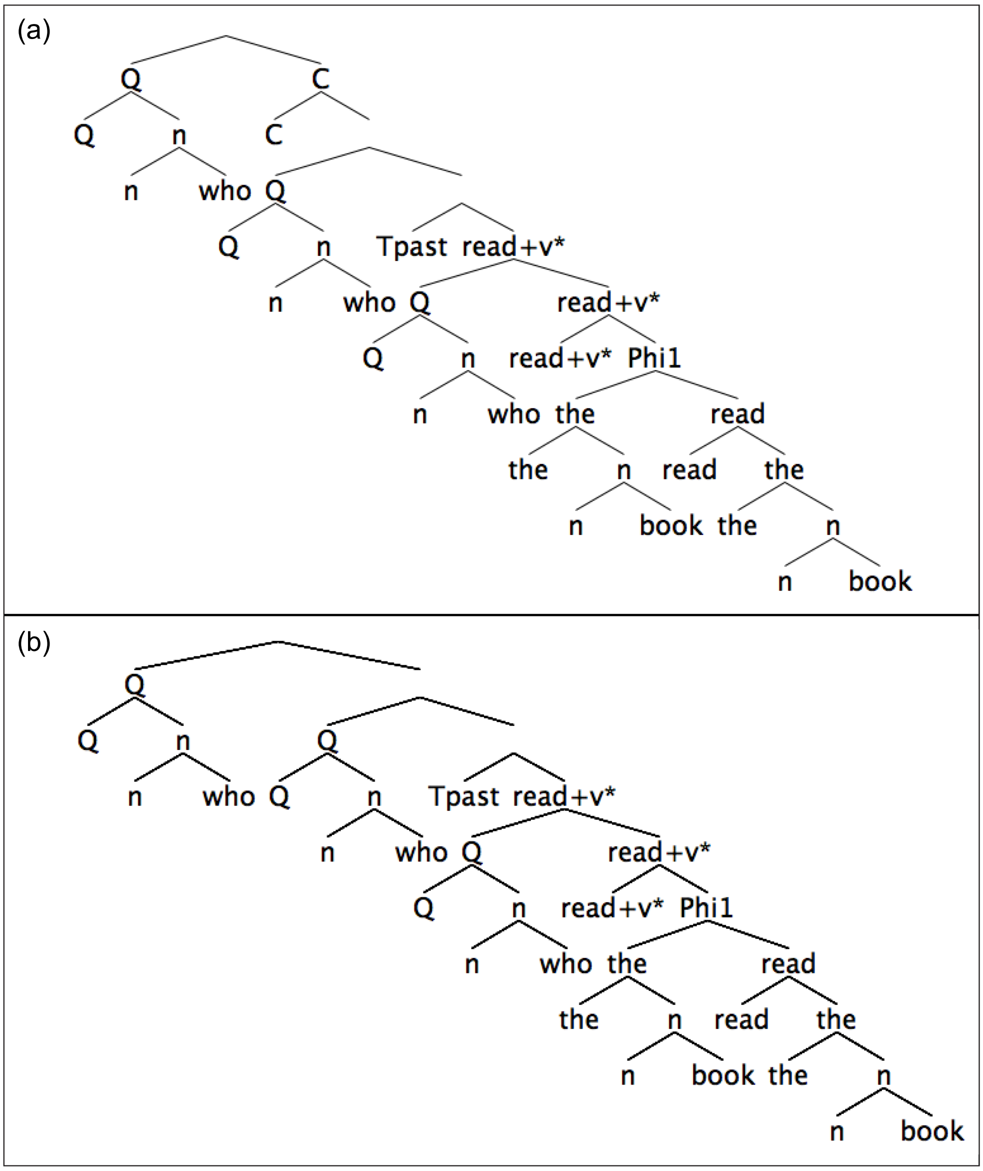


Figure 10: Chomsky's (2015) account of "Who do you think read the book?"

Example	Wh-movement	Explanation
(13a) What do you think (that) John read?	*what* must remerge with C, regardless of whether C is pronounced. See Figure 8.	If *what* does not remerge with C, *what* will be transferred.
(13b) *Who do you think that read the book?	*who* remerges with C. See Figure 9.	There is a labeling failure; *who* is not visible in the C phase, so shared phi-features cannot label {who, Tpast read the book}.
(13c) Who do you think read the book?	Possibility 1: *who* remerges with C before C deletes. See Figure 10.	The result is a configuration in which there are two instances of *who* in the clause edge. This might result in a labeling failure.
	Possibility 2: *who* does not remerge with C before C deletes. See Figure 11.	If *who* does not remerge with C, there is no labeling failure.

Table 1: Summary of *that*-trace effects in (13a–c).

To summarize, (13a) can be accounted for because there are no labeling failures regardless of whether or not C is deleted. Crucially, *what* must remerge with C. (13b) is accounted for as resulting from a labeling failure, but only if labeling occurs after *who* has moved out of the C phase. (13c) can be accounted for by assuming that deletion of C avoids a labeling failure. The latter case is complex in that it either a) results in a complex structure with two instances of *who* at the edge of the embedded clause (Figure 10), or b) it results from deletion of C before *who* remerges with C (Figure 11). The latter option in which *who* does not remerge with the C projection seems to be optimal, since then there are no problems for the labeling algorithm. However, the latter option raises the following question: if a wh-phrasal object must remerge with C (*that*) before C deletes, then why can't a wh-phrasal subject also remerge with C? If the derivation in Figure 11 is correct, then there appears to be a wh-subject vs. wh-object remerge asymmetry that requires an explanation.

In this paper, we propose an analysis of (13a–c) that accounts for the relevant data, but that avoids these complications inherent in Chomsky (2015). We develop an analysis in which there is no requirement that labeling be delayed until after *who* moves out of the embedded clause as in Figure 9, and the complexities of Figures 10–11 are avoided.

The derivation of (13a), as proposed by Chomsky is unproblematic. Figure 12a shows the structure after v* has been merged. At the v* phase level, features of v* are inherited by

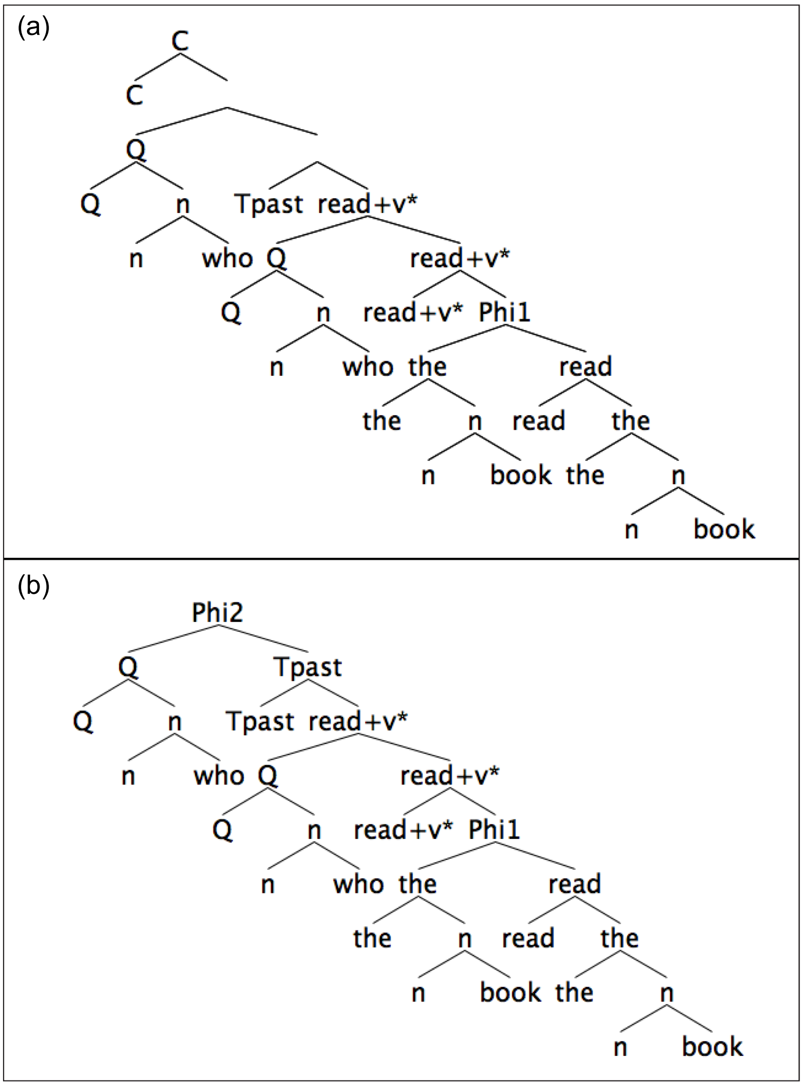


Figure 11: Chomsky's (2015) account of "Who do you think read the book?"

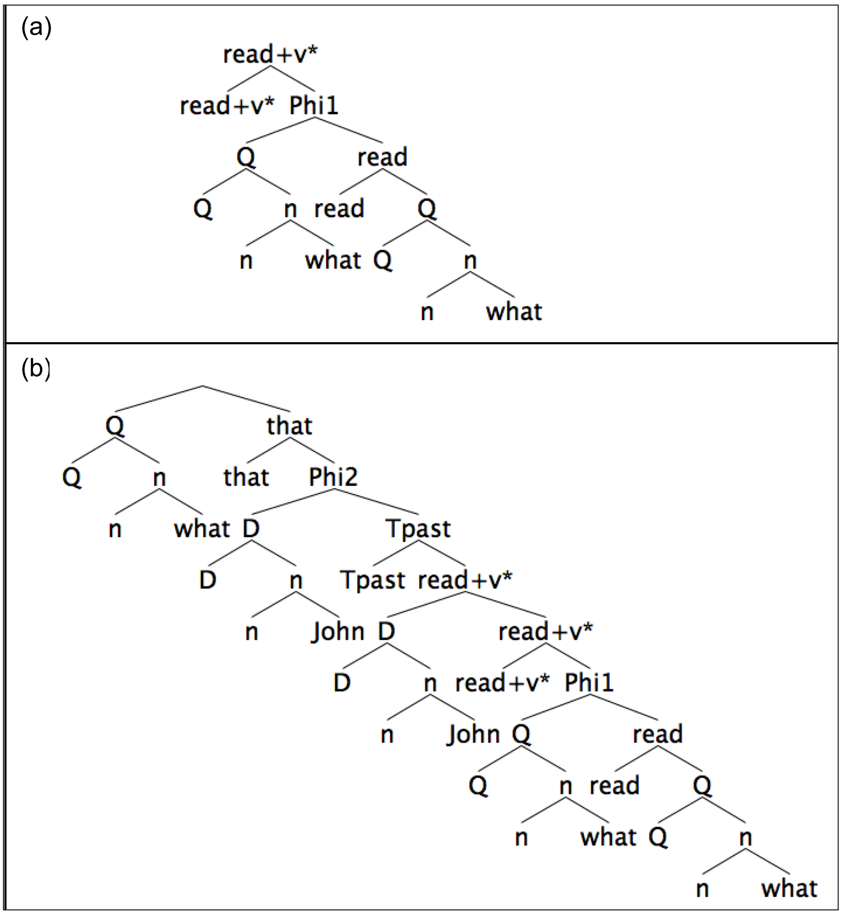


Figure 12: Extraction of an object wh-phrase from "What do you think that John read?"

the verbal root *read*. Feature checking between *read* and *what* occurs, and *read* remerges with v*, resulting in dephasing of v*. The structure {what, read what} is labeled by shared phi-features. Normally, at this point, the complement of v* would be transferred. However, because v* is dephased (see (12)), transfer of the complement of v* does not occur. Rather, the complement of the verbal root *read* is transferred, and thus the lower instance of *what* is transferred. Note that the higher instance of *what* is outside of the transferred domain, so it remains accessible to higher operations. After merge of C *that*, shown in Figure 12b, features of *that* are inherited by Tpast. Then labeling occurs. At this point, the Movement operation (8c) requires an unlicensed SO that is about to be transferred to remerge with the root node. As a result, *what* remerges with the root node, thereby escaping transfer. When transfer occurs, the complement of *that* is transferred. There are no labeling failures (the lower copy of *what* is in a structure that has already been labeled), and the higher copy of *what* remains accessible to further operations, so that it can eventually remerge with the matrix C_Q. The derivation successfully converges.

Consider what happens when there is deletion of C *that* in (13a). After merge of v*, there is feature inheritance and labeling. Just as in Figure 12 above, the complement of v* is labeled by the shared phi-features of *what* and *read*. After merge of C, there is feature inheritance and labeling, shown in Figure 13a. Then, there is deletion of C, shown in Figure 13b. After deletion of C, phasehood is transferred to Tpast, and the complement of Tpast will be transferred. In order to avoid transfer of the unlicensed wh-phrase, the Movement operation (8c) requires *what* to remerge with the root node, as shown in Figure 13c. Then the complement of Tpast is transferred. Crucially, there is no labeling failure, and the higher

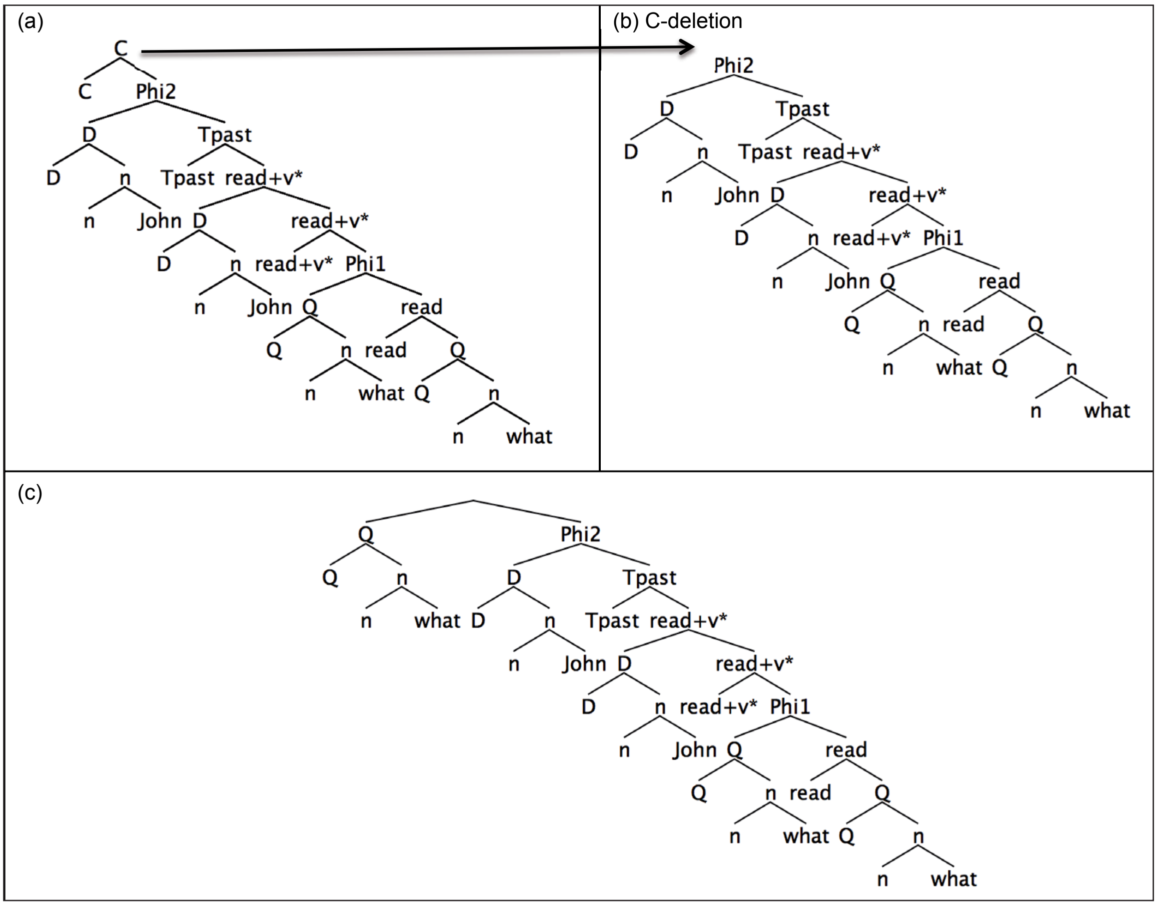


Figure 13: Extraction of an object wh-phrase from "What do you think John read?"

instance of *what* remains accessible to further operations so that it can eventually agree and remerge with the matrix C. Note that C is deleted as soon as its features are inherited.[20]

We next turn to *that*-trace effects. Consider how the derivation of (13b) works in POP, as discussed above in Figure 9. In the embedded clause, *that* is merged, followed by remerge of *who* to produce the structure in Figure 14a. After feature inheritance and feature checking, labeling occurs. But in this case, there is a labeling failure – the SO formed from *who* and the T structure is unable to label via shared phi-features because *who* is no longer visible – there is a higher copy of *who* merged with the *that* projection. There is also another possibility. Assume that *who* does not remerge with *that*. Instead, shown in Figure 14b, there is no labeling failure. However, when the complement of *that* is transferred, *who* becomes inaccessible to higher operations, since it is contained within this transferred complement. This derivation also crashes. Thus, POP seems to be compatible with either version, that shown in Figure 14a (in which *who* remerges with *that*) and Figure 14b (in which *who* remains in subject position).

Our model implements this latter approach, shown in Figure 14b, in which *who* does not remerge with the root, which we propose is required due to the following Remerge condition, given in (14). The Remerge condition prevents an SO from remerging with another SO with which it shares features. Assume that an SO Y is contained within the SO X.

[20] We thank an anonymous reviewer for pointing out that this type of example can be accounted for with immediate deletion of C. It is important to note that Chomsky (2015) appears to require *what* to remerge with C before C deletes (see Figure 8 above).

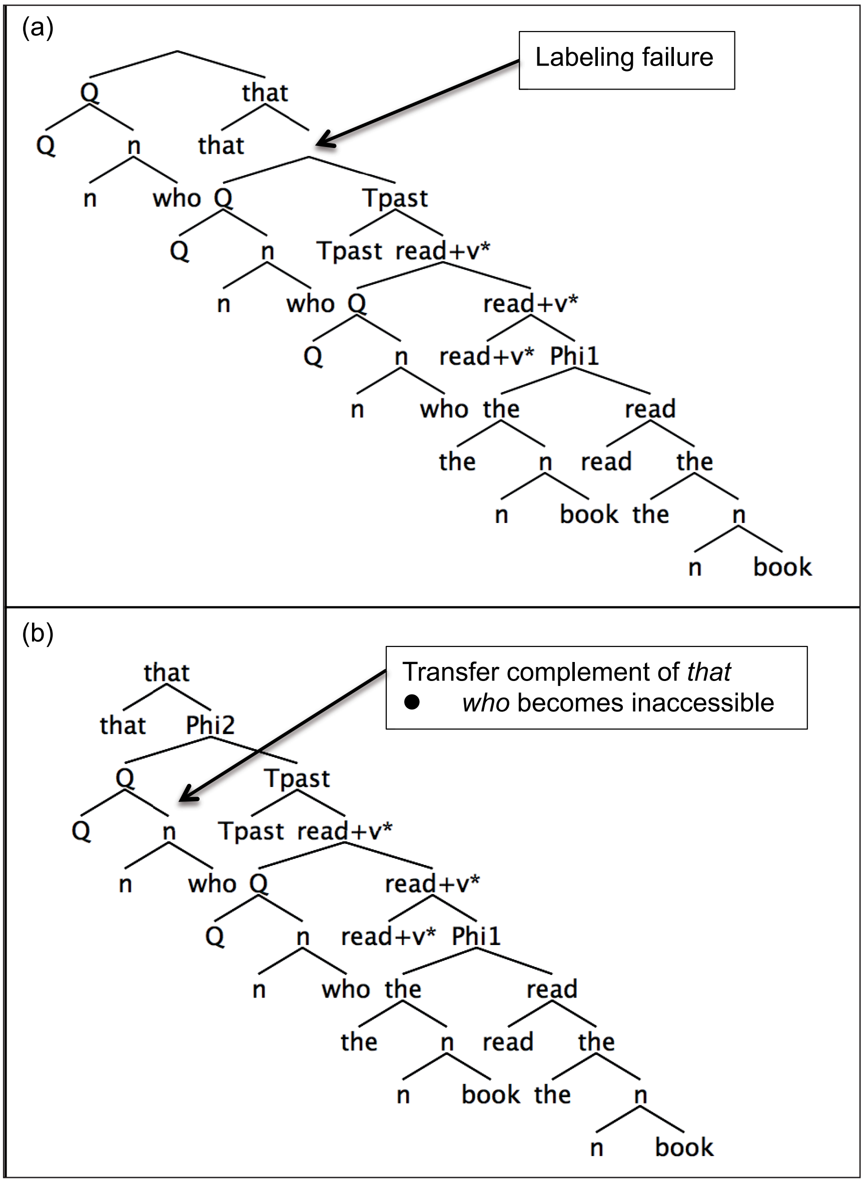


Figure 14: Crash timing in "*Who do you think that read the book?"

If these two SOs already share features, then remerge of one SO with the other will not result in a new feature sharing relation, and thus remerge is blocked.

(14) **Remerge condition**
Remerge of a Syntactic Object X with a Syntactic Object Y cannot occur if X and Y share features.

The Remerge condition in (14) accounts for the *that*-trace effect as follows; a wh-phrasal object can remerge with C, but a local wh-phrasal subject cannot. In Figure 15a, uPhi-features from C are inherited by Tpast, and the uPhi are checked via agreement with the subject *who*.[21] When the uPhi on C are checked, C and the subject share phi-features. As a result, the subject cannot remerge with C, or there will be a violation of (14). If C is not deleted, then *who* is transferred together with the complement of C, resulting in ill-formedness (the *that*-trace effect). In the case of extraction of an object wh-phrase, after

[21] Note that we assume that feature inheritance occurs as soon as C is merged.

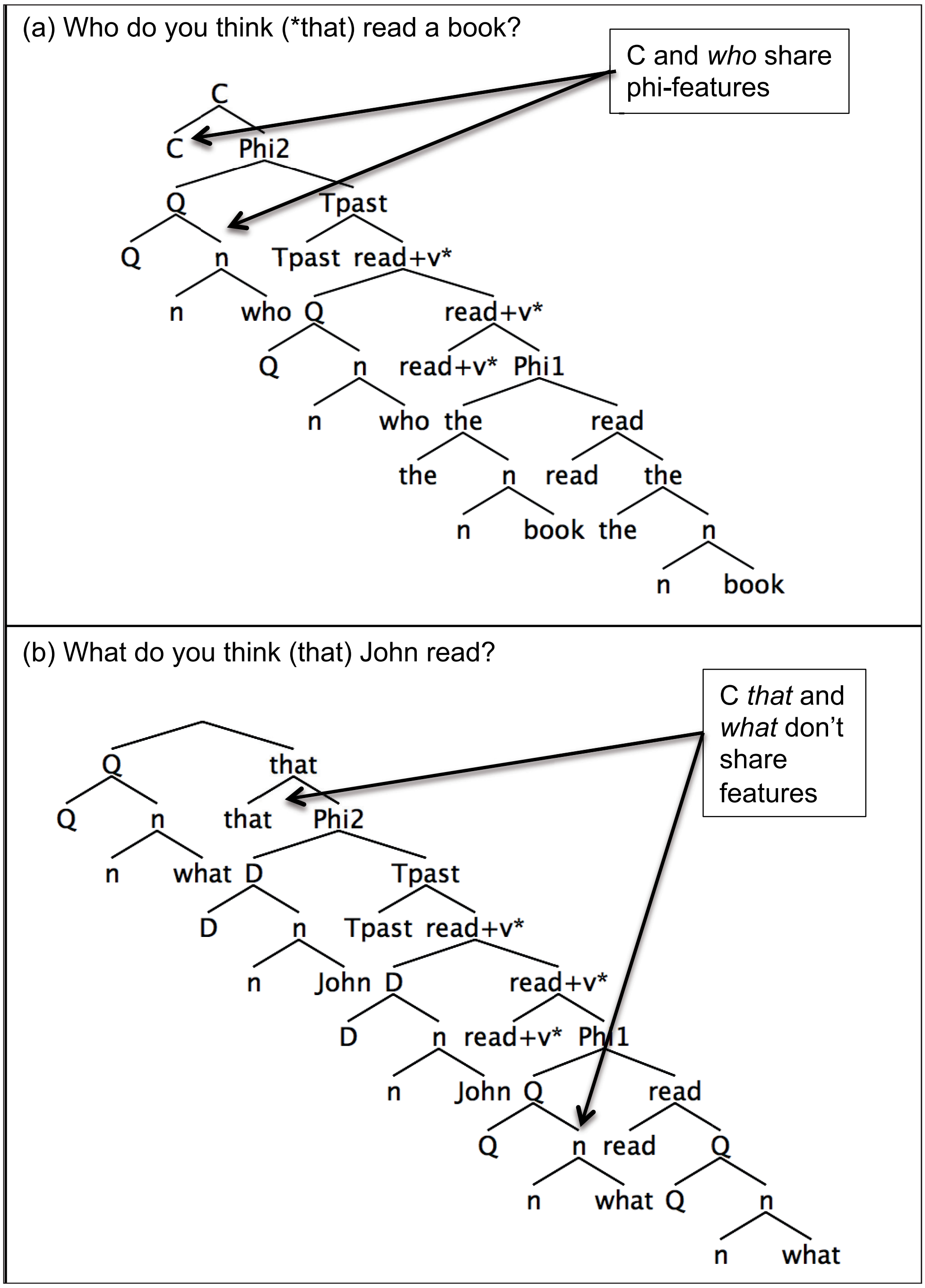


Figure 15: Remerge constraint and *that*-trace effects.

C is merged and features are inherited from C, the wh-phrase is located within the verbal projection, so it does not share features with C (Figure 15b). Therefore, *what* is able to successfully remerge with C, without violating (14), bringing *what* outside of the transfer domain, regardless of whether or not C *that* deletes. This analysis thus accounts for the *that*-trace effect as resulting from an inability for a subject wh-phrase (but not an object wh-phrase) to remerge with a local C.

Deletion of C saves (13c). Consider how this works. The C head is merged, followed by feature inheritance and labeling, shown in Figure 16a. Then C is deleted, resulting in Figure 16b when labeling occurs. Since all instances of *who* are in the relevant labeling

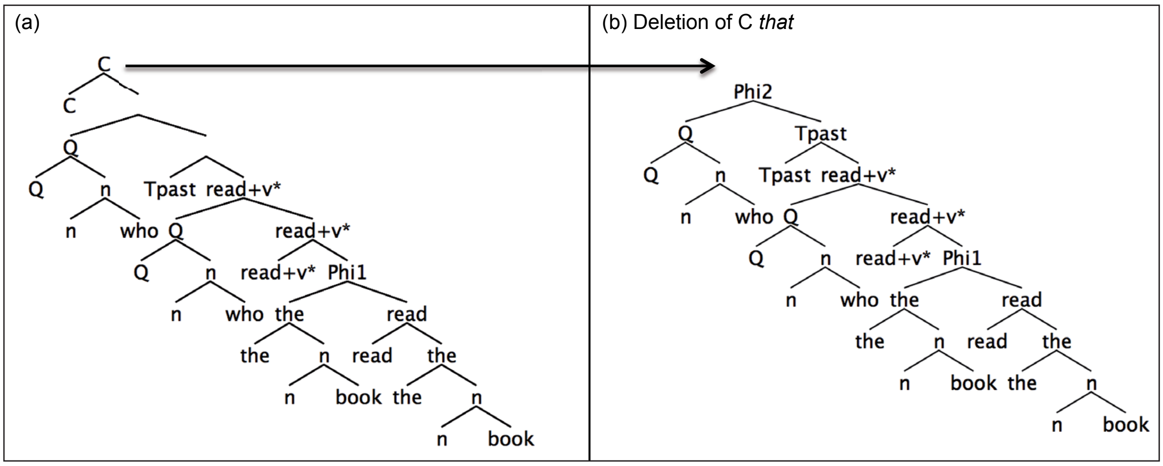


Figure 16: Dephasing of embedded clause in "Who do you think read the book?"

domain, labeling by shared phi-features proceeds, followed by transfer of the complement of Tpast. Crucially, *who* remains accessible to higher operations.

2.7 The remerge condition, labeling, and cyclicity

The remerge condition has two other primary benefits. First, it enables simplification of the labeling process, so that there is never any need to delay labeling until the phase level. Second, it provides an explanation (or at least a partial explanation) for why derivations must be cyclic.

In Chomsky (2015), if we understand correctly, labeling occurs at the phase-level, after feature inheritance and feature checking. In Figure 17, originally shown as Figure 9 above, after *that* is merged, uPhi from *that* are inherited by Tpast. Crucially, labeling does not occur until after *who* remerges with the root, which results in a labeling failure, because *who* is not visible to the labeling algorithm in the T projection, which requires shared phi-features for labeling.

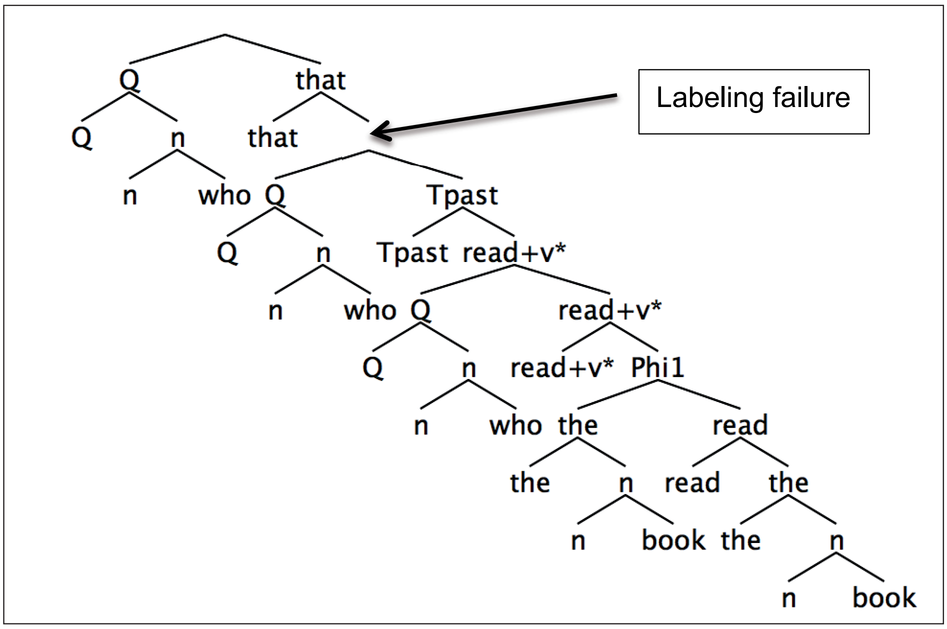


Figure 17: Crash timing in "*Who do you think that read the book?"

If our analysis is correct, there is no reason to delay labeling. As a result of the Remerge condition (14), the wh-phrase *who* cannot remerge with the projection labeled by *that* in the first place. As soon as inherited features are checked and an intermediary projection becomes labelable, the labeling algorithm should apply. Therefore, in (13b), as soon as uPhi are inherited by Tpast, and the uPhi are checked, labeling should take place. We propose the following:

(15) **Immediate labeling**
Labeling occurs as soon as a configuration is labelable.

For example, as soon as a head that is strong enough to label (v*, C, etc) merges with a phrasal element, the head labels. As soon as an {XP, YP} structure has identical shared features (or a single shared feature) that are capable of labeling, then the shared feature(s) label, as shown in Figure 18. Labeling is not delayed until after a phase is complete. The situation in Figure 17, in which a wh-phrase moves before labeling occurs in the embedded C projection, does not arise.

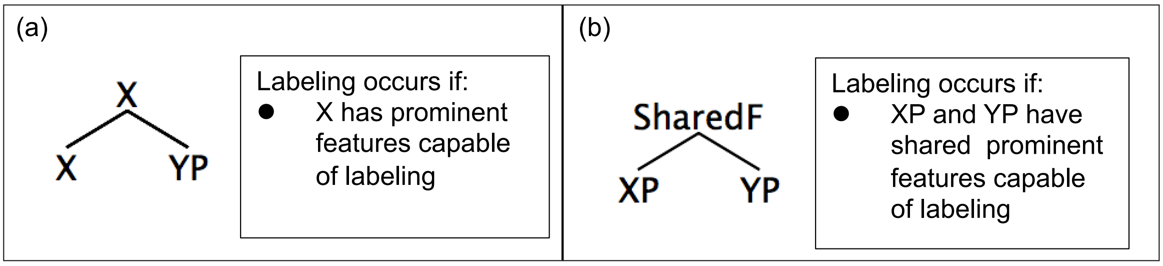

(a)
X
X
YP
Labeling occurs if:
● X has prominent features capable of labeling
(b)
SharedF
XP
YP
Labeling occurs if:
● XP and YP have shared prominent features capable of labeling

Figure 18: Immediate labeling.

Consider how Immediate labeling (15) works in our model to account for (13b). As shown in Figure 19a, when C *that* is merged, phi-features are inherited and checked, followed by labeling, via shared phi-features, of the projection containing the subject *who* and the projection of Tpast, and the strengthened Tpast also labels. The situation in Figure 19b, in which *who* remerges with *that*, does not arise because *who* cannot remerge with *that* due to the Remerge condition (14). As a result, *who* is transferred together with the complement of C *that*, so that *who* becomes inaccessible and the derivation crashes, as desired. In this manner, we can explain this example without needing to delay labeling until the phase level, contra Chomsky (2015).

Also note that Immediate labeling (15) does not create problems for other examples. Consider (13a), repeated below.

(16) What do you think (that) John read?

In (16), shown in Figure 20a, when *that* is merged, the uPhi-features of *that* are inherited by Tpast, and uPhi on *that* are checked via agreement with *John*, so that the unified uPhi on Tpast are also checked. The result is that the subject and Tpast share phi-features, so the shared phi-features, as well as the strengthened Tpast, label. Then, shown in Figure 20b, *what* remerges with C *that*. Delaying labeling until after *what* remerges with the root *that* makes no difference for this derivation, and thus seems unnecessary.

The Remerge condition in (14) also has a very important function. It requires derivations to be cyclic, an observation that we thank an anonymous reviewer for pointing out.[22]

[22] The constraint in (14) was originally proposed to account for why a subject wh-phrase cannot remerge with C (see the discussion of Figure 15). However, an anonymous reviewer points out that counter-cyclic derivations violate (14), and thus (14) requires derivations to be cyclic.

Another anonymous reviewer suggests that the Remerge condition is not necessary to force cyclicity,

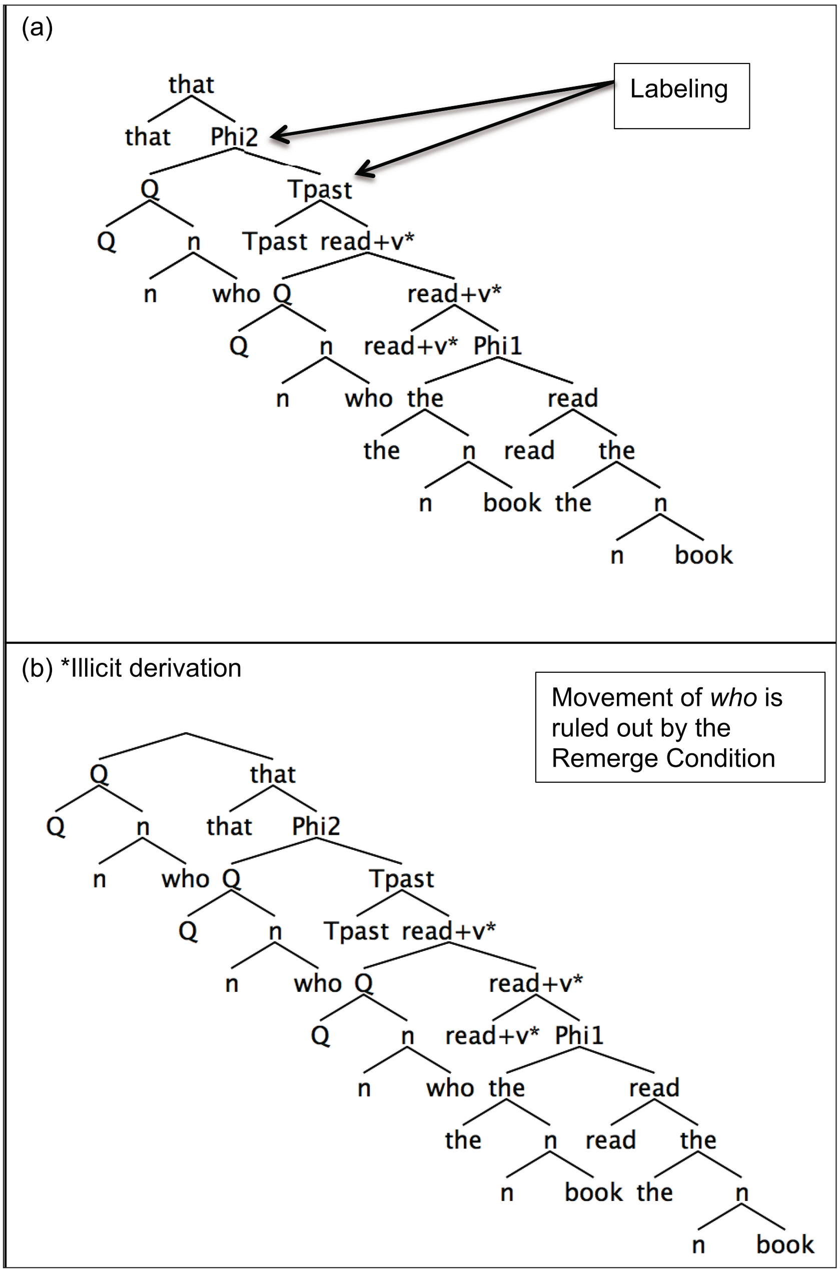


Figure 19: Derivation of "*Who do you think that read the book?"

Consider how this works. In Figure 21a, assume the counter-cyclic version of POP. When v* is merged, the uPhi of v* are inherited by *read*, and then there is an agree relation, whereby the uPhi on v* are checked by the iPhi on *a book*, and the shared uPhi on *read* are simultaneously checked. The result is that *read* and *a book* now share phi-features. Thus, in accord with (14), *a book* can't remerge with *read*. However, if a derivation is cyclic, shown in Figure 21b, then *a book* remerges with *read* before v* is merged. At this point, *a book* and *read* don't share any features, so there is no violation of (14). This is followed by merge of v*, at which point feature inheritance, feature checking, and ultimately feature sharing, can occur.

since cyclicity has other motivations, namely the need to avoid "a substitution operation that is ternary (Chomsky 2015: 13)". We leave open the possibility that factors other than the Remerge condition also play a role in forcing cyclicity. In other words, it may be that the Remerge condition is not the sole factor that forces cyclicity.

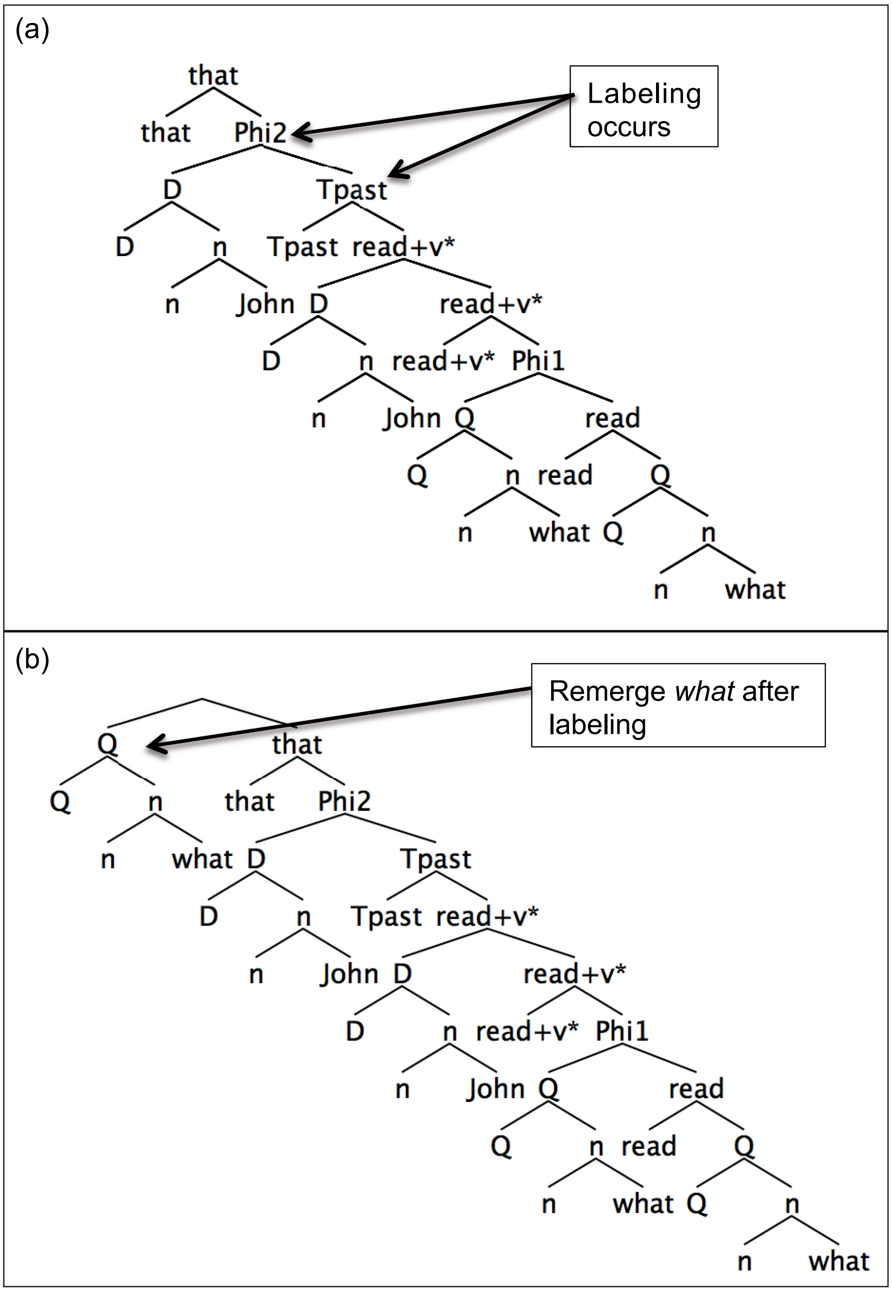


Figure 20: Immediate labeling in “What do you think that John read?”

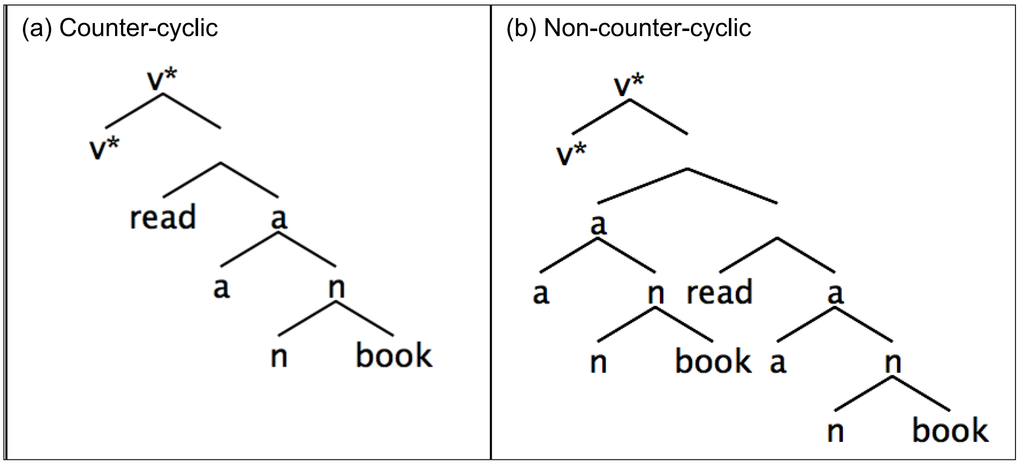


Figure 21: Remerge condition and cyclicity.

The Remerge condition also provides an explanation (or at least a partial explanation) for dephasing of v*.[23] Consider (10), repeated below as (17). Chomsky proposes that dephasing of v* enables *who* to remain accessible to higher operations; without dephasing, *who* would be transferred after merge of v*.

(17) Who do you expect to win? (Chomsky 2015: 10)

The movement operation (8c) requires an SO that is about to be transferred to remerge with the root node if possible. If there were no dephasing of v*, then the question arises of why *who* couldn't just remerge with v* and escape transfer. This is blocked by the Remerge condition, because v* and *who* already share features, so *who* cannot remerge with C. The result is that dephasing of v* is necessary.

2. 8 Labeling of verbs that take clausal complements and non-finite T

In constructions with a matrix verb that takes an embedded clausal complement, issues regarding how labeling occurs arise. Consider (13c), repeated as (18a).

(18) a. Who do you think read the book?
b. [$_T$ who do you v* [$_\varepsilon$ think [$_\delta$ C [$_\alpha$ t T β]]]] (Chomsky 2015: 13)

The matrix verbal root *think* is too weak to label. A transitive verbal root is labeled by sharing phi-features with a raised object. Chomsky points out that the object of *think* should raise and remerge with the projection labeled ε in (18b), in the same way that an object raises and remerges with a verbal root in a simple transitive clause, such as in Figure 1 above. Chomsky (2015: 13) writes:

> "Prior to raising of *who* to matrix SPEC, the object of *think*, δ, should raise to SPEC of ε, with the root *think* then raising to v*, the ordinary raising to object analysis. But questions arise about how ε should be labeled, since the raised object in this case lacks the relevant features; there is no agreement."

Chomsky points out that if δ remains in situ, then these problems don't arise. However, if δ were to remain in situ, then the question arises of how the projection of *think* is labeled, considering that a lexical root is too weak to label.

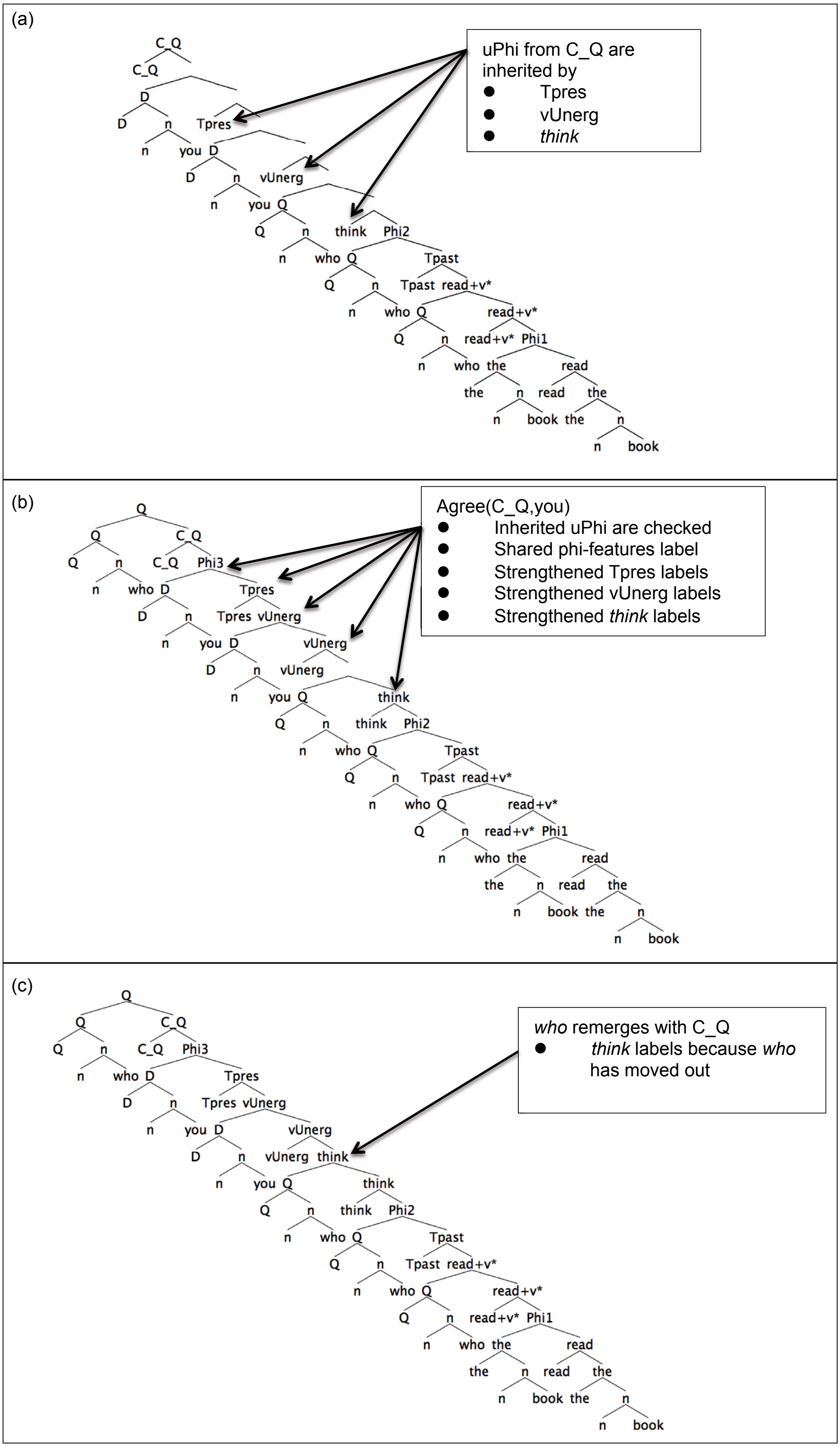
First, we assume that a verbal root that takes a sentential complement does not merge with a typical v*. Assuming that v* comes with uPhi and also assigns case, then v* cannot be present in the matrix clause in (18a), since there is no object for v* to agree with and assign case to. But the verbal projection is accounted for if it contains an unergative v, which we refer to as vUnerg. This a verb that requires a subject, but that lacks uPhi and does not assign case.[24]

This vUnerg does not have uPhi features that are checked, and thus it is different from v*. It does not seem to be the type of element that is capable of labeling on its own. Even if it were capable of labeling, it isn't clear how the root verb that it takes as a complement, *think* in (18), would end up labeled, since vUnerg does not pass any uPhi to its complement. Thus, the question arises of how labeling occurs in a configuration with a verb that takes a clausal complement.

We rely on feature inheritance, originally presented in (3) above. To account for feature inheritance in Figure 1, it was necessary for a phase head to pass its uPhi only to the head

[23] We thank an anonymous reviewer for pointing this out.

[24] Our model makes use of a v* and a vUnerg to account for the derivations in this paper. There also need to be other types of little vs. For example, there must be a little v, occurring in passive constructions, that takes an object but has no subject position. Also, to account for expletive constructions with *be*, along the lines of Sobin (2014), there may be a variety of other little vs.



uPhi from C_Q are inherited by

- Tpres
- vUnerg
- *think*

Agree(C_Q,you)

- Inherited uPhi are checked
- Shared phi-features label
- Strengthened Tpres labels
- Strengthened vUnerg labels
- Strengthened *think* labels

who remerges with C_Q

- *think* labels because *who* has moved out

Figure 22: Super-inheritance in "Who do you think read the book?"

of its complement. But assume that feature inheritance can travel farther; i.e. there can be "super-inheritance" (a term proposed by an anonymous reviewer), whereby uPhi from a phase head travel down a tree until they cannot travel any farther. According to (19), a phase head passes its uninterpretable features (uFs) to its complement X, if the complement X is unlicensed. Furthermore, X will pass these uFs down to its complement Y, if Y is unlicensed, and so on. An unlicensed complement is unlabeled.

(19) **Feature inheritance (revised)**
Uninterpretable features are passed to an unlicensed complement.

Consider how feature inheritance works in the derivation of (18). When the matrix C_Q is merged, note that the projections of vUnerg and *think* are unlabeled, shown in Figure 22a. C_Q passes its uPhi features down the tree, from Tpres to vUnerg and then to *think*. Further feature inheritance is blocked by the labeled embedded clause.[25] Then there is feature checking. All of the unified uPhi are checked via agree between the uPhi on C_Q and the matching phi-features on *you*. Then the projections with these valued phi-features are labeled. Note that there is still one projection, consisting of an {XP, YP} structure, with *who* merged with the projection of *think*, that remains unlabeled in Figure 22b. This projection, however, is labeled after *who* remerges with C_Q in order to be licensed, shown in Figure 22c.

This analysis also extends naturally to non-finite T. Chomsky (2015) takes the position that in English, T is labeled via shared phi-features. The question then arises of what happens to non-finite T *to*. Consider (20) with a nonfinite T in the embedded clause. Note that there is no local C from which the non-finite T could inherit uFs.

(20) They expected John to win. (Chomsky 2015: 10)

We can account for the labeling of non-finite T, which we refer to as toT, as resulting from feature inheritance, as in Figure 22 above. When v* is merged, its uPhi are inherited by the verbal root *expect*, toT, and vUnerg, shown in Figure 23a. Then after the uPhi on v* agree with the subject *John*, the unified uPhi are checked, so that labeling occurs, whereby the relevant projections are labeled by shared phi-features or by strengthened heads, resulting in Figure 23b.[26]

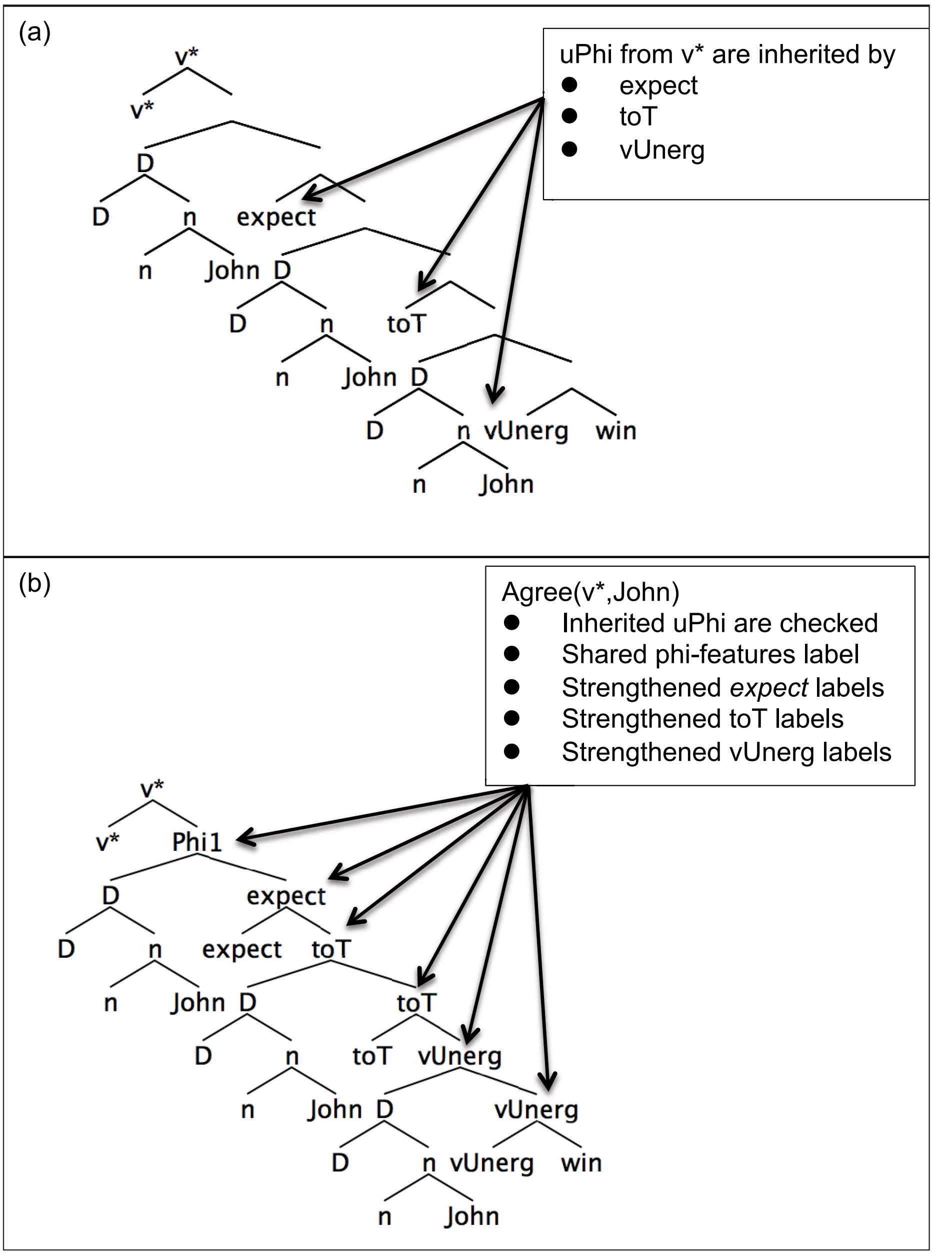
[25] Note that it is necessary to assume that feature inheritance spreads to all successively lower projections, even if a goal is encountered in one branch of the projection. In Figure 22a, uPhi from C_Q are passed all the way down to the projection with the root *think*. However, the goal *you* is higher in the tree than the projections with the roots vUnerg and *think*. Thus, uPhi are passed down the main spine of the tree, and if a goal is encountered on a branching spine of the tree, this does not block uPhi from traveling further down the main spine. In Figure 22, feature inheritance stops fully when the labeled embedded clause is reached, because at this point there are no further paths for the uPhi to traverse down.

[26] Epstein, Kitahara & Seely (2014) account for the contrast in (ia–b) as resulting from the inability of labeling to occur in the embedded non-finite clause if *a man* remains in-situ. Specifically, in (ia) the embedded non-finite clause α has an {XP, YP} structure, but *a man* and non-finite T do not share any features. As a result, α cannot be labeled and (ia) is ruled out. (ib) is fine because the embedded clause α can be labeled by the projection of non-finite T *to* after *there* moves out. (ii) should also be fine because *a man* moves out of the embedded clause, thereby enabling labeling by the non-finite projection.

(i) a. *There is likely [$_{\alpha}$ a man to be t in the room]
b. There is likely [$_{\alpha}$ t to be a man in the room] (Epstein, Kitahara & Seely 2014: 471)
(ii) A man is likely [$_{\alpha}$ ~~a man~~ to be ~~a man~~ in the room]

Chomsky (2000) proposes that (ia) is blocked because of a preference for Merge of *there* over Move of *a man*, and importantly, *there* and *a man* are contained within the same lexical subarray. But Epstein et al. (2014) point out that (ia) is independently ruled out because the embedded clause can't be labeled. Thus, Epstein et al. (2014) suggest that their analysis does away with the need for Merge over Move, which also leads to the conclusion that phases are not necessary.

An anonymous reviewer points out that our view of labeling faces a potential problem in accounting for (ia). In our account, non-finite T inherits uPhi from a phase head. Assuming that there is no preference for Merge over Move, then in (ia), non-finite T inherits the uPhi of the matrix C, and uPhi on C are checked


(a)

uPhi from v* are inherited by
- expect
- toT
- vUnerg

(b)

Agree(v*,John)
- Inherited uPhi are checked
- Shared phi-features label
- Strengthened *expect* labels
- Strengthened toT labels
- Strengthened vUnerg labels

Figure 23: Labeling of non-finite T.

via agree with *a man*. Once the uPhi are checked, the phi-features that are shared between *a man* and toT should label α. The result would be that (ia) is predicted to be well-formed, contrary to fact. There are at least two possible ways of accounting for (ia). One solution would be to follow Chomsky (2000), in which case the possibility of external merge of *there* blocks remerge of *a man* in the embedded non-finite clause; this would reinstate the preference for Merge over Move, contra Epstein et al. (2014). Another possibility is to assume that non-finite T is only able to inherit a subset of uPhi (e.g., uPerson only) from a phase head. This would be in line with Chomsky's view (2001) that non-finite T is defective, and only has uPerson. After uPhi on the closest phase head (in this case the matrix C) agree with *a man*, the inherited uPerson of *to* would be checked. The result is that in the embedded clause there would be an {XP, YP} structure consisting of the XP *a man*, containing a full set of phi-features (e.g., phi-features for person, number, and gender), and the YP *to be in the room*, containing only a subset of phi-features (only person), as shown in (iii). Assume that a shared person feature alone is not sufficient for labeling via shared phi-features. As a result, there is a labeling failure. Thus, (ia) is predicted to be ill-formed, and Epstein et al's (2014) proposal can be maintained.

(iii) {[a man$_{[iPer,iNum,iGen]}$] [to$_{[iPer]}$ be t in the room]} → A shared Person feature only is not sufficient for labeling.

An anonymous reviewer wonders how this latter analysis can deal with a construction such as (iii).

(iv) They expected John to win. (Chomsky 2015: 10)

2.9 The halting problem

Chomsky (2015) discusses (21a), in which the issue arises of why *which dog* can't raise from the embedded interrogative clause. Chomsky refers to this as "the halting problem", following Rizzi (2013).

(21) a. *Which dog do you wonder John likes? (Chomsky 2015: 8)
b. [$_{\alpha}$ which dog do you wonder [$_{\beta}$ ~~which~~ ~~dog~~ Q John likes]]

Initially, *which dog* raises to the edge of the embedded clause, and then it raises to the edge of the matrix clause. Both the matrix and embedded clauses are projections of interrogative Cs. Assume that β is labeled after the higher matrix phase is constructed. At this point, *which dog* has raised to the matrix clause. Therefore, the embedded clause β is labeled by the interrogative C alone. This produces a yes/no construction. The matrix clause α is then labeled by the Q feature that is shared between the matrix interrogative C and the wh-phrase *which dog*. This shared Q feature produces a wh-question interpretation. The result is a wh-construction that contains a yes/no construction. Chomsky proposes that (21a) is ill-formed because a wh-question containing an embedded yes/no question is semantically ill-formed, so that it crashes at the Conceptual-Intentional Interface.

2.10 Summary

We have discussed the core components of POP. We have tried to clarify the crucial algorithms that are required to construct derivations in the POP model. On the way, we have examined the primary proposals of POP and tried to clarify how these proposals can actually be implemented, and we have discussed some issues/problems and developed some solutions to these problems.

3 The model and derivations

We created a computer model of POP that automatically constructs detailed derivations of the core examples from Chomsky (2015). The core components of this model consist of a lexicon and a set of operations/principles.

Table 2 shows the basic elements of the lexicon. Phase heads have uPhi features and the ability to assign case. The phase heads also have the ability to label. The other heads that label differ from the phase heads in that they do not have uPhi and cannot assign case. Heads that cannot label are lexical roots (which lack any category label), vUnerg and tense heads. Following Chomsky (2015), we assume that a lexical root merges with a category label, thus obtaining a category.[27] For example, when the lexical root *book* merges with an n head, a noun is formed.[28]

Phase heads	Other heads that label	Heads that can't label
v*, C, C_Q	D, n	TPres, TPast, toT, vUnerg, roots

Table 2: Lexicon.

Crucially, in (ia), when the uPerson features are inherited by non-finite T, we assume that *a man* has remained within the embedded clause merged with the non-finite T projection, where a labeling failure occurs. But in (iv), *John* raises out of the embedded clause and remerges with the matrix verbal root before the closest phase head, in this case v*, is merged. After v* is merged, *John* is no longer within the embedded clause. Thus, because *John* is no longer merged with the non-finite T, the non-finite T will be able to label, after it is strengthened. See Figure 25 and the discussion of example (23) for details.

[27] Chomsky follows work by Borer (2005a; 2005b; 2013), Marantz (1997); Embick and Marantz (2008).

[28] Note that if we assume that the nominal, which consists of a lexical root that has merged with n, is a complex element (i.e., an XP), then when the head D merges with the XP *book*, the D, being a head, must label. This is the approach that we take. However, if we assume that the nominal is a simplex element (i.e., a head), then there is merge of an N head with a D head and it isn't clear how the labeling algorithm should determine the label.

The core operations/principles that merge elements from the lexicon to form derivations, which were introduced in section 2, are summarized in Table 3, in the order in which these operations tend to occur. These consist of a) operations/principles taken directly from Chomsky (2013; 2015), b) modified versions of operations/principles from Chomsky (2013; 2015), and c) revised/new operations/principles designed to resolve potential problems with POP.

#	Core POP Operations/principles	Source	Motivation
(1)	**Merge** Merge syntactic objects X and Y to form the syntactic object Z = {X, Y}.	POP	Needed to construct an SO.
(8)	**Movement operations** a. When an unlabeled SO is formed, remerge the closest available SO with phi-features. b. Movement of an SO occurs to create a structure that can be labeled for semantic reasons. c. An unlicensed SO with an uninterpretable feature that is about to be transferred remerges with the root node (if possible).	POP (modified)	Explains when an SO moves. See section 2.4.
(19)	**Feature inheritance** Uninterpretable features are passed to an unlicensed complement.	POP (modified)	Accounts for feature checking and labeling of otherwise weak heads such as T, non-finite T, vUnerg, and lexical roots. See discussion of (3) and section 2.8.
(6)	**Probing from the root node, feature unification, and feature checking** a. Only a root node can probe. b. Feature inheritance leads to unified instances of a feature on multiple SOs. c. An uninterpretable feature is checked when it is valued by a matching interpretable feature.	Probing from a root node (6a) and feature unification (6b) are based on Richards (2007b) and Fong (2014), respectively. Feature checking (6c) follows POP.	Counter-cyclic probe-goal agree relations do not occur. See section 2.3.
(4)	**Strengthening** The process in which an unlabeled SO obtains prominent features that are capable of labeling.	POP	Necessary for a previously unlabeled SO to become labelable. See Figures 1–2.
(14)	**Remerge condition** Remerge of a Syntactic Object X with a Syntactic Object Y cannot occur if X and Y share features.	New	(a) Accounts for *that*-trace effects. See Figures 15–16. (b) Permits labeling to occur as soon as possible. See Figures 18–20. (c) Requires cyclicity. See section 2.7. (d) Requires dephasing of v*. See section 2.7.
(2)	**Labeling** a. When a strong head X is merged, the label is X. b. If {XP, YP} share prominent features that are capable of labeling, the shared features label. c. If YP moves out of {XP, YP}, then XP labels. If XP moves out of {XP, YP}, YP labels.	POP	Necessary for interpretation.

#	Core POP Operations/principles	Source	Motivation
(15)	**Immediate labeling** Labeling occurs as soon as a configuration is labelable.	Revision	There does not appear to be any reason to delay labeling until the phase level, counter POP. See Figures 18–20.
(12)	**Dephasing** a. A verbal root remerges with the local v* (when present) and phasehood is transferred to the verbal root. b. Deletion of C transfers phasehood to T.	POP	Necessary to account for ECM constructions and *that*-trace effects. See section 2.5.
(9)	**Transfer** a. The complement of a phase head is transferred. b. A derivation is transferred when all operations are completed.	POP	Necessary to account for when portions of a derivation become inaccessible.

Table 3: Summary of operations/rules.

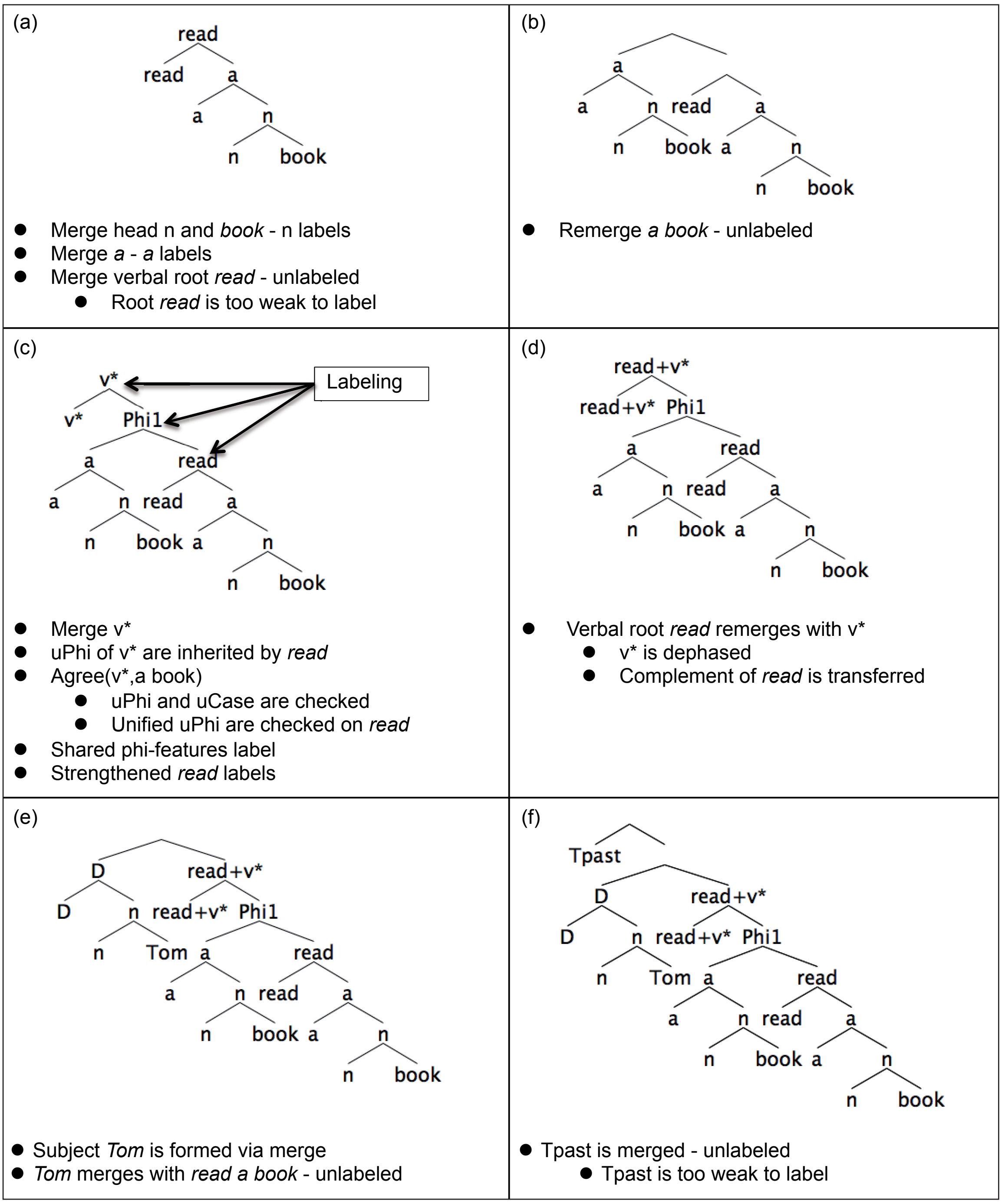
Utilizing the lexicon (Table 2) and the operations/principles from Table 3, we created a model that automatically produces detailed structures of the following examples (22–28), which are primarily from Chomsky (2015).[29]

(22) Tom read a book. (Chomsky 2015: 10)
(23) They expected John to win. (Chomsky 2015: 10)
(24) Who do you expect to win? (Chomsky 2015: 10)
(25) What do you think (that) John read?
(26) *Who do you think that read the book? (Chomsky 2015: 10)
(27) Who do you think read the book? (Chomsky 2015: 10)
(28) *Which dog do you wonder John likes? (Chomsky 2015: 8)

The derivation of the simple sentence (22) is shown in Figure 24. Initially, as shown in Figure 24a, the object is formed. The n *book* merges with the D head *a* and the *a* labels, followed by merge of the object with the verbal root *read*, which is too weak to label. Since this structure is unlabeled, the closest available SO with phi-features, in this case the object, raises and remerges with *read*, shown in Figure 24b. Movement occurs in accord with the Movement operation (8a). Note that even though *a book* has moved, the unlabeled "read a book" structure remains unlabeled since *read* is a verbal root that is too weak to label. When v* is merged (Figure 24c), feature inheritance and feature checking occur in accord with (19) and (6). The uPhi-features of v* are inherited by the verbal root *read*, so that there are unified uPhi on both v* and *read*. The uPhi on the root node v* probe and agree with matching phi-features on *a book*. An agree relation results in checking of uPhi on v* and checking of uCase on *a book* – *a book* obtains accusative case. When uPhi on v* are checked, the unified uPhi on *read* are also simultaneously checked, so that the phi-features shared between *read* and *a book* label. The unlabeled lower projection with the root *read* then labels because *read* has been strengthened, in accord with (4), since *read* now contains checked phi-features. Next, shown in Figure 24d, the verbal root *read* remerges with v*, resulting in v* being affixed onto *read*, so that v* is dephased, in accord with (12); instead of the structure labeled by *read* being transferred, the complement of *read*, which is the lower instance of *a book*, is transferred. The subject *Tom* is formed via merge and *Tom* merges with "read a book" (Figure 24e). Since this is an {XP, YP} structure, there is no label. The past tense element Tpast is merged (Figure 24f), and

[29] See the Appendix (in Supplementary Files 1) for the actual output derivations that are automatically generated by our computer model.

again, the structure is unlabeled because Tpast is too weak to label. Since this structure is unlabeled, at the next stage (Figure 24g), the subject *Tom* remerges with Tpast. Movement of the subject out of the lower {XP, YP} structure enables labeling by *read* + v* - after movement, not all copies of *Tom* are contained within this structure, so the prominent element *read* + v* labels. At the last stage (Figure 24h), C is merged. At this point, the uPhi-features of C are inherited by Tpast. C and the subject agree, resulting in checking of uPhi on C and uCase on *Tom*. The unified uPhi on Tpast are simultaneously checked, thereby enabling labeling by shared phi-features. The strengthened Tpast then is able to label the lower projection, and the derivation converges successfully.



(a)

- Merge head n and *book* - n labels
- Merge *a* - *a* labels
- Merge verbal root *read* - unlabeled
  - Root *read* is too weak to label

(b)

- Remerge *a book* - unlabeled

(c)

- Merge v*
- uPhi of v* are inherited by *read*
- Agree(v*,a book)
  - uPhi and uCase are checked
  - Unified uPhi are checked on *read*
- Shared phi-features label
- Strengthened *read* labels

(d)

- Verbal root *read* remerges with v*
  - v* is dephased
  - Complement of *read* is transferred

(e)

- Subject *Tom* is formed via merge
- *Tom* merges with *read a book* - unlabeled

(f)

- Tpast is merged - unlabeled
  - Tpast is too weak to label

contd.

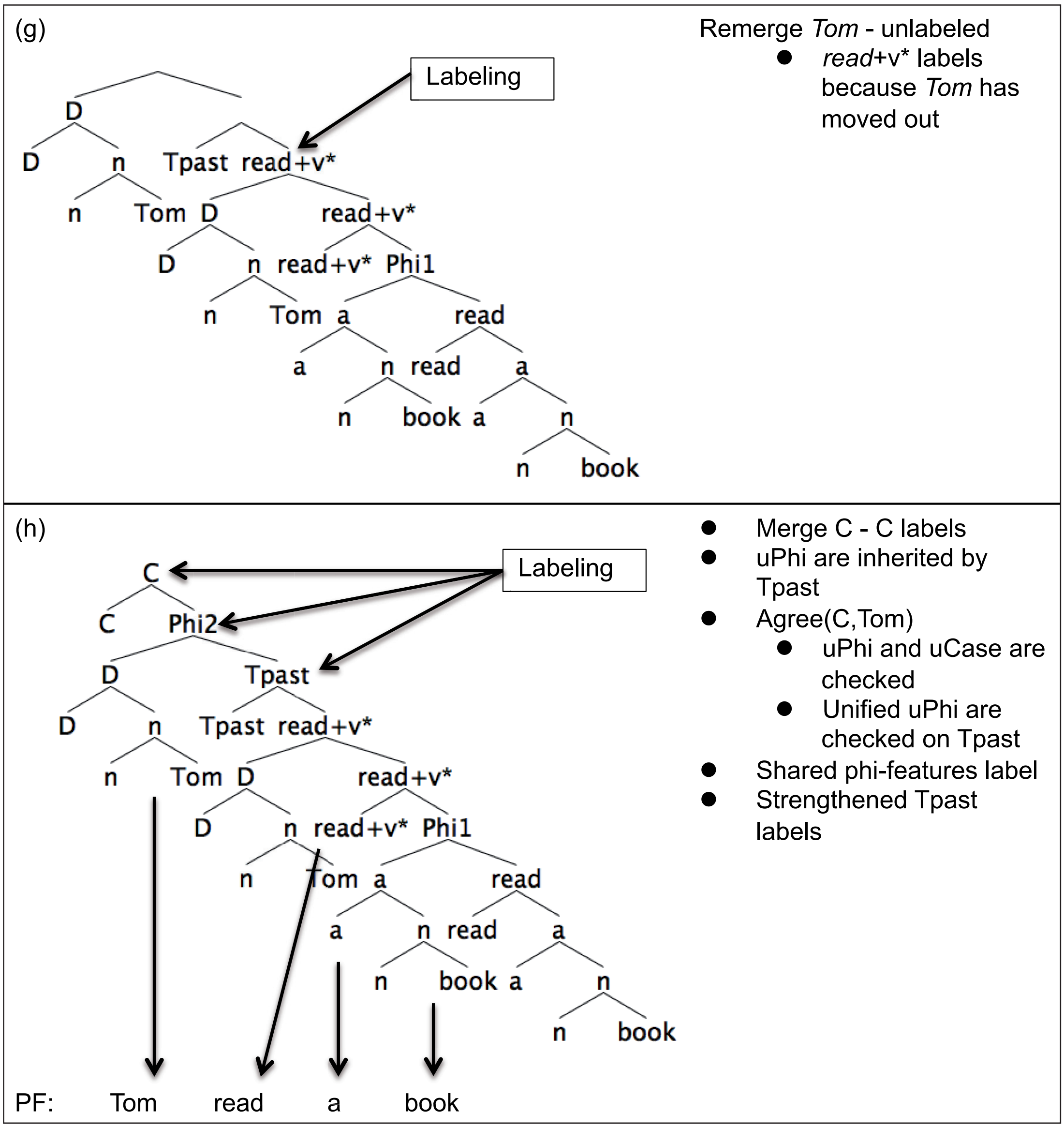
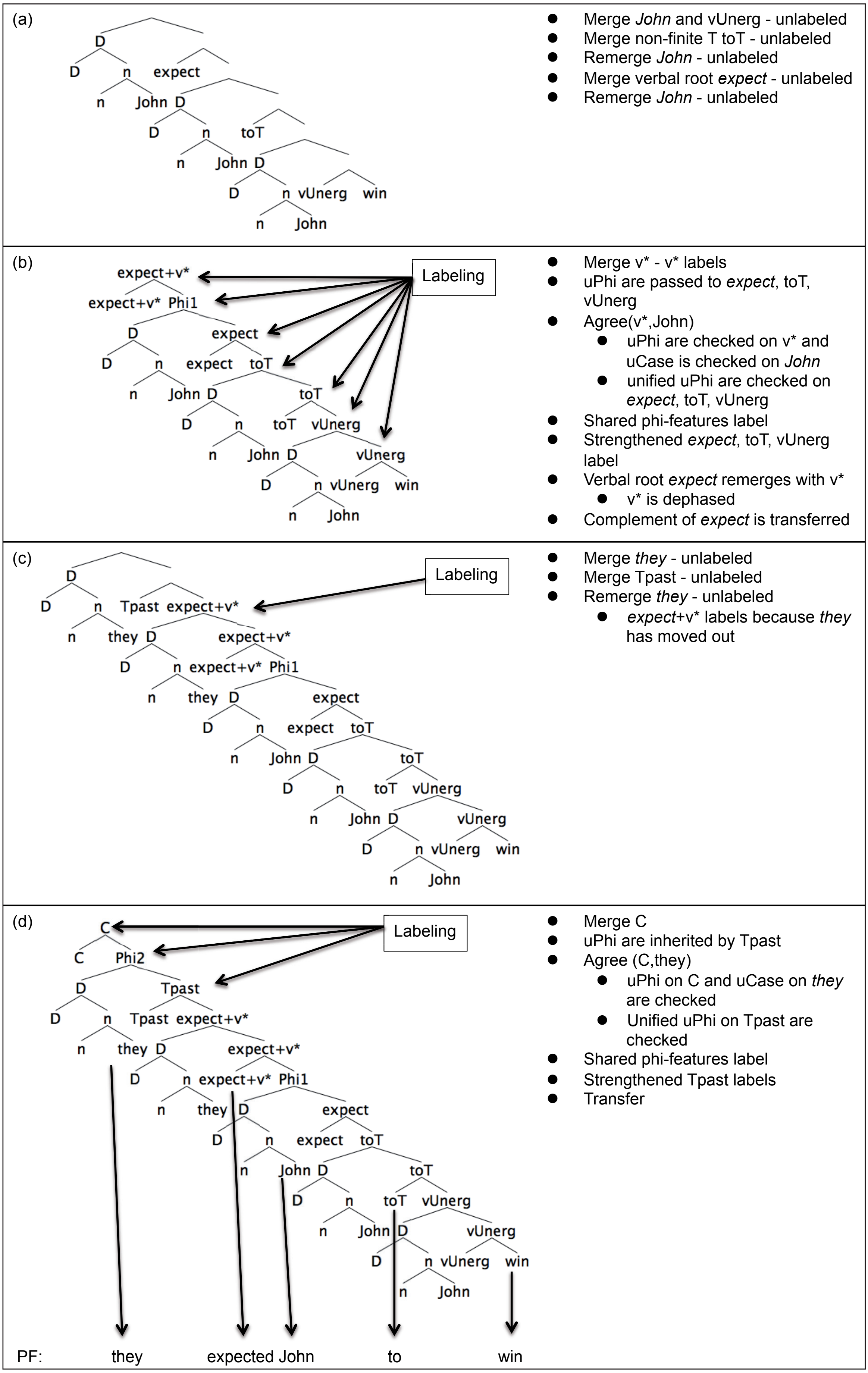


Figure 24: Derivation of "Tom read a book."

We next turn to the derivation of the exceptional case-marking (ECM) construction (23), shown in Figure 25. First, as shown in Figure 25a, the unergative verb *win* is formed from a vUnerg head and a verbal root *win* (see section 2.8 above for discussion of vUnerg). The unergative construction merges with the subject, resulting in an unlabeled structure, followed by merge of toT, and then remerge of the subject *John*. Then the root *expect* is merged, and the subject remerges again. Neither toT nor *expect* are strong enough to label. Remerge of the subject is forced by the Movement operation (8a), which forces remerge of the closest available SO with phi-features. At the next stage, Figure 25b, the phase head v* is merged and labels. The uPhi of v* are inherited by the verbal root *expect*, as well as by toT and vUnerg. The phase head v* probes and agrees with *John*, which crucially is within its search domain. As a result, uPhi of v* are checked and uCase on *John* is valued as accusative. When uPhi on v* are checked, the unified uPhi on *expect*, toT, and vUnerg are also valued. The subject *John* and its complement share phi-features, which label. Furthermore, the now strengthened *expect*, toT, and vUnerg all label. Then, the verbal root *expect* raises and remerges with v*, resulting in dephasing of v* and transfer of the



(a)

- Merge *John* and vUnerg - unlabeled
- Merge non-finite T toT - unlabeled
- Remerge *John* - unlabeled
- Merge verbal root *expect* - unlabeled
- Remerge *John* - unlabeled

(b)

- Merge v* - v* labels
- uPhi are passed to *expect*, toT, vUnerg
- Agree(v*,John)
  - uPhi are checked on v* and uCase is checked on *John*
  - unified uPhi are checked on *expect*, toT, vUnerg
- Shared phi-features label
- Strengthened *expect*, toT, vUnerg label
- Verbal root *expect* remerges with v*
  - v* is dephased
- Complement of *expect* is transferred

(c)

- Merge *they* - unlabeled
- Merge Tpast - unlabeled
- Remerge *they* - unlabeled
  - *expect*+v* labels because *they* has moved out

(d)

- Merge C
- uPhi are inherited by Tpast
- Agree (C,they)
  - uPhi on C and uCase on *they* are checked
  - Unified uPhi on Tpast are checked
- Shared phi-features label
- Strengthened Tpast labels
- Transfer

Figure 25: Derivation of "They expected John to win."

complement of *expect*. In Figure 25c, the matrix subject *they* merges with the *expect* + v* projection, resulting in an unlabeled structure. This operation is followed by merge of the past tense Tpast, which is too weak to label. Then *they* remerges with the Tpast projection. Remerge of *they* enables *expect* + v* to label. The final stages of the derivation are shown in Figure 25d. C is merged, followed by feature inheritance, feature checking, and labeling operations. C labels the entire structure. The uPhi of C are inherited by Tpast. The uPhi on C are checked via agree with *they* (simultaneously resulting in checking of the unified uPhi on Tpast), and uCase of *they* is valued nominative. The shared phi-features of the subject and the Tpast projection label, and the strengthened Tpast labels. The output is transferred and the derivation converges.

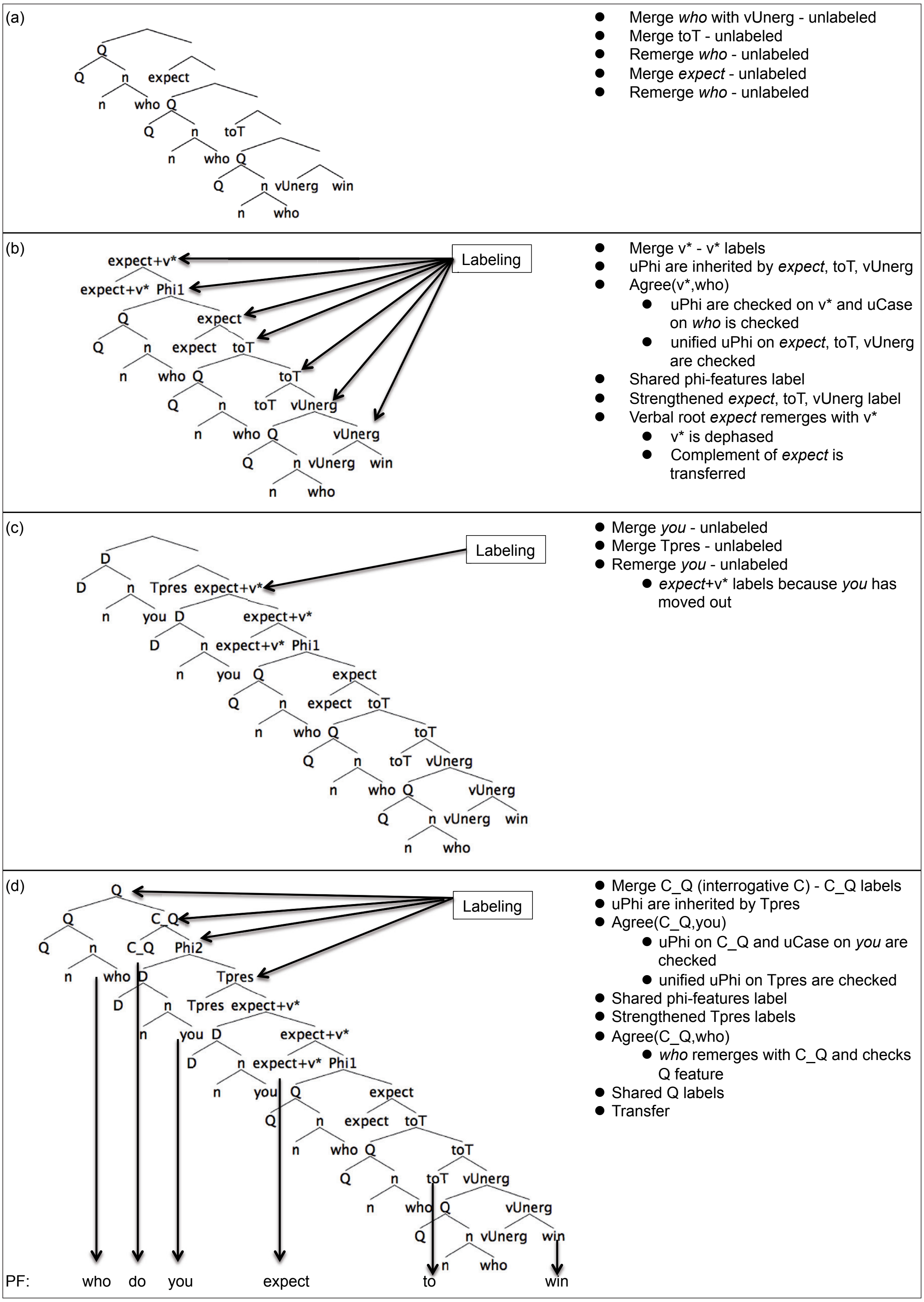
The derivation of the ECM construction with wh-movement (24) is shown in Figure 26. Figure 26a shows the unlabeled structure before merge of v*, which is formed via merge of vUnerg, toT, and *expect*, which are all too weak to label. The underlying embedded wh-subject *who* initially merges with vUnerg, and remerges with the projections of toT and *expect*, forming unlabeled {XP, YP} projections. Again, remerge is forced by the Movement operation (8a). When v* is merged (Figure 26b), uPhi are inherited by the verbal root *expect*, toT, and vUnerg. The uPhi on v*, as well as the unified uPhi on *expect*, toT, and vUnerg are checked. Shared phi-features label, and the now strengthened *expect*, toT, and vUnerg label. The verbal root *expect* undergoes movement, resulting in dephasing of v*, and transfer of the complement of *expect*. In Figure 26c, the subject *you* and Tpres are merged, followed by remerge of the subject, which enables labeling by *expect* + v*. At the final stage, Figure 26d, the interrogative C_Q is merged. The present tense Tpres inherits uPhi of C_Q. The uPhi of C_Q are checked via agree with the subject *you* and *you* obtains nominative case. The phi-features shared between Tpres and *you* label. The wh-phrase remerges with the C_Q projection, in accord with the Movement operation (8b) to create a structure that can be appropriately labeled for semantic reasons. In this case, the wh-phrase checks a Q feature on C_Q, and the Q-feature shared between C_Q and *who* labels the entire structure.[30] Note that we leave aside the issue of how T to C movement occurs, resulting in pronunciation of *do* (cf. Pesetsky & Torrego 2001, among others). Also, in Figure 26b, note that the dephasing step occurs in accord with (12). The step in which the verbal root *expect* raises and remerges with v* is crucial (see section 2.5) because dephasing of v* enables *who* to remain in the non-transferred search domain of C_Q.

We next turn to long distance wh-movement and *that*-trace effects. To account for these, the Remerge condition (14), which blocks an SO from remerging with another SO with which it shares features, is crucial.

Long distance movement of a wh-object in (25) is accounted for as shown in Figure 27. The lower v* projection is shown in Figure 27a. In this structure, *what* remerges with the verbal root projection to form an unlabeled structure. After merge of v*, uPhi-features are inherited by *read*, and v* and *what* undergo agreement, resulting in feature checking, labeling by shared phi-features, and labeling by the strengthened *read*. Remerge of *read* with v* results in dephasing of v* and transfer of the lower instance of *what*. At the

[30] An anonymous reviewer points out that if C_Q and the wh-phrase end up in a feature sharing relation, via agree, before the wh-phrase moves, then the Remerge condition (14) would actually block remerge, since C_Q and the wh-phrase would share features. Our view is that the Movement operation (8b) requires the wh-phrase and C_Q to be in a local configuration for semantic reasons. In this case, movement of the wh-phrase results from agreement between the wh-phrase and C_Q. So it is not the case that there is first an agree relation, resulting in feature sharing, followed by remerge of the wh-phrase. Rather, agreement and movement occur together, thus avoiding a violation of the Remerge condition.

Note that the idea that certain agreement relations trigger movement may not be all that different from saying that there is an EPP feature or Edge Feature associated with C_Q. Thus, this component of the model may be an imperfection that requires further investigation.



(a)

- Merge *who* with vUnerg - unlabeled
- Merge toT - unlabeled
- Remerge *who* - unlabeled
- Merge *expect* - unlabeled
- Remerge *who* - unlabeled

(b)

- Merge v* - v* labels
- uPhi are inherited by *expect*, toT, vUnerg
- Agree(v*,who)
  - uPhi are checked on v* and uCase on *who* is checked
  - unified uPhi on *expect*, toT, vUnerg are checked
- Shared phi-features label
- Strengthened *expect*, toT, vUnerg label
- Verbal root *expect* remerges with v*
  - v* is dephased
  - Complement of *expect* is transferred

(c)

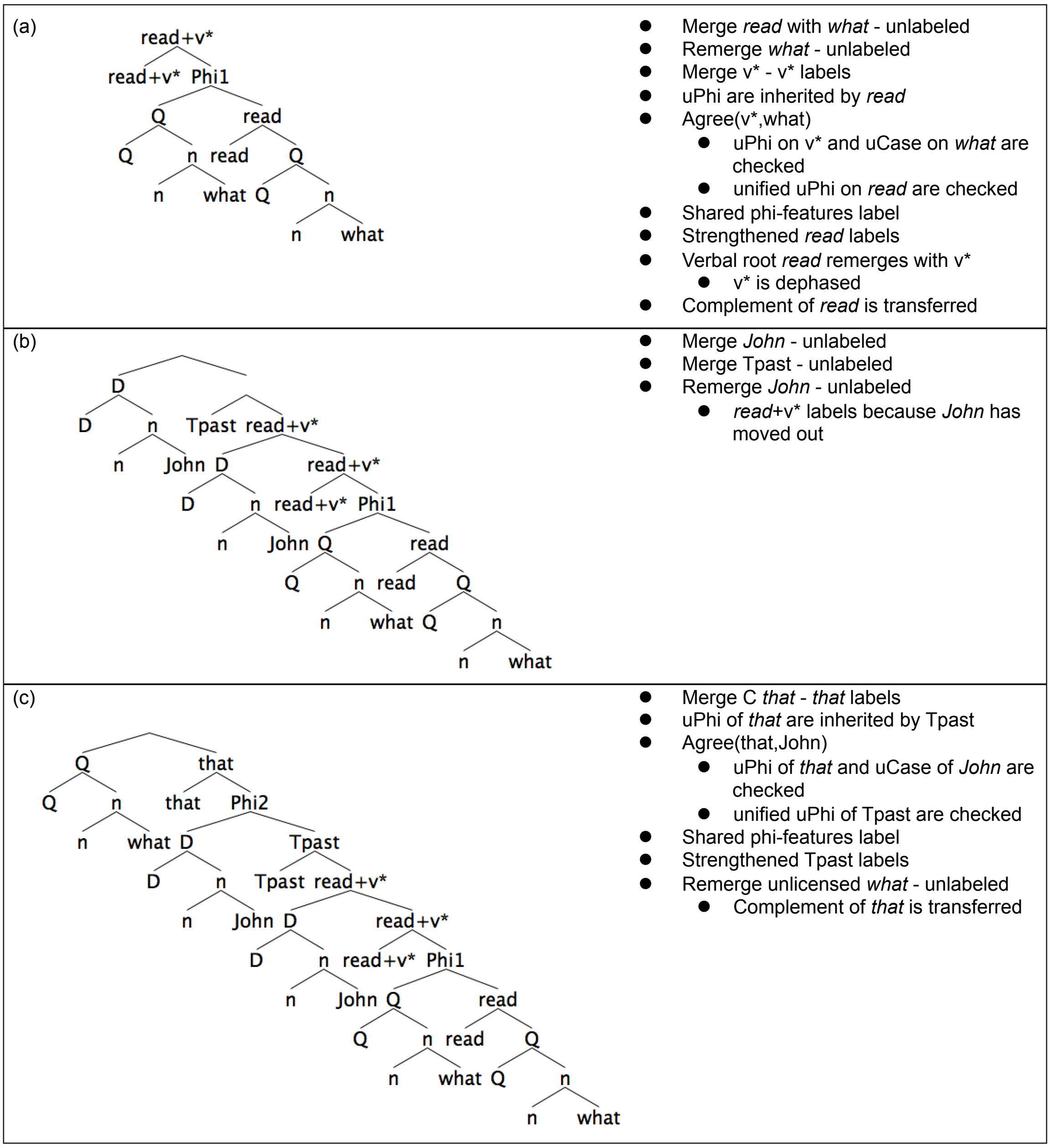
- Merge *you* - unlabeled
- Merge Tpres - unlabeled
- Remerge *you* - unlabeled
  - *expect*+v* labels because *you* has moved out

(d)

- Merge C_Q (interrogative C) - C_Q labels
- uPhi are inherited by Tpres
- Agree(C_Q,you)
  - uPhi on C_Q and uCase on *you* are checked
  - unified uPhi on Tpres are checked
- Shared phi-features label
- Strengthened Tpres labels
- Agree(C_Q,who)
  - *who* remerges with C_Q and checks Q feature
- Shared Q labels
- Transfer

Figure 26: Derivation of "Who do you expect to win?"

next stage, Figure 27b, the subject *John* and the Tpast structure are merged, followed by remerge of the subject, resulting in an unlabeled structure. Then, shown in Figure 27c, C *that* is merged. The uPhi of *that* are inherited by Tpast, *that* agrees with the subject, checking uPhi on *that* as well as the unified uPhi on Tpast. The phi-features shared between Tpast and the subject label, and the strengthened Tpast labels the lower projection. At this point, the wh-phrase *what* is still unlicensed, since it must be licensed via an agree relation with an interrogative C. Thus, the wh-phrase remerges with the embedded C, in accord with the movement operation (8c), bringing the higher copy of *what* out of the transfer domain; after merge of the phase head C, the complement of C, in this case corresponding to [$_{\text{Phi}}$ John read what] is transferred, but the higher copy of *what* is no longer in the transfer domain. Note that labeling occurs in the embedded clause before *what* is remerged with *that*, in accord with Immediate labeling (15); there is no need to wait until after remerge of *what*. Moving into the matrix clause, in Figure 27d, the verbal root *think* is merged. Then the wh-phrase remerges with the root node. This remerge operation is



(a)

- Merge *read* with *what* - unlabeled
- Remerge *what* - unlabeled
- Merge v* - v* labels
- uPhi are inherited by *read*
- Agree(v*,what)
  - uPhi on v* and uCase on *what* are checked
  - unified uPhi on *read* are checked
- Shared phi-features label
- Strengthened *read* labels
- Verbal root *read* remerges with v*
  - v* is dephased
- Complement of *read* is transferred

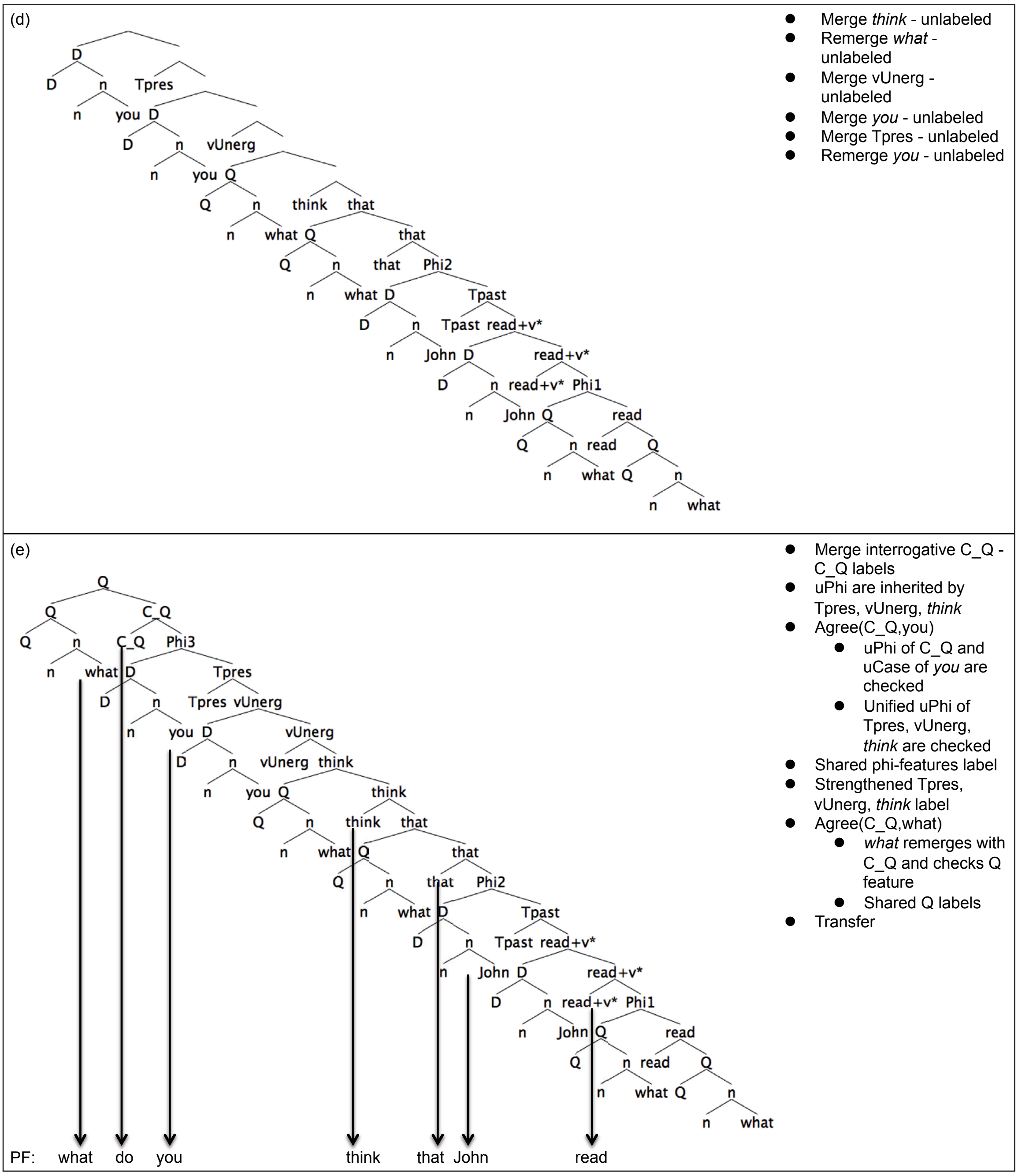
(b)

- Merge *John* - unlabeled
- Merge Tpast - unlabeled
- Remerge *John* - unlabeled
  - *read*+v* labels because *John* has moved out

(c)

- Merge C *that* - *that* labels
- uPhi of *that* are inherited by Tpast
- Agree(that,John)
  - uPhi of *that* and uCase of *John* are checked
  - unified uPhi of Tpast are checked
- Shared phi-features label
- Strengthened Tpast labels
- Remerge unlicensed *what* - unlabeled
  - Complement of *that* is transferred

contd.



(d)

- Merge *think* - unlabeled
- Remerge *what* - unlabeled
- Merge vUnerg - unlabeled
- Merge *you* - unlabeled
- Merge Tpres - unlabeled
- Remerge *you* - unlabeled

(e)

- Merge interrogative C_Q - C_Q labels
- uPhi are inherited by Tpres, vUnerg, *think*
- Agree(C_Q,you)
  - uPhi of C_Q and uCase of *you* are checked
  - Unified uPhi of Tpres, vUnerg, *think* are checked
- Shared phi-features label
- Strengthened Tpres, vUnerg, *think* label
- Agree(C_Q,what)
  - *what* remerges with C_Q and checks Q feature
  - Shared Q labels
- Transfer

Figure 27: Derivation of "What do you think that John read?"

forced by the movement operation (8a). The wh-phrase is an unlicensed element with phi-features that is available (it is not within a transferred domain). Therefore, the wh-phrase moves. Movement of the wh-phrase enables *that* to label the embedded clause. Then the vUnerg head is merged, followed by merge of the subject *you*.[31] The final steps of the

[31] Note that we must assume that external merge of *you* blocks remerge (internal merge) of *what*. According to the Movement operation, (8a), when an unlabeled SO is formed, the closest available SO with phi-features is to be remerged. In Figure 27d, when vUnerg is merged, an unlabeled SO results, and the closest available SO with phi-features is *what*. But if *what* were to remerge with vUnerg, then the subject *you* would not be merged into the derivation, and would not obtain a theta-role. Therefore, the possibility of external merge of *you* blocks remerge of *what*.

Epstein et al. (2014) propose that there are no subarrays. If there are no subarrays, then the question arises of why external merge of *you* doesn't block remerge of *what* throughout the derivation. The solution derivation are shown in Figure 27e. The interrogative C_Q is merged, followed by inheritance of uPhi-features by Tpres, and agreement resulting in feature checking between C_Q and the subject. Then there is labeling by shared phi-features and strengthening. C_Q also has a Q feature that needs to be checked. Thus, C_Q agrees with *what*, which is still in the relevant search domain. As a result, the Q feature on C_Q is checked, and *what* remerges with the C_Q structure. Movement of *what* is driven by the need for the structure to be appropriately labeled as an interrogative, following the movement operation (8b). At this point, *what* and C_Q share a Q-feature, which becomes the final label, and the derivation converges.

Next, consider the derivation of (25) with a null C. Figure 28a shows the embedded clause after C has merged. Features are inherited from C, labeling occurs, and then C is deleted, shown in Figure 28b. The Movement operation (8c) forces *what* to remerge with the root to produce the structure in Figure 28c. After dephasing, the complement of Tpast is transferred. The final stages of the derivation are shown in Figure 28d. Note that *what* remerges with the unlabeled SO that is formed after merge of the verbal root *think*, so that shared phi-features label in the embedded clause. After merge of the matrix C_Q, *what* again remerges and the total construction is labeled by a shared Q feature.

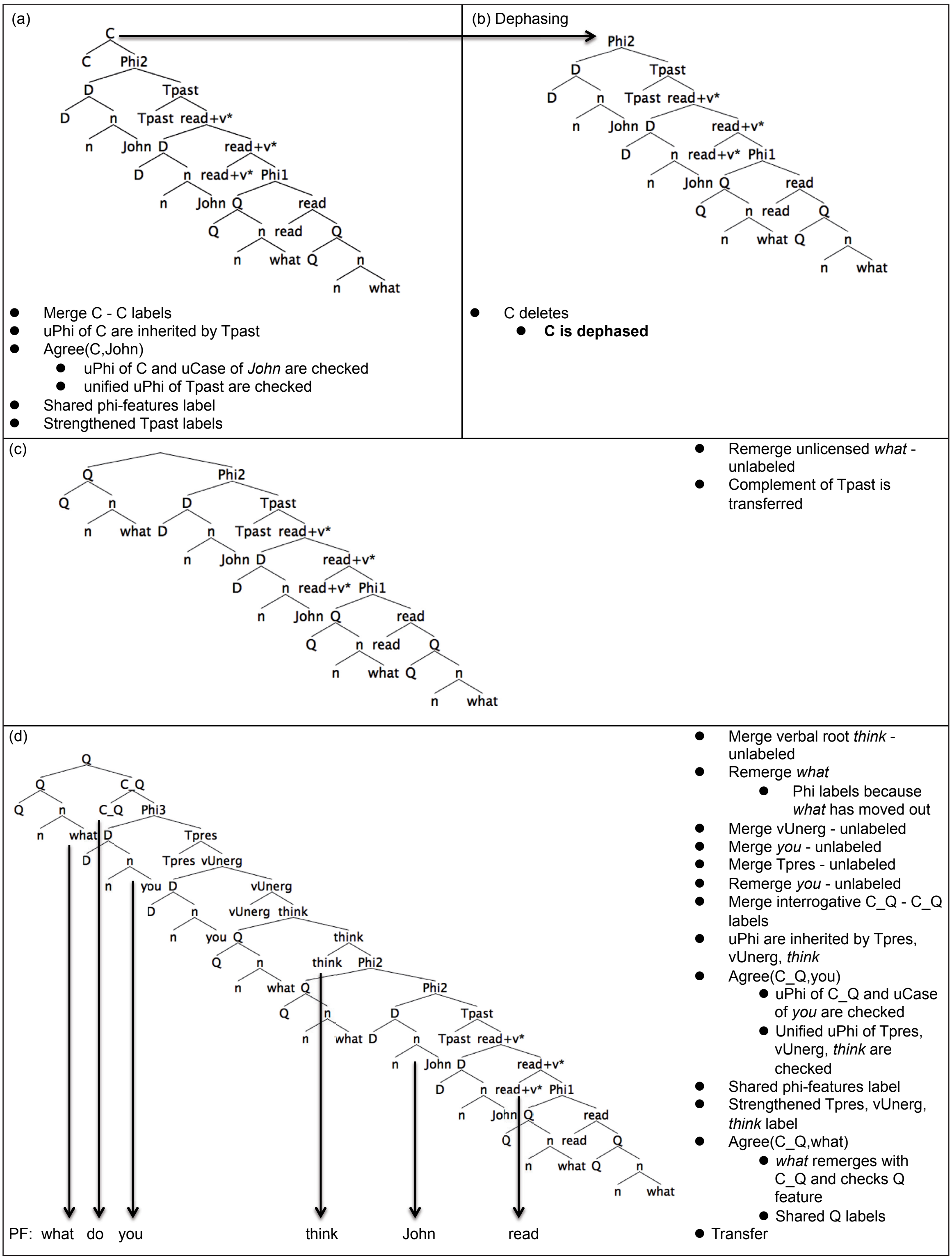
We next turn to the *that*-trace effect of example (26). The embedded clause is shown in Figure 29. After merge of C *that*, the uPhi of *that* are inherited by Tpast, and uPhi of C as well as uCase of the subject are checked. Shared phi-features then label, and the strengthened Tpast labels. At this point, the derivation differs from a construction in which there is extraction of a wh-object. In Figure 27c above, the unlicensed wh-object remerges with the root SO, thereby removing it from the transfer domain. But in Figure 29, the wh-subject is already in a feature checking relation with C *that*; the subject *who* has checked the uPhi of C, so *who* and C share phi-features. Thus, in accord with the Remerge condition (14), *who* cannot remerge with *that*. The complement of the phase head *that* is transferred, and *who* is among those transferred elements, causing the derivation to crash because *who* is no longer available for agreement with the matrix C_Q. Note that there are two possibilities for the crash timing: 1) when the complement of C is transferred, the presence of the unlicensed *who* in the transfer domain leads the derivation to crash, or 2) the derivation continues until the matrix C_Q is merged, at which point the derivation crashes because there is no wh-phrase available to check the relevant interrogative feature of C_Q.

The derivation of the well-formed (27) is successful because of deletion of C, as shown in Figure 30. When the embedded C is merged, Figure 30a, uPhi are inherited by Tpast, and C and *who* undergo a feature checking relation. The phi-features shared between *who* and Tpast label. Then C deletes (Figure 30b), resulting in dephasing of C. As a result, the complement of C is not transferred, meaning that *who* remains accessible to higher operations. The matrix verbal root *think*, vUnerg and Tpres are merged, shown in Figure 30c. After the matrix C_Q is merged, uPhi are inherited by Tpres, and labeling proceeds in the usual manner. Crucially, C_Q is able to agree with *who*, resulting in remerge of *who* and sharing of a Q feature, which labels the resulting structure.

Lastly, we demonstrate what our model does with example (28), which exemplifies the halting problem – the issue of why a wh-phrase that is licensed in an embedded interrogative clause can't move into a higher clause, as discussed in section 2.9. Our model

here seems to be theta-role assignment. When the root node of an SO is a theta-role assigning head, there is a preference for external merge of an SO that lacks a theta-role. So in this example, external merge of *you* blocks remerge of *what* to the theta-role assigning vUnerg.

Whether or not external merge is required for theta-role assignment depends on whether or not an SO can obtain more than one theta-role and/or appear in more than one theta-position (cf. Hornstein 1999; 2001, among others), an issue which is beyond the scope of this work.



(a)

- Merge C - C labels
- uPhi of C are inherited by Tpast
- Agree(C,John)
  - uPhi of C and uCase of *John* are checked
  - unified uPhi of Tpast are checked
- Shared phi-features label
- Strengthened Tpast labels

(b) Dephasing

- C deletes
  - **C is dephased**

(c)

- Remerge unlicensed *what* - unlabeled
- Complement of Tpast is transferred

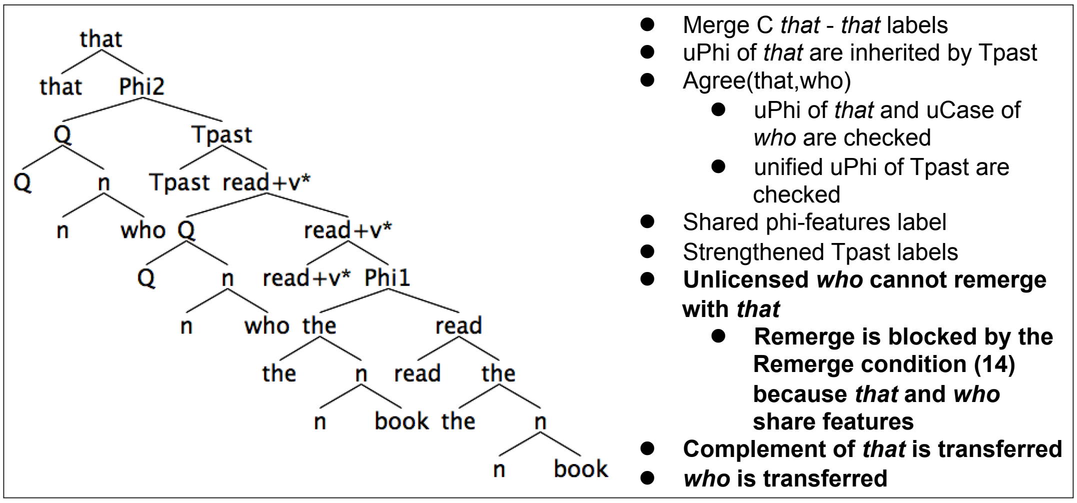
(d)

- Merge verbal root *think* - unlabeled
- Remerge *what*
  - Phi labels because *what* has moved out
- Merge vUnerg - unlabeled
- Merge *you* - unlabeled
- Merge Tpres - unlabeled
- Remerge *you* - unlabeled
- Merge interrogative C_Q - C_Q labels
- uPhi are inherited by Tpres, vUnerg, *think*
- Agree(C_Q,you)
  - uPhi of C_Q and uCase of *you* are checked
  - Unified uPhi of Tpres, vUnerg, *think* are checked
- Shared phi-features label
- Strengthened Tpres, vUnerg, *think* label
- Agree(C_Q,what)
  - *what* remerges with C_Q and checks Q feature
  - Shared Q labels
- Transfer

Figure 28: Derivation of "What do you think John read?"

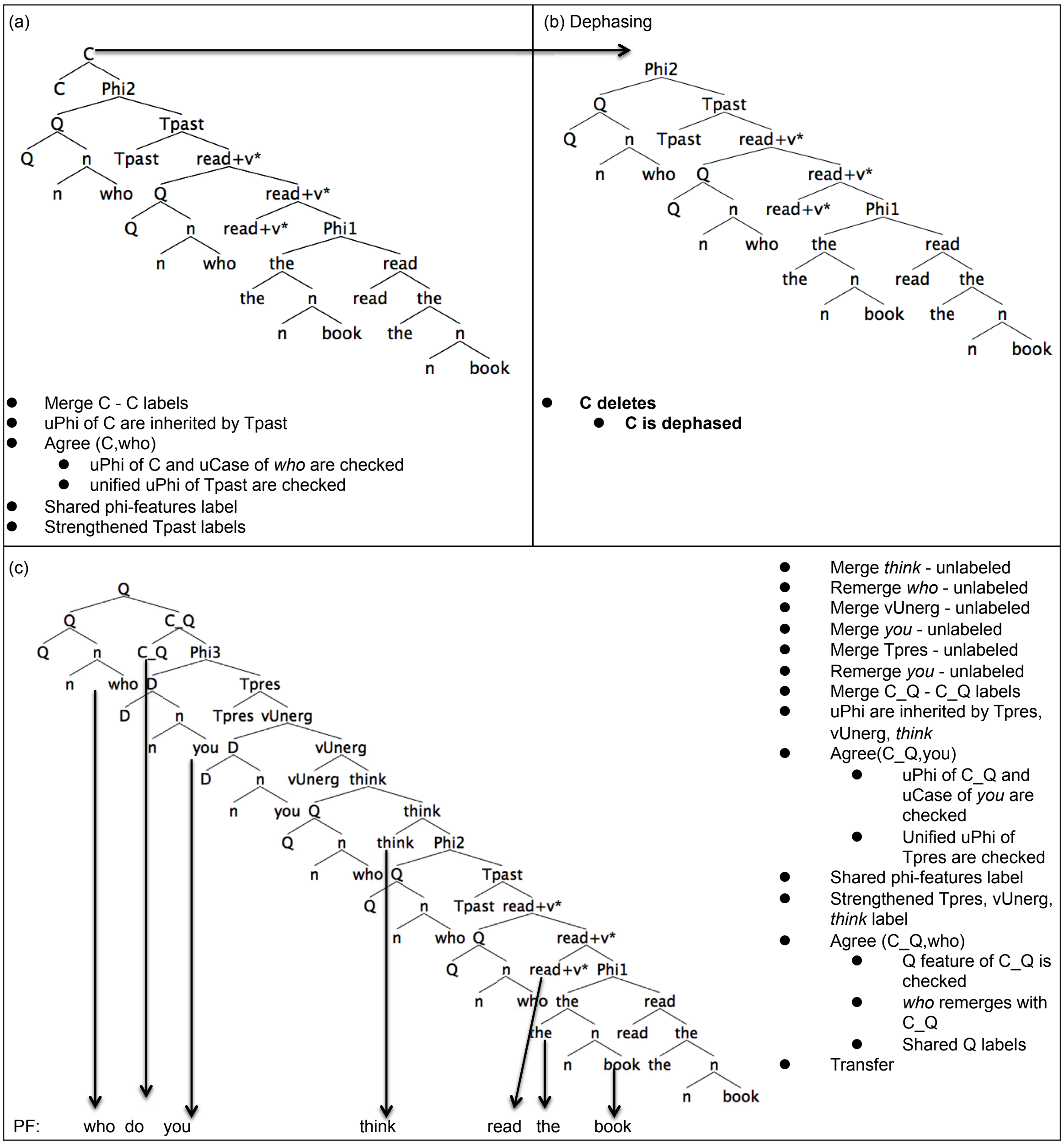
predicts that this example converges, although not in the expected way. The full embedded clause is shown in Figure 31a. After the embedded C_Q is merged, features are inherited by Tpres, and labeling occurs in the usual manner. Then the unlicensed wh-phrase *which dog* agrees with C_Q and remerges. The resulting structure is labeled by a shared Q feature. In the matrix clause,[32] Figure 31b, when the interrogative C_Q is merged, there is

[32] Note that after the verbal root *wonder* is merged, *which dog* does not remerge with *wonder*. Compare this



- Merge C *that* - *that* labels
- uPhi of *that* are inherited by Tpast
- Agree(that,who)
  - uPhi of *that* and uCase of *who* are checked
  - unified uPhi of Tpast are checked
- Shared phi-features label
- Strengthened Tpast labels
- **Unlicensed *who* cannot remerge with *that***
  - **Remerge is blocked by the Remerge condition (14) because *that* and *who* share features**
- **Complement of *that* is transferred**
- ***who* is transferred**

Figure 29: Derivation of "*Who do you think that read the book?"


(a)

- Merge C - C labels
- uPhi of C are inherited by Tpast
- Agree (C,who)
  - uPhi of C and uCase of *who* are checked
  - unified uPhi of Tpast are checked
- Shared phi-features label
- Strengthened Tpast labels

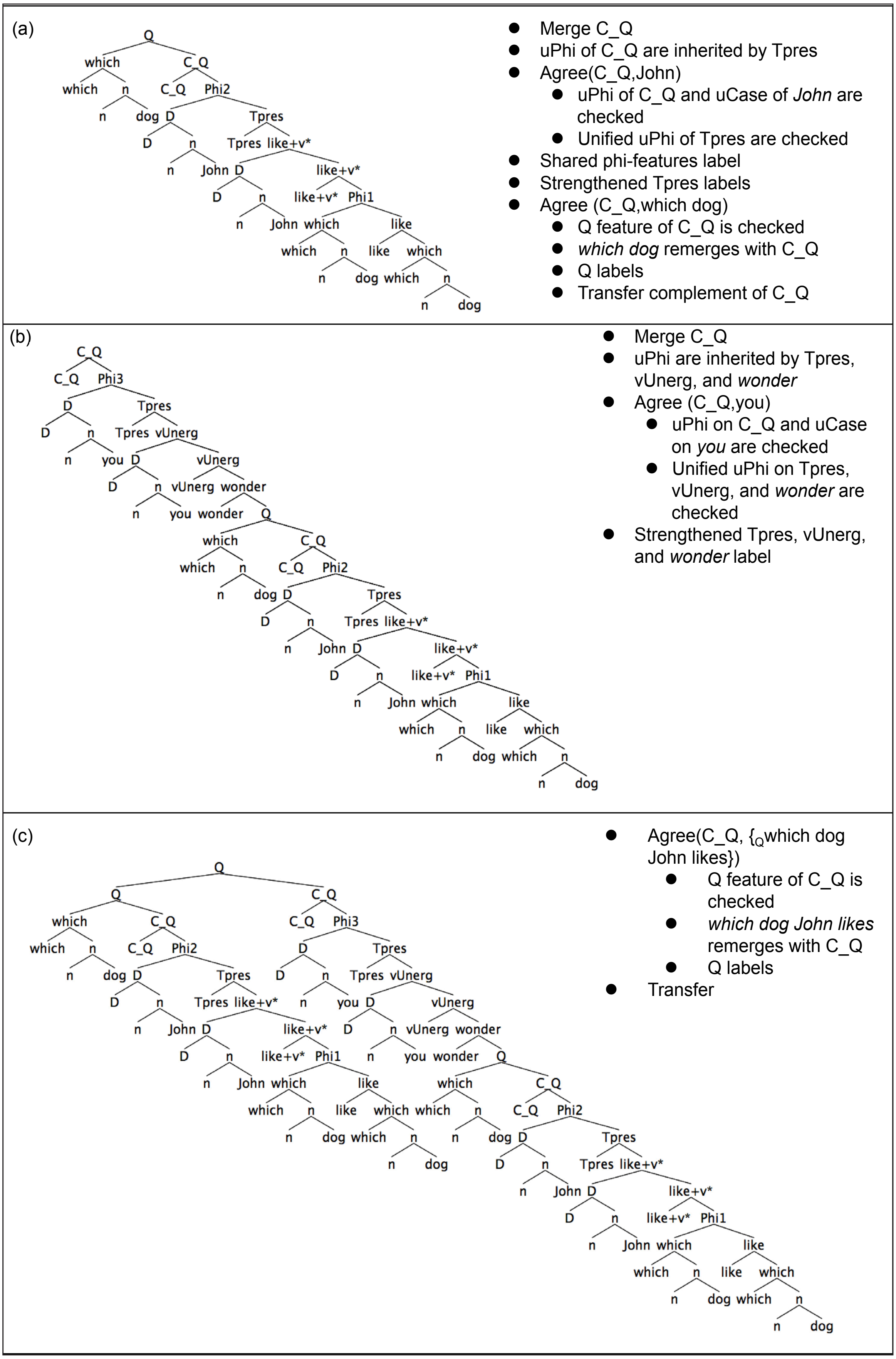
(b) Dephasing

- **C deletes**
  - **C is dephased**

(c)

- Merge *think* - unlabeled
- Remerge *who* - unlabeled
- Merge vUnerg - unlabeled
- Merge *you* - unlabeled
- Merge Tpres - unlabeled
- Remerge *you* - unlabeled
- Merge C_Q - C_Q labels
- uPhi are inherited by Tpres, vUnerg, *think*
- Agree(C_Q,you)
  - uPhi of C_Q and uCase of *you* are checked
  - Unified uPhi of Tpres are checked
- Shared phi-features label
- Strengthened Tpres, vUnerg, *think* label
- Agree (C_Q,who)
  - Q feature of C_Q is checked
  - *who* remerges with C_Q
  - Shared Q labels
- Transfer

Figure 30: Derivation of "Who do you think read the book?"



- Merge C_Q
- uPhi of C_Q are inherited by Tpres
- Agree(C_Q,John)
  - uPhi of C_Q and uCase of *John* are checked
  - Unified uPhi of Tpres are checked
- Shared phi-features label
- Strengthened Tpres labels
- Agree (C_Q,which dog)
  - Q feature of C_Q is checked
  - *which dog* remerges with C_Q
  - Q labels
  - Transfer complement of C_Q

- Merge C_Q
- uPhi are inherited by Tpres, vUnerg, and *wonder*
- Agree (C_Q,you)
  - uPhi on C_Q and uCase on *you* are checked
  - Unified uPhi on Tpres, vUnerg, and *wonder* are checked
- Strengthened Tpres, vUnerg, and *wonder* label

(c)
Q
Q C_Q
which C_Q C_Q Phi3
which n C_Q Phi2 D Tpres
n dog D Tpres D n Tpres vUnerg
D n Tpres like+v* n you D vUnerg
n John D like+v* D n vUnerg wonder
D n like+v* Phi1 n you wonder Q
n John which like which C_Q
which n like which which n C_Q Phi2
n dog which n n dog D Tpres
n dog D n Tpres like+v*
n John D like+v*
D n like+v* Phi1
n John which like
which n like which
n dog which n
n dog

- Agree(C_Q, {$_{Q}$which dog John likes})
  - Q feature of C_Q is checked
  - *which dog John likes* remerges with C_Q
  - Q labels
- Transfer

Figure 31: Derivation of "*Which dog do you wonder John likes?"

feature inheritance and labeling. Then the matrix C_Q searches for a goal that contains a Q feature. In this case, the closest goal with a Q feature is not the specifier of the embedded clause *which dog*, but rather, it is the entire embedded clause *which dog John likes*, since the label of this clause is Q. Thus, C_Q agrees with *which dog John likes*, the Q feature of C_Q is checked, and *which dog John likes* remerges with C_Q, as shown in Figure 31c.

Note that if this analysis is on the correct track, then the target construction, repeated below as (29a) may actually not be derivable. The real question may be that of why (29b) ill-formed.

(29) a. *Which dog do you wonder John likes?
b. *Which dog John likes do you wonder ~~which dog John likes?~~

It also may be the case that for the matrix C_Q, the embedded phrase *which dog* is available for remerge via a different method of determining closeness.

Our model does not directly rule out (29b), and it is not entirely clear if our model can explain the ill-formedness of (29a).[33,34] While we do not offer a clear solution, one possibility is that in the derivations of (29a–b), ill-formedness results from the label of the matrix clause being too complex. As shown in Figure 32, there is merge of a phrase that is labeled via a shared Q feature (formed from X and Y which contain Q features) with another phrase Z that has a Q feature. In other words, the final label is a Q feature that is shared among X, Y, and Z, so that there is a Q within a Q. This complex label could possibly create problems for semantic interpretation, or possibly for the labeling algorithm. We leave this issue for further research.

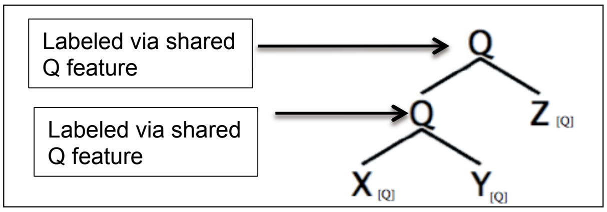


Figure 32: Labeling in (29a–b).

with the examples shown in Figures 26, 27, 28, and 30 above in which the wh-phrase from the embedded clause remerges with the matrix verbal root. The wh-phrase *which dog* does not remerge at this point because it is not visible to the movement operation (8a) since it does not have any uFs. Due to a lack of uFs, we assume that it is not available for remerge in this case.

[33] Keeping in mind that our model generates (29b) and not (29a), an anonymous reviewer points out that our analysis is not able to account for (29a) in the way intended by Chomsky. In Chomsky (2015), semantic ill-formedness of (29a) results due to a wh-question containing an embedded yes/no construction (see section 2.9 above). In the analysis proposed here, the Immediate labeling operation (15) results in the embedded clause being labeled by a Q feature that is shared between *which dog* and C_Q, as shown in Figure 31a above. This means that the embedded clause actually is interpreted as a wh-construction. If labeling information cannot be lost, which Chomsky (2015) suggests is the case, then the embedded clause cannot be interpreted as a yes/no construction.

[34] In (29b), the wh-phrase does not move out of the embedded clause, so the result is not a wh-construction that contains a yes/no construction, which would be semantically ill-formed according to Chomsky (2015). Rather, it is a wh-construction that contains another wh-construction. Thus, semantic ill-formedness does not rule this example out.

Our analysis does not rule out (29b), but an anonymous reviewer points out that Chomsky (2015) can rule this out, given Chomsky's views about agreement in wh-questions. We assume that C_Q has a uQ feature that is checked via agreement with a wh-phase (really a Q phrase) that contains a Q feature. Under our analysis, (29b) should converge, contrary to fact. Chomsky, however, assumes that an interrogative C has a Q feature that values an unvalued Q feature on a wh-phrase. See footnote 17 for discussion of this.

Following Chomsky's (2015) view of feature checking in a wh-construction, in the relevant portions of (29b), shown in (i), the unvalued Q feature of *which dog* is checked via agreement with the valued Q feature of C_Q1. As a result, the Q label of *which dog John likes* does not have an unvalued Q feature. Both C_Q2 and the Q label of *which dog John likes* have valued Q features. Therefore, there is no agreement relation that can be established between C_Q2 and *which dog John likes*. As a result, there should be a labeling failure.

(i) [[$_Q$ which dog C_Q1 John likes]$_1$ [C_Q2 [you wonder t$_1$]]]

4 Conclusion

This paper has described a model of recent work in the POP approach to language. We have developed a revised POP model in which we have attempted to make explicit and clarify the core elements that are required for POP, and we have demonstrated how this model accounts for the main examples from Chomsky (2015), while overcoming certain problems/issues for POP having to do with cyclicity, probing from a non-root node, labeling, and the mechanisms of wh-movement. In addition to overcoming problems with POP, we have attempted to clarify the exact mechanisms that are at work in POP. While POP does away with endocentricity and possibly specifiers, the core labeling algorithm results in complex derivations that require (as modeled in this paper) all of the operations/principles listed in Table 3. This raises the following question, which we leave for further research: can the POP model be further simplified, and if so how? Also, we have only examined a small set of English data, raising the issue of how well the POP model can be extended to account for other data.

We also want to point out that the proposals in this paper were implemented via a computer model (see the Appendix, in Supplementary Files 1), thus enabling others to verify the fine details of the derivations. We have used this model to produce step-by-step derivations of all target sentences. Drawing complete trees that show all relevant details of target sentences is tedious work, and when done by hand, is prone to error. That is why using a computer for this task is extremely useful. We invite other researchers to consider the merits of using computer models for syntactic research.

Supplementary Files

The supplementary files for this article can be found as follows:

- **Supplementary File 1:** Appendix. http://dx.doi.org/10.5334/gjgl.22.s1

Acknowledgements

I want to thank two anonymous reviewers for extremely helpful comments, which I think greatly improved the quality of this paper. Much thanks also to Sandiway Fong. This project is an outgrowth of joint work with Sandiway to develop a Phase Theory-based Minimalist Machine (Fong & Ginsburg, in progress), as well as of an initial computational model based on Chomsky (2013). Some of the ideas in this paper, especially those regarding feature unification and use of an unergative vUnerg were developed jointly with Sandiway for the Minimalist Machine. Trees in this paper were produced with Treedraw software developed by Sandiway. All errors are my own.

Competing Interests

The author declares that he has no competing interests.

References

Borer, Hagit. 2005a. *In name only. Structuring sense, Volume 1*. Oxford: Oxford University Press. DOI: http://dx.doi.org/10.1093/acprof:oso/9780199263905.001.0001

Borer, Hagit. 2005b. *The normal course of events: Structuring sense, Volume 2*. Oxford: Oxford University Press. DOI: http://dx.doi.org/10.1093/acprof:oso/9780199263929.001.0001

Borer, Hagit. 2013. *Taking form: Structuring sense, Volume 3*. Oxford: Oxford University Press. DOI: http://dx.doi.org/10.1093/acprof:oso/9780199263936.001.0001

Cable, Seth. 2007. *The grammar of Q: Q-particles and the nature of wh-fronting, as revealed by the wh-questions of Tlingit*. Cambridge, MA: Massachusetts Institute of Technology dissertation.

Cable, Seth. 2010. Against the existence of pied-piping: Evidence from Tlingit. *Linguistic Inquiry* 41. 563–594. DOI: http://dx.doi.org/10.1162/LING_a_00013

Chomsky, Noam. 1995. *The Minimalist Program*. Cambridge, MA: MIT Press.

Chomsky, Noam. 2000. Minimalist inquiries: The framework. In Roger Martin, David Michaels & Juan Uriagereka (eds.), *Step by step*, 89–155. Cambridge: MA: MIT Press.

Chomsky, Noam. 2001. Derivation by phase. In Michael Kenstowicz (ed.), *Ken Hale: a life in language*, 1–52. Cambridge, MA: MIT Press.

Chomsky, Noam. 2008. On phases. In Robert Freidin, Carlos Otero & Maria-Luisa Zubizaretta (eds.), *Foundational issues in linguistic theory; essays in honor of Jean-Roger Vergnaud*, 133–166. Cambridge, MA: MIT Press. DOI: http://dx.doi.org/10.7551/mitpress/9780262062787.003.0007

Chomsky, Noam. 2013. Problems of projection. *Lingua* 130. 33–49. DOI: http://dx.doi.org/10.1016/j.lingua.2012.12.003

Chomsky, Noam. 2015. Problems of projections: Extensions. In Elisa di Domenico, Cornelia Hamann & Simona Matteini (eds.), *Structures, strategies and beyond: Studies in honour of Adriana Belletti*, 1–16. Amsterdam: John Benjamins.

Embick, David & Alec Marantz. 2008. Architecture and blocking. *Linguistic Inquiry* 39. 1–53. DOI: http://dx.doi.org/10.1162/ling.2008.39.1.1

Epstein, Samuel David, Hisatusugu Kitahara & T. Daniel Seely. 2012. Structure building that can't be! In Myriam Uribe-Etxebarria & Vidal Valmala (eds.), *Ways of structure building*, 253–270. Oxford: Oxford University Press. DOI: http://dx.doi.org/10.1093/acprof:oso/9780199644933.003.0011

Epstein, Samuel David, Hisatusugu Kitahara & T. Daniel Seely. 2014. Labeling by Minimal Search: Implications for successive cyclic A-movement and the conception of the postulate "Phase". *Linguistic Inquiry* 45. 463–481. DOI: http://dx.doi.org/10.1162/LING_a_00163

Fong, Sandiway. 2014. Unification and efficient computation in the Minimalist Program. In Francis Lowenthal & Laurent Lefebvre (eds.), *Language and recursion*, 129–138. New York: Springer. DOI: http://dx.doi.org/10.1007/978-1-4614-9414-0_10

Fong, Sandiway & Jason Ginsburg. In Preparation. *The Minimalist Machine*.

Frampton, John & Sam Gutmann. 2000. Agreement is feature sharing. MS, Northeastern University.

Ginsburg, Jason. 2009. *Interrogative features*. Tucson, AZ: University of Arizona dissertation.

Hagstrom, Paul. 1998. *Decomposing questions*. Cambridge, MA: Massachusetts Institute of Technology dissertation.

Hornstein, Norbert. 1999. Movement and control. *Linguistic Inquiry* 30. 69–96. DOI: http://dx.doi.org/10.1162/002438999553968

Hornstein, Norbert. 2001. *Move! A Minimalist Theory of construal*. Malden, MA: Blackwell.

Marantz, Alec. 1997. No escape from syntax: Don't try morphological analysis in the privacy of your own lexicon. *University of Pennsylvania Working Papers in Linguistics* 4(2), 201–225.

Moro, Andrea. 2000. *Dynamic antisymmetry*. Cambridge, MA: MIT Press.

Pesetsky, David & Esther Torrego. 2001. T-to-C movement: Causes and consequences. In Michael Kenstowicz (ed.), *Ken Hale: A life in language*, 355–426. Cambridge, MA: MIT Press.

Pesetsky, David, & Esther Torrego. 2007. The syntax of valuation and the interpretability of features. In Simin Karimi, Vida Samiian & Wendy Wilkins (eds.), *Phrasal and clausal architecture: Syntactic derivation and interpretation. In honor of Joseph E. Emonds*, 262–294. Amsterdam: John Benjamins. DOI: http://dx.doi.org/10.1075/la.101.14pes

Richards, Marc. 2007a. On feature inheritance: An argument from the Phase Impenetrability Condition. *Linguistic Inquiry* 38. 563–572. DOI: http://dx.doi.org/10.1162/ling.2007.38.3.563

Richards, Marc. 2007b. Object shift, phases, and transitive expletive constructions in Germanic. In Pierre Pica, Johan Rooryck, & Jeroen Van Craenenbroeck (eds.), *Linguistics Variation Yearbook* 6, 139–159. Amsterdam: John Benjamins. DOI: http://dx.doi.org/10.1075/livy.6.07ric

Rizzi, Luigi. 2013. Cartography, criteria and labeling. Ms., Universities of Sienna and Geneva.

Sobin, Nicholas. 2014. Th/Ex, Agreement, and Case in expletive sentences. *Syntax* 17. 385–416. DOI: http://dx.doi.org/10.1111/synt.12021

How to cite this article: Ginsburg, Jason. 2016. Modeling of problems of projection: A non-countercyclic approach. *Glossa: a journal of general linguistics* 1(1): 7. 1–46, DOI: http://dx.doi.org/10.5334/gjgl.22

Published: 01 June 2016

]u[*Glossa: a journal of general linguistics* is a peer-reviewed open access journal published by Ubiquity Press.

OPEN ACCESS