Chapter 12

Un-Principled Syntax: The Derivation of Syntactic Relations

Samuel David Epstein

Nash (1963, 173–174) writes,

> [H]aving learned to reject, as delusive, the hope that theoretical premises are, or can be made, self-evident—we cannot but recognize that always our explanations are *incomplete*. Hall [(1956, 177)] attributes to Galileo and Newton the opinion that:
>
> > The explanation of phenomena at one level is the description of phenomena at a more fundamental level, . . .
>
> Complete understanding then fails by the margin of those theoretical premises which are *stipulated*, perhaps "described," but certainly not themselves explained or explicable for so long as they remain our ultimate premises.
>
> . . . Resolved to maximize our understanding, we find ourselves committed to a highly characteristic effort to minimize the *number* of theoretical premises required for explanation. Einstein [(1954, 282)] speaks of:
>
> > . . . the grand aim of all science, which is to cover the greatest possible number of empirical facts by logical deductions from the smallest number of hypotheses or axioms.
>
> Some centuries earlier Newton had expressed the same "grand aim" in the first of his Rules of Reasoning:
>
> > *We are to admit no more causes of natural things than such as are both true and sufficient to explain their appearances.*
>
> . . . Each "quality" imputed to a premised entity figures as an additional postulate. Our desire for parsimony of postulates thus evokes a search for theoretical posits having the slenderest possible qualitative endowment.

12.1 Introduction: Syntactic Relations

Alongside "syntactic feature" and "permissible syntactic feature bundle" (i.e., a possible syntactic category), perhaps the most fundamental construct postulated within syntactic theory is "syntactic relation." In fact, broadly conceived, syntactic theory is precisely a theory of relations and the elements that enter into them. For example, each of the following is considered to be a core syntactic relation:

(1) a. (Subject-verb) agreement — The dog kicks walls.
b. (Object-verb) θ-relation — The dog kicks walls.
c. (Accusative) Case relation — The dog kicks them.
d. (Reflexive) binding relation — The dog kicks herself.
e. (Passive) movement relation — (Sadly) the dog was kicked t.
f. The "is a" relation — The dog $[_{VP}[_{V}$ kicks] $[_{DP}$ walls]]. (*kicks*, a V, and *walls*, a DP, together constitute—i.e., are—a VP)

Such relations are apparently very heavily constrained. That is, we do not find empirical evidence for all of the logically possible syntactic relations; instead, at least from the perspective of unified theories (to be discussed momentarily), we find only one, or perhaps "a few," to be distinguished on a principled basis.

Given that syntactic relations are heavily constrained, the questions we confront include "What are they?" "How are they to be formally expressed?" and, more deeply, "Why do we find *these* and not any of the infinite number of other logically possible relations?" Within the framework of Chomsky 1981, 1982, there were, by hypothesis, at least two fundamental syntactic relations. The first, *government* (see also Chomsky 1986), was a unified construct, a binary relation, intended to subsume all the seemingly disparate phenomena illustrated in the English examples (1a–e) (but not (1f)). The second, "*is a*" (1f), was a relation created by the base component, upon which government relations were defined. For example, the trinary relation in (1f)—"V and DP *are a* VP"—is not a government relation.

This type of theory of syntactic relations arguably faces several conceptual (and empirical) problems:

1. *Unification.* Unification is, in a sense, precluded in that the "is a" relation is divorced from the government relation. Government relations are defined on base-generated representations already exhibiting "is a" relations. Why isn't there just one relation? And if there is indeed more than one, why are (1a–e) the cases that are unified, under government, and why is (1f), the "is a" relation, the one that is left out?
2. *Explanation.* Explanation at a certain level is lacking to the extent that the (or one of the) fundamental relations, government (Chomsky 1981, 1986)—or more recently minimal domain (Chomsky 1993), a binary relation defined on representations—is merely a definition. That is, the following question is unanswered: "Why is government as defined in (3) or minimal domain (see (38)) the fundamental syntactic relation and not any of the infinite number of other logically possible syntactically definable relations?"
3. *Primitive constructs.* Government and minimal domain are not in fact primitives; instead, each incorporates a more fundamental binary relational construct, *command.* Thus, contrary to standard assumptions, government or minimal domain is not the fundamental unexplained definition; rather, the relation *command* is. Of course, if command is to express a fundamental syntactic relation, and it remains an (unexplained) *definition*, then, in the same sense as in point 2, explanation is lacking: "Why is *this* relation, so defined, syntactically significant?"
4. *Complexity.* Government and minimal domain definitions are complex. Of course, this claim has no substance in the absence of an explicit, principled complexity metric. Hence, I will leave it to the reader's intuition that the alternative theory of syntactic relations proposed here achieves significant simplification and (I hope) does not lose (perhaps even gains) in empirical adequacy.

In this chapter, I will address each of these (closely related) problems confronting the fundamental construct "syntactic relation" as it is expressed in contemporary syntactic theories. The analysis will be couched within the Minimalist Program (Chomsky 1993, 1994), which makes the following important innovations:

- D-Structure is eliminated and, along with it, the bifurcation of the D-Structure-generating base component and the transformational component.
- The concept of generalized transformation (= Merge), arguably unifiable with the concept of singulary transformation (= Move α), as proposed by Kitahara (1994, 1995, 1997), is reinstated.

The central hypothesis I will propose here can be expressed in an informal and preliminary fashion as follows:

(2) *Preliminary hypothesis*
a. The fundamental concept "syntactic relation" (e.g., "government" or "minimal domain") is not an unexplained definition defined on representations (i.e., already built-up phrase structure representations). Rather, syntactic relations are properties of

independently motivated, simple, and minimal transformations. That is, syntactic relations are established between a syntactic category X and a syntactic category Y when (and only when) X and Y are transformationally concatenated (thereby entering into sister relations with each other) by either Merge or Move during the tree-building, iterative, universal rule application that constitutes the derivation.

b. The fundamental structure-building operation is neither Move α (Chomsky 1981, 1982) nor Affect α (Lasnik and Saito 1992) but "Concatenate X and Y, thereby forming Z."

The analysis I will propose is entirely natural in that concatenation and only concatenation establishes syntactic relations between categories. Given this hypothesis, and given that any syntactic system, by definition, requires concatenation, the fundamental construct "syntactic relation" should be deducible from (hence explained by appeal to) the independently and strongly motivated postulate "Concatenate," as simply expressed by Merge and Move in the Minimalist Program. To the extent that "syntactic relation" is indeed (contra contemporary theories of syntax) an explicable derivational construct, not a definitional (unexplained) representational notion, the four central obstacles 1–4 that confront all current syntactic explanations will be overcome.

Before proceeding, I would like to briefly place this hypothesis—that fundamental, representational, unexplained definitions can be replaced by derivational explanations—in a broader historical context. As is well known, the "rule-less," representation-based principles-and-parameters theory evolved from earlier rule-based systems. The construction specificity and language specificity of the postulated phrase structure and transformational rules represented a serious obstacle to explanatory adequacy: "How does the learner select *this* grammar with *these* rules, on the basis of exposure to degenerate data?" An entirely natural development was the gradual abandonment of rule-based grammars and the concomitant postulation of universal constraints on representations or principles (expressing the properties common to rules), which were consequently neither construction- nor language-particular entities. The residue—the language-specific properties of rules, to be fixed by experience—was ascribed the status of parameters with the hope that construction specificity would be altogether eliminated from core grammar.

Although the abandonment of rule systems and the adoption of principles (i.e., filters or well-formedness conditions on rule-generated representations, such as binding theory, Case theory, X-bar theory, and θ-theory) was an entirely natural development, there is an alternative, which I believe is reflected (perhaps only implicitly) in Chomsky 1991, 1993, 1994. Given that it was the language specificity and construction specificity of rules and not the fact that they were rules per se that apparently threatened explanatory adequacy, the alternative approach is to retain a rule-based framework, but eliminate from the rules their language-particular and construction-particular formal properties. That is, instead of universal principles, such an approach postulates universal iteratively applied rules, thereby maintaining a strongly derivational theory of syntax. The approach is like Standard Theory in that it incorporates iterative application of rules but unlike Standard Theory in that the rules are "universalized" as generalized transformation (Merge) and Move; that is, they are purged of language-specific and construction-specific properties, their apparent crosslinguistic differences being attributed, by hypothesis, to irreducible morphological variation.

In this chapter, I will argue that this strongly derivational universal-rule approach, in which iterative rule application characterizes syntactic derivations while constraints on output representations and on levels of representation (hence levels themselves) are altogether eliminated (see Chomsky 1994, 1995), exhibits vast explanatory advantages over the existing representational, rule-free, principle-based (hence representation-based) theories—at least in the domain of accounting for the central construct "syntactic relation."

12.2 Syntactic Relations in a Principle-Based Theory

In the pre-minimalist, principle-based framework of Chomsky 1981, 1986, representations (phrase structure trees) are built freely by an unconstrained (implicit) base. The output is constrained by the X-bar schema, a filter on output representations. The unifying construct "government" is a binary, unidirectional (asymmetric) syntactic relation holding between two syntactic categories in a derived representation. It is defined as follows:

(3) *Government*

a. X *governs* Y iff
   i. X m-commands Y *and*
   ii. There is no Z, Z a barrier for Y, such that Z excludes X.

b. X *m-commands* Y iff the minimal maximal projection dominating X dominates Y (see Aoun and Sportiche 1983).

c. X *excludes* Y iff no segment of X dominates Y.
d. X *dominates* Y only if every segment of X dominates Y (see Chomsky 1986, n. 10).
e. Z is a *barrier* for Y iff
   i. Z immediately dominates W, W a blocking category for Y, *or*
   ii. Z is a blocking category for Y and Z $\neq$ IP.
f. A maximal projection X *immediately dominates* a maximal projection Y iff there is no maximal projection Z such that X dominates Z and Z dominates Y.
g. Z is a *blocking category* for Y iff
   i. Z is not L-marked *and*
   ii. Z dominates Y.
h. X *L-marks* Y iff X is a lexical category that θ-governs Y.
i. X *θ-governs* Y iff
   i. X is a zero-level category *and*
   ii. X θ-marks Y *and*
   iii. X and Y are sisters.

Although such an approach offers an impressive, highly explicit, and unified analysis of a number of seemingly disparate syntactic phenomena, it arguably suffers from the four problems noted above:

1. It is not wholly unified since the "is a" relation is not a government relation.
2. "Government" is a definition; hence, it is entirely unexplained why syntactic phenomena, by hypothesis, conform to this particular relation and not to any of the infinite number of alternative syntactically definable relations.
3. "Government" is not really a primitive relation since it incorporates the more primitive relation "m-command."
4. The definition of "government" is arguably complex (but see section 12.1 regarding the unclarity of such inexplicit claims).

Following Chomsky (1993, n. 9), I will assume that m-command is not a primitive. I will however assume that c-command is. In the next section, I will show that contrary to syntactic analyses from Reinhart 1979 to Kayne 1994, c-command need not be expressed as an unexplained representational definition but can instead be expressed as a natural, explicable, derivational construct, assuming Chomsky's (1993, 1994) elimination of a distinct base component (and, along with it, the elimination of a base-generated D-Structure level of representation) and the postulation of a syntactic component in which derivations are characterized by iterative,

bottom-up application of universalized, simple, and perhaps unifiable rules, Merge and Move.

12.3 C-Command

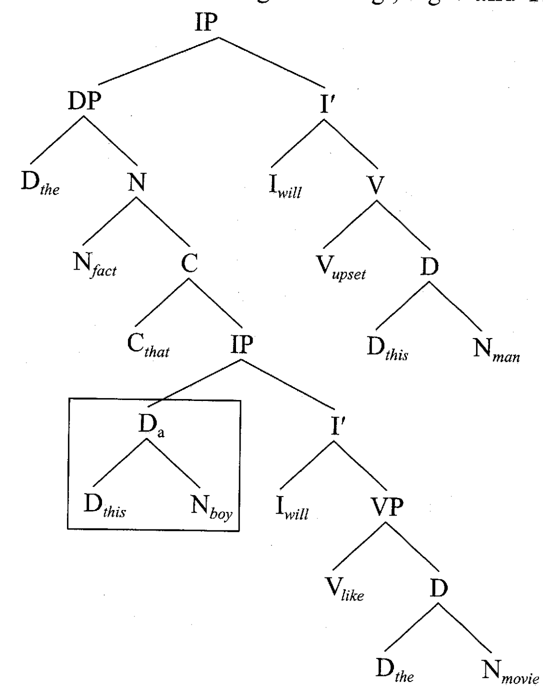
12.3.1 Representational C-Command

Consider the following representational definition of c-command (Reinhart 1979):

(4) *The representational definition of c-command*
A c-commands B iff
a. The first branching node dominating A dominates B, *and*
b. A does not dominate B, *and*
c. A $\neq$ B.

Four important properties of c-command should be noted here. First, (4) constitutes a *definition*, hence explanation is lacking. That is, we have no answer to the question, "Why is this particular binary relation syntactically significant?" As an illustration, consider (5).

(5) *A schematic illustration of c-command*
(certain irrelevant categories—e.g., AgrP and TP—are omitted)



In (5), [Spec, IP] = D_a c-commands I′, I_{will}, VP, V_{like}, D, D_{the}, N_{movie}, and no other categories. It is as if the other categories in (5) are inexplicably invisible with respect to D_a; hence, D_a enters into no relations with them. Why? Second, c-command is pervasive and fundamental, apparently playing a unifying role throughout the different subcomponents of the syntax. Third, it is persistent; that is, despite substantive changes in the theory of syntax, Reinhart's definition, proposed two decades ago, remains, by hypothesis, linguistically significant. Fourth, it is representational; that is, it is a relation defined on representation.

Thus, c-command faces at least these unanswered questions:

(6) a. Why does it exist at all? Why doesn't A enter into relations with all constituents in the tree?
b. Why is the *first* branching node relevant? Why not "The first or second or third (*n*th?) node dominating A must dominate B?"
c. Why is *branching* relevant?
d. Why doesn't A c-command the first branching node dominating A, instead c-commanding only categories dominated by the first branching node?
e. Why must A not dominate B?
f. Why must A not equal B?

I will advance the hypothesis that these properties of c-command are not accidental, but are intimately related. First, I believe c-command is fundamental, pervasive, and persistent because it is, by hypothesis, indeed a syntactically significant relation. Second, I propose that it is definitional (nonexplanatory) precisely because it has been formulated or construed as a representational relation. Third, I propose that it is in fact derivational—that is, a relation between two categories X and Y established in the course of a derivation (iterative universal-rule application) when and only when X and Y are paired (concatenated) by transformational rule (i.e., Merge or Move). When c-command is construed derivationally, the unanswered questions confronting the representational definition receive natural answers.

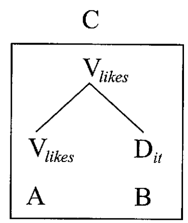
12.3.2 The Derivation of C-Command

To begin with, I will assume that

(7) a. Merge and Move (Chomsky 1993, 1994) are at least partly unifiable (as proposed in Kitahara 1993, 1994, 1995, 1997) in that each pairs (concatenates) exactly two categories, A and B, rendering them sisters immediately dominated by the same (projected) mother, C (where C = the head of A or of B (Chomsky 1994)).
b. Given (7a), there is a fundamental operation, common to or shared by Merge and Move: "Concatenate A and B, forming C (C = the head of A or of B)."

Crucially, then, each universalized transformational rule, Merge and Move, establishes a syntactic relation between two concatenated syntactic categories A and B by virtue of placing them in the "is a" relation with C, the projected category. I will also assume, with Chomsky (1994), that Merge operates bottom-up—that is, applies cyclically (cyclic application being an independently motivated universal constraint on universal-rule application)—and that Move does so as well. Consider, for example, the derivation in (8).

(8) Merging V_{likes} and D_{it} yields, informally:



The lower V_{likes} (= A) and D_{it} (= B) are, by virtue of undergoing Merge, in a relation: they are the sister constituents of a V-phrase/projection C, labeled V_{likes}.

Thus, what Merge does is create *sisters*. Crucially, A and B cannot be or become sisters without having a common mother. Conversely, if nonbranching projection is disallowed, and only binary branching is permitted, then there cannot be a mother C without exactly two daughters (the sisters A and B). In a nutshell, both the sisterhood relation and the motherhood ("is a") relation are created simultaneously, by a single application of Merge.

Viewed in terms of Standard Theory transformations, A and B (in (8), V_{likes} and D_{it}) constitute the structural description of Merge. The structural change (perhaps deducible, given the structural description; see Chomsky 1995) specifies the categorial status of the mother or output tree/set. Since the two entities in the structural description are rendered sisters (i.e., are placed in the "is a" relation to the projected (perhaps

In (5), [Spec, IP] = D_a c-commands I′, I_{will}, VP, V_{like}, D, D_{the}, N_{movie}, and no other categories. It is as if the other categories in (5) are inexplicably invisible with respect to D_a; hence, D_a enters into no relations with them. Why? Second, c-command is pervasive and fundamental, apparently playing a unifying role throughout the different subcomponents of the syntax. Third, it is persistent; that is, despite substantive changes in the theory of syntax, Reinhart's definition, proposed two decades ago, remains, by hypothesis, linguistically significant. Fourth, it is representational; that is, it is a relation defined on representation.

Thus, c-command faces at least these unanswered questions:

(6) a. Why does it exist at all? Why doesn't A enter into relations with all constituents in the tree?
b. Why is the *first* branching node relevant? Why not "The first or second or third (*n*th?) node dominating A must dominate B?"
c. Why is *branching* relevant?
d. Why doesn't A c-command the first branching node dominating A, instead c-commanding only categories dominated by the first branching node?
e. Why must A not dominate B?
f. Why must A not equal B?

I will advance the hypothesis that these properties of c-command are not accidental, but are intimately related. First, I believe c-command is fundamental, pervasive, and persistent because it is, by hypothesis, indeed a syntactically significant relation. Second, I propose that it is definitional (nonexplanatory) precisely because it has been formulated or construed as a representational relation. Third, I propose that it is in fact derivational—that is, a relation between two categories X and Y established in the course of a derivation (iterative universal-rule application) when and only when X and Y are paired (concatenated) by transformational rule (i.e., Merge or Move). When c-command is construed derivationally, the unanswered questions confronting the representational definition receive natural answers.

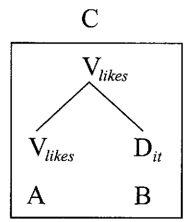
12.3.2 The Derivation of C-Command

To begin with, I will assume that

(7) a. Merge and Move (Chomsky 1993, 1994) are at least partly unifiable (as proposed in Kitahara 1993, 1994, 1995, 1997) in that each pairs (concatenates) exactly two categories, A and B, rendering them sisters immediately dominated by the same (projected) mother, C (where C = the head of A or of B (Chomsky 1994)).
b. Given (7a), there is a fundamental operation, common to or shared by Merge and Move: "Concatenate A and B, forming C (C = the head of A or of B)."

Crucially, then, each universalized transformational rule, Merge and Move, establishes a syntactic relation between two concatenated syntactic categories A and B by virtue of placing them in the "is a" relation with C, the projected category. I will also assume, with Chomsky (1994), that Merge operates bottom-up—that is, applies cyclically (cyclic application being an independently motivated universal constraint on universal-rule application)—and that Move does so as well. Consider, for example, the derivation in (8).

(8) Merging V_{likes} and D_{it} yields, informally:



The lower V_{likes} (= A) and D_{it} (= B) are, by virtue of undergoing Merge, in a relation: they are the sister constituents of a V-phrase/projection C, labeled V_{likes}.

Thus, what Merge does is create *sisters*. Crucially, A and B cannot be or become sisters without having a common mother. Conversely, if nonbranching projection is disallowed, and only binary branching is permitted, then there cannot be a mother C without exactly two daughters (the sisters A and B). In a nutshell, both the sisterhood relation and the motherhood ("is a") relation are created simultaneously, by a single application of Merge.

Viewed in terms of Standard Theory transformations, A and B (in (8), V_{likes} and D_{it}) constitute the structural description of Merge. The structural change (perhaps deducible, given the structural description; see Chomsky 1995) specifies the categorial status of the mother or output tree/set. Since the two entities in the structural description are rendered sisters (i.e., are placed in the "is a" relation to the projected (perhaps predictable) mother C, all in one application of Merge), there is no need for a representational definition of "sister" or "mother" ("is a"); these two relations are clearly expressed (and unified) within the independently motivated, universal structure-building rules themselves. Representational definitions would therefore be redundant and (being definitions) nonexplanatory.

The tree in (8) is formally represented as $\{V_{likes}, \{V_{likes}, D_{it}\}\}$. This object (set) consists of three terms:

(9) a. The entire tree/set (= C)
  b. V_{likes} (= A)
  c. D_{it} (= B)

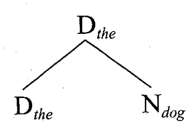
That is, following Chomsky (1994, 12; 1995), I assume:

(10) a. Definition of *term* ("constituent")
   i. K is a term of K (i.e., the entire set or tree is a term).
   ii. If L is a term of K, then the members of the members of L are terms of K.
  b. The terms in (8) are as follows:
   i. $K = \{V_{likes}, \{V_{likes}, D_{it}\}\}$ = one term
   ii. K has two members:
    member 1 = V_{likes} = "the label"
    member 2 = a two-membered set = $\{V_{likes}, D_{it}\}$
   iii. The V_{likes} and D_{it} that are each members of member 2 are thus members of a member. Therefore, each is a term.

Thus, "[t]erms correspond to nodes of the informal representations, where each node is understood to stand for the subtree of which it is the root" (Chomsky 1994, 12).

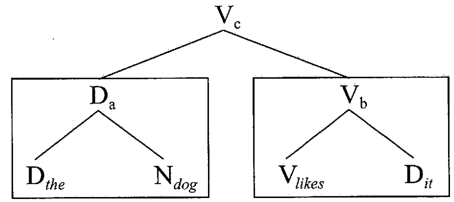
Continuing with the derivation, suppose that concurrent with the construction of (8), we construct the separate phrase marker (11) (recall that separate phrase markers may be constructed in parallel, as long as a single phrase marker results by the end of the derivation (see Collins 1997)).

(11) Merge D_{the} and N_{dog}, yielding informally:



The tree is formally represented as $\{D_{the}, \{D_{the}, N_{dog}\}\}$, similarly consisting of three terms: the entire two-membered set and each of the two categories that are members of a member of the two-membered set (D_{the} and N_{dog}). Now, having constructed the two three-membered trees in (8) and (11), suppose we merge them, yielding (12) (*a*, *b*, and *c* are purely heuristic: $D_a = D_{the}$; V_b and V_c each $= V_{likes}$).

(12)



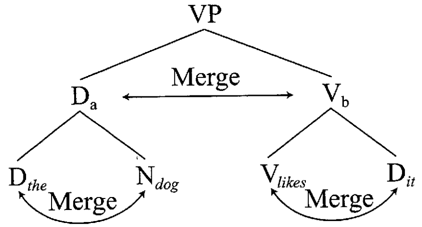
Now notice that there exists a massive redundancy. The representational definition of c-command (4) stipulates c-command relations between sisters in the derived representation (12). But sisters are precisely the objects A and B that invariably undergo Merge in building the representation. Thus:

(13) In (12):
  a. i. D_{the} representationally c-commands N_{dog}; they were merged.
   ii. N_{dog} representationally c-commands D_{the}; they were merged.
  b. i. V_{likes} representationally c-commands D_{it}; they were merged.
   ii. D_{it} representationally c-commands V_{likes}; they were merged.
  c. i. D_a representationally c-commands V_b; they were merged.
   ii. V_b representationally c-commands D_a; they were merged.
  d. V_c representationally c-commands nothing; it has not been merged with any category.
  e. In (12), the 10 binary dominance relations ("X dominates Y") are, by pure stipulation in (4b), not c-command relations; they were not merged.
  f. By pure stipulation in (4c), no category representationally c-commands itself; no category is merged with itself.

Thus, Merge—an entirely simple, natural, minimal, and independently motivated structure-building operation (i.e., transformational rule)—seems to capture representational c-command relations. In other words, if X and Y are concatenated, they enter into (what have been called) c-command relations. Consequently, it would seem that we can eliminate the stipulated, unexplained representational definition of c-command (4).

There is a problem with this suggestion, however. When Merge pairs two categories, the pairing establishes only *symmetrical* (reciprocal) c-command relations. Consider, for example, (14).

(14)

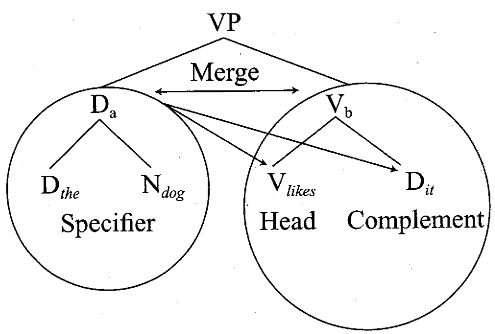


Correctly, each arrow in (14) indicates a c-command relation. But Merge does not totally subsume the representational definition of c-command, precisely because there exist c-command relations between categories that have not been merged. Thus, (15a) is true, but (15b) is false.

(15) a. If A and B were merged, then A c-commands B and B c-commands A.
b. If A c-commands B, then A and B were merged.

To see the falsity of (15b), consider (16).

(16) D_a and V_b are merged:



The specifier of V_{likes}, D_a, c-commands the head V_{likes} and the complement D_{it}, but D_a was not merged with either of them.

To solve this problem confronting the attempt to entirely deduce representational, definitional c-command from Merge, notice that although D_a was not merged with V_{likes} or with D_{it}, it was merged with V_b. But now recall that $V_b = \{V_b, \{V_{likes}, D_{it}\}\}$ ("each node is understood to stand for the subtree of which it is the root"). That is, V_b consists of three

terms: $\{V_b, \{V_{likes}, D_{it}\}\}$ (the whole V_b subtree in (16)), V_{likes}, and D_{it}. Given that a syntactic category is a set of terms (in dominance/precedence relations), we can propose the following, natural derivational definition of c-command:

(17) *Derivational c-command (preliminary version)*
X c-commands all and only the terms of the category Y with which X was merged in the course of the derivation.

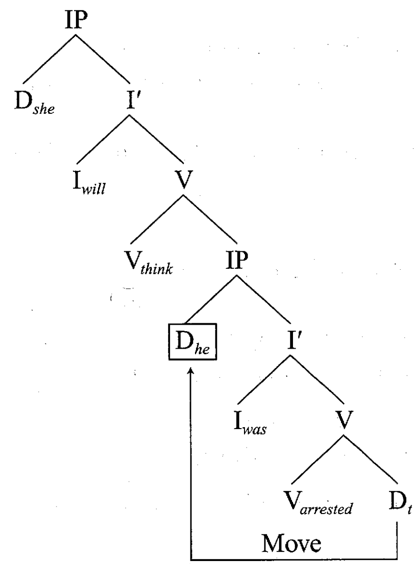
Thus, D_a ([Spec, VP]) c-commands V_b (X′) and all terms of V_b.

Now recall that Move, the other structure-building operation, also pairs/concatenates exactly two categories, projecting the head of one, and in this respect is identical to Merge. Since "is a" relations are created by Move in the same manner as they are created by Merge, we can now propose the final version of the derivational definition.

(18) *Derivational c-command (final version)*
X c-commands all and only the terms of the category Y with which X was paired *by Merge or by Move* in the course of the derivation.

Given (18), consider the case of Move in (19).

(19)



By definition (4), in (19) D_{he} representationally c-commands the five categories (subtrees) I′, I_{was}, V, $V_{arrested}$, D_t, and nothing else. This state of affairs, unexplained representationally, can be explained derivationally.

D_{he} was paired/concatenated (in this case by Move) with I′, and I′ is a five-term category/tree/set consisting precisely of I′, I_{was}, V, $V_{arrested}$, and D_t. It is entirely natural then that, since D_{he} was paired with a five-term object, and since pairing/concatenation is precisely the establishment of syntactic relations, D_{he} enters into a relation (what has hitherto been called c-command) with each of these five terms, and with nothing else.

This analysis also captures a certain (correct) asymmetry. Although D_{he} c-commands each of the five terms of I′, the converse is not true. For example, I_{was} is a term of I′, but I_{was} does not c-command D_{he}; rather, since in the course of the derivation I was paired (this time by Merge) with V, the derivational analysis rightly predicts that I_{was} c-commands each of the three terms of V (V itself, $V_{arrested}$, D_t) and nothing else.

The derivational definition of c-command (18) enables us to answer questions that the representational definition (4) did not.

Q: (Really an infinite number of questions) Why is it that X c-commands Y if and only if the *first* branching node dominating X dominates Y?
A: The first (not, e.g., the fifth, sixth, or *n*th, for *n* any positive integer) node is relevant because it is the projected node created by pairing of X and Y by Merge and Move.

Q: Why doesn't X c-command the first branching node dominating X, instead of c-commanding only the categories dominated by the first branching node?
A: Merge or Move did not pair X with the first branching node dominating X.

Q: Why is branching relevant?
A: Assuming bare phrase structure (Chomsky 1994), no category is dominated by a nonbranching node. In other words, free projection (as in Chomsky 1993) is eliminated: structure building (Merge and Move) consists of pairing, hence invariably generates binary branching.

Q: Why must X not equal Y; that is, why doesn't X c-command itself?
A: X is never paired with itself by Merge or Move.

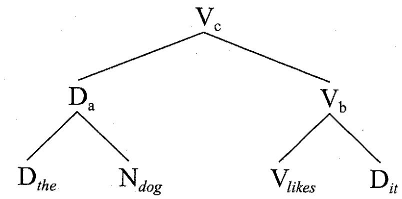
Q: Why is it that in order for X to c-command Y, X must not dominate Y?
A: If X dominates Y, X and Y were not paired by Merge or Move.

Thus, as is entirely natural, pairing/concatenating of X and Y, by application of the universal transformational rules Move and Merge, expresses syntactic relations such as c-command.

Thus far, I have provided what I believe to be strong explanatory arguments for derivational construal of c-command. However, I have provided no arguments that representational c-command is empirically inadequate. I will now present one such argument, showing that representational c-command is inconsistent with an independently motivated hypothesis (and that derivational c-command is not). (For more extensive discussion, see Epstein et al. 1998.)

Consider again a tree such as (20).

(20) V_b and V_c each = V_{likes}; D_a = D_{the}



Recall that in the input to (i.e., the structural description of) Merge, there were two categories:

(21) a. D_a = three terms:
 i. D_a itself (K is a term of K; see (10))
 ii. D_{the}
 iii. N_{dog}
b. V_b = three terms:
 i. V_b itself
 ii. V_{likes}
 iii. D_{it}

Given that D_a and V_b were merged, derivational c-command (18) entails that

(22) a. D_a c-commands V_b, V_{likes}, and D_{it}.
b. V_b c-commands D_a, D_{the}, and N_{dog}.

But assuming a relational analysis of a syntactic category's phrase structure status (Muysken 1982; Freidin 1992), in the representation (20) V_b, being neither a minimal nor a maximal projection of V, is not a term (or is an "invisible term") of (20) (Chomsky 1994). Therefore, algorithmically speaking, V_b is "stricken from the record" in (22); that is, it is not a c-commander at all. Consequently, Kayne's (1994) reanalysis of the specifier as an X′ adjunct is not required for Linear Correspondence

Axiom compatibility, exactly as Chomsky (1994) proposed. Nor is V′ (more generally, X′) c-commanded by any category. Thus, the informal representation (20) includes only the following relations, a proper subset of those in (22):

(23) a. D_a asymmetrically c-commands V_{likes} and D_{it}.
b. D_{the} symmetrically c-commands N_{dog}.
c. V_{likes} symmetrically c-commands D_{it}.

These are, by hypothesis, the desired results. Importantly, V_b (V′), although representationally invisible (i.e., not a term in the resulting representation (20)), nonetheless blocks c-command of D_a ([Spec, VP]) by V_{likes}, the head, and by D_{it}, the complement (see (23a)). But given that V′ is representationally invisible, the representational definition of c-command fails to even stipulate the apparent *fact* stated in (23a). That is, neither V_{likes} nor D_{it} is a term of some other visible term that excludes D_a ([Spec, VP]) in the resulting representation (20). By contrast, since V_{likes} and D_{it} were merged with each other, derivational c-command (18) entails that they c-command each other and nothing else. Notice that at one derivational point, V_{likes} and D_{it} were members of a term, V_b, which was a maximal term (V_{max}) that excluded [Spec, VP] immediately after merging V_{likes} and D_{it}. However, given the invisibility of X′, in the resulting representation neither V_{likes} nor D_{it} is a member of a term (other than itself) that excludes D_a ([Spec, VP]); that is, there is no (visible) node (term) that dominates V_{likes} and D_{it} and also excludes D_a ([Spec, VP]). This suggests that the derivational construal of c-command proposed here is not only natural and explanatory but also empirically preferable to the extent that the representational definition wrongly predicts that categories immediately dominated by a representationally invisible single-bar projection (e.g., the complement) c-command the specifier and (worse yet) all members of the specifier.

12.3.3 Discussion

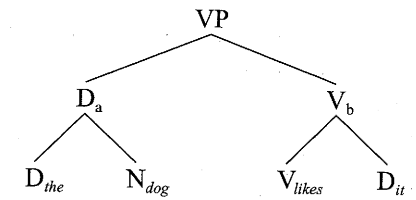
The derivational definition of c-command (18) eliminates massive redundancy (see (13)), provides principled answers to an infinite number of unanswered questions confronting the definition of representational c-command, and overcomes empirical inadequacies resulting from the interaction of the X′-invisibility hypothesis (Chomsky 1993) and representational c-command (Reinhart 1979).

Moreover, the derivational definition is an entirely natural subcase of a more general hypothesis (explored below): all syntactic relations are formally expressed by the operation "Concatenate A and B (the structural description), forming C (the structural change)" common to both structure-building operations (transformational rules), Merge and Move. Thus, Merge and Move establish relations, including "is a" and c-command, by virtue of concatenating categories. Nonetheless, despite its significant advantages over representational c-command and despite its being so natural, the derivational definition is just that: a definition. It (albeit naturally) asserts that X enters into c-command relations with all and only the terms of the category with which X is transformationally concatenated. But it still does not answer at least one very deep question: "Why does c-command exist at all? That is, why doesn't a category X simply enter into relations with *all* constituents in the tree?" I now turn to this question.

12.3.4 Toward Deducing the Derivational Definition of C-Command

First, consider the case of two categories neither of which c-commands the other, illustrated in (24).

(24) V_b and VP each = V_{likes}



Here, neither D_{the} nor D_{it} c-commands the other, illustrating the generalization that members of the specifier do not c-command members of X′ and members of X′ do not c-command members of the specifier. The first conjunct of this generalization is illustrated by, for example, the binding violation in (25).

(25) *[$_{Spec}$ this picture of *John*] [X′ upsets *himself*]

The derivational definition (18) correctly entails that *John* fails to c-command *himself* in (25). But the nonexistence of such c-command relations is, I think, deducible. Consider what I will call the *First Law*: The largest syntactic object is the single phrase structure tree. Interestingly, this hypothesis is so fundamental that it is usually left entirely implicit. The standard (i.e., representational) construal can be stated as follows:

(26) *The First Law* (representationally *construed*)
A term (tree/category/constituent) T_1 can enter into a syntactic relation with a term T_2 only if there is at least one term T_3 of which both T_1 and T_2 are member terms.

Informally, by the most fundamental definition of "syntax," there are no syntactic relations that hold between trees. In other words, the laws of syntax are intratree laws; X and Y can enter into syntactic relations only if they are both in the same tree. In (24), the Merge-derived representation, there is indeed a tree (the entire tree in (24)) such that D_{the} (a member of the specifier) and D_{it} (the complement) are both in it. But as shown in (12), derivationally prior to cyclic Merge, D_a (the specifier tree) and V_b (the X′ tree) were two unconnected trees; hence, no syntactic relation, including c-command, can hold between their members (the trees themselves can enter into a relation later, if they are merged together). Generally, there can be no relations between members of two unconnected trees.

To capture this, I reformulate the implicit First Law as a derivational law, not a representational one.

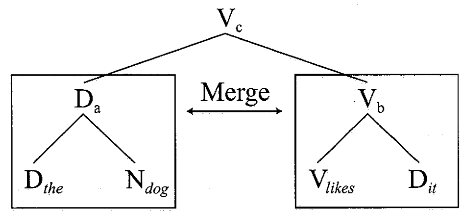
(27) *The First Law (*derivationally *construed)*
T_1 can enter into c-command (perhaps, more generally, syntactic) relations with T_2 only if there exists no derivational point at which
a. T_1 is a term of K_1 ($K_1 \neq T_1$), *and*
b. T_2 is a term of K_2 ($K_2 \neq T_2$), *and*
c. There is no K_3 such that K_1 and K_2 are both terms of K_3.

Informally stated: No relations hold between members of two trees that were unconnected at any point in the derivation. (For a formal explication of the First Law, making the intuition presented here explicit, see Groat 1997; Epstein et al. 1998, chap. 6.) Assuming Cyclicity (a universal constraint on universal-rule application), deducible for Move as hypothesized by Kitahara (1993, 1994, 1995), in the derivation of (24) there was necessarily a point at which D_{the} was a member of D_a ([Spec, VP]) and D_{it} was a member of V_b (X′) but there did not yet exist a tree containing both the branching D_a tree and the V_b tree. It follows from the derivational construal of the First Law that there is no relation between D_{the} and D_{it}. More generally, there are no relations between members of the specifier and members of X′. We thus at this point partially derive

fundamental syntactic relations like c-command and entirely derive the nonexistence of an infinite number of logically possible but apparently nonexistent syntactic relations, each of which is representationally definable (e.g., the relation *from* X *to* X's great-great-great(. . .)aunt). We do so with no stipulations, no technicalia, nothing ad hoc—only by appeal to the First Law, derivationally construed.

Notice, incidentally, that in (28) the two merged trees, D_a and V_b themselves, *can* enter into syntactic relations even though at one derivational point they were unconnected. That is, (27) entails that since neither is a member of a term/tree other than itself (i.e., each equals a root node), neither has undergone Merge or Move. Hence, like a lexical entry, each is not yet a participant in syntactic relations.

(28)



To summarize, for two nodes (trees/terms/categories) X and Y, neither of which c-commands the other, we do not need to stipulate representational c-command (4) to account for the fact that this relation does not hold. In fact, we do not even need to appeal to the far more natural (redundancy-eliminating, X′-invisibility-consistent) derivational definition of c-command (18). The derivational construal of the First Law is sufficient: no syntactic relations hold between X and Y if they were, at any derivational point, members of two unconnected trees.

As a simple illustration, again consider (25), repeated here as (29).

(29) *$[_{\text{Spec}(=D_a)}$ this picture of *John*$]$ $[_{X'(=V_b)}$ upsets *himself*$]$

This type of binding phenomenon is now easily accounted for. A reflexive requires an antecedent of a particular morphosyntactic type (by hypothesis, an irreducible lexical property). "To have an antecedent" is "to enter into a syntactic relation." However, the First Law, derivationally construed, precludes the reflexive from entering into any syntactic relation with the only morphosyntactically possible candidate, *John*, since, given cyclic Merge, there existed a point in the derivation at which *John* was a member of D_a ([Spec, VP]), *himself* was a member of V_b (X′), and D_a and V_b were unconnected trees.

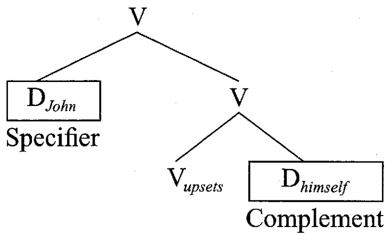
This completes the discussion of deducing those aspects of the derivational definition of c-command that pertain to two categories X and Y, neither of which c-commands the other.

Next, consider the case of asymmetric c-command, illustrated by (30)—by contrast to (29), a grammatical sentence.

(30) $[_{\text{Spec}}$ *John*$]$, $[_{X'}$ upsets *himself*$]$

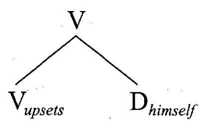
Here, X c-commands Y, but Y does not c-command X, as shown in the tree representation (31), where the specifier representationally c-commands the complement but not conversely.

(31)

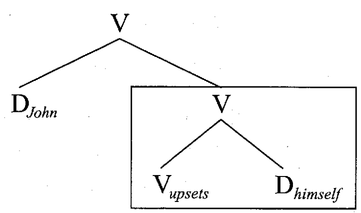


The generalization to be accounted for is that the specifier asymmetrically c-commands the complement. Given cyclic Merge, the derivation of (31) is as shown in (32).

(32) a. *First application of Merge*



b. *Second application of Merge*



Notice that D_{John} ([Spec, V]) was never a member of a tree that did not contain $D_{himself}$; in other words, there never were two unconnected trees, one containing *John* and the other *himself*. Rather, the second application of Merge pairs/concatenates D_{John} itself (a member of the numeration) with a tree containing $D_{himself}$. Thus, correctly, the First Law allows (i.e., does not block) a c-command relation from *John* (T_1 of (27)) to *himself* (T_2 of (27)).

In fact, since there never were two unconnected trees in this derivation, the First Law, a relationship blocker, is altogether inapplicable here. Rather, the first application of Merge merges two members of the numeration (V_{likes} and $D_{himself}$), forming $\{V_{likes}, \{V_{likes}, D_{himself}\}\}$, after which the second application of Merge merges yet another element of the numeration, D_{John} (not a set/tree) with this object, yielding (33).

(33) $\{V_{likes}, \{D_{John}, \{V_{likes}, \{V_{likes}, D_{himself}\}\}\}\}$

Since the First Law is inapplicable, a problem arises: *all* relations are now allowed—not only the empirically supported (c-command) relation from the specifier to the complement, but also, incorrectly, a c-command relation from the complement to the specifier. That is, in the absence of any supplementary constraints (relationship blockers), the inapplicability of the First Law allows the complement to c-command the specifier.

As a possible solution, recall that in the Minimalist Program all concatenation/pairing is performed by either Merge or Move. As claimed above, Merge and Move express syntactic relations, including the "is a" relation. Now, if the universal rules Merge and Move are the sole relationship establishers, and in addition apply cyclically, it is altogether natural, if not necessary, that a relation between X and Y is established exactly at the derivational point at which X and Y are concatenated. As a result, complements (and members of complements) never bear any relation to (e.g., never c-command) specifiers, because (a) when a complement (e.g., $D_{himself}$ in (32a)) is transformationally introduced, the specifier does not yet exist, and (b) an entity X can never bear a relation to a nonexistent entity (derivational preexistence).

Crucially, then, the matter of "timing" is the issue at hand; when a category X undergoes Merge/Move, it comes into a relation with everything *in the tree with which it is concatenated*. If a category Y isn't yet in the tree, the relation from X to Y does not arise. Hence, the asymmetry of the relation parallels the asymmetry of the iterative derivational procedure.

Thus, derivational c-command—and perhaps more generally the fundamental concept "syntactic relation"—appears to be deducible by appeal only to

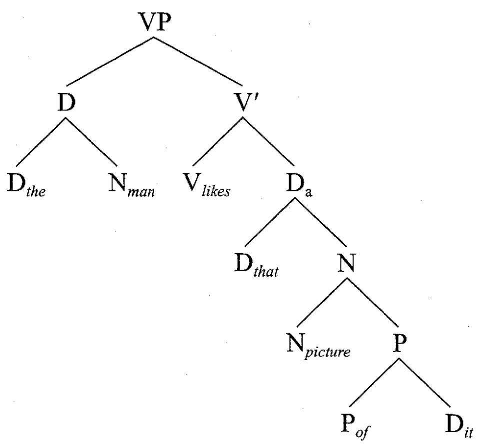
- the independently motivated, quite simple, formal properties of two (perhaps unifiable) universalized transformational rules,
- these rules' universalized, similarly simple, and perhaps explicable mode of cyclic application, and
- the fundamental, perhaps irreducible First Law, derivationally construed.

In sections 12.4 and 12.5, I propose a derivational approach to two other apparently fundamental relations, the head-complement and specifier-head relations.

12.4 The Head-Complement Relation

To begin exploring a derivational approach to the head-complement relation, consider (34).

(34)



D_a is a category consisting of the following seven terms:

(35) a. The D_a tree/set itself
b. The branching N tree/set
c. The branching P tree/set
d. D_{that}
e. $N_{picture}$
f. P_{of}
g. D_{it}

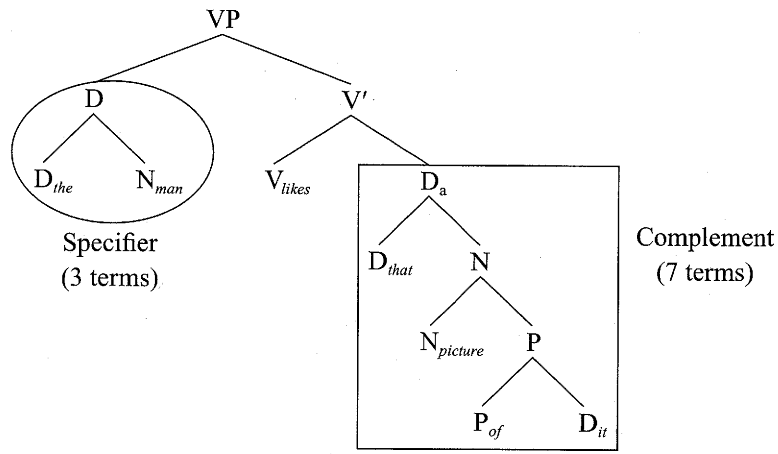
In the derivation of (34), D_a was paired with V_{likes}. V_{likes} therefore c-commands all seven terms of D_a; it enters into relations with (c-commands) nothing else, since nothing else existed when the merger took place.

In fact, if a syntactic category/tree/term is in part defined as a set of terms (in dominance/precedence relations), then the theory predicts that there should exist two types of relations (using (34) as an illustration):

(36) a. A relation between V_{likes} and each of the seven terms of D_a, including D_a itself (c-command), *and*
b. A relation between V_{likes} and D_a, the seven-term tree itself (the head-complement relation).

D_a itself is special among the seven terms that constitute D_a, since V_{likes} was paired with D_a itself (i.e., V_{likes} and D_a constituted the structural description of Merge). This completely natural analysis, couched in derivational/transformational terms, captures part of the representational definition of minimal domain proposed in Chomsky 1993. Consider the "enriched" representation of (34) in (37).

(37)



In order to account for (among other things) the head-complement relation, Chomsky (1993) offers the following definitions:

(38) *A representational definition of (specifier-head and) head-complement relations*
a. The *domain* of a head α = the set of nodes contained in Max(α) that are distinct from and do not contain α.
b. *Max(α)* = the least full category maximal projection dominating α. (In (37), for $\alpha = V_{likes}$, Max(α) = the 10-member set consisting of all categories in the circle and all categories in the square.)

c. The *complement domain* of a head α = the subset of the domain reflexively dominated by the complement of the construction. (In (37), the complement domain = the 7 terms in the square that constitute D_a.)

d. The *minimal complement domain* of a head α = all members of the complement domain that are not dominated by a member of the complement domain. (In (37), the minimal complement domain = D_a itself.)

Like its predecessor, the representational definition of government in (3), this representational definition is just that, a definition. Hence, it is not explanatory, and we still lack answers for the questions "Why is the complement domain of a head α as defined in (38c) significant?" "Why is the minimal complement domain of a head α as defined in (38d) significant?" and "Why are these and not any of the infinite number of other logically possible, syntactically definable relations linguistically significant?"

By contrast, the derivational approach reveals the fundamental nature of the head-complement relation. The syntax—more specifically, Merge and Move—establishes syntactic relations by pairing (two) categories. Derivationally, V_{likes} in (37) was paired with D_a, a seven-term category. Thus, it is entirely natural, if not an inherent property of the concatenative system, that

(39) a. V_{likes} bears a relation to D_a itself, namely, the head-complement relation. Thus, the representational (nonexplanatory) definition (38d) is unnecessary.

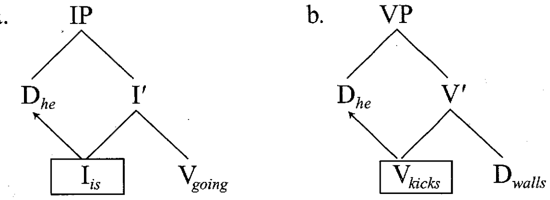
b. V_{likes} bears a relation to each member of D_a (= the complement domain "unminimized" as defined in (38c)) since these members constitute D_a. This is the relation that has been called c-command.

c. The converse of (b) does not hold. That is, correctly, it is not the case that each member of D_a bears a relation to V_{likes}; certain members of D_a underwent pairing prior to the syntactic introduction of V_{likes}, thereby permanently fixing their derivationally established relations.

12.5 The Specifier-Head Relation

To begin exploring a derivational approach to the specifier-head relation, consider (40).

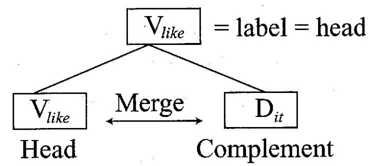
(40) a. b.



In (40a), I (the head) is assumed to check agreement and nominative Case on [Spec, IP].[1] In (40b), V assigns the agent θ-role to [Spec, VP]. The hypothesized generalization is thus that Case and agreement (and external θ-role) can be (perhaps "can only be") assigned from the head to the specifier. However, a problem has long confronted the expression of this relation: the head does not c-command the specifier, given the "first branching node" definition (4). To solve this problem, the notion of m-command was developed (Aoun and Sportiche 1983). Under the derivational analysis proposed here, the specifier c-commands the head, but the head, having been cyclically merged with the complement, is created prior to and in the absence of the specifier. Therefore, the head bears a relation only to the complement and members of the complement; that is, there is no relation from the head to the specifier.

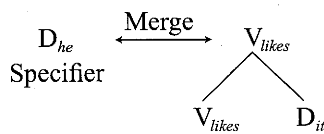
There is at least one possible solution to this apparent problem. As illustrated in (41), when two categories A and B are merged, they form a new category whose label is identical to the head of either A or B ("... Merge ... is asymmetric, projecting one of the objects to which it applies, its head becoming the label of the complex formed"; Chomsky 1994, 11).

(41)



This reflects a more general and heavily restricted hypothesis concerning the inventory of syntactic entities: "There are ... only lexical elements and sets constructed from them" (Chomsky 1994, 27). Therefore, when [Spec, VP] is paired with the entire tree/set in (41), as shown in (42), it is paired with (thus, the structural description contains) a category "head-labeled" V_{likes}.

(42)



Thus, the specifier is indeed merged with a (complex) category bearing the morphological features (label) of the head V_{likes}. If this analysis is maintainable, the specifier-head relation can also be captured as a relation established by Merge/Move—a result that would represent a clear advance over nonunified, nonexplanatory theories invoking not only a representational definition of c-command but also a representational definition of m-command (postulated precisely to capture the relation from head to specifier, inexpressible as a representationally defined c-command relation).[2]

12.6 Summary and Discussion

In this chapter, I have proposed a syntactic theory in which arguably (at least some of) the most fundamental syntactic relations posited—including c-command, "is a," specifier-head, and head-complement—are not formally expressed as unexplained representational definitions. I have proposed instead that such syntactic relations are derivational constructs expressed by the formally simple ("virtually conceptually necessary"; Chomsky 1994) and unified (Kitahara 1993, 1994, 1995, 1997) universalized transformational rules Merge and Move (each motivated on entirely independent grounds in Chomsky 1993).

This theory of syntactic relations, seeking to eliminate central representational definitions such as "government," "minimal domain," and "c-command," is entirely natural and, I think, explanatory. Concatenation operations are by (minimal) hypothesis a necessary part of the syntax; that is, a concatenative procedure (the application of which, by hypothesis, yields representations of sentences) must exist. By contrast, it is not the case that, in the same sense, principles (i.e., filters or well-formedness conditions on representation) must exist.

The question I have investigated here is thus, "Are the simple, independently motivated, virtually conceptually necessary, structure-building operations themselves—specifically, the universalized transformational rules Merge and Move, iteratively applied in conformity with the cycle—sufficient to capture fundamental syntactic relations?" The tentative answer is that they seem to be. If they are, a theory of syntax that expresses this will attain a much more unified, nonredundant, conceptually simple, and correspondingly explanatory account of the most fundamental syntactic construct, "syntactic relation," known in advance of experience by virtue of the human biological endowment for grammar formation.

Notes

This is a revised version of a draft originally written in the summer of 1994. Portions of this material were presented at the Harvard University Linguistics Department Forum in Synchronic Linguistic Theory in December 1994. I thank the members of that audience for very helpful discussion, in particular Naoki Fukui, Masatoshi Koizumi, and Ken Wexler. A later version was presented in April 1995 at the Linguistics Department at the University of Maryland. I thank members of that department as well for their hospitality and for very insightful comments, especially Norbert Hornstein, Juan Carlos Castillo, and Jairo Nunes. I am especially grateful to the following people for extensive discussion of the ideas presented here: Robert Berwick, Maggie Browning, Noam Chomsky, Robert Frank, Robert Freidin, Günther Grewendorf, Erich Groat, Sam Gutmann, Hisatsugu Kitahara, David Lieb, Elaine McNulty, Joachim Sabel, Esther Torrego, and Larry Wilson. I am also particularly indebted to Suzanne Flynn and to Höskuldur Thráinsson for their help during this project. Finally, I also thank Matthew Murphy, Elizabeth Pyatt, and Steve Peter for indispensable editorial assistance.

A modified version of this work appears as chapter 1 of Epstein et al. 1998. I gratefully acknowledge Oxford University Press and in particular Peter Ohlin for permission to publish this material here.

1. Here, for the purposes of illustration, I assume a pre–Pollock 1989 unsplit I. In fact, the unsplit I may not be simply illustrative; as Thráinsson (1994) argues, it may be empirically correct for English. By contrast, Icelandic would display a truly split I, $Agrs^0$ and T^0 (see Jonas and Bobaljik 1993; Bobaljik and Jonas 1996; Bobaljik and Thráinsson 1998).

2. For a different analysis of specifier-head relations, see Epstein et al. 1998.

References

Aoun, Joseph, and Dominique Sportiche. 1983. On the formal theory of government. *The Linguistic Review* 2, 211–235.

Bobaljik, Jonathan David, and Dianne Jonas. 1996. Subject positions and the roles of TP. *Linguistic Inquiry* 27, 195–236.

Bobaljik, Jonathan David, and Höskuldur Thráinsson. 1998. Two heads aren't always better than one. *Syntax* 1, 37–71.

Chomsky, Noam. 1981. *Lectures on government and binding*. Dordrecht: Foris.

Chomsky, Noam. 1982. *Some concepts and consequences of the theory of government and binding*. Cambridge, Mass.: MIT Press.

Chomsky, Noam. 1986. *Barriers*. Cambridge, Mass.: MIT Press.

Chomsky, Noam. 1991. Some notes on economy of derivation and representation. In *Principles and parameters in comparative grammar*, ed. Robert Freidin. Cambridge, Mass.: MIT Press. [Reprinted in *The Minimalist Program*, Noam Chomsky. Cambridge, Mass.: MIT Press, 1995.]

Chomsky, Noam. 1993. A minimalist program for linguistic theory. In *The view from Building 20: Essays in linguistics in honor of Sylvain Bromberger*, eds. Kenneth Hale and Samuel Jay Keyser. Cambridge, Mass.: MIT Press. [Reprinted in *The Minimalist Program*, Noam Chomsky. Cambridge, Mass.: MIT Press, 1995.]

Chomsky, Noam. 1994. Bare phrase structure. (MIT Occasional Papers in Linguistics 5.) MITWPL, Department of Linguistics and Philosophy, MIT, Cambridge, Mass. [Published in *Evolution and revolution in linguistic theory: Essays in honor of Carlos Otero*, eds. Héctor Campos and Paula Kempchinsky. Washington, D.C.: Georgetown University Press, 1995, and *Government and Binding Theory and the Minimalist Program*, ed. Gert Webelhuth. Oxford: Blackwell, 1995.]

Chomsky, Noam. 1995. Categories and transformations. In *The Minimalist Program*, 219–394. Cambridge, Mass.: MIT Press.

Collins, Chris. 1997. *Local economy*. Cambridge, Mass.: MIT Press.

Einstein, Albert. 1954. *Ideas and opinions*. New York: Crown.

Epstein, Samuel David, Erich Groat, Ruriko Kawashima, and Hisatsugu Kitahara. 1998. *A derivational approach to syntactic relations*. Oxford: Oxford University Press.

Freidin, Robert. 1992. *Foundations of generative syntax*. Cambridge, Mass.: MIT Press.

Groat, Erich. 1997. A derivational program for syntactic theory. Doctoral dissertation, Harvard University, Cambridge, Mass.

Hall, Alfred R. 1956. *The scientific revolution*. London: Longmans, Green.

Jonas, Dianne, and Jonathan David Bobaljik. 1993. Specs for subjects. In *Papers on Case and agreement I*, eds. Jonathan David Bobaljik and Colin Phillips. (MIT Working Papers in Linguistics 18.) MITWPL, Department of Linguistics and Philosophy, MIT, Cambridge, Mass.

Kayne, Richard. 1994. *The antisymmetry of syntax*. Cambridge, Mass.: MIT Press.

Kitahara, Hisatsugu. 1993. Deducing strict cyclicity from principles of derivational economy. Paper presented at the 16th GLOW Colloquium, Lund.

Kitahara, Hisatsugu. 1994. Target α: A unified theory of movement and structure-building. Doctoral dissertation, Harvard University, Cambridge, Mass.

Kitahara, Hisatsugu. 1995. Target α: Deducing strict cyclicity from derivational economy. *Linguistic Inquiry* 26, 47–77.

Kitahara, Hisatsugu. 1997. *Elementary operations and optimal derivations*. Cambridge, Mass.: MIT Press.

Lasnik, Howard, and Mamoru Saito. 1992. *Move α: Conditions on its application and output*. Cambridge, Mass.: MIT Press.

Muysken, Pieter. 1982. Parameterizing the notion "head." *Journal of Linguistic Research* 2, 57–75.

Nash, Leonard K. 1963. *The nature of the natural sciences*. Boston: Little, Brown.

Pollock, Jean-Yves. 1989. Verb movement, Universal Grammar, and the structure of IP. *Linguistic Inquiry* 20, 365–424.

Reinhart, Tanya. 1979. The syntactic domain for semantic rules. In *Formal semantics and pragmatics for natural languages*, eds. F. Guenther and S. J. Schmidt. Dordrecht: Reidel.

Thráinsson, Höskuldur. 1994. On the (non-)universality of functional categories. In *Minimal ideas*, eds. Werner Abraham, Samuel David Epstein, Höskuldur Thráinsson, and C. Jan-Wouter Zwart. Amsterdam: John Benjamins.