Chapter 10

Rethinking phrase structure

Howard Lasnik
University of Maryland at College Park

Zach Stone
University of Maryland at College Park

We investigate structural properties of two set-theoretic models of phrase structure, namely the phrase markers of LSLT and bare phrase structure. We demonstrate that neither set-theoretic model has a nice notion of "substructure" which is well-behaved with respect to the extension condition. We compare these with graph- and order-theoretic representations which have well-behaved structure-preserving maps for characterizing both the extension condition and the operation Agree.

1 Introduction

We review two models of phrase structure in Generative Grammar and survey their structural properties with respect to substructures and isomorphism. We especially look at how these structural notions bear on the extension condition. Specifically, we show that neither formal representation captures a sufficiently general form of the extension condition, while the correct properties are captured straightforwardly both by graph- and order-theoretic representations.

We use standard set-theoretic notation: we sometimes indicate a set by writing its elements in braces $A = \{a_i\}_{i \in I}$; we use the symbol $A \subset B$ to represent that every element of A is an element of B, called a (potentially improper) subset; we use $A \cong B$ to indicate that there is some bijection between the sets; we use $A \cup B$ to represent the union of two sets; we use A^* to represent the set of all *words*, or

Howard Lasnik & Zach Stone. 2020. Rethinking phrase structure. In András Bárány, Theresa Biberauer, Jamie Douglas & Sten Vikner (eds.), *Syntactic architecture and its consequences II: Between syntax and morphology*, 191–206. Berlin: Language Science Press. DOI: 10.5281/zenodo.4280645

strings of finite length spelled from symbols of A; we represent a set-function $f : A \rightarrow B$, or sometimes just $A \rightarrow B$.

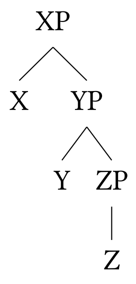
We discuss substructures and isomorphism somewhat informally, though all forms of them discussed can be made precise in the language of model theory or category theory.

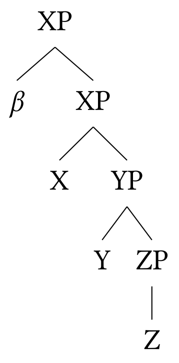
2 Phrase markers and reduced phrase markers

Lasnik (2006) briefly points out an issue that arises with respect to the extension condition (EC), the Minimalist version of the principle of the cycle proposed by Chomsky (1993), or, more precisely, the deduction of it by Chomsky (2000). Chomsky (1993: 22) formulated EC as follows:

(1) GT [generalized transformation] and Move α extend K to K′ which includes K as a proper part.

The Chomsky (2000) rationale for EC is that derivations conform to a condition demanding that there be no tampering by a transformation with already existing structure. If an item is newly attached at the "top" of a tree, the former tree is assumed to be completely preserved as a sub-tree by external merge, and also by internal merge on the copy theory of movement. Here's a simplified toy illustration. Start with the tree in (2).

(2)


(3)


Now suppose β is adjoined to XP in accord with (1). The resulting tree is (3), which clearly includes (2) as a sub-tree, the intended consequence.

But now consider these structures in terms of their set-theoretic representations, for example, as in LSLT (Chomsky 1975 [1955]). The picture in (2) stands for the actual object in (4), a set of strings:

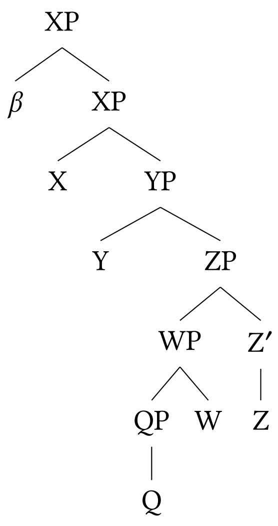
(4) {XP, X YP, X Y ZP, X Y Z}

And the picture in (3) stands for the actual derived object in (5):

(5) {XP, *β* XP, *β* X YP, *β* X Y ZP, *β* X Y Z}

Notice that (4) is in no respect a sub-object, i.e., a subset, of (5). And this is not because of any special property of the example chosen. It is invariably true that if we adjoin something to the "top" of an LSLT-style phrase marker (PM), the resulting set is never a superset of the original. That is, we have dramatically "tampered" with the original set: It is gone.

It is important to realize that the same conclusion follows on any "purely" set theoretic implementation of syntactic theory. One other such implementation is that of Lasnik & Kupin (1977). In that framework as in that of LSLT, a PM is a set of strings. The difference is that for L&K the PM consists entirely of the terminal string and "monostrings" (strings comprised of exactly one non-terminal symbol surrounded by any number of terminal symbols). L&K called their PMs reduced phrase markers (RPMs). To see that the same conclusion outlined above happens with RPMs, we need to slightly complicate the example discussed, since there, it turns out that the PM and RPM are the same. So consider the slightly more complex tree in (6):

(6)


The initial RPM is (7):

(7) {XP, X YP, X Y ZP, X Y WP Z, X Y QP W Z, X Y Q W Z′, X Y Q Z}

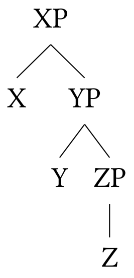
And the derived RPM is (8):

(8) {XP, *β* XP, *β* X YP, *β* X Y ZP, *β* X Y WP Z, *β* X Y QP W Z, *β* X Y Q W Z′, *β* X Y Q W Z}

Once again, the initial set is not a subset of the derived set. In fact, as with the LSLT PMs, there is no obvious simple set-theoretic relation at all between them.

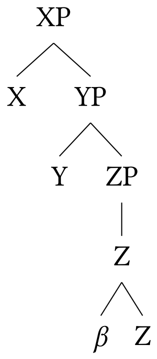
This is a special case of a more pervasive limitation of such purely set-theoretic formalizations: constituents are never sub-structures (subsets in this instance), nor are many core syntactic configurations, such as the template for a specifier.

Surprisingly, attaching at the very "bottom" does yield a superset of the initial set, the exact opposite of the evidently desired result. We illustrate this beginning with the simple structure in (2), repeated here, followed by the RPM (which, as noted earlier, is identical to the LSLT PM in this case):

(9)


(10) {XP, X YP, X Y ZP, X Y Z}

This time, adjoin *β* at the bottom, in extreme violation of EC:

(11)


The new set is (12):

(12) {XP, X YP, X Y ZP, X Y Z, X Y *β* Z}

But surprisingly this time the original object is not tampered with as (10) ⊂ (12). It is safe to conclude, then, that if Chomsky's deduction of EC is to be maintained, neither classic set-theoretic formalization of phrase structure is appropriate.

In summary, while producing the "wrong" result, RPMs have a well-defined notion of substructure. For example, (10) ⊂ (12) is a subset relation, and the defining relations of an RPM – precedence and dominance – are "preserved" by this inclusion (for example, the monostring X YP dominates the monostring X Y ZP in (10), as does the corresponding monostring in (12)).

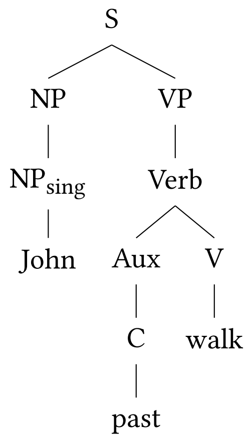
There is also a clear notion of *isomorphism* between RPMs, which will be important in §3.2. Roughly, if N and M are two sets of nonterminals and T and S sets of terminals, a pair of bijections $f : N \to M$ and $g : T \to S$ extends to a bijection between sets of strings $(f + g)^* : (N \cup T)^* \to (M \cup S)^*$ (replacing each nonterminal symbol A in a string from $(N \cup T)^*$ with $f(A)$ and each terminal symbol t with $g(t)$) and hence between monostrings. Given such bijections, we can compare RPMs F and G consisting of monostrings from $(N \cup T)^*$ and $(M \cup S)^*$, respectively, by using the bijection $(f + g)^*$ restricted to $F \subset (N \cup T)^*$ and $G \subset (M \cup S)^*$ (if possible). We could say that two RPMs F and G over (N, T) and (M, S) respectively are isomorphic if we can rename monostrings from F as monostrings in G along the bijection, and vice-versa (using the inverse of $(f + g)^*$ restricted to $G \to F$), extending to a bijection $F \cong G$, such that two monostrings ϕ and ψ in F are in a precedence or dominance relation exactly when the corresponding monostrings in G are.

Before proceeding, we note in passing that it is not only the case that in the LSLT model, attachment at the top does not "preserve structure", but also that attachment at the top is literally impossible, at least for a transformation. Transformations in that framework consist of a structural analysis (SA) and a SC (structural change). The former determines whether the T is applicable to a particular PM, while the latter indicates the operation to be performed. An SA is a sequence of "terms", each term a (string) variable, a constant (i.e., a syntactic symbol), or a linear combination of any of the preceding. Consider Chomsky's auxiliary transformation "affix hopping" as presented by Chomsky (1957). The following is one of a family of 20 SAs embodied by the T:

(13) X – *past* – V – Y

Applicability is determined by comparing the SA with the members of the set to establish satisfaction. Any string satisfies a variable, while a constant is satisfied only by that very symbol. The T with SA in (13) is applicable to the PM pictorially represented in (14).

(14)



The PM is in (15).

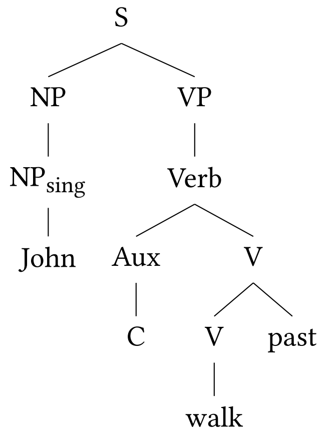
(15) {S, NP VP, NP Verb, NP Aux V, NP Aux walk, NP C V, NP C walk, NP past V, NP past walk, NP$_{sing}$ VP, NP$_{sing}$ Verb, NP$_{sing}$ Aux V, NP$_{sing}$ Aux walk, NP$_{sing}$ C V, NP$_{sing}$ C walk, NP$_{sing}$ past V, NP$_{sing}$ past walk, John VP, John Verb, John Aux V, John Aux walk, John C V, John C walk, John past V, John past walk}

In this case, applicability of the transformation is established by any of 3 members of the set:

(16) NP past V NP$_{sing}$ past V John past V

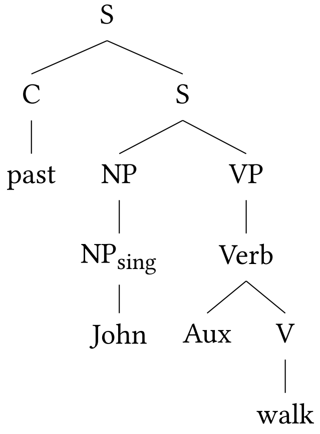
Notice that every member of any PM has symbols in a linear order; every pair of symbols in a member are in the precedence relation. Thus, the symbols in any SA are likewise necessarily in a linear order. Thus, a symbol can adjoin to one that follows it (as in affix hopping, where past will adjoin to V), or to one that precedes it. The result of the operation is in (17).

(17)



An operation that would adjoin a symbol to a dominating symbol is literally unstatable. But any singular movement T satisfying the extension condition would have to do exactly this. Suppose, for example, we wanted to apply a C fronting type operation (something like Chomsky's T_q) to (14), but which would left-adjoin C to S (in accord with EC), as pictured in (18):

(18)



In the LSLT formalism C and S would both have to be mentioned in the SA. So perhaps the SA could be (19a) or (19b):

(19) a. X–S–C–Y
b. X–C–S–Y

But now look again at the PM (15) to which we would want to apply (19). There is no member of that set that contains both S and C, so the transformation could never apply. This example is completely representative. No movement transformation in the LSLT framework would ever be able to apply in accord with EC.[1]

Interestingly, the L&K framework also forbids EC-satisfying operations, but only by stipulation. Within that model, as noted above, the phrase markers are RPMs, sets consisting of the terminal string and monostrings. Determination of transformational applicability then has to be somewhat different. In particular, it is small sets of monostrings, rather than single ones, that are relevant. L&K provide a definition of precedence between monostrings, and then simply stipulate in their definition of "basic analyzability" that any qualifying set of monostrings

[1]As a reviewer observes, older formulations of the cyclic constraint, as in Chomsky (1965) or the strict cycle condition of Chomsky (1973), do not run into this difficulty, since they only required operations to target topmost domains, and not the root per se.

must be pairwise in the precedence relation. That line of their definition can be eliminated leaving the remainder intact. The effect of this simplification would be to allow a set of monostrings not in the precedence relation, and hence in the dominance relation, to qualify. And this, of course, would allow EC-satisfying operations.

3 Bare phrase structure

Bare phrase structure (BPS, Collins & Stabler 2016 (C&S); Chomsky 2000; Fukui 2011) takes an alternative approach to phrase-markers. BPS uses the set-theoretic $\in$-relation to describe constituency. We fix the instantiation of BPS described in Chomsky (2000; 2008) and formalized in C&S.

In these models, MERGE is a structure-building operations which takes two objects A and B and forms $\{A, B\}$.[2] From this definition, we can recover an "immediately contains" relation between the objects A and B and $\{A, B\}$ by using the elementhood relation. Explicitly, we say that X is *immediately contained* in Y if and only if $X \in Y$.[3] General *containment* is defined as the transitive closure of this relation. Explicitly, we can inductively define containment by saying that X is contained in Y if $X \in Y$ or $X \in Z$ for some Z contained in Y.

Strictly speaking, this is a relation which is defined on the entire model of the ambient set theory, not on a single set X which represents a single syntactic object, as in the case of the precedence and dominance relations between elements of an RPM. That is, containment is a relation between sets in the entire class of sets, not between elements ("nodes") of a single syntactic object. Accordingly, a substructure with respect to the $\in$ relation refers not to a subset of any object in the model, but rather to a *submodel* of the model of set theory.[4]

It is straightforward to show that constituents are not in general subsets of a BPS syntactic object X.

(20) Let A, B, C, and D be lexical items or complex syntactic objects. Construct $X = \textsc{merge}(A, \textsc{merge}(B, \textsc{merge}(C, D))) = \{A, \{B, \{C, D\}\}\}$. Then, $\{C, D\}$ is contained in X, but $\{C, D\} \not\subset X$.

As syntactic objects X are also not models of set theory, but rather the elements of such a model, the submodel relationship which preserves the $\in$ relation also cannot be the correct notion of substructure for syntactic objects.

[2] C&S, Def. 13.

[3] C&S, Def. 8.

[4] Chang & Keisler (1990).

We now present arguments that the $\in$ relation, and its transitive closure, while providing an accurate characterization of the *containment* relation,[5] do not provide a *substructure* relation between syntactic objects. Unfortunately, constituency cannot be used to determine the appropriate notion of substructure, since, in trees, "A contains B" is coextensive with "the constituent dominated by B is a substructure of the constituent dominated by A". In other words, we cannot tell the containment relation apart from substructure inclusions between constituents. However, in slightly relaxed notions of substructures, $\in$ is clearly behaving as a primitive containment relation between nodes, and not a substructure inclusion. We turn to some motivating examples.

In C&S, lexical items are treated as a triple of sets of features (SEM, SYN, and PHON). The features of a syntactic object X are formalized externally with a TRIGGERS function. C&S keep track of which features have been satisfied by removing elements from the sets of features associated to X via TRIGGER. Chomsky suggests in *Categories and transformations* (CT, 1995: Chapter 4) that certain formal features may be *erased* upon satisfaction, or at the interfaces.[6] We first look at how C&S formalize their calculus of features. C&S's feature calculus is meant to capture this intuition.

(21) (C&S Def. 26) TRIGGERS is any function from each syntactic object A to a subset of the trigger features of A, meeting the following conditions:
   i. If A is a lexical item with n trigger features, then TRIGGERS(A) returns all of those n trigger features. (So when $n = 0$, TRIGGERS(A) $= \{\}$.)
   ii. If A is a set, then $A = \{B, C\}$ where TRIGGERS(B) is nonempty, and TRIGGERS(C) $= \{\}$, and TRIGGERS(A) $=$ TRIGGERS(B) $- \{\text{TF}\}$, for some trigger feature TF $\in$ TRIGGERS(B). Otherwise, TRIGGERS(A) is undefined.
   iii. Otherwise, TRIGGERS(A) is undefined.

This goes hand in hand with their definition of triggered merge.

(22) (C&S Def. 27) Given any syntactic objects A, B, where TRIGGERS(A) $\neq \{\}$ and TRIGGERS(B) $= \{\}$, MERGE(A, B) $= \{A, B\}$.

The idea is that two items may only merge when one has remaining trigger features, and the other does not. If defined, the trigger features of $\{A, B\}$ are

[5]Ignoring issues relating to "occurrences" of lexical items – i.e. non-tree structures resulting from the elementhood graphs of sets.

[6]Chomsky (1995: 280): "Erasure is a 'stronger form' of deletion, eliminating the element entirely so that it is inaccessible to any operation, not just to interpretability at logical form (LF)."

just those of the triggering object A with the triggering feature removed. Notice, however, that TRIGGER keeps track of the feature changes externally, in that no features of heads contained in A or B are changed. Under such a method, the set-theoretic structure of syntactic objects alone does not encode the featural changes. We want to "internalize" the feature calculus so that MERGE actually results in changes in the structure of the objects it combines.

We have at least two reasonable options for formally realizing these notions of erasure/deletion within a syntactic object itself: by removing the element in question from the syntactic object, or by changing the element in some way which marks it as inoperative. We will show that either method results in an object which the $\in$ relation and its transitive closure both fail to treat as related to the original object in any straightforward way. We will extend the argument to cases of AGREE.

3.1 Method one: Removal of the feature

For any sets A and B, we can construct a set $A - B = \{a \in A : a \notin B\}$, their *difference*, which removes B-elements from A.

Let A be a lexical item and X and Y be syntactic objects (lexical items or otherwise). We treat lexical items as in CT, where A is literally a *set* of features. Take the syntactic object MERGE$(A, Y) = \{A, Y\}$.[7] Suppose that when this object is merged with X, a feature of the head A is checked, removing $f \in A$, resulting in the object $\{X, \{A - \{f\}, Y\}\}$. Alternatively, if features are not deleted in syntax, we may say that some interface only sees the structure $\{X, \{A - \{f\}, Y\}\}$, which should be a substructure of $\{X, \{A, Y\}\}$.

In the first case, we should like to describe in what sense $\{A - \{f\}, Y\}$ is a substructure of $\{A, Y\}$ in that they have the same phrase structure, with the former simply missing a feature of the latter, so that we can state a form of the EC. In the second case, we should like to describe how $\{X, \{A - \{f\}, Y\}\}$ is a substructure of $\{X, \{A, Y\}\}$.

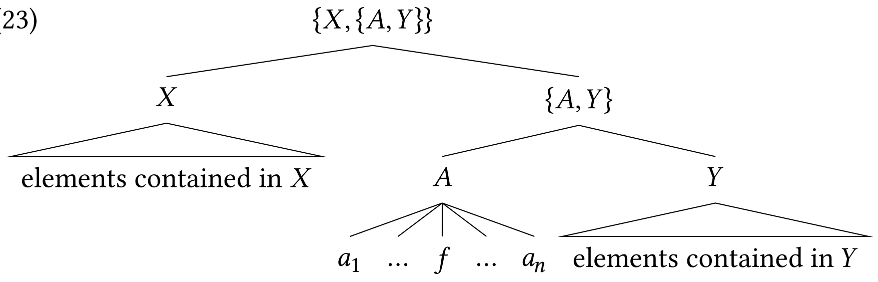
As expected, a subset relation fails to hold in both cases: $\{X, \{A - \{f\}, Y\}\} \not\subset \{X, \{A, Y\}\}$, and $\{A - \{f\}, Y\} \not\subset \{A, Y\}$. However, there is also no containment relation between the syntactic objects. In fact, there is no straightforward set-theoretic relation between these objects. While a subset relation $\{A - \{f\} \subset A\}$ does hold, $\{A - \{f\}, Y\} \not\subset \{A, Y\}$. More generally, for any constituent M containing a head A from which we remove a feature, the resulting constituent M' will simply be a distinct set from M (often with the same number of elements as M). In

[7]For simplicity, we delete no features in the first step, though the argument still holds if we do remove a feature of A (or Y) during this first step.

this example, $\{X, \{A - \{f\}, Y\}\}$ and $\{X, \{A, Y\}\}$ have the same number of elements, though A and $A - \{f\}$ do not, assuming A is finite.

On the other hand, there are canonical ways to draw graph-theoretic objects from well-founded sets. One method produces trees: draw a set X as a root, and write all of its elements as immediate daughters. We repeat the process at each child, writing the same element multiple times if necessary. This process is described in Aczel (1988).

(23)



A *graph* can be defined as a set X together with a relation $R \subset X \times X$. For syntactic objects K, we can define a set X of occurrences of contained elements, with $R \subset X \times X$ being the immediate containment relation between the appropriate occurrences; see C&S (§4, Def. 18) for a formal treatment.

We can define a *subgraph* relation between two graphs (X, R) and (Y, S) if $X \subset Y$ and we have a relation xRx' for $x, x' \in X$ if and only if xSx' in Y. We can then form the graph-theoretic tree associated to $\{X, \{A - \{f\}, Y\}\}$, which is clearly a subgraph of the graph in (23). We could similarly use the containment relation in place of the immediate containment relation, which would describe the syntactic objects as partially ordered sets, with the substructure relation being a subspace inclusion of finite partial orders.

3.2 Method two: Changing (the value of) a feature

Changing the "value" or otherwise adding diacritical marks to an element is another way to formally represent the status of a feature in a syntactic object.

In this case, suppose that we have again constructed $\{A, Y\}$ which we intend to merge with X in a way which will alter a feature $f \in A$. This alteration could be realized as a bijection $m : A \to A'$, where A' is the same set as A, except the feature f has been replaced by $\overline{f}$, the "inoperative" form of f.

However, $\{A, Y\}$ is not a subset of $\{X, \{A', Y\}\}$, nor do we have a containment relation between the two sets. Much like subsets are not the relevant notion of substructure for BPS sets, neither will bijection be the appropriate notion of isomorphism. For, depending on whether we allow MERGE to combine identical sets or not, every BPS set will have cardinality 1 or 2, and hence be in a bijection with the set $1 = \{0\}$ or $2 = \{0, 1\}$. So while $\{A', Y\}$ and $\{A, Y\}$ are "isomorphic" in that there is a bijection between them, so are they both isomorphic to $\{X, \{A, Y\}\}$, showing that this is not the correct notion of "isomorphism" between the objects, in that it totally ignores constituency.

Again, we may convert $\{A, Y\}$ and $\{X, \{A', Y\}\}$ into graph- or order-theoretic trees. We can define an isomorphism between graphs (X, R) and (Y, S) as a bijection $m : X \to Y$ such that xRx' in X if and only if $(mx)S(mx')$ in Y (or similarly, an isomorphism of partial orders as a bijection $m : P \to Q$ such that $x \leq x'$ in P if and only if $m(x) \preceq m(x')$ in Q). Using these definitions, two graph- or order-theoretic trees (X, R) and (Y, S) will be isomorphic if and only if they have the same number of nodes with the same constituency relations.[8] Using this definition, the graphs associated to $\{A, Y\}$ and $\{A', Y\}$ will be isomorphic, such that $\{A, Y\}$ is isomorphic to a subgraph of $\{X, \{A', Y\}\}$ in the appropriate way.

Alternatively, we might think of this "value" or "activity" as a property of a feature which is explicitly part of its structure. This again has a straightforward formalization when the syntactic objects are graphs: we define a graph-with-value as a graph (X, R) together with a function $v : X \to \{\top, \bot\}$ where we interpret $v(x) = \top$ as meaning "x is inactive". We define a homomorphism between graphs-with-values $f : (X, R, v) \to (X', R', v')$ as a graph homomorphism such that if $v(x) = \top$, then $v'(f(x)) = \top$, i.e. inactive features stay inactive, but active features may be deactivated. Using this structure, the inclusion of an operand A into larger object X, while deactivating a feature in A, would be a homomorphism.

3.3 Agree

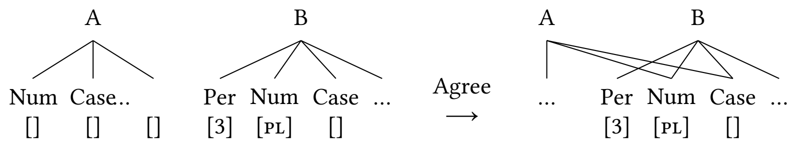
The above examples showed that the feature-deletion and feature-valuation methods of modeling MERGE do not lead to substructure embeddings or homomorphisms between BPS sets in any obvious sense. In contrast, relations between derived syntactic objects are straightforward when represented as graphs (possibly with extra structure). Chomsky (1999) has a "valuation" version of agreement, which is subject to similar analysis as the valuation case for selection above. We look now at a feature-sharing approach to agreement, and similarly show that the structural relation between the input structures and output structures is given

[8]Though, this ignores the "occurrence" relations which indicate which nodes are "copies" of others. On the other hand, the multidominant picture of a tree, called the *canonical picture* in Aczel (1988), and given in Fig. 3 in C&S, would not have this issue, and could be used instead.

straightforwardly by graph homomorphisms, while there is no clear associated notion for sets.

Frampton & Gutmann (2000) give an explicit architecture for agreement as feature-sharing using the set-theoretic structure of BPS:

> Consider [(24)] and suppose that Agree applies to the pair of nodes.
>
> (24) {Num$_1$, Case$_2$, ...}, {Per$_3$, Num$_4$, Case$_5$,...}
>
> [...] suppose that Agree induces feature sharing, so that matching features coalesce into a single shared feature, which is valued if either of the coalescing features is valued. So [(24)] produces:
>
> (25) {Num$_6$, Case$_7$, ...}, {Per$_3$, Num$_6$, Case$_7$,...}
>
> The value of Num$_6$ is the coalescence of the values of Num$_1$ and Num$_4$. The value of Case$_7$ is the coalescence of the values of Case$_2$ and Case$_5$. New indices were chosen, but index 6, for example, could just as well have been 1 or 4. The choice of index is not a substantive question, assuming that it is suitably distinguished.
>
> If the two coalescing features are both valued to start with, it is not clear that the result is coherent. But this will never arise, because Agree is driven by an unvalued feature. A picture will make the idea clearer. Agree takes [(26a)] into [(26b)], assuming that none of the features indicated by the ellipsis marks match.
>
> (26) a. b.



> (Frampton & Gutmann 2000)

The arrow “Agree” in Frampton & Gutmann’s figure can clearly be viewed as a pair of graph homomorphisms from each graph on the lefthand side to the graph on the righthand side, or as single graph homomorphism from the “structured disjoint union"[9] of the graphs on the lefthand side to the graph on the righthand side. If we view the valuations as properties attached to the nodes of the graph, then we can additionally view this map Agree as a graph homomorphism which preserves those properties (e.g. a PL node gets taken to a PL node).

However, it is again difficult to describe the relationship above when we view the objects as BPS sets. Usually at least one of A or B above will be in a phrase when agreement is applied. Suppose it is B, and we have $B \in \ldots \in X$. We intend to construct from A and X an object $\{A', X'\}$, where A' and X' are exactly A and X, but where the number and case features have been replaced accordingly. Again, we will have no subset, containment, or other obvious set-theoretic relation between A or X and $\{A', X'\}$.

Another application of isomorphism appears implicitly here. Frampton & Gutmann note that the specific index for the element representing the shared feature does not matter, so long as it is suitably distinguished. Again, while the set-theoretic statement of this is somewhat complex (and relies on knowing the specific indices used elsewhere in syntactic objects contained in the current one), the graph-theoretic notion is quite elegant: the righthand side above is determined up to isomorphism of graphs (possibly with values assigned to nodes).

4 The extension condition in the theory of phrase structure

Using LSLT phrase markers, constituents do not arise as substructures in any straightforward way. Accordingly, even if we allow operations which have the effect of the EC, it will not be strictly true that the inputs to the operation are substructures of the output.

In BPS, if we represent feature-changes at all in syntactic objects, either by means of deletion or alteration, it is no longer straightforward in what sense the inputs to MERGE are substructures of or are contained in the output. C&S and Chomsky (2000) only avoid this problem by not annotating the "feature-updates" in the syntactic objects themselves, the former by keeping track of the features in "scoreboard" sets external to the syntactic object (though relevant to determining properties of it, such as labeling), where the latter does not address the treatment of features in syntactic objects formally at all.

If we choose the second method which "alters" features, and implement it in the syntax, then the input $\{A, Y\}$ to MERGE will not be a substructure of the output

[9]Formally, this is the *coproduct* of graphs in the category of directed graphs.

$\{X,\{A',X\}\}$ or immediately contained in it. Similarly, no substructure of $\{A,Y\}$ will be contained in $\{X,\{A-\{f\},Y\}\}$ if the first method is used in syntax. Both lead to complications in stating the extension condition for BPS.

However, BPS sets can be viewed as an "encoding" of graphs or partial orders using some canonical translation of them. These graphs essentially arise from constructing a set of elements contained in a syntactic object X (possibly with occurrences), and restricting the $\in$ relation between sets to this set. In C&S, many of the important structural properties of syntactic objects – e.g. c-command, relative minimality and maximality (of projections), and specifiers – are similarly defined not on a syntactic object X itself but the associated graph of occurrences of elements contained in X, using a relation based on $\in$ as its "edge relation". Accordingly, the graph- and order-theoretic representation of BPS objects provides a coherent notion of substructure and isomorphism, which makes statement of the EC straightforward using either method described above.

Pure set-theoretic representations limit the distinctions that can be made. To the extent that human language does not rely on the encoding of the mathematically unavailable distinctions, we should favor a theory based on such representations, as we want to limit the descriptive power of the theory as much as is empirically possible, in line with the general Chomskian program. But where we do need to make such distinctions in a full account of human language, we must move to a richer theory of representations, as we have explored here. Studying substructures and isomorphism as they can be used to state the EC provide just one example of how understanding formal properties of the representation of syntactic objects can clarify the relationship between structure-building operations and the properties of the syntactic objects themselves.

Abbreviations

3	third person	PM	phrase marker
BPS	bare phrase structure	RPM	reduced phrase marker
EC	extension condition	SA	structural analysis
LF	logical form	SC	structural change
PL	plural		

Acknowledgements

We are very pleased to help honor Ian Roberts, who has encouraged the field to rethink so many topics in syntax. We are indebted to the two reviewers, whose comments helped us substantially improve the presentation.

References

Aczel, Peter. 1988. *Non-well-founded sets* (CSLI Lecture Notes 14). Stanford, CA: CSLI Publications.

Chang, Chen Chung & H. Jerome Keisler. 1990. *Model theory*. Elsevier.

Chomsky, Noam. 1957. *Syntactic structures*. The Hague: Mouton.

Chomsky, Noam. 1965. *Aspects of the theory of syntax*. Cambridge, MA: MIT Press.

Chomsky, Noam. 1973. Conditions on transformations. In Stephen Anderson & Paul Kiparsky (eds.), *A Festschrift for Morris Halle*, 232–286. New York: Academic Press.

Chomsky, Noam. 1975 [1955]. *The logical structure of linguistic theory*. New York: Plenum.

Chomsky, Noam. 1993. A minimalist program for linguistic theory. In Kenneth L. Hale & Samuel Jay Keyser (eds.), *The view from Building 20: Essays in linguistics in honor of Sylvain Bromberger*, 1–52. Cambridge, MA: MIT Press.

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

Chomsky, Noam. 1999. Derivation by phase. *MIT Occasional Papers in Linguistics* 18.

Chomsky, Noam. 2000. Minimalist inquiries: The framework. In Roger Martin, David Michaels & Juan Uriagereka (eds.), *Step by step: Essays on minimalist syntax in honor of Howard Lasnik*, 89–155. Cambridge, MA: MIT Press.

Chomsky, Noam. 2008. On phases. In Robert Freidin, Carlos P. Otero & Maria Luisa Zubizarreta (eds.), *Foundational issues in linguistic theory: Essays in honor of Jean-Roger Vergnaud*, 133–166. Cambridge, MA: MIT Press.

Collins, Chris & Edward Stabler. 2016. A formalization of minimalist syntax. *Syntax* 19(1). 43–78. DOI: 10.1111/synt.12117.

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

Fukui, Naoki. 2011. Merge and bare phrase structure. In Cedric Boeckx (ed.), *The Oxford handbook of linguistic minimalism*, 73–95. Oxford: Oxford University Press.

Lasnik, Howard. 2006. Conceptions of the cycle. In Lisa Lai-Shen Cheng & Norbert Corver (eds.), *Wh-movement: Moving on*, 197–216. MIT Press.

Lasnik, Howard & Joseph J. Kupin. 1977. A restrictive theory of transformational grammar. *Theoretical Linguistics* 4(1–3). Reprinted in Lasnik, Howard. 1990. *Essays on restrictiveness and learnablity*, 17-41. Dordrecht: Kluwer, 173–196. DOI: 10.1515/thli.1977.4.1-3.173.