The antisymmetry of syntactic structure: a view from automorphisms

Robert Frank, William Min, Abhisar Mittal & Billy Zhong

Yale University

Abstract. Kayne (1994) and Moro (2000) argue that an asymmetric c-command relation between syntactic constituents X and Y maps onto linear precedence relations between the lexical items in X and Y respectively, the Linear Correspondence Axiom (LCA). Kayne and Moro demonstrate that this assumption, together with the properties of linear orderings (totality, asymmetry and transitivity), imposes linguistically significant limits on the class of possible syntactic structures. In this short paper, we explore an alternative approach to deriving restrictions on syntactic structure that depends not on a specific assumption about linearization, but instead focuses directly on the non-existence of structural symmetry. Specifically, we consider the class of trees, NoAut, which admit no (non-trivial) automorphism (nonidentity mappings between the nodes that preserve structural relations). We demonstrate that NoAut imposes restrictions similar to the ones derived from the LCA, which have proven useful in syntactic theory. However, we prove that the LCA class is a proper subset of NoAut class. We briefly consider the linguistic relevance of the extra flexibility of the NoAut class.

1 Introduction

Natural language sentences have hierarchical structure that can be encoded as a tree, a mathematical object that Keenan & Moss 2015 characterize as follows:

Definition 1. A **simple tree** is a pair $\langle N, D \rangle$ where N is a finite set of nodes and D is a weak partial order over N (i.e., it is reflexive, antisymmetric and transitive) obeying the following conditions:

1. *Rootedness*: there is a y such that for all x, $y D x$;
2. *Chain Condition*: for all x, y, z, if $x D z$ and $y D z$, then $x D y$ or $y D x$.

A significant discovery of syntactic theory is that this general class of representations is too broad: not all simple trees correspond to possible syntactic structures. Chomsky 1970 and Jackendoff 1977 posit the X-bar template as a means of limiting the local syntactic configurations that can relate syntactic heads to their associated arguments/satellites both across constructions and across languages. Kayne 1984 argues that syntactic structures across languages are restricted to binary branching. Subsequent work has attempted to show that such restrictions stem from more basic principles. Under one approach, Kayne 1994 derives limits on possible syntactic structures, at least in part, from the necessity of mapping between hierarchical structure and the linear ordering among the words in the sentence.[1] To do this, he first defines the set TERM of *terminal nodes* in a tree

1. Within the Minimalist Program, an alternative line of explanation has explored the potential of "third factor" principles of efficient computation for restricting possible syntactic structures (Chomsky 2005). Binary branching has been argued to be a consequence of the nature of a computationally simple Merge operation or the nature of search in the establishment of probe-goal relations. Configurational restrictions such as those deriving from X-bar theory might derive from the properties of the labeling algorithm. In the absence of a precise characterization of the relevant notion of computational efficiency, it remains difficult to evaluate such proposals or to precisely characterize their formal consequences. For this reason, we put them aside here.

and a function d that maps nodes in a tree to the terminal nodes that they dominate.

$$\begin{aligned}
\text{TERM} &= \{x \mid \text{if } x D y \text{ then } y = x\} \\
d(x) &= \{y \in \text{TERM} \mid x D y\} \\
d(\langle x, y\rangle) &= \{\langle u, v\rangle \mid u \in d(x), v \in d(y)\} \\
d(R) &= \bigcup_{\langle x,y\rangle \in R} d(\langle x, y\rangle)
\end{aligned}$$

Given these definitions, Kayne's proposal, the Linear Correspondence Axiom, can be stated as follows:

Definition 2. Linear Correspondence Axiom (LCA). in a well-formed syntactic structure, d(ACC) is a strict linear ordering (a total, asymmetric and transitive relation) on TERM, where

$$\begin{aligned}
\text{CC} &= \{\langle x, y\rangle \mid \neg y D x \text{ and if } z \neq x \text{ and } z D x, \text{ then } z D y\} \\
\text{ACC} &= \{\langle x, y\rangle \mid x \text{CC} y \text{ and } \neg y \text{CC} x\}
\end{aligned}$$

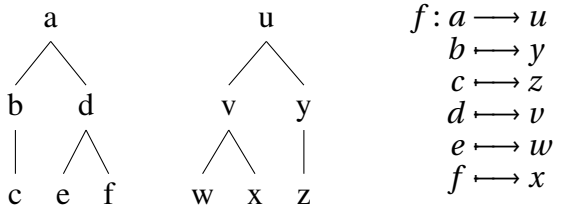
The LCA instantiates the idea that hierarchical asymmetry, as instantiated in the ACC relation, is crucial in producing an ordering of the terminals. Structures that exhibit symmetry do not yield a linear ordering and are ruled out. As Kayne and Moro 2000 demonstrate, the LCA derives linguistically significant restrictions on the class of trees, like those imposed by binary branching and the X-bar template. Yet, the character of the mapping the LCA assumes between hierarchy and linear order is to a certain degree stipulative. In this paper we propose an alternative to the LCA that retains the idea that syntactic structure avoids symmetry. However, instead of adopting a parochial derivation of precedence, we directly characterize the notion of structural (a)symmetry, using the notion of automorphism from abstract algebra.

2 Tree Automorphisms and their Absence

Given two simple trees, we define the notion of isomorphism as a map between them that establishes a kind of structural equivalence.

Definition 3. An **isomorphism** between trees $T = \langle N, D \rangle$ and $T' = \langle N', D' \rangle$ is a function $f : N \rightarrow N'$ such that (i) f is a bijection, and (ii) for all $x, y \in N$, $x D y$ iff $f(x) D f(y)$.

For the pair of trees given below, the function f is an isomorphism between them: it relates each node of one tree to exactly one in the other (i.e., it is a bijection) and dominance relations are preserved. For example, node d dominates nodes d, e and f in the left tree, and f(d) = v dominates f(d), f(e), and f(f), i.e., v, w, and x, in the right tree.

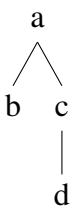


If an isomorphism exists between two trees, we say that the trees are isomorphic. Isomorphisms have the interesting characteristic of preserving structural properties of nodes in a tree. For example, each node in the lefthand tree above is mapped to a node in the right tree that is identical with respect to depth, i.e., the length of its path to the root. In fact, this preservation of node depth must hold for any isomorphism. To see why, let T_1 and T_2 be simple trees and let f be a isomorphism between them. The set of nodes dominating a node n in T_1 is $\{x \mid x D_1 n\}$. Because f is a bijection, this set must be of the same cardinality as the set of dominators of the node $f(n)$ corresponding to n in T_2: $\{y \mid y D_2 f(n)\}$; otherwise we would be able to find at least one node m that dominates n but for which $f(m)$ does not dominate $f(n)$ (or conversely), in violation of the definition of isomorphism. Similar arguments can be constructed for the preservation of other properties under bijection, such as the number of c-commanding nodes, or whether a node is a terminal node or the root. An even stronger example of a property preserved under an isomorphism f is subtree structure: given simple trees T_1 and T_2 related

by f, for every m, n such that $f(m) = n$, there is an isomorphism between the subtrees rooted in m and n as well, which is defined by restricting the domain of f to the nodes dominated by m. Essentially any property that can be defined from the dominance relation is preserved under isomorphism, using arguments similar to the one given here.[2]

We now define an *automorphism* to be an isomorphism between a tree and itself. For example, for the left tree given above, the function which swaps nodes e and f and maps other nodes to themselves, is an automorphism. Because nodes e and f stand in the dominance relation with precisely the same nodes in the tree (i.e., only themselves), swapping one with the other preserves the relevant structural properties in the original tree. Automorphisms like this one highlight points of structural symmetry in a tree structure: nodes that can be swapped in an automorphism are ones that have equivalent structural properties. The identity function, which maps every node to itself, meets the definition of automorphism for any tree at all. We call this a *trivial automorphism* because it does not establish any symmetries between distinct nodes in the tree.

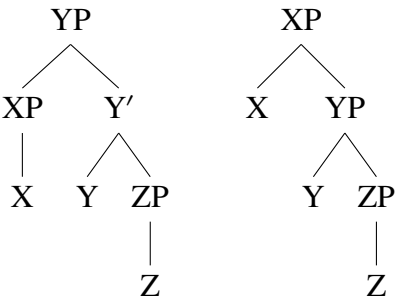
Interestingly, there are some trees for which this trivial automorphism is the only one available. The following is the simplest such structure that includes branching nodes:



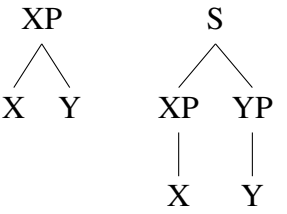
Here, none of the nodes can be "swapped" by an automorphism. To see why, observe that each node in this tree has a distinctive property that must be retained under automorphism: the number of dominated

2. Note that since linear order is not represented in simple trees as we have defined them, ordering between nodes that can be read from these diagrams is not preserved.

and dominating nodes. An automorphism must map terminal nodes like b and d to other terminal nodes, but in this case these nodes cannot be exchanged as they are dominated by distinct numbers of nodes. The non-terminal nodes a and c cannot be exchanged either, since a is dominated by no other nodes (as the root), whereas c is dominated by one other. Consequently, the only automorphism for this tree is the trivial automorphism. We define *NoAut* to be the class of trees for which no non-trivial automorphisms exist. These are trees which have no points of structural symmetry. Like the set of LCA-satisfying trees, the NoAut class imposes linguistically significant constraints on what configurations are possible. NoAut permits trees that accord with the X-bar template, such as the following:[3]



At the same time, and like the LCA, NoAut rules out structures that include symmetries that run afoul of X-bar theory, with a head as complement of another head, or two phrasal sisters.



3. We label the nodes of the tree to facilitate linguistic interpretation. However, these labels play no role in determining membership of a tree in NoAut. While this is also the case for LCA-satisfying structures, trees with specifiers do require special treatment under the LCA, as discussed in the text above, which may reflect a sensitivity to a difference in labeling.

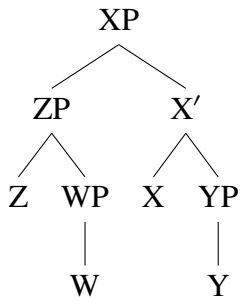
For the left tree, the bijection that swaps X and Y, i.e., mapping X to Y, Y to X, and XP to itself, is a non-trivial automorphism. For the right tree, the bijection swapping both branches, i.e., mapping X to Y, XP to YP, Y to X, YP to XP, and S to itself, is a non-trivial automorphism.

Given the connection that the LCA posits between structural asymmetry and linearizability, it is interesting to understand the connection between the class of LCA-admissible trees and the NoAut class. Our main result, whose proof is given in the appendix, is the following:

Theorem 1. *LCA* $\subseteq$ *NoAut*

This theorem tells us that any tree excluded by NoAut is also excluded by the LCA. However, it allows for the possibility that there are trees that satisfy NoAut, but which run afoul of the LCA. One case in which we see this involves structures with specifiers. Kayne 1994 is forced to assume that specifiers are adjoined to the phrase to which they attach. By adopting the assumptions concerning c-command relations in adjunction proposed by May 1985 (see also Frank, Hagstrom & Vijay-Shanker 2002), Kayne is able to render structures with specifiers compatible with the LCA. However, if we assume instead that symmetric syntax is regulated by the NoAut restriction, there is nothing inherently anomalous about structures with specifiers, as already noted above. Thus, any reasons to put aside the adjunction analysis of specifiers would support the use of NoAut in its place.

Nonetheless, NoAut does impose some constraints on trees that are, at least at first glance, surprising. When a tree includes a specifier has a structural configuration which mirrors the structure of its sister X′, it is no longer in NoAut, as it admits a non-trivial automorphism:



$$
\begin{aligned}
f: \mathrm{XP} &\longrightarrow \mathrm{XP}\\
\mathrm{ZP} &\longmapsto \mathrm{X}'\\
\mathrm{X}' &\longmapsto \mathrm{ZP}\\
\mathrm{Z} &\longmapsto \mathrm{X}\\
\mathrm{X} &\longmapsto \mathrm{Z}\\
\mathrm{WP} &\longmapsto \mathrm{YP}\\
\mathrm{YP} &\longmapsto \mathrm{WP}\\
\mathrm{W} &\longmapsto \mathrm{Y}\\
\mathrm{Y} &\longmapsto \mathrm{W}
\end{aligned}
$$

Interestingly, such an automorphism would fail to exist for this tree if Z itself had a specifier or if either of the heads W or Y had a phrasal complement. At present, we are unaware of patterns in natural language that support such restrictions on possible syntactic structures. These restrictions implicate a sensitivity to global structure, as they turn on the existence of isomorphic subtrees that are sisters (the trees rooted in ZP and X′ in the case here). Nonetheless, we leave it as an open empirical question whether this consequence of NoAut has empirical support.

If we wish to permit structures with such global symmetries, one possibility would be to consider only automorphisms that preserve aspects of the node labeling. In the automorphism for the tree just given, we have ZP and X′ that are exchanged by the automorphism. By allowing only automorphisms that preserve, say, the bar-level of a node's label, we could eliminate this automorphism from consideration and retain the well-formedness of this tree. Defining this restriction precisely would necessitate having an independent theory of tree labeling. We leave for future work carrying this out in detail.

3 Concluding Remarks

To sum up, we have formalized the assumption that trees do not tolerate symmetry in terms of the absence of non-trivial automorphisms and shown that it leads to specific predictions about well-formed structures, corresponding to a slight relaxation of the restrictions imposed by the LCA. Before concluding, we offer a few remarks to

situate this work in a broader context and to lay out areas for future exploration.

Note first that we have not provided empirical evidence in favor of the NoAut pattern of restrictions. However, it is important to note that many of the arguments originally marshalled in favor of the LCA-licensed structures can also be used as arguments in favor of NoAut, as the relevant structural distinctions are also provided by NoAut. And even within the class of trees that they both permit, we take it to be an advantage that NoAut permits specifiers without the need for specific assumptions about adjunction and its consequences for c-command.

One interesting property of NoAut (shared with the LCA) is that it is insensitive to the labeling assigned to the nodes of the tree. Consequently, if it is necessary for nodes to be labeled with grammatical categories or features, this will require an independent theory of labeling. As noted in the previous section, it is possible to allow the existence of labels to interact with what mappings over trees constitute automorphisms (if some properties of the labels must be preserved under automorphism). Furthermore, nothing in our formalization depends on the presence or absence of phonological content in the nodes in a syntactic tree. In contrast, under Moro's (2000) dynamic antisymmetry proposal, the absence of phonological content, either as the result of movement or phonologically null elements, renders a piece of structure immune to the effects of the LCA. It is possible to modify the NoAut proposal to directly accommodate this intuition, though a more intriguing alternative (at least for cases associated with movement) might explore the use of multi-dominance structures (i.e., rooted directed acyclic graphs), so that movement introduces asymmetries between nodes so that they can no longer be "swapped" by an automorphism.

Given the abstractness of our proposal, the restrictions imposed by NoAut may be applied to any kind of linguistic representation. If there is reason to believe that symmetry must be avoided across linguistic levels, we should expect to see the impact of the NoAut restrictions in the domains of morphology (Di Sciullo 2005) and phonology (Raimy 2003), though the representations it restricts need not

be simple trees.

NoAut is a restriction on fully-formed linguistic objects, and can be seen as a representational, as opposed to derivational, constraint on syntactic structures. It remains an open question whether there is a derivational system that yields exactly the NoAut class. If we require that the derivation operate in a local manner without access to unbounded amounts of derivational context, this seems unlikely, though such a conclusion will depend on the specifics of the possible derivational operations Frank & Hunter 2021.

Finally, we would like to comment on the connection of the current proposal with the work of Keenan & Stabler 2004, K&S, who also employ the notion of automorphism in syntactic theory. For K&S, automorphisms operate at the level of the grammar: an automorphism is a bijection f mapping the set of grammatical atoms and rules to itself, with the requirement that structures x and y can be combined via rule R of the grammar to produce $R(x, y)$ if and only if $R(f(x), f(y))$ can also be generated. K&S consider the elements or properties that are invariant under such automorphisms and hypothesize that these are co-extensive with the grammatically relevant ones (though see Paperno 2012). While intriguing, this approach is orthogonal to the one taken here, where automorphisms must preserve the structure-defining dominance relation. Nothing in K&S's approach rules out symmetries in syntactic structures.

Appendix: Proof of Main Result

In this appendix, we present a proof of our main result concerning the relationship between the class of LCA-satisfying trees and NoAut. Note that we consider only simple trees, and not structures that admit the adjunction structures of the sort exploited by Kayne 1994. Before proceeding with the main theorem, we first establish several helpful results.

Lemma 1. *For every non-trivial automorphism φ over simple tree T, there must exist distinct nodes a and b that share a parent such that $\varphi(a) = b$.*

Proof. Suppose that φ is a non-trivial automorphism on T. Consider any pair of nodes a, b for which $\varphi(a) = b$. Because these nodes are related by an automorphism, they cannot stand in a domination relation to one another, otherwise they would be of distinct depths and cannot be related by isomorphism. This leaves two cases:

> *Case 1.* If a and b share a parent, the lemma holds.
>
> *Case 2.* Suppose that a and b do not share a parent. Let r be the least node that dominates both a and b, i.e., any node that dominates a and b will also dominate r, and let r_a and r_b be the children of r that dominate a and b respectively. It must be the case that $\varphi(r_a) = r_b$: $\varphi(r_a)$ must be a node of the same depth as r_a and it must dominate b. r_b is the unique node having these properties. Since r_a and r_b share a parent by construction, the lemma holds. □

The following corollary now follows from this Lemma together with the fact that nodes related by isomorphism must be the roots of isomorphic subtrees.

Corollary 1. *A simple tree T admits a non-trivial automorphism if and only if there exist two isomorphic subtrees in T that share a parent.*

Lemma 2. *A simple tree T satisfies the LCA if and only if for all distinct $x, y \in$ TERM, either x ACC y or y ACC x.*

Proof. We proceed with the forward direction. Let $T = \langle N, D \rangle$ satisfy the LCA. Suppose, for the sake of contradiction, that there exist distinct terminals x, y such that $\langle x, y \rangle \notin$ ACC and $\langle y, x \rangle \notin$ ACC. Then, there exists a common ancestor w such that $w D x$ and $w D y$. Since neither x nor y asymmetrically c-command each other, it must be the case that either both are the child of this common ancestor or neither are.

> *Case 1.* Suppose that both terminals are the children of w. This means that $\langle x, y\rangle \notin d(\text{ACC})$ and $\langle y, x\rangle \notin d(\text{ACC})$. It follows that $d(\text{ACC})$ is not total and thus not a strict linear ordering. This is a contradiction.
>
> *Case 2.* Suppose that both terminals are not the children of the common ancestor. Then the common ancestor has two children a, b, such that $a D x$ and $b D y$. Then it follows that a asymmetrically c-commands y and b asymmetrically c-commands x. This means that $\langle x, y\rangle \in d(\text{ACC})$ and $\langle y, x\rangle \in d(\text{ACC})$. It follows that $d(\text{ACC})$ is not antisymmetric and thus not a strict linear ordering. This is a contradiction.

Since both cases yield a contradiction, our original assumption must have been incorrect. Thus, if T satisfies the LCA, it must be the case for all distinct terminals x, y that $x \text{ACC} y$ or $y \text{ACC} x$.

Next, we show the converse. Again let $T = \langle N, D\rangle$ be a simple tree and suppose that for any two distinct terminals x, y, either $x \text{ACC} y$ or $y \text{ACC} x$. From this it follows that $d(\text{ACC})$ is total, since $d(x)$ will always contain x. Additionally, since ACC, as defined, is an irreflexive, antisymmetric, and transitive relation, it follows that $d(\text{ACC})$ is a strict linear ordering. Hence, T satisfies the LCA.

This Lemma, together with the observation that terminals at the same depth in a tree cannot stand in a ACC relation, yields the following corollary:

Corollary 2. *If a tree satisfies LCA, then its terminals have distinct depths.*

We now prove our main result.

Theorem 1. *LCA* $\subseteq$ *NoAut*

Proof. Suppose that a simple tree T is not in *NoAut*, i.e., it has a non-trivial automorphism f. To establish the theorem, it suffices to prove that T does not satisfy the LCA. By Corollary 1, we know that any T that is not a member of *NoAut* must have two isomorphic subtrees

that share a parent, say node r. Now suppose that x is a terminal in one such subtree (one must exist by the definition of subtree) and let $f(x)$ be the node in the other subtree that isomorphism f maps x to. Because f is an isomorphism, it follows that $f(x)$ is also a terminal. Further, again because f is an isomorphism, x and $f(x)$ must have the same depth. By Corollary 2, this means that T containing terminals x and $f(x)$ does not satisfy the LCA. □

References

Chomsky, N. 1970. Remarks on nominalization. In R.A. Jacobs & P.S. Rosenbaum (eds.), *Readings in English Transformational Grammar*, 184–221. Ginn & Co.

Chomsky, N. 2005. Three factors in language design. *Linguistic Inquiry* 36(1). 1–22.

Di Sciullo, A.M. 2005. *Asymmetry in Morphology*. MIT Press.

Frank, R., P. Hagstrom & K. Vijay-Shanker. 2002. Roots, constituents, and c-command. In A. Alexiadou (ed.), *Theoretical Approaches to Universals*, 109–137. John Benjamins.

Frank, R. & T. Hunter. 2021. Variation in mild context-sensitivity: Derivational state and structural monotonicity. *Evolutionary Linguistic Theory* 3(2). 181–214.

Jackendoff, R. 1977. *X Syntax: A Study of Phrase Structure*. MIT Press.

Kayne, R.S. 1984. Unambiguous paths. In *Connectedness and Binary Branching*, 129–163. Foris.

Kayne, R.S. 1994. *The Antisymmetry of Syntax*. MIT Press.

Keenan, E.L. & L.S. Moss. 2015. *Mathematical Structures in Language*. CSLI Publications.

Keenan, E.L. & E.P. Stabler. 2004. *Bare Grammar*. CSLI Publications.

May, R. 1985. *Logical Form*. MIT Press.

Moro, A. 2000. *Dynamic Antisymmetry*. MIT Press.

Paperno, D. 2012. A note on invariance of grammatical categories. In T. Graf, D. Paperno, A. Szabolsci & J. Tellings (eds.), *UCLA Working Papers in Linguistics, Volume 17,* 325–329.

Raimy, E. 2003. Asymmetry and linearization in phonology. In A.M. Di Sciullo (ed.), *Asymmetry in Grammar. Volume 2: Morphology, Phonology, Acquisition*, 129–146. John Benjamins.