STRUCTURE DEPENDENCE AND LINEAR ORDER: CLARIFICATIONS AND FOUNDATIONS

Jordi Fortuny

Universitat de Barcelona

According to Chomsky (2010, 2013) and Berwick and colleagues (2011), the structure-dependence principle suggests that linear order is a reflex of the sensory-motor system and plays no role in syntax and semantics. However, when these authors use the expression *linear order*, they seem to refer exclusively to the literal precedence/temporal relation among terminals in linguistic objects. This narrow use, which is very common within linguistics, differs from the technical use in a noninnocuous way and does not allow us to exploit the unificational force that the concept of order can have for minimalist investigations. Here I follow Fortuny and Corominas-Murtra's (2009) formal definition of the syntactic procedure, which capitalizes on the foundational set-theoretical concept of nest. I show how the structure-dependence principle can be derived from a local definition of syntactic domain while retaining the idea that central concepts of configurational and transformational syntactic theories are orders.*

Keywords: structure-dependence principle, linear order, order relations, nests, syntactic domain, set theory, syntactic theory, minimalism

1. The structure-dependence principle. In this article I focus on a plausible constraint on possible grammars, the so-called structure-dependence principle, postulated by Chomsky (1972), which can be informally expressed in the following terms.

(1) Structure-dependence principle (SDP): Syntactic operations are defined on hierarchical representations.

Consider, for concreteness, the following well-known auxiliary-fronting case study, where the interrogative sentence (2b) is formed from the declarative sentence (2a).

(2) a. Peter is tall.
b. Is Peter tall?

The paradigm given in 2a and 2b is consistent with the following three rules, since the interrogative sentence is formed by commuting two words of the declarative sentence (3a), or more particularly by fronting the first occurrence of *is* (3b), which is also the second word of the declarative sentence (3c).

(3) a. Given a declarative sentence, form the appropriate interrogative sentence by commuting any two contiguous words.

* This work has been supported by the Secretary for Universities and Research of the Ministry of Economy and Knowledge of the Government of Catalonia and the Cofund programme of the Marie Curie Actions of the 7th R&D Framework Programme of the European Union, as well as the research project FFI2014-56258-P of the Spanish Ministerio de Economía y Competitividad. I would like to thank the audience of the mini-seminar I gave at the University of Cambridge in 2013, where I first presented the main thesis of this article; I am particularly indebted to Ian Roberts, Theresa Biberauer, Michelle Sheehan, and Jenneke van der Wall for being willing to hear about these issues and for the multiple questions they raised. I have also benefited from presenting this work at the Incontro di Grammatica Generativa held in Trento in 2014. I am especially grateful to Bernat Corominas-Murtra and Jan-Wouter Zwart, for their collaborations in the past and for their contributions to the understanding of how syntactic structures can be related to nests, and to Adriana Fasanella for carefully reading a previous version of this article and providing several improvements. Finally, I would like to express my gratitude to Andries W. Coetzee and Line Mikkelsen for their help and advice during the revision process, as well as to two anonymous referees for raising many important comments and indicating unclear aspects of the manuscript.

b. Given a declarative sentence, form the appropriate interrogative sentence by fronting the first occurrence of *is* in the declarative sentence.
c. Given a declarative sentence, form the appropriate interrogative sentence by fronting the second word in the declarative sentence.

Each of these three rules makes different predictions about the formation of an interrogative sentence from the novel declarative sentence in 4, as illustrated, respectively, by the three strings in 4a, 4b, and 4c.

(4) The man who is talking is tall.
a. *The man who is talking tall is?
b. *Is the man who talking is tall?
c. *Man the who is talking is tall?

None of these strings is grammatical, however, and none of the three rules can be generalized adequately to form the grammatical question in 5.

(5) Is the man who is talking tall?

The three rules under discussion share the property of being expressed in terms of precedence/temporal relationships among words; in order to attain the correct grammatical rule, however, the hierarchical structure of sentences must be brought into consideration. The approximative hierarchical representation of 4 is provided in 6.

(6) [[the man [who is talking]] is tall]

We can now informally express the following rule, which is intuitively structure-dependent and accounts satisfactorily for the question formation of both 2a and 4.

(7) Given a declarative sentence, form the appropriate question by fronting the hierarchically most prominent auxiliary in the declarative sentence.

This leads us to the generally accepted conclusion that in order to determine the appropriate grammatical system with the correct predictions, it is necessary to consider the hierarchical properties of linguistic expressions, instead of the position of words in the precedence relation. Therefore, the metric for movement operations is structurally determined.

It is controversial, though, whether such a generalization can be learned from the linguistic evidence (see Berwick et al. 2011 and references cited therein for discussion), or whether it can be derived from economy conditions. Here we are not directly concerned with learnability considerations, although I provide reasons to think that structure dependence is not a primitive component of UNIVERSAL GRAMMAR (UG), since it derives from the locality conditions that the syntactic procedure must satisfy (see particularly §3). But note also that, if structure dependence derives from economy conditions, then it does not need to be learned from the linguistic evidence, so we obtain it for free. Thus, by investigating efficiency conditions, we reduce both the content of UG and the amount of grammatical postulates that must be learned. This should be a welcome conclusion, given the ideals of the MINIMALIST PROGRAM and the importance that the logical problem of language acquisition has for theoretical linguistics in general.

I emphasize that the formal study of central linguistic mechanisms offered here may shed light on fundamental questions; beyond the technical apparatus developed, I am ultimately interested in characterizing the content of UG and determining to what extent this content derives from optimality conditions. Indeed, the answer to questions of the type ‘can an alleged principle of UG be learned?’ is dependent on how the hypothetical principle is related to the optimal design of the underlying grammatical mechanisms. But first, it is mandatory to provide definitions of the grammatical mechanisms that are general enough. In relation to this particular case study, that is, structure-dependence effects, the definition of the syntactic component given here leads us to the idea that they derive from optimality considerations. If structure-dependence effects are a consequence of the optimal functioning of syntactic operations, then there is nothing like a structure-dependence principle—that is, a primitive component of UG—and structure dependence does not need to be learned from the linguistic evidence.

1.1. A clarification. Before developing the proposal in §§2 and 3, I must consider in more detail the connection between the literal precedence/temporal relationship among the terminals of an expression and the SDP, informally expressed in 1 above.

It is crucial to be aware that we cannot properly conclude from the need to claim that syntactic operations are structure-dependent, and not dependent on the precedence relation, that linear order plays no role in syntax and semantics, but is relevant only to the sensory-motor system. This seems to be the view contended, for instance, by Chomsky (2010, 2013) and Berwick, Pietroski, Yankama, and Chomsky (2011). For example, Chomsky claims that 'the best explanation for the choice of structural rather than linear distance would be that linear order is simply not available to the operations of the I-language—that it is a secondary phenomenon imposed by the sensory motor system' (2010:11), and Berwick and colleagues state that 'linear order seems to be a reflex of the sensory-motor system, and so unavailable to the syntax and semantics we describe there' (2011:1217). According to this position, the syntactic and semantic components have hierarchy and structure, but not linear order.

In the view just sketched (and more generally, in generative grammar), the use of the expressions *order*, *linear*, and *linear order* differs from their technical meaning: they refer exclusively to the precedence/temporal relationship among terminals, which is indeed an order. However, it is clear that there are order relations that are not precedence/temporal relations. I emphasize this elementary but important terminological issue because, as will become clear, it is not merely a particular informal use that innocuously differs from the technical use, but rather the source of obscurity about an aspect of the theory of grammar that has a prominent place in current generative linguistics, namely, how abstract hierarchical representations are assigned a precedence/temporal relationship. This aspect is usually called (again, misleadingly) the linearization problem.

What is essential about order relations (in the technical sense) is that they are transitive and antisymmetric.

(8) An order is a transitive and antisymmetric relation.

A relation R is defined as transitive or antisymmetric as follows.

(9) a. R is transitive iff, for any objects x, y, z, if $\langle x, y\rangle \in R$ and $\langle y, z\rangle \in R$, then $\langle x, z\rangle \in R$.

b. R is antisymmetric iff, for any two objects x, y, if $\langle x, y\rangle \in R$ and $\langle y, x\rangle \in R$, then $x = y$.

Let us consider, for clarity, some typical order relations. The 'greater than' ($>$) and the 'greater than or equal to' ($\geq$) relations defined in the set $\mathbb{N}$ of natural numbers are linear orders in $\mathbb{N}$; they are orders because they are transitive and antisymmetric, and they are linear in $\mathbb{N}$ because they are connected in $\mathbb{N}$.[1] The statement in 10 holds in general.

[1] It may be useful to clarify why $>$ is antisymmetric. For $>$ to be antisymmetric, the following conditional must be true for any two numbers x, y: if $x > y$ and $y > x$, then $x = y$. However, there are no distinct numbers x, y such that $x > y$ and at the same time $y > x$; accordingly, the antecedent of the conditional ($x > y$ and $y > x$) is false, and thus the conditional is (vacuously) true. Therefore we conclude that $>$ is antisymmetric.

(10) A relation R is connected in A iff, for any two objects $x, y \in A$, either $\langle x, y \rangle \in R$ or $\langle y, x \rangle \in R$.

The terms *linear* and *connected* are thus synonymous when they refer to order relations. The difference between $>$ and $\geq$ is that the former is a strict linear order in $\mathbb{N}$ and the latter is a reflexive linear order in $\mathbb{N}$. The usual definitions for 'strict' and 'reflexive' hold.

(11) a. A relation R is strict (or irreflexive) iff, for all objects $x \in A$, $\langle x, x \rangle \notin R$.
  b. A relation R is reflexive iff, for all objects $x \in A$, $\langle x, x \rangle \in R$.

Other instances of orders are the strict inclusion relationship ($\subset$) and its reflexive associate ($\subseteq$). We can observe that they are nonlinear in the set of subsets of $\mathbb{N}$, since we can find sets of natural numbers such as $\{1, 2\}$ and $\{7, 8, 9\}$ that are not connected by these relations. The two pairs of relations that have briefly been considered ($>/\geq$ and $\subset/\subseteq$) are orders, but none of them involves a precedence/temporal relationship.

Once we bear in mind these technical definitions, we can grasp that syntactic structures are full of linear orders; below I argue for the idea that syntactic constituents (generated via external merge) and chains (generated by internal merge) are precisely linear orders. We can also understand that the evidence for structure dependence does not suggest that linear order plays no role in syntax. Admitting that the metric for movement operations is not defined on the basis of a precedence relationship, but rather on the basis of structural prominence, does not lead us to the conclusion that linear order is unavailable in syntax. Indeed, the main objective of this article is to provide a principled account of structure-dependence effects that retains the centrality of order in syntax; I attain this goal by providing a local definition of domain, which restricts the categories that are available for movement operations.

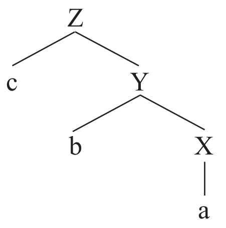
What this article contributes to previous work in Fortuny & Corominas-Murtra 2009 is an argument for making two theses compatible: linear orders are pervasive in syntax, and syntactic operations are defined on structural relationships (and not on precedence relationships). If we abandon the very narrow—and incorrect—conception of linear order as a precedence relation among terminals, and replace it with the standard mathematical notion, we arrive at the conclusion that it is meaningless to claim that linear order is a by-product of the ARTICULATORY-PERCEPTUAL SYSTEM, contrary to the guiding conception within the minimalist program; it becomes evident that both structural relations and precedence relations are linear orders. Therefore, the fundamental syntactic notion is that of linear order, from which both structural and temporal relationships derive.

1.2. ORGANIZATION AND GENERAL OBJECTIVE. The remainder of this article is organized as follows. Section 2 investigates the emergence of structure-dependence effects. I choose the Fortuny & Corominas-Murtra 2009 formalism among the multiple options offered by contemporary syntactic theory in order to study the generation of expressions such as 5. What seems appealing about this choice is that it is based on the mathematical concept of NEST. We must recall in this regard that the notion of linear order is reduced to the notion of nest in set theory (Kuratowski 1921); therefore I attempt to apply foundational set-theoretical tools in a minimalistic investigation of syntactic structures and operations. Given that my aim is a methodological one, it is important to construct the syntactic theory on the basis of foundational notions.

In §2.2 the mechanics of the nesting machine when one single derivational space is involved are defined, along with the concept of syntactic constituent (**Definition 1**) as a linear order of occurrences. In §2.3 I describe the mechanics of the nesting machine when there is more than one derivational space involved, and I define the concept of syntactic domain (**Definition 3**) as well as the concept of syntactic chain as a linear order of copies of an occurrence (**Definition 4**) in §2.4. Finally, §3 summarizes the derivation of structure-dependence effects from efficiency considerations and emphasizes, consequently, that structure dependence is not a primitive or unmotivated principle of UG. Needless to say, the character of this article is clearly programmatic. My strategy is to clarify certain basic concepts in order to find a solid ground for structure dependence before addressing more complex issues within this novel framework.

2. NESTS. I take as a starting point the Fortuny & Corominas-Murtra 2009 formalization of certain core aspects of current generative syntax (cf. among others Stabler 2011 for a different perspective). This work is a particular technical implementation of the minimalistic approach to phrase structure initiated by Chomsky (1995), which capitalizes on the intuition that the simplest way to generate hierarchically structured expressions is by means of a successive operation that takes two objects as input and merges them together, thereby yielding as output a new object composed solely of the two objects taken as input. Chomsky's earliest minimalist considerations also argued for the methodological virtue of dispensing with strictly grammatical idiosyncrasies (such as X′-theory or trace-theory). A further though narrowly related possibility was indicated by Epstein (1999) in his study of c-command: namely, that syntactic relationships could simply derive from the computational procedure; they would constitute not primitive syntactic elements, but rather computational by-products (cf. also Epstein 1998 for a broader derivational study of syntactic relations). Fortuny & Corominas-Murtra 2009 is a particular execution of the reductionist program initiated by Chomsky and Epstein: it provides a precise definition of both the computational procedure and the basic hierarchical notions of the generated outcomes (such as the concepts of syntactic constituent, chain of copies, and syntactic domain, as well as the dominance relationship) that does not rely on idiosyncratic grammatical elements but solely on the most fundamental elements of set theory. This proposal also confirms Kayne's (1994) influential intuition that an ordering among the terminals of a linguistic expression can be directly obtained from hierarchical syntactic relations, although X′-theory does not need to be assumed. The original idea from Fortuny 2008 that nests may offer a useful tool for theoretical syntax has also been adopted by Zwart's (2009, 2010) work on syntactic dependencies and De Belder and van Craenenbroeck's (2011) proposal on root insertion.

2.1. SKETCH OF THE PROPOSAL. At this point it may be useful to provide a nontechnical and preliminary sketch of my proposal, with no aim of being exhaustive at all. The main insight is that we can explore the similarities between tree diagrams, such as 12, and a particular kind of sets, commonly called *nests*, *towers*, or *chains* (Kelley 1955), such as 13.

(12)


(13) $N = \{\{a\}, \{a, b\}, \{a, b, c\}\}$

In 12 we observe a tree with three nodes: *X*, *Y*, and *Z*, and in 13 we observe the set *N*, which has exactly three elements: $\{a\}$, $\{a, b\}$, and $\{a, b, c\}$. The three elements of *N* correspond to the sets of terminals dominated by the three nodes in 12; indeed, the set of nodes dominated by X, Y, and Z are, respectively, $\{a\}$, $\{a, b\}$, and $\{a, b, c\}$, which, as noted, are precisely the three elements of N.

This basic observation opens up certain methodological questions relevant for minimalist inquiries. For instance, can we reformulate standard syntactic relations, commonly defined on the basis of tree diagrams, using nests and following the central tenets of bare phrase structure, thereby dispensing with X′-theory? Can we define a syntactic procedure that generates nests? What do we need to add to nests in order to deal with syntactic phenomena? These are some of the general research questions that underlie this investigation about structure dependence.

2.2. Simple nesting. Let us thus study in a precise way the generation of expression 5, repeated in 14.

(14) Is the man who is talking tall?

Let the alphabet for the generation of 14 be the set A of minimal syntactic categories, each viewed as a singleton whose single element is a primitive element, also called a terminal.

(15) $A = \{\{the\}, \{tall\}, \{talking\}, \{man\}, \{who\}, \{is\}\}$

I emphasize that minimal syntactic categories are being defined as singleton sets of terminals, and not simply as terminals. It is easy to understand why. In the framework being developed, minimal syntactic categories need to be singletons, because the basic syntactic operation (what is commonly called 'merge') is defined on the basis of the set-theoretical operation of union formation, as will become clear immediately. Since union formation takes sets (and not primitive elements) as input, then our basic syntactic operation must take sets (and not terminals) as input; this entails that minimal syntactic categories cannot be terminals but must be singletons of terminals.

Note also that, for the sake of simplicity, I ignore the inflectional structure of words such as *talking* and *is* and treat $\{talking\}$ and $\{is\}$ as syntactic primitives. I avoid introducing in our alphabet singleton sets whose single elements are inflectional suffixes, since the internal structure of words is not the focus of our attention.

We must be aware that the declarative sentence *The man who is talking is tall* (as well as the interrogative sentence in 14) presents two occurrences of the lexical item *is*: one appears between *who* and *talking*, and the other between *talking* and *tall*. In general, different occurrences of a lexical item may have distinct semantic properties and thus must be distinguished by syntactic computations; in the case of the lexical item *is* in the sentence *The man who is talking is tall*, the first occurrence is an auxiliary encoding aspect and selecting a gerund, whereas the second occurrence is a copula and has no predicative function: it encodes only grammatical features, and the predicate is the adjective *tall*.

Occurrences of a lexical item are understood to be no more than the result of selecting the lexical item at different times, that is, at different moments of the syntactic derivation. I thus keep track of the moment when occurrences are merged in the syntactic workspace in order to distinguish them.

Let us now consider how the nesting machine $\mathcal{N}$ proceeds in order to generate 14. At the first step, call it s_0, $\mathcal{N}$ selects an element $\{k\}$ of a given alphabet and generates the set $\{k_0\}$, which I refer to as the set M_0. Accordingly, we shall say that when an element $\{k\}$ of an alphabet comes into the computation at step s_0, its element, k, becomes the occurrence k_0. Assume for concreteness that in the derivation of 14 $\mathcal{N}$ generates the singleton $\{tall_0\}$, as in 16.

(16) $\{tall_0\} = M_0$

Essentially, this treatment of occurrences is a particular adaptation of Chomsky's (1995:227) insights: occurrences are distinguished crucially by keeping track of the derivational step where they are introduced. However, this adaptation differs from Chomsky's (1995) original view in one respect: whereas Chomsky provides an index *i* specifying how many times a lexical item will be selected and introduced into the derivation, I use numbers to represent the step at which each occurrence is introduced in the derivation.[2]

At any further derivational step (symbolically at any step $s_{n>0}$), $\mathcal{N}$ selects an element $\{k\}$ of an alphabet, forms the set $\{k_n\}$ (namely, the set whose element is the *n*-occurrence of *k*), and generates the set M_n, which is recursively defined as the union of $\{k_n\}$ and the output M_{n-1} of the immediately previous step s_{n-1}. Assume that $\mathcal{N}$ continues the derivation started in 17a with the union-formation operation represented in 17b.

(17) a. $\{tall_0\} = M_0$

b. $M_0 \cup \{is_1\} = \{tall_0\} \cup \{is_1\} = \{tall_0, is_1\} = M_1$

$\mathcal{N}$ is thus defined as an operation that generates at the first step s_0 the set $M_0 = \{a_0\}$ for $\{a\}$, an element of a given alphabet A, and forms, at any further step $s_{n>0}$, the union of the output M_{n-1} and $\{k_n\}$ for $\{k\} \in A$. When an arbitrary element of A, namely $\{e\}$, comes into the computation at step $s_{i\geq 0}$, its element, e, becomes an occurrence e_i. This procedure can be symbolically summarized as follows.[3]

(18) $M_0 = \{a_0\}$ $(\{a\} \in A)$

$M_{n>0} = M_{n-1} \cup \{k_n\}$ $(\{k\} \in A)$

Thus, a syntactic derivation is viewed as a sequence of steps, or lines: in the first line, we pick up an element of the alphabet; and in all successive steps we form the union of a set of the alphabet and the set generated in the previous step. Occurrences are distinguished by keeping track of the step where they are introduced in the derivation. Note also that the number of steps of a syntactic derivation is unboundedly large but finite; this means that a nesting computation must always be finite, although there is no fixed natural number that constrains the length of nesting computations.

It is important to distinguish the narrow study of grammatical mechanisms from the problem of choice of action using Chomsky's terminology (see Chomsky 1995:226–27). Here we are concerned strictly with the first problem, and not with the second one. As a matter of fact, it is standardly assumed that the main objective of generative grammar is to unearth the basic mechanisms responsible for creating linguistic structures without even addressing any kind of question related to the problem of choice of action, such as how a particular alphabet 'is formed rather than another—or rather than none, so that we have silence' (Chomsky 1995:227), why a given syntactic category is selected instead of another, or why a grammatical derivation starts, for instance. This is not an idiosyncrasy of generative grammar, but a common approach to the study of computation: asking any of these questions

> would be like asking that a theory of some formal operation on integers—say, addition—explain why some integers are added together rather than others, or none. Or that a theory of the mechanisms of vision or motor coordination explain why someone chooses to look at a sunset or reach for a banana. The problem of choice of action is real, and largely mysterious, but does not arise within the narrow study of mechanisms. (Chomsky 1995:226–27).

[2] Accordingly, the formalization here includes not only sets but also numbers, as noted by a referee. Chomsky (1975 [1955]) already argued for the need for a notation to distinguish among occurrences; at that time Chomsky's notation was based on a device proposed by Quine (1940).

[3] Note that $M_{n\leq 0}$ is the set of occurrences of the lexical items introduced up to step *n*; recall the informal comparison between tree diagrams and nests of §2.1, where the set $M_{n\leq 0}$ would correspond to a particular node of a tree diagram.

Before continuing with the investigation of the generation of expression 14, it will be useful to introduce a precise definition of the notion of *constituent* (**Definition 1**), which plays a crucial role in configurational approaches to syntax in determining how semantic relations are set on the basis of syntactic representations, and to spell out some of its fundamental properties (**Remarks 1**, **2**, and **3**).

Syntactic constituents are nests. Intuitively, syntactic constituents are no more than the parts of a syntactic object. This part-whole relationship, which is essential to the notion of constituent, can be properly formulated within our framework, thanks to the purely incremental nature of the nesting machine. What we need to do is simply pick up a particular derivational step, say s_k, and then form the set of all sets previously generated by the nesting machine.

Illustration. Consider again the derivation we are studying, schematized in 19.

(19) $M_0 = \{tall_0\}$
$M_1 = \{tall_0, is_1\}$

We can determine that in this nesting derivation there are two constituents so far, one for each step.

(20) $C_0 = \{M_0\} = \{\{tall_0\}\}$
$C_1 = \{M_0, M_1\} = \{\{tall_0\}, \{tall_0, is_1\}\}$

The following general definition of syntactic constituent can now be provided.

Definition 1. *A* constituent *C_k is the outcome of $\mathcal{N}$ at a particular step s_k, which is the set of those sets M_j $(0 \leq j \leq k)$ successively generated,*

$$C_k = \{M_0, M_1, \ldots, M_k\}.$$

Let us prove by induction that any outcome of the nesting machine is a nest (**Remark 1**).

Remark 1. *Any constituent C_k generated by $\mathcal{N}$ is a nest, that is, a set of sets linearly ordered by inclusion,*

$$M_0 \subset M_1 \subset \ldots \subset M_k.$$

Proof. **Remark 1** means that, for any two distinct elements M_i, $M_j \in C_k$, either $M_i \subset M_j$, or $M_j \subset M_i$. Consider first that we stop $\mathcal{N}$ at $s_0 : \{a_0\}$; accordingly, the outcome of this derivation will be, by **Definition 1**, the constituent

$$C_0 = \{\{a_0\}\},$$

which is a trivial nest, since it is vacuously true that all distinct elements of C_0 are pairwise related by inclusion. Consider now that we stop $\mathcal{N}$ at an arbitrary s_n when $n > 0$ and assume, by inductive hypothesis, that C_{n-1} is a nest, that is, $M_0 \subset \ldots \subset M_{n-1}$; since M_n is the union of M_{n-1} and a singleton, we conclude that $M_{n-1} \subset M_n$. Accordingly, $M_0 \subset \ldots \subset M_{n-1} \subset M_n$, whereby C_n is also a nest when $n > 0$. Therefore, any constituent is a nest. □

The following remark would be easy to verify by induction as well. I skip this demonstration, since it is quite straightforward.

Remark 2. *Given a nesting derivation of n-steps, the set $\mathcal{C}_n$ of constituents generated during this derivation,*

$$\mathcal{C}_n = \{C_i : i \leq n\},$$

is a nest (i.e. it is a set of sets linearly ordered by inclusion).

In relation to **Remark 2**, observe that the set in 21 contains all of the constituents generated in the syntactic derivation under discussion, sketched in 19 above.

(21) $\mathcal{C}_1 = \{\{\{tall_0\}\}, \{\{tall_0\}, \{tall_0, is_1\}\}\}$

And indeed, it is a nest, since the constituent $\{\{tall_0\}\}$ is included in $\{\{tall_0\}, \{tall_0, is_1\}\}$, but not vice versa.

Therefore, the proposed formal definitions of constituent and of set of constituents of a syntactic object are nests.

Nests are linear orders. A further interesting consequence of introducing nests into syntactic theory is that the precedence/temporal ordering among the terminals of an expression is directly obtained from the hierarchical or nested representation.

As proved by Kuratowski (1921), a family F of sets linearly orders a set S iff F is saturated as to the property of being a nest of S.

Definition 2. *F is saturated as for the property of being a nest of S iff all the elements of F are subsets of S and F is not a proper subset of a nest of S.*

Consider, for concreteness, the following set S and the families of sets F_1, F_2, and F_3.

(22) $S = \{a, b, c\}$
$F_1 = \{\{a\}, \{a, b\}, \{a, b, c\}\}$
$F_2 = \{\{a\}, \{a, b, c\}\}$
$F_3 = \{\{a, b\}, \{a, b, c\}\}$

The family F_1 is a linear order of $S = \{a, b, c\}$, since (i) all of the elements of F_1 are subsets of S and (ii) there is no nest of S that is a proper superset of F_1; therefore, F_1 satisfies the two conditions specified in **Definition 2**, which means that F_1 is a linear order of S, as proved by Kuratowski. However, the families F_2 and F_3 are not linear orders of S: although their respective elements are subsets of S, F_2 and F_3 are proper subsets of the nest F_1.

In order to perceive in what sense nests are linear orders, we may observe that saturated nests satisfy the essential property of linear orders. This essential property can be expressed in the following terms.

(23) $(<a_1, a_2, \ldots, a_n> = <b_1, b_2, \ldots, b_n>) \rightarrow (a_1 = b_1 \wedge a_2 = b_2 \wedge a_n = b_n)$

In other words, if two orders are equal, then their respective first components are equal to each other, and their respective second components are equal to each other, and in general, their respective nth components are equal to each other. Thus, in order to conclude that nests are equivalent to orders, we need to show the following.

(24) $(\{\{a_1\}, \{a_1, a_2\}, \{a_1, a_2, \ldots\}, \{a_1, a_2, \ldots, a_n\}\} = \{\{b_1\}, \{b_1, b_2\}, \{b_1, b_2, \ldots\},$
$\{b_1, b_2, \ldots, b_n\}\}) \rightarrow (a_1 = b_1 \wedge a_2 = b_2 \wedge a_n = b_n)$

Assume the truth of the antecedent; that is, assume that the two nests are equal to each other. If this is the case, then we conclude, first, that $a_1 = b_1$, because $\{a_1\}$ is the sole singleton of the first nest and $\{b_1\}$ is the sole singleton of the second nest. Second, $a_2 = b_2$, since $\{a_1, a_2\}$ and $\{b_1, b_2\}$ are, respectively, the only sets of two elements of the first and the second nest, and we already concluded that $a_1 = b_1$. If we continue applying this reasoning, we conclude finally that $a_n = b_n$, since $\{a_1, a_2, \ldots, a_n\}$ and $\{b_1, b_2, \ldots, b_n\}$ are, respectively, the only sets of n elements of the first and the second nest. Consequently, we derive the truth of the consequent from the truth of the antecedent.[4]

[4] I adapt this reasoning from Fortuny 2008 in order to ensure that the presentation here is self-contained. See Hallett 1986 for a detailed discussion of Kuratowski's method of reducing the notion of order to set theory and its precedents.

I introduce this very general result into our syntactic theory by claiming that the constituent C_n, generated at step s_n, linearizes the set M_n, whose elements are all of the occurrences that have been introduced so far.

Remark 3. *For a given constituent C_n, we can identify the set M_n as the set of occurrences of syntactic terminals; note that C_n is always saturated as to the property of being a nest of its element M_n, whereby C_n is a linear order of M_n. In other words, a constituent is a linear order of the occurrences of its terminals. This linear ordering can be straightforwardly interpreted as the precedence/temporal relation among the occurrences of syntactic terminals of a constituent at the sensory-motor system.*

Returning to the study of expression 14, we can observe that $M_1 = \{tall_0, is_1\}$ is the set of terminals for the constituent $C_1 = \{\{tall_0\}, \{tall_0, is_1\}\}$. Since C_1 is a nest saturated of M_1, it can be interpreted as the ordering $\langle is_1, tall_0\rangle$.

In sum, we arrive at the conclusion that the nesting machine generates syntactic constituents, which are linear orders on the basis of which the precedence/temporal relation can be directly read at the sensory-motor system.

I emphasize that there is a very narrow relation between the direction of syntactic derivations and the direction of the precedence/temporal relation among terminals; in Appendix A I discuss this issue and argue that the nesting machine must be a bottom-up procedure and that the orders it generates must be read in a particular temporal direction if we want to obtain the appropriate syntactic constituents.

Now it becomes crucial to observe that the next element to be introduced in the nesting derivation of 14 is not a terminal, but a complex syntactic object.

(25) [the [man [who [is [talking]]]]]

This requires us to enrich $\mathcal{N}$ in order to merge into the syntactic workspace not only atomic syntactic terminals but also complex syntactic objects.[5]

2.3. Complex nesting. On both empirical and conceptual grounds it is necessary to allow syntactic computations to manipulate not only minimal syntactic categories but also complex syntactic objects (see Zwart 2009 for a broader discussion on layered derivations). With this purpose Fortuny & Corominas-Murtra 2009 allows the nesting machine to be compounded of multiple derivational spaces, labeled as $D_1, D_2, \ldots, D_n$, and defines necessary conditions as well as the required notation.

As argued in Fortuny & Corominas-Murtra 2009 in more detail, there are two minimal conditions imposed on nesting computations involving multiple derivational spaces, which are given in 26.

(26) a. The number of derivational spaces involved in a nesting derivation must be bounded.

b. At the end of a given derivation, only one derivational space can remain open.

These two conditions restrict the class of computations performed by a nesting machine with multiple derivational spaces. They clearly have an economy flavor, but it must also be remarked that they are virtual conceptual necessities. Condition 26a is necessary in order to ensure that the memory of $\mathcal{N}$ is finite; if we want to decide whether a given object is the outcome of a computation performed by the machine, this condition also avoids the halting problem (see Fortuny & Corominas-Murtra 2009 and references cited therein).

[5] It is plausible, given our current understanding, that [*the man who is talking*] constitutes a small clause with [tall] and moves up to the specifier of the inflected auxiliary is_1. I do not follow this analysis here, because how to represent movement operations within this framework has not yet been shown. This is explained below in §2.4.

Condition 26b simply implies that all derivational spaces used in the computation generated their outcomes as inputs for other derivational spaces; otherwise, we would not be dealing with a single derivation with multiple spaces but with different (unconnected) derivations.[6]

As for notation, subindices and superindices refer, respectively, to derivational steps and derivational spaces. For instance, the terminal k^g_f would have been introduced at the derivational space D_g and at the step s_f, the set M^d_r would have been formed at D_d and at s_r, the constituent C^j_i would have been generated at D_j at step s_i, and the set $\mathcal{C}^j_i$ would contain all of those constituents generated at D_j at some $s_{f \leq i}$. When the constituent C^i_s is the final outcome of D_i, and D_i feeds D_j at step s_r, then C^i_s becomes $C^{i/j}_r$. Thus, we use numbers, as before, to recall what has happened in the derivation, to recall the step and also the space where a syntactic object has been introduced. In order to keep our representations as simple as possible, I introduce indices only when they are required for practical reasons, that is, only when we need to distinguish different occurrences of the same element.

Let $M^1_1 = \{tall, is^1_1\}$ and assume that the constituent depicted in 25 is generated at D_2:

$$\begin{array}{l} \qquad\qquad D_2 \\ s_0 : \{talking\} \\ s_1 : \{talking, is^2_1\} \\ s_2 : \{talking, is^2_1, who\} \\ s_3 : \{talking, is^2_1, who, man\} \\ s_4 : \{talking, is^2_1, who, man, the\} \end{array}$$

$$C^2_4 = \{\{talking\}, \{talking, is^2_1\}, \{talking, is^2_1, who\}, \{talking, is^2_1, who, man\}, \{talking, is^2_1, who, man, the\}\}$$

At this point D_2 must feed D_1, which means that the final outcome C^2_4 of D_2 is introduced into D_1. Accordingly, $\mathcal{N}$ takes as input $\{C^2_4\}$, forms at s_2 of D_1 the set $\{C^{2/1}_2\}$, and performs the operation $\{tall^1_0, is^1_1\} \cup \{C^{2/1}_2\}$, thereby generating the new constituent C^1_2. See Table 1 for a representation of the whole derivation involving two spaces of C^1_2.[7]

	D_2	D_1
s_0	$M^2_0 = \{talking\}$	$M^1_0 = \{tall\}$
s_1	$M^2_1 = \{talking, is^2_1\}$	$M^1_1 = \{tall, is^1_1\}$
s_2	$M^2_2 = \{talking, is^2_1, who\}$	$M^1_2 = \{tall, is^1_1, C^{2/1}_2\}$
s_3	$M^2_3 = \{talking, is^2_1, who, man\}$	
s_4	$M^2_4 = \{talking, is^2_1, who, man, the\}$	
	$C^2_4 = \{M^2_0, M^2_1, M^2_2, M^2_3, M^2_4\}$	
		$C^1_2 = \{M^1_0, M^1_1, M^1_2\}$

TABLE 1. Nesting derivation of C^1_2. Derivational space D_2 consists of five derivational steps; its final outcome is the constituent C^2_4. In D_1 at s_2, $\mathcal{N}$ takes as input C^2_4, forms the set $\{C^{2/1}_2\}$, that is, the final outcome of D_2 reintroduced at D_1 at s_2, and performs the operation $\{tall^1_0, is^1_1\} \cup \{C^{2/1}_2\}$. The outcome of D_1 is the set C^1_2.

Since C^1_2 is the linear order 27a and $C^{2/1}_2$ the linear order 27b, the linear ordering among terminals (27c) is obtained by composition of relations. This composed linear order yields the literal precedence/temporal relationship among terminals of a linguistic expression. See Appendix B for a more detailed discussion on composition of relations.

[6] This condition is equivalent to the SINGLE ROOT CONDITION (Partee et al. 1990:439): 'In every well-formed constituent structure tree there is exactly one node that dominates every node'. This condition allows us to distinguish a tree from a forest of trees, as condition 26b allows us to distinguish a complex derivation with multiple spaces from a multiplicity of unconnected derivations.

[7] Here I assume for simplicity that *the man* is generated in D_2 by successive applications of external merge; alternatively, for instance, we could consider it to be generated in a third derivational space, D_3, and introduced in D_2. Note that this question does not affect the argumentation here, since we have just allowed the nesting machine to contain several derivational spaces.

(27) a. ⟨$C_2^{2/1}$, is_1^1, *tall*⟩
b. ⟨*the*, *man*, *who*, is_1^2, *talking*⟩
c. ⟨*the*, *man*, *who*, is_1^2, *talking*, is_1^1, *tall*⟩

In other words, the final outcome of D_2, that is, the constituent generated at step 4 of the derivational space D_2, symbolically dubbed C_4^2, provides us with an ordering of the terminals of the subject of the expression, *the man who is talking*. This ordering is expressed in 27b. The constituent C_4^2 is reintroduced from D_2 to the main space D_1 at step 2; we now call this constituent $C_2^{2/1}$. The constituent C_2^1 is the final outcome of D_1; it provides us an ordering affecting the constituent C_4^2 and the occurrences is_1^1 and *tall*, which is expressed in 27a. If we combine this last ordering (27a) with that affecting the occurrences of the subject (27b), then we obtain a linear ordering of all terminals of the expression (27c).

It is not necessary to introduce any complication in order to obtain an ordering among the terminals of a linguistic expression; X′-theory does not need to be assumed, and Kayne's (1994) so-called LINEAR CORRESPONDENCE AXIOM (LCA) becomes a mere by-product of the generative procedure, as argued in Fortuny 2008. It is also remarkable that the problems posed to Kayne's approach by nonbranching complements and by branching specifiers do not appear once we dispense with X′ notation and bring into consideration appropriate set-theoretical tools, as illustrated. We need to resort to a union-formation operation that takes as input sets of an alphabet but also constituents generated separately, and also to a device that recalls the step and the space where categories are introduced. Some variant of these elements must be assumed to ensure that syntax combines small units into greater units and to distinguish among occurrences.

2.4. INTERNAL MERGE AND EXTERNAL MERGE. I contend, following Chomsky's (2001, 2008) terminology, that *X* is 'externally merged' to *Y* when *X* is either a minimal category selected from the alphabet or a complex syntactic object generated at a different derivational space, and 'internally merged' to *Y* when it is selected from the syntactic domain of *Y*. I emphasize, adapting Chomsky's (2001, n. 29) view, that the virtue of allowing $\mathcal{N}$ to perform internal merge operations is a matter of applicability and of conceptual necessity: it is a matter of applicability because it is a simple analytical tool that is constructively used to capture the property of displacement, which seems ubiquitous in natural languages; and it is a matter of conceptual necessity because only by stipulation could $\mathcal{N}$ be banned from performing internal merge operations, a stipulation that would seem unmotivated, given our current understanding of the syntactic patterns of natural languages.

Recall that, as noted in §1, we form a question by fronting the hierarchically most prominent occurrence of an auxiliary (7), and not by fronting the first occurrence of an auxiliary in the precedence/temporal relationship (3b). In other words, locality conditions on movement operations are structure-dependent; they are not defined on the basis of the precedence/temporal relationship. The definition of (syntactic) domain must ensure that the occurrence is_1^1 (created at the main derivational space D_1) can be fronted to form the appropriate question, whereas the occurrence is_1^2 (created at the secondary derivational space D_2) cannot. I thus provide the following local definition of domain, which will be applied immediately to the study of structure-dependence effects. Note that the elements of a syntactic domain (according to the following definition) are always singletons.

Definition 3. *Given a set M_j^i, $\{x\}$ belongs to the domain of M_j^i ($\Delta(M_j^i)$) iff one of the following conditions is fulfilled: (i) x is an element of M_j^i, or (ii) x is a constituent gen-*

erated at D_i at some step previous to s_j (or in other words, x belongs to the set $\mathcal{C}^i_{j-1}$ of constituents). Symbolically,

$$\Delta(M^i_j) = \{\{x\} : x \in M^i_j \vee x \in \mathcal{C}^i_{j-1}\}.$$

Let us thus apply this definition to the study of structure-dependence effects. We need to determine the elements of the domain of $M^1_2 = \{tall, is^1_1, C^{2/1}_2\}$ in the relevant syntactic derivation, which has been detailed in Table 1.

On the one hand, the domain of M^1_2 contains the singleton sets whose respective members are the three elements of M^1_2, that is, the singletons of the two minimal syntactic categories *tall* ($\{tall\}$) and is^1_1 ($\{is^1_1\}$), and the singleton of the complex syntactic object $C^{2/1}_2$.

(28) $C^{2/1}_2 = \{\{\{talking\}, \{talking, is^2_1\}, \{talking, is^2_1, who\}, \{talking, is^2_1, who, man\}, \{talking, is^2_1, who, man, the\}\}\}$

And on the other hand, the domain of $M^1_2 = \{tall, is^1_1, C^{2/1}_2\}$ also contains the singletons whose elements are the smaller constituents generated at the main derivational space D_1 (which are specified in 29).

(29) $\{C^1_0\} = \{\{\{tall\}\}\}$
$\{C^1_1\} = \{\{\{tall\}, \{tall, is^1_1\}\}\}$

In sum, this is the domain of M^1_2:

$$\Delta(M^1_2) = \cup M^1_2 \cup \mathcal{C}^1_{2-1} = \left\{ \begin{array}{l} \{tall\}, \\ \{is^1_1\}, \\ \{C^{2/1}_2\}, \\ \{\{\{tall\}\}\}, \\ \{\{\{tall\}, \{tall, is^1_1\}\}\} \end{array} \right\}$$

Given that $\{is^1_1\} \in \Delta(M^1_2)$ and that $\{is^2_1\} \notin \Delta(M^1_2)$, we can perform at s_2 of D_1 the operation 30a but not the operation 30b.

(30) a. $M^1_2 \cup \{is^1_1\}$
b. $M^1_2 \cup \{is^2_1\}$

This accounts for the grammaticality contrast between 14 and 4b, that is, for structure-dependence effects, as desired.

Note that, if the syntactic domain of M^1_2 were the linear order

$$C^1_2 = \langle the, man, who, is^2_1, talking, is^1_1, tall \rangle,$$

we would expect is^2_1 to be a legitimate target for a fronting operation, if not the favorite one, given that it is closer to the landing position than is^1_1. The empirical observation behind the SDP simply reveals that the syntactic domain of a set M^1_2 within a given syntactic derivation cannot be the linear order C^1_2, but the set $\Delta(M^1_2)$.

I thus emphasize that the SDP does not indicate that linear order is irrelevant for the syntactic component whatsoever, but rather that the domain for internal merge operations is not a linear order of occurrences but a set of syntactic categories defined in such a way that it bans internal merge operations from crossing derivational spaces.[8] If this remark is neglected, then we prevent ourselves from grounding several basic syntactic notions on a single general concept, namely that of order. Below, in §3, I argue for a

[8] I refer the interested reader to Zwart 2009 for a broader study of opacity effects in terms of layered derivations, that is, in terms of derivations involving multiple derivational spaces, using our terminology. Zwart (2009, 2010) argues as well for the derivation of the LCA from a syntactic algorithm that generates nests and studies how the notion of dependency can be defined from an ordered syntactic representation. The procedure described therein differs from mine in being top-down (see also Fortuny & Corominas-Murtra 2009).

principled account of why the local notion of syntactic domain (**Definition 3**) must be defined in this way.

I conclude this study on the relationship between linear order and structure dependence by observing that the chains of copies generated by internal merge operations can be defined as nests, that is, as linear orders (cf. Nunes 2004). Whereas we say that when the element x is externally merged at s_j it becomes an occurrence x_j, I shall say that it becomes a copy $x_{j/i}$ when it is selected from s_j and remerged at $s_{i>j}$. I thus define the notion of *chain* in the following terms:

Definition 4. *A chain $CH(x_j)$ is a linear order of the copies of an occurrence x_j,*

$$CH(x_j) = \{\{x_j\}, \{\ldots, x_j\}, \{x_{j/k}, \ldots, x_j\}\}.$$

The copy x_j is the tail of $CH(x_j)$, the copy $x_{j/k}$ the head, and any $x_{j/n(j<n<k)}$ an intermediate copy. Multiple copies of x are identified by virtue of the subindex referring to the derivational step where the occurrence has been introduced to the derivation, and distinguished with respect to each other by virtue of the subindexical suffix referring to the derivational step where they are subsequently merged ($_{/i}$). For a given constituent C_i^j, M_i^j is the set of occurrences and copies involved in D_j. The head of the chain is pronounced by the sensory-motor system, whereas the tail and usually the intermediate copies are not externalized, although they remain active at the conceptual-intentional system.

Since $\{is_1^1\} \in \Delta(M_2^1)$ and $\{is_1^2\} \notin \Delta(M_2^1)$, $\mathcal{N}$ can generate outcome 31a but not outcome 31b.

(31) a. $C_2^1 = \{\{tall\}, \{tall, is_1^1\}, \{tall, is_1^1, C_2^{2/1}\}, \{tall, is_1^1, C_2^{2/1}, is_{1/3}^1\}\}$
b. $C_2^1 = \{\{tall\}, \{tall, is_1^1\}, \{tall, is_1^1, C_2^{2/1}\}, \{tall, is_1^1, C_2^{2/1}, is_{1/3}^2\}\}$

The chain of the occurrence is_1^1 formed by internal merge at D_1 is:

$$CH(is_1^1) = \{\{is_1^1\}, \{is_1^1, is_{1/3}^1\}\},$$

where is_1^1 and $is_{1/3}^1$ are, respectively, the tail and the head.

Note finally that this structure-dependent rule also accounts for the observation that 32 is unambiguously the question corresponding to 32a (Berwick et al. 2011).

(32) Can eagles that swim fly?
a. Eagles that swim can fly.
b. Eagles that can swim fly.

The occurrence of the auxiliary *can* generated in the matrix clause in 32a is hierarchically more prominent than the one generated in the relative clause in 32b. The former is externally merged in the main derivational space D_1, whereas the latter is externally merged in a secondary derivational space D_2 that feeds D_1. As a consequence, the former auxiliary, but not the latter, belongs to the search domain, and thus can be internally merged in D_1 in order to form the question in 32, which is, consequently, unambiguous.

3. A principled account of structure dependence.

In the previous section the Fortuny & Corominas-Murtra 2009 theory of hierarchical expressions constructed on the basis of the concept of *nest* was applied to the study of a well-known condition on syntactic operations: the structure-dependence principle (SDP). The observation behind this principle reveals that syntactic operations in natural languages are not carried out by scanning the literal precedence/temporal relationships among the terminals of a sentence but by taking into account the structures into which they enter (Chomsky 1972).

The structure-dependence effects have been accounted for in terms of a local definition of syntactic domain that reduces computational complexity by prohibiting internal merge operations to cross derivational spaces. As a consequence, the syntactic domain for internal merge operations is not given by a precedence relationship among terminals; however, this does not entail that linear order plays no role in syntax. Indeed, the concepts of constituent and chain (**Definition 1** and **Definition 4**) can be defined on the basis of nests, the foundational set-theoretical notion to which order can be reduced (Kuratowski 1921); it is also worth noting, in this regard, that a constituent is a linear order of occurrences and copies, whereby the literal precedence/temporal-order relation among terminals can be directly derived from our definition of constituent. As these considerations reveal, the technical notion of order has a remarkable unificational force for theoretical linguistics. In this sense, it is not the case that order is a secondary phenomenon relevant only to the sensory-motor system, but it is an essential property of syntax and of the syntactic representations that feed the conceptual-intentional system.

Under the light of this proposal, the SDP does not seem to be a primitive or unmotivated principle of UG, contrary to the view originally expressed by Chomsky:

> The structure dependent operation has no advantages from the point of view of communicating efficiency or 'simplicity'. If we were, let us say, designing a language for formal manipulations by a computer, we would certainly prefer structure independent operations. These are far simpler to carry out, since it is only necessary to scan the words of sentences paying no attention to the structure into which they enter, structures that are not marked physically in the sentence at all. (Chomsky 1972:28)

I mention here three clear advantages of structure dependence that reflect an interconnection between semantic/syntactic richness and simplicity, which lead me to argue that structure dependence constitutes neither a primitive of UG nor an imperfection.

First, structure-independent operations based solely on the relation of literal precedence among 'words' are clearly insufficient for setting grammatical operations: they would fail to account for the manipulation of a complex constituent, as in the external merge of a complex phrase, for which it is necessary to bring into account the structure into which words enter. In brief, an artificial language that preferred structure-independent operations would not be able to generate a sentence like *the man who is tall is talking*, whose subject is an internally organized syntactic unit. This point, in fact, has been more forcefully argued for by Chomsky (1965, 1975 [1955]), among many others. Therefore, structure dependence, or the capacity—in a technical sense—of manipulating linearly ordered structures, has a very clear advantage at least in terms of syntactic and semantic richness.

Second, if external merge takes into account linearly ordered (or hierarchical) structures to ensure syntactic and semantic richness, then we expect internal merge to behave the same way. In other words, if, as just argued in the preceding paragraph, external merge allows the possibility of combining complex phrases with an internal hierarchical organization, then we would expect internal merge operations to be dependent on this hierarchical organization as well, and not to be dependent on a different type of organization, like a precedence relation among words. The null hypothesis is that external and internal applications of merge have essentially the same nature, whereby postulating any substantial difference between them would require strong argumentation; and by default, we would expect internal merge applications to be dependent on the structure that external merge applications create. This explains why the notion of local domain (**Definition 3**) of a given outcome M_i^j cannot be the linear order of terminals C_i^j, but rather the set of the members of M_i^j and of the constituents previously generated in D_j.

Importantly, there are independent reasons to claim that internal and external merge behave alike. In this regard, it is important to observe that, if a local domain for internal merge was characterized by a linear order of terminals, then the constituents generated by external merge would be destroyed; they would be unavailable for internal merge. If the domain for internal merge is a precedence relationship among terminals, then only terminals could move; for instance, complex DPs such as *the two candidates* could not undergo movement to an A-position, nor could WH-phrases such as *which of the two candidates* be fronted to the appropriate A′-position. Expressions such as *The two candidates didn't pass the exam*, where the DP has arguably moved to an A-position, and *Which of the two candidates do you prefer?*, where the WH-phrase has been fronted to the appropriate A′-position, would be grammatically unavailable. This reasoning provides a principled account of why internal merge operations must be structure-dependent, just like external merge operations.

And third, as already argued, this local definition of syntactic domain is related to simplicity considerations: derivations involving multiple spaces may be required in order to ensure a certain level of syntactic and semantic richness, but intuitively, internal merge operations crossing derivational spaces seem excessively complex. The structure dependence of internal merge operations (which is ensured by **Definition 3**) is, thus, advantageous from the point of view of structural or derivational 'simplicity', in the sense that it narrows down the search space.

The technical proposals here thus have an interesting consequence concerning the content and the nature of UG: the SDP, which seemed to be an idiosyncratic element of UG that could not be motivated on independent grounds, can now be viewed as a natural consequence of how the generative procedure of language forms constituents and chains by minimizing the search space.

I thus believe that the methodological virtue of this theory of hierarchical expressions contributes substantially to a deeper understanding of the nature of the syntactic component of UG. Last but not least, as noted in Fortuny & Corominas-Murtra 2009, the concept of nest is a powerful abstract entity postulated in different domains, like theoretical biology, statistical physics, and genetics, where a recursive algorithm or an evolutionary process is involved.

Appendix A: On the direction of derivations

In principle, a linear order such as A1 could be read as a precedence relationship or as a successor relationship.

(A1) $\{\{John\}, \{John, kisses\}, \{John, kisses, Mary\}\} = \langle John, kisses, Mary\rangle$

If it is read as a precedence relationship, then *John* precedes *kisses*, *kisses* precedes *Mary*, and by transitivity *John* precedes *Mary*. If it is read as a successor relationship, then *John* follows *kisses*, *kisses* follows *Mary*, and by transitivity *John* follows *Mary*. If A1 is interpreted as a precedence relationship, then $\mathcal{N}$ is a top-down procedure that first merges the specifier of a projection and forms the trivial constituent C_0 containing only the specifier, then introduces the head and forms the constituent C_1 containing solely the specifier and the head, and finally introduces the object and forms the final constituent C_2 containing the specifier, the head, and the object.

(A2)
s_0 : $\{John\}$
$C_0 = \{\{John\}\}$
s_1 : $\{John, kisses\}$
$C_1 = \{\{John\}, \{John, kisses\}\}$
s_2 : $\{John, kisses, Mary\}$
$C_2 = \{\{John\}, \{John, kisses\}, \{John, kisses, Mary\}\}$

Accordingly, the constituents we obtain if $\mathcal{N}$ is read as a precedence relationship are in contradiction with standard constituency considerations. Crucially, there is a constituent C_1 consisting of the specifier and the head to the exclusion of the complement.

However, if $\mathcal{N}$ is assumed to be a bottom-up procedure, then we obtain the correct expected constituency (cf. Fortuny & Corominas-Murtra 2009). For this simple reason I adopt here the bottom-up definition of the nesting machine. Note that the top-down procedure represented in A2 differs from Phillips's (2003) left-to-right procedure for generating syntactic structures, which creates a new constituent by destroying a previous one. Indeed, Phillips's procedure is not top-down, but rather left-to-right.

It must be said that it is possible to define a top-down procedure that generates nests and yields the expected constituency effects; such a device is based not on a successive operation of union formation, but rather on a successive operation of complementary subset formation (Fortuny & Corominas-Murtra 2009, Zwart 2009). An apparent disadvantage of this top-down device for generating nests is that it does not allow us to generate chains of copies by means of internal merge operations, as argued in Fortuny & Corominas-Murtra 2009:106–8.

Appendix B: On composition of relations

I would like to clarify in what sense composition of relations is crucial for obtaining a linear order of expressions containing branching specifiers (see §2.3).

Resorting to composition of relations means that we need to invoke some algorithm to combine order relations 27a and 27b in order to obtain the composed order 27c. Accordingly, we could raise the question of whether this algorithm is operative in narrow syntax or whether it is a mere PF (phonetic form) side effect resulting from assigning a temporal ordering to 27a which contains a component that is also a linear order. The former possibility entails that a linear order of all terminals of a linguistic expression is part of syntax, whereas the latter possibility would support the view that the linear order of terminals is a reflex of the sensory-motor system.

Note that even if it were concluded that a linear order of all terminals were a reflex of the sensory-motor system, this would not necessarily lead us to the view defended in Chomsky 2010, 2013 and Berwick et al. 2011, and in general in mainstream minimalism: for independent reasons we can be sure that the concept of linear order is crucial for syntax, given that—as I argue—central syntactic notions are linear orders, although a linear order of terminals of a linguistic expression would be straightforwardly obtained at PF by means of composition of ordering relations on the basis of the syntactic representations.

The question of which of the two possibilities just mentioned is more appropriate seems to me entirely immaterial; the insight that is important to retain is that, if we define the syntactic procedure in such a way that it generates nests (i.e. orders), then we can understand how central notions of configurational and transformational approaches to syntax emerge from the syntactic procedure responsible for generating linguistic expressions. And once we bring the appropriate set-theoretical concepts, the problem of how a linear order of terminals is obtained from a constituent is easily solved.

REFERENCES

Berwick, Robert C.; Paul M. Pietroski; Beracah Yankama; and Noam Chomsky. 2011. Poverty of the stimulus revisited. *Cognitive Science* 35.1207–24. DOI: 10.1111/j.1551-6709.2011.01189.x.

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

Chomsky, Noam. 1972. *Problems of knowledge and freedom: The Russell lectures*. New York: Vintage Books.

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

Chomsky, Noam. 1995. Categories and transformations. *The minimalist program*, 219–394. Cambridge, MA: MIT Press.

Chomsky, Noam. 2001. *Beyond explanatory adequacy*. (MIT occasional papers in linguistics 20.) Cambridge, MA: MIT Press.

Chomsky, Noam. 2008. On phases. *Foundational issues in linguistic theory: Essays in honor of Jean-Roger Vergnaud*, ed. by Robert Freidin, Carlos P. Otero, and Maria Luisa Zubizarreta, 133–66. Cambridge, MA: MIT Press.

Chomsky, Noam. 2010. Poverty of stimulus: Unfinished business. Lecture presented in the lecture series 'Sprache und Gehirn-Zur Sprachfahigkeit des Menschen', organized by Angela D. Friederici in the context of the Johannes Gutenberg endowed professorship, Johannes Gutenberg University Mainz.

Chomsky, Noam. 2013. Problems of projection. *Lingua* 130.33–49. DOI: 10.1016/j.lingua.2012.12.003.

De Belder, Marijke, and Jeroen van Craenenbroeck. 2011. How to merge a root. *Linguistic Inquiry* 46.625–55. DOI: 10.1162/LING_a_00196.

Epstein, Samuel D. 1998. *A derivational approach to syntactic relations*. Oxford: Oxford University Press.

Epstein, Samuel D. 1999. Un-principled syntax. *Working minimalism*, ed. by Samuel D. Epstein and Norbert Hornstein, 317–45. Cambridge, MA: MIT Press.

Fortuny, Jordi. 2008. *The emergence of order in syntax*. (Linguistik aktuell/Linguistics today 119.) Amsterdam: John Benjamins.

Fortuny, Jordi, and Bernat Corominas-Murtra. 2009. Some formal considerations on the generation of hierarchically structured expression. *Catalan Journal of Linguistics* 8.99–111. Online: https://www.raco.cat/index.php/CatalanJournal/article/view/168906/221175.

Hallet, Michael. 1986. *Cantorian set theory and limitation size*. Oxford: Oxford University Press.

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

Kelley, John L. 1955. *General topology*. New York: Springer.

Kuratowski, Kazimierz. 1921. Sur la notion de l'ordre dans la théorie des ensembles. *Fundamenta Mathematicae* 2.161–71.

Nunes, Jairo. 2004. *Linearization of chains and sideward movement*. Cambridge, MA: MIT Press.

Partee, Barbara H.; Alice ter Meulen; and Robert E. Wall. 1990. *Mathematical methods in linguistics*. Dordrecht: Kluwer.

Phillips, Colin. 2003. Linear order and constituency. *Linguistic Inquiry* 34.37–90. DOI: 10.1162/002438903763255922.

Quine, Willard Van Orman. 1940. *Mathematical logic*. Cambridge, MA: Harvard University Press.

Stabler, Edward P. 2011. Computational perspectives on minimalism. *The Oxford handbook of linguistic minimalism*, ed. by Cedric Boeckx, 617–43. Oxford: Oxford University Press.

Zwart, Jan-Wouter. 2009. Prospects for top-down derivations. *Catalan Journal of Linguistics* 8.161–87. Online: https://www.raco.cat/index.php/CatalanJournal/article/view/168909/221178.

Zwart, Jan-Wouter. 2010. Structure and order: Asymmetric merge. *The Oxford handbook of linguistic minimalism*, ed. by Cedric Boeckx, 96–119. Oxford: Oxford University Press.

Departament de Filologia Catalana
i Lingüística General
Facultat de Filologia
Universitat de Barcelona
Gran Via de les Corts Catalanes, 585
Barcelona 08007, Spain
[jordifortuny@ub.edu]

[Received 18 May 2016;
revision invited 16 April 2017;
revision received 10 October 2017;
accepted pending revisions 30 December 2017;
revision received 13 January 2018;
accepted 13 January 2018]