An introduction to Phase-based Minimalist Grammars: why *move* is Top-Down from Left-to-Right

Cristiano Chesi
CISCL - University of Siena
chesi@media.unisi.it

This paper is an introduction to a grammatical formalism (Phase-based Minimalist Grammar, elaboration of Stabler's 1997 Minimalist Grammar) that includes a revised version of the standard minimalist structure building operations *merge*, *move* and the notion of *derivation by phase* (Chomsky 1999-2005). The main difference with respect to the standard theory is that these devices strictly operate Top-Down and from Left-to-Right. In these pages I will argue that long distance dependencies, such as successive cyclic A'-movement, are better understood within this unconventional (at least within the Minimalist Program) phase-based directional perspective[1].

1. Introduction

The Government and Binding approach (Chomsky 1981) broke with the Transformational Grammar tradition (Chomsky 1957) by shifting the focus of inquiry from the generative procedure of phrase structure building to the configurational patterns apt to discard/license bad/well-formed sentences given their underlying Structural Description(s). Within the Minimalist Program (Chomsky 1995), structure building operations (such as *merge* and *move*) re-gain a prominent position but certain well know problems of directionality of application of these operations, with respect to specific formal tasks such as *generation* and *recognition* (Aho and Ullman 1972) and their performance counterpart, *production* and *parsing* (in the sense of Berwick and Weinberg 1986), are only marginally discussed within the standard framework (Phillips 1996, 2003). This is essentially because the definition of these structure building operations implicitly includes a directionality assumption without discussion.

In this paper I will tackle this assumption and argue that the directionality issue is crucial for understanding certain aspects of the grammar, especially long distance dependencies such as A'-movement: if we assume that a long-distance dependency necessarily includes a unique base thematic position and a unique topmost criterial position (e.g. operator, subject or topic, Rizzi 2004a), from a purely formal viewpoint we might expect that the order of computation of these positions be irrelevant, that is,

[1] This work is a distilled version of the ideas discussed in my Ph.D. thesis defended at the University of Siena (January 2005). I am especially grateful to Valentina Bianchi for the careful discussion of every aspect of this paper. Parts of this work have been discussed at CISCL (2001-07), MIT (2001, 2004), Harvard (February 2006), UPENN (February 2006), University of Geneva (March 2005, 2006), University of Nanzan (February 2007), University of Gerona (June 2007). Thanks to all these audiences for their precious suggestions and remarks.

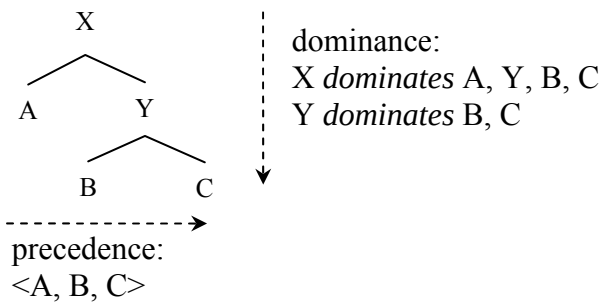
the computation might equally well start from the bottom position (foot) or from the topmost position of the syntactic chain. However, even though this representational approach correctly characterizes the structural constraints on the form of the chain, it does not guarantee that an algorithm exists to build such a chain and that it is computable. The problem lies on the fact that in between the basic/thematic position and the topmost criterial one, there might be a potentially infinite number of intermediate positions which are required to satisfy locality constraints, and which must be created by a recursive procedure. Computationally speaking, when we deal with recursive procedures we need some special care: the grammatical formalism, Phase-based Minimalist Grammar (PMG)[2], outlined in these pages allows for an explicit formalization of a class of computationally tractable minimalist grammars[3]. In particular I will provide an algorithmic definition of the movement operation that is computationally tractable, mainly deterministic and potentially cognitively motivated. In particular, to achieve these results, I will focus on three important properties:

1. structure building operations can be included in the grammar but only if they apply Top-Down, from Left-to-Right;
2. using a Linearization Principle (inspired by Kayne 1994's LCA) and fixing the functional structure by means of a universal hierarchy (cartographic approach[4]) makes the algorithm mostly deterministic;
3. the adoption of the notion of phase (Chomsky 1999) allows us to achieve computational tractability in the computation of a relevant subset of long distance relations.

2. Formalizing grammars

In standard generative approaches, we are used to thinking of a sentence as a bidimensional structure bearing information on both *precedence* and *dominance* relations among lexical items and categorial features, where precedence is a total order among the pronounced elements (namely *words*, which are groups of morpho-phonological features) while dominance expresses the constituency/dependency relations among pronounced and other (abstract/categorial) elements. These two kinds of information are represented by *tree* structures[5] such as the following one:

(1)



A *language* can be extensionally characterized as an infinite set of grammatical expressions each associated with at least one structural description. We assume that an intentional procedure characterizes this set by productively restricting the theoretically

[2] Based on Stabler 1997; see Chesi 2004 for a full preliminary discussion of this approach.
[3] The notion of "grammatical formalism" is thus the formally explicit counterpart of the informal notion of "framework" which has wide currency in syntactic theorizing.
[4] Belletti ed. 2002, Cinque 1999, ed. 2002, Rizzi 1997, ed. 2004b and related works.
[5] Rooted, labeled, oriented trees, in fact (McCawley 1968).

possible precedence/dominance relations that can co-occur within the same sentence/structural description. We refer to this intensional procedure as speaker's *competence* (or *I-language*, Chomsky 1986).

Formally speaking, this competence is represented by a *grammar*: this includes by standard hypothesis[6], at least a specification of a *lexicon* (a finite set of *words*/*morphological units*, built from an *alphabet*, with associated specific abstract/semantic *features*) plus some universal properties (usually encoded as *principles*/*rules*) and language specific options (*parameters*) to derive the combinatorial/recursive potentiality of any natural language. Within this framework, the specification of the *Structure Building Operations* and their order of application is controversial: namely, it could be impossible[7] or at least superfluous[8] to specify once and for all one precise algorithm that recursively defines the procedure for assembling bigger and bigger meaningful units, starting from the lexicon and its properties. Recent minimalist inquiries pay serious attention to this problem, and provide interesting solutions but also raise puzzling problems that need heedful discussion.

In order to do that, we need to make fully explicit assumptions: a complete formalization is a necessary step since it forces us to specify everything we need to describe a language. On the other hand, when choosing a specific formalization (§2.1) we must consider to which extent it transparently encodes linguistic intuitions (§2.2) and at which (computational) cost it does (§2.3, §3).

2.1. A simple formalism: Minimalist Grammars

Stabler (1997) proposes a simple formalization of a Minimalist Grammar (MG) as outlined in Chomsky 1995. Following his work, a MG can be defined as a 4-tuple {**V**, **Cat**, **Lex**, **F**} such that:

(2) **Minimalist Grammar** (**MG**, from Stabler 1997)

V is a finite set of non-syntactic features, (P ∪ I) where
P are phonetic features and *I* are semantic ones;

Cat is a finite set of syntactic features, *Cat* = (*base* ∪ *select* ∪ *licensors* ∪ *licensees*) where
base are standard categories {*complementizer*, *tense*, *verb*, *noun* ...},
select specify selection requirements {=*x* | *x* ∈ *base*} where =x means *base*-driven selection of an x phrase;
licensees specify requirements forcing phrasal movement {-*wh*, -case ...}, -x triggers covert movement, -X triggers overt movement;
licensors are the features that can satisfy licensee requirements {+*wh*, +case ...}

Lex is a finite set of expressions built from *V* and *Cat* (the lexicon);

F is a set of the two partial functions from tuples of expressions to expressions{*merge*, *move*};

[6] I will assume by "standard hypothesis" the Principle and Parameter framework presented in Chomsky 1981.

[7] In a transformational grammar (Chomsky 1957), rewriting rules have to be applied in a precise order, crucially different depending on the sentence to be processed.

[8] As shown by Fong (1991), principles can be applied as filters/generators in many orders and all of them will lead to the same structural representation. The difference would be made in terms of the number of states the algorithm should evaluate before reaching the correct result.

The language defined by such a grammar is the closure of the lexicon (*Lex*) under the structure building operations (*F*). (3) is an example of MG able to deal with simple (i.e. not cyclic) *wh*- movement[9]:

(3) MG example

V = *P* = {/what/, /did/, /you/, /see/}, *I* = {[what], [did], [you], [see]}

Cat = *base* = {D, N, V, T, C}, *select* = {=D, =N, =V, =T, =C}, *licensors* {+wh}, *licensees* {-wh}

Lex = [$_{\text{-wh D}}$ what], [$_{\text{=V T}}$ did], [$_{\text{D}}$ you], [$_{\text{=D =D V}}$ see], [$_{\text{=T +wh C}}$ ∅] [10]

F = {*merge*, *move*} such that:

merge (X, Y) = is a function that takes two adjacent nodes X and Y and returns a modified node X if and only if X has as a first selecting feature, *=F*, and Y has the selected base feature *F*[11]. This operation deletes, by pairing them, *=F* on X and *F* on Y and linearize X and Y as <X, Y> iff *=F* triggers the first merge operation, <Y, X> otherwise.[12]

move (X, Y) = is a function taking a tree [$_{+g\ \text{X}}$ X] and a proper subtree of X of the form [*-g* Y] such that <[$_{+g\ \text{X}}$ X [W [$_{-g}$ Y]]]> (where W can be any possible subtree, even null, with no selecting/selector feature *g* on it) and creates a modified X of the form [[$_{\text{X}}$ Y X [W [t$_{\text{Y}}$]]]

Following Chomsky, a derivation proceeds from bottom to top and *licensees* trigger movement as shown in (4)[13]:

(4) 1. merge ([$_{\text{=}\mathbf{D}\ \text{=D V}}$ see], [$_{\text{-wh}\ \mathbf{D}}$ what]) → [$_{\text{=D V}}$ see [$_{\text{-wh}}$ what]]

2. merge ([$_{\mathbf{D}}$ you], [$_{\text{=}\mathbf{D}\ \text{V}}$ see [$_{\text{-wh}}$ what]]) → [$_{\text{V}}$ [you] [see [$_{\text{-wh}}$ what]]]

3. merge ([$_{\text{=}\mathbf{V}\ \text{T}}$ did], [$_{\mathbf{V}}$ [you] [see [$_{\text{-wh}}$ what]]]) →
[$_{\text{T}}$ did [[you] [see [$_{\text{-wh}}$ what]]]]

4. merge ([$_{\text{=}\mathbf{T}\ \text{+wh C}}$ ∅], [$_{\mathbf{T}}$ did [[you] [see [$_{\text{-wh}}$ what]]]]) →
[$_{\text{+wh C}}$ ∅ [$_{\mathbf{T}}$ did [[you] [see [$_{\text{-wh}}$ what]]]]]

5. move ([$_{\mathbf{+wh}\ \text{C}}$ ∅ [$_{\mathbf{T}}$ did [[you] [see [$_{\mathbf{-wh}}$ what]]]]], [$_{\mathbf{-wh}}$ what])→
[$_{\text{C}}$ [what] ∅ [$_{\mathbf{T}}$ did [[you] [see [*t*$_{what}$]]]]]

It has been shown that the generative power of such a grammar is weakly equivalent to Multiple Context-Free Grammars (thus MGs are included in the Mildly Context-Sensitive class, Michaelis 1998, Harkema 2001), even though it is more powerful than Tree Adjoining Grammars (Stabler 1997), and that several recognizer algorithm can be defined (moreover guaranteeing a polynomial time recognition, Harkema 2001). These results should guarantee that this formalism is powerful enough to express natural language and that an algorithmic way to use it to generate/recognize a language exists and it is computationally decidable. However, it is difficult to draw

[9] For the sake of simplicity, I ignored the options, proposed by Stabler, to distinguish simple merge Vs. incorporation (=x Vs. =X) and covert Vs. overt movement (-f Vs -F).

[10] Whatever *P* and *I* features will be, they are simplified/collapsed in this notation on the same lexical string. The single lexical item is conventionally represented as follows: [$_{\textit{=select* (+licensors) =select* (base) -licensees*}}$ *P**/*I**],

[11] Notice that the history of the derivation is a subtree, in this case of the form [$_{\text{X}}$ X Y].

[12] This straightforwardly creates a right-branching tree.

[13] This is a very simplified derivation for the purpose of exemplification. It would clearly be possible to include subject movement too, but this would have been required extra steps in the derivation.

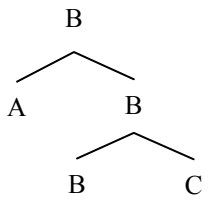
any empirical conclusion from these results, because they are based either on a deductive parsing perspective (Shieber and al. 1995) or on a *weak equivalence* between MGs and other formalisms: namely, such grammatical formalisms can produce the very same set of strings MGs will produce, but either they fail to associate the same structure to these strings[14] or they encode lexical items, features and structure building operations in a non transparent way with respect to the linguistic intuitions that introduced them. In these pages I will argue that these two factors are indeed crucial both in computational and empirical terms. Especially concerning the empirical argument, very little can be said about **V** and **Lex**, which are largely underspecified within the Minimalist Program: Stabler's formalization of these makes the simplest possible assumptions, worth to be kept the way he defined them. On the contrary, the organization of **Cat** in four subclasses of features can be further refined: *base* and *select*, the sets of standard categories, mesh together functional and lexical features (Tense, V, D, N...), but there are strong empirical reasons to believe that this distinction is both theoretically (Belletti, Rizzi 1996) and cognitively justified (Leonard 1998). For instance, the Cartographic approach (Belletti ed. 2002, Cinque 1999, ed. 2002, Rizzi 1997, ed. 2004b and related works) suggests a rich and cross linguistically robust hierarchy of functional projections that redefine the CP, IP and DP geometry adding many levels of projection (§4.2). Moreover a single-level categorization would not be able to predict the correct locality constraints in movement (§5.2). Then I will propose a modification that will lead to a more precise definition of the structure building operations of *merge* and *move* (§4.2 and §5.3 respectively).

Before tackling these issues, I will discuss the relevant relations defined on a *Structural Description* (SD) that will be used in the rest of the paper.

2.2. Structural Descriptions (SDs)

As sketched in (1), we assume that a sentence is structurally described in terms of *dominance* and *precedence*. In a label free system[15] the labels X and Y should be replaced by the projecting head, as shown below:

(5)



In such a tree the following strictly local relations can be defined: first, *immediate dominance*[16] encodes in a transparent way the result of *merge*: B $\lhd$ A means that B merged with A and *projected over* A (namely B is the head of the constituent resulting from merge):

[14] From a strict derivational perspective, deriving different intermediate structures at some point of the computation is a signature of weak equivalence, even thought the resulting tree will be the same in the end (cf. Phillips 1996 for a discussion of the relevance of intermediate structures).

[15] Namely in a representational system that only uses lexical items to describe phrase structures, complying then with a strict version of the *inclusiveness condition* (Chomsky 1995) as discussed in Collins (2001).

[16] This notion of "immediate dominance" corresponds to Stabler's notion of "*project over*". Henceforth I will use the notation "$X \lhd Y$" to express the relation "*X immediately dominates / projects over Y*".

(6) $B \lhd A \equiv merge(A, B) = [_B\ A\ B]$

Second, *immediate precedence* is a restricted version of the classical total strict order (*precedence*) which is defined only between two adjacent nodes: in (5) A *immediately precedes* B, henceforth $A \prec B$, but no immediate precedence relation is defined between A and C. If we consider the classical *precedence* order as a temporal sequence, immediate precedence could underline the instant at which a certain lexical item is linearized. It is clear that the two relations are not redundant since, as shown in (5), given two arbitrary nodes the two relations can either "diverge" (e.g. $A \prec B$; $B \lhd A$) or "converge" (e.g. $B \prec C$; $B \lhd C$): so one does not trivially subsume the other[17]. Obviously these two relations do not exhaust all the empirically relevant relations: since we need to describe phrase structures also in terms of discontinuous constituency groups[18], we will use the notion of *movement* (Chomsky 1995-2005) to capture the idea that (a part of) an element is present in more than one position in the phrase structure, even if it is (fully) pronounced only in one of these positions (usually the structurally highest one). In the next section, I will give a simple definition of movement based on the interplay between immediate dominance and immediate precedence.

2.3. Movement as a Long Distance Dependency

Given a precise formalization of grammar (§2.1) and the two relevant local relations (§2.2), we can define discontinuous constituency relations as follows:

(7) **Long Distance Dependency** (definition)
two non-empty elements enter a long distance dependency (thus forming a *discontinuous constituency relation*) when an immediate dominance relation but no immediate precedence relation is defined between them.

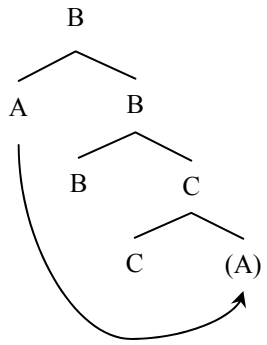
Note that a phonologically null element is not necessarily an *empty element* (since σ and other formal features can be present). For instance, given the information in (8), A and C are linked by Long Distance Relation since $C \lhd A$ is defined but neither $A \prec C$ nor $C \prec A$ are present:

(8)		**L**(exical items)	{A, B, C}
	(immediate)	**P**(recedence)	$\{A \prec B, B \prec C\}$
	(immediate)	**D**(ominance)	$\{B \lhd A, B \lhd C, C \lhd A\}$

Marking as (A) the non linearized copy of A, the SD we want to represent can be visually depicted with the tree in (9):

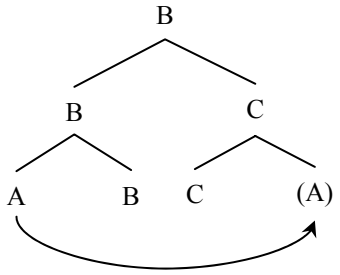
[17] Among other interesting formal properties, notice that both relations are *partial*, *binary*, *intransitive* and *asymmetric*.
[18] This is both because of the "dual semantics property of the conceptual-intentional system", Chomsky (2005:7), and because of the necessity of establishing binding/scope relations.

(9)

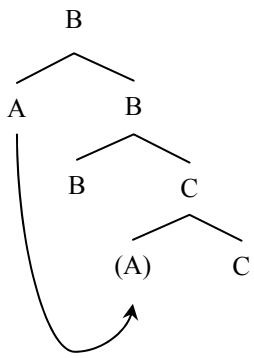


The Long Distance Relation represented in (8)-(9) is essentially an instance of movement (the directionality of the arrow indicates that the highest element provides feature values for interpreting the underspecified features of the lowest one)[19]. The relation between A and (A) is identical to the relation between a moved element and its trace. Note, however, that the information in (8) is ambiguous between the right branching structure given in (9) and other structures such as the ones shown below[20]:

(10) a.



b.



It is possible to rule out the unwanted representations by posing universal constraints on the occurrence of long distance relations and/or on the general shape of the SD; the Linear Correspondence Axiom (Kayne 1994), for instance, does the right job by providing a deterministic mapping between dominance and precedence. Following the spirit of the LCA I assume the following principle:

(11) **Linearization Principle** (**LP**)
if A $\lhd$ B, then either
a. A $\prec$ B if B is a complement of A (that is, A selects B), or
b. B $\prec$ A if B is a functional projection[21] of A

[19] Notice that, formally speaking, this definition is general enough to capture any kind of long distance dependency (e.g. binding, raising and control) whenever we conclude that these relations require a local evaluation, namely a re-merge of some distal element (see Kayne 2002, Hornstein 1999 and Boeckx & Hornstein 2004 for such minimalist proposals; but see also Landau 2003 for a serious critical analysis of any naive attempt to subsume control under movement). From this perspective the relation of *Agree*, as discussed in Chomsky 2005, is not formally distinct from this specific notion of long-distance relation.

[20] The SD in (8) does not exclude multi-dominance relations, thus the *nontangling condition* (Partee and al. 1993: 440) has to be stipulated (or derived) independently (if needed).

[21] Assume *functional projection* to be synonym both of *specifier* and of functional projection, following Starke 2002.

This principle is weaker than the LCA[22] since it does not automatically predict the leftward position of the specifier with respect to its head. On the other hand, if we reject the head-specifier distinction, following Starke's (2002) intuition, there is no clear necessity for this antisymmetric prediction. Notice that both the LCA and the LP predict the very same asymmetric relations among syntactic nodes (antisymmetric C-command in LCA-based theories corresponds to standard dominance, namely immediate dominance plus transitivity, in LP-based theories). In fact, by using the notion of *merge*, here subsumed by the immediate dominance relation as suggested in (6), we can define a (potentially useful) asymmetric relation among the nodes of a tree:

(12) **Asymmetric C-command** (definition)
When two elements A and B merge, each one asymmetrically C-commands the internal constituents of the other.

This definition is sufficient to discard (10.a) under a relatively standard constraint on movement:

(13) **C-command constraint on movement** (definition)
A moved element always asymmetrically C-commands its trace(s)

On the other hand, assuming that the lower position(s) in a chain is(/are) always selected, (11.a) would discard (10.b)[23].

3. Phases and complexity

In the current Minimalist Framework, the operation Move is understood in terms of probe-goal relation between a functional head and an element within its (C-command) domain endowed with "active" features (Chomsky 2005). But the search of a goal can be extremely "expensive" in computational terms. In order to appreciate this point, it is worth making a digression on complexity theory.

3.1. Phases as chunks

Since Miller 1956, it is common to consider relevant portion of information (chunks) instead of unstructured input tokens to account for a reduction of the overall complexity of a cognitive process (e.g. remembering objects). The idea of using *phases* in linguistic computation (Chomsky 1999-2005) is coherent with this intuition. This is at least the abstract motivation for the adoption of the derivation by phase, but the absence of a precise computational specification of this notion makes it difficult to explicitly calculate the actual complexity reduction we could attain, and thus to appreciate the real advantages of this insight.

Recall that the complexity of a problem is an expression of the resources (essentially *time* and *memory*) that are needed to solve it. More precisely, it is a function of the size of the problem, determined by at least three factors:

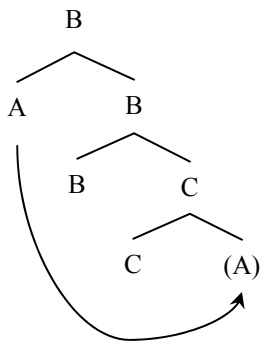
[22] Thanks to Valentina Bianchi for discussions on this point.
[23] But see §5 for a thorough discussion of this point

(14) **complexity factors**:
   a. the length of the input (*n*);
   b. the space of the problem (all the states the system can reach by correctly applying any legal rule);
   c. the algorithm used to explore this space.

While (14.a) is largely independent from the grammar, (14.b) and (14.c) are strictly determined by the linguistic theory. Considering MGs, (14.b) is mostly determined by the lexicon and by the structure building operations Merge and Move; (14.c) is usually assumed to be a bottom-to-top algorithm (namely an algorithm that starts building structures from the inner most constituent, adding piecemeal more peripheral elements[24]) constrained by some economy conditions (shortest move/attract closer, earliness Vs. procrastinate, Merge preempts Move etc.). For example, establishing a long distance dependency by successive cyclic A'-movement is a "complex" problem, because it is necessary to evaluate many (potentially infinite) structural positions (this creates a big space of the problem, (14.b)) and to decide whether or not to stop in each of these position (if the algorithm, (14.c), allows for both options, then it is non-deterministic). The aim of using phases is to reduce non-determinism: more precisely, by chunking the derivation in phases we can reduce the space of most problems by reducing the number of states to be explored, then the options to be considered.

Let us consider successive cyclic A'-movement as a simple combinatorial problem: minimally speaking, any movement operation introduces at least an extra immediate dominance relation[25] (i.e. "C $\lhd$ A" in the example repeated below):

(9)



Combinatorially speaking, exploring which dominance relations have to be associated to a given set of precedence relations has, at least, the complexity order of $\mathbf{O(2^{n-1})}$, where *n* is the length of the totally ordered sequence (namely the number of words in the processed sentence): this is because among *n* items we should define *n-1* relations at best (the minimum number of relations that would make a tree of *n* leafs fully connected) and every relation could be ambiguous as to which element projects (e.g. A $\lhd$ B or B $\lhd$ A). Even if we limit ourselves to the discussion of simple movements[26] the complexity increases significantly: at worse, any item could have been potentially

[24] This is different from Bottom-Up, which technically means processing a rewriting rule (Chomsky 1957) from right to left.
[25] This is, again, largely independent from the specific characterization of the movement operation.
[26] Also without considering successive cyclic movements the argument is sound and significant.

moved from/to any lower (licensed/selected) position, so that the complexity order of the problem increases up to **O(** $2^{(n^2-n)/2}$ **)**: this is because, potentially, any element could establish a dominance relation with any other element that follows it: e.g. with 4 elements {A, B, C, D} we could have 6 possible dominance relations {A-B, A-C, A-D, B-C, B-D, C-D}; with 5 elements {A, B, C, D, E} we could have 10 possible dominance relations {A-B, A-C, A-D, A-E, B-C, B-D, B-E, C-D, C-E, D-E} and so on. Then (((n-1)+1)·(n-1))/2. This is clearly not a satisfactory result, since the growing rate of any of these functions would make the evaluation problem quickly intractable[27].

Interestingly enough, intractability can be tackled by adopting the idea of phase: following Chomsky (2001), let us assume that if movement takes place, it is limited within the phase boundaries: then, given a limited context, either we introduce an extra dominance relation and we license the moved element within the current phase[28], or we put it aside and consider this element within the next phase to be processed. This procedure can be reiterated recursively (and it is potentially non-directional). Only when an extra dominance relation is introduced within the currently processed phase, the element is frozen in place and cannot move further. This mechanism produces a remarkable reduction of complexity: for any moved element, we would have only two possible "landing site" positions within a phase (one *criterial* position and one *selected* position, Rizzi 2004a)[29]. Assuming k to be the fixed maximal size of the phase, then for any phase the number of dominance relations required would be, at worst, $\mathbf{2^{2k}}$ (in case anything would have been moved[30]). The complexity order of the problem, considering any dominance as ambiguous, would be $\mathbf{O(2^{2k})}$.

Interestingly, if we assume that we can process phases "in parallel", we obtain an asymmetry in complexity growth: processing a phase before having closed the previous one leads to a duplication of the number of possible dominance relations and so on as long as new phases are opened. The order of the problem is then $\mathbf{O(2^{p2k})}$ with p representing the number of open phases at the same time. The relation between the length of the input (n) and this function is expressed in terms of phases, since any phase represents a non-overlapping partition of the input; in particular, the number of lexical items in n would determine the number of phases (which could be n, at worst).

[27] We do not need to argue that complexity is, in fact, even worse than that, since unpronounced elements can enter dominance relations (possibly without being linearized) and there is no way to determine how many empty elements could be present, in principle, in a sentence of length n.

[28] For the present discussion the actual position/status of this dominance relation is irrelevant.

[29] It has been sometimes assumed that moved constituents stop by intermediate positions which are different from the "base position" and from the "edge" of each phase (see Sportiche 1988 for a discussion on floating quantifiers). Despite the fact that in order to account for these positions we need to consider a weakening of some stronger assumptions about movement, as long as the number of landing site within a phase is finite, these considerations do not affect the soundness of the reasoning.

[30] Phases are non-overlapping partitions of the input since every node univocally belongs to a single phase and a lexical item is fully inserted/lexicalized in a single (subset of strictly adjacent) node(s).

Functions:	*N° of relations to be evaluated* *n=6* *(p=2, k=3)*	 *n=9* *(p=3, k=3)*[31]
Any option $2^{\frac{(n^2-n)}{2}}$	≅ 32K	≅ 68.000M
Nested Phases 2^{p2k}	≅ 4K	≅ 262K
Sequential Phases $p \cdot 2^{2k}$	128	192

Note that long distance dependencies within a phase would not produce any remarkable effect on the complexity of the problem; discontinuous constituency relations among phases would increase the complexity of the problem in a linear way if each phase is completed before the next one starts (sequential phases), whereas there is an exponential increase of complexity whenever we open a phase before having closed the previous one (nested phases). We will see in §5 that this expected increase of the complexity, caused by the movement of elements across open phases allows for an account of the notion of (strong) islands (Huang 1982).

3.2. Formalizing phases

In more formal terms, what we need from a computational point of view, is a finite structural context within which to calculate the relevant syntactic relations such as movement. As far as I can tell, standard phase theory does not provide any explicit characterization of this point[32].

Let us start from the minimal assumption: the main lexical categories (N(ouns) and V(erbs)) are phase heads[33]: according to the cartographic approach, a finite set of functional categories is rigidly projected above VP and NP (Belletti 2002, Cinque 1997, 2002 and Rizzi 2004b). The left-periphery (in the sense of Rizzi 1997) can be included in this set with one proviso: positions such as Topic cannot be recursive; if we had a recursive category within a phase, we would loose the computational advantages we discussed in the previous paragraph. Then we can only include a finite number of non recursive functional categories[34].

Following Grimshaw (1991), let us assume that functional projections are structured as extended projections of a lexical head; let us consider complement projections as larsonian VP-shells. We obtain the following schematic structure for a phase projected by a lexical V head:

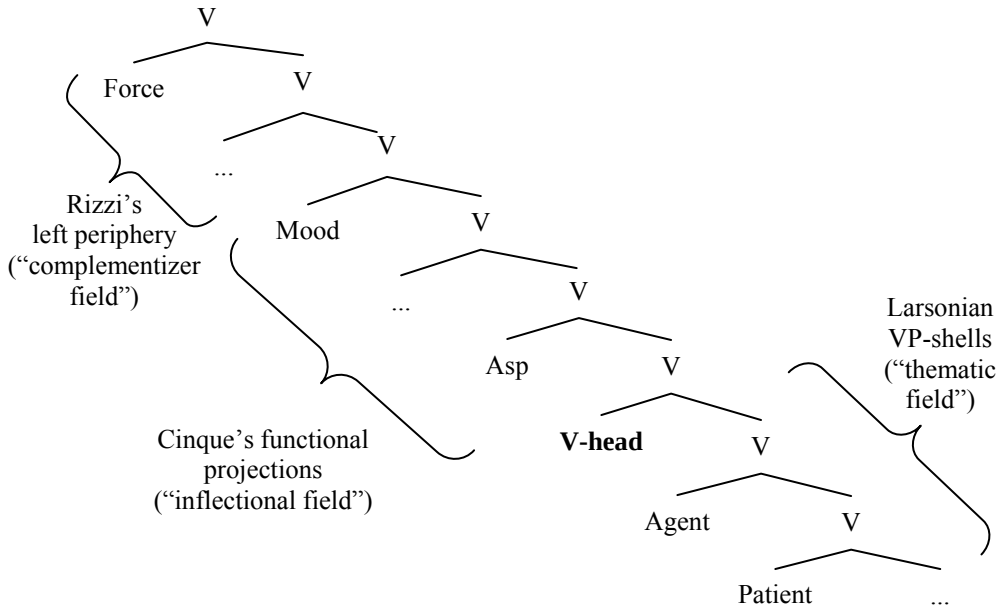
[31] ***p*** is the number of phases in the input, ***k*** is the maximum size of the set of items per phase. Both numbers can be arbitrarily fixed to any finite natural number.

[32] Matushansky 2005 argues for the opposite conclusion, namely that phases are arbitrarily bigger. This conclusion is completely legitimate but it makes the notion of phase irrelevant with respect to the complexity problem (which it was originally meant to solve).

[33] From a formal point of view, nothing prevent us from including Adjectives and other categories in the list of phase heads; the argument is still sound also assuming one single phase-head.

[34] Notice that this way we do not discard the possibility that a single topic position can be filled with complex ph(r)ases each composed by an indefinite number of DPs for instance.

(15)



The verbal head projects upward, dominating (in a label-free grammar, Collins 2001) every functional projection that precedes it, and downward, dominating its complements[35]. This seems to be a step back with respect to Abney's influential work (Abney 1987), but in fact, assuming standard selection among functional projections would obscure the intuition that they are all part of the same phase-projection and they are related to one and the same lexical head.

As for the thematic field, according to Pesetsky (1998) heads can take at most three arguments. This guarantees a maximum finite number of elements per phase: let us say at most *k* functional elements plus at most three complements[36].

It is clear that every element of such a phase can be constituted by another phase: for instance, a DP in a preverbal argumental position would be a nominal phase embedded within a verbal phase; the same holds for PP adjuncts which should be attached/related to a precise functional specification: they are nominal phases within verbal phases[37] (and so on, in a potentially recursive fashion). Then a phase is the minimal computational space defined as follows:

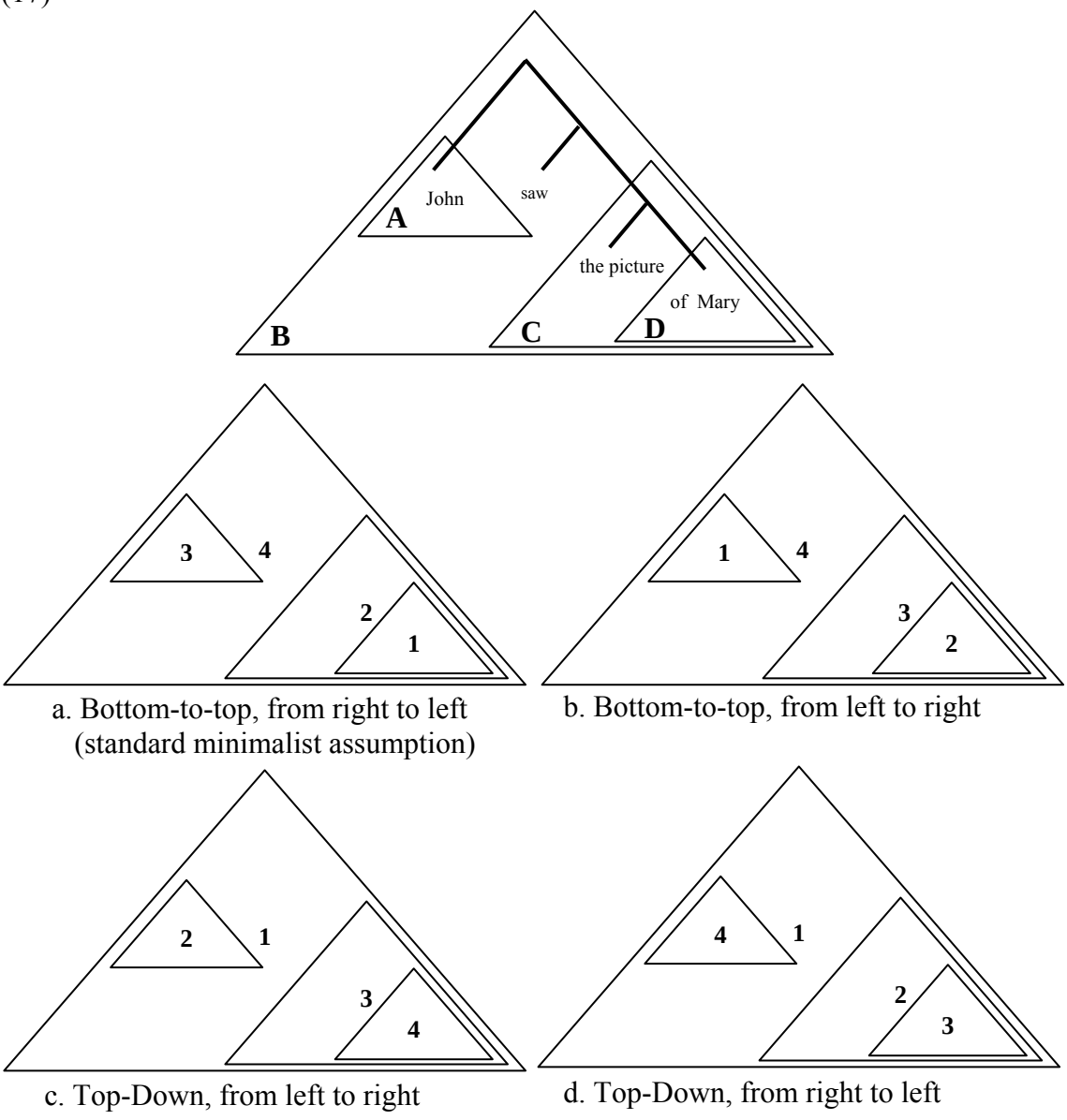
(16) **Phase** (first preliminary definition)
a phase is the minimal set of structural positions that includes:
i. the phase head obligatory filled by a nominal or a verbal lexical category;
ii. a finite (not bigger than *k*, the number of possible functional categories) set of functional specifications of this head;
iii. the minimal (finite, potentially not bigger than three set of argumental positions satisfying the selectional requirements of the phase head.

[35] Larsonian shells are projected in accordance with selectional requirements that should be encoded on the verbal head in the lexicon. Mutatis mutandis, we can assume that nominals project a parallel structure. For an attempt to draw some parallelism between nominal and verbal functional domains see Ogawa 2001, Grohmann 2003 and Laenzlinger 2005 among others.
[36] If we accept Borgonovo and Neeleman's idea (2000), namely that the predicated event can take arguments too, the number of arguments in a phase can be bigger, yet crucially finite.
[37] From this perspective, the preposition is a case marker that take a case peripheral position above the classic DP (e.g. *K* position discussed in Ogawa 2001).

We will say that a phase is complete/saturate when all the mandatory selectional requirements of this head are satisfied, that is, when the relevant set of structural positions (i.e. dominance relations) are introduced in the Structural Description.

Before exploring how phases are introduced/evaluated in the derivation, it is important to consider the logical ways we could have to combine/explore phases within a tree. Ignoring movement, we can take a very simple sentence such as the one in (17) and examine the fourth main strategies of phases composition summarized below:

(17)



a. Bottom-to-top, from right to left (standard minimalist assumption)

b. Bottom-to-top, from left to right

c. Top-Down, from left to right

d. Top-Down, from right to left

Triangles within the tree, tagged from *A* to *D* (depending on the order in which their elements are linearized in the sentence), correspond to nominal and verbal phases as defined in (16); the numbers within the triangles represent some logically possible sequences of computation. In principle, in the example above, every strategy is potentially equivalent since complete (every phase can be enumerated then processed at some point) and sound (given a SD, there are no ambiguities in determining

dominance and precedence relations among phases, so that every strategy is computable and deterministic). But this is not a general property:

(17') Probably John suddenly saw the picture of Mary
[$_B$ Probably [$_A$ John] suddenly saw [$_C$ the picture [$_D$ of Mary]]]

In (17') the adverbial *probably* belongs to the verbal matrix phase B which dominates the nominal phase *John* (A). The set of (immediate) dominance relations is unchanged from (17) to (17') (i.e. B $\lhd$ A, B $\lhd$ C, C $\lhd$ D) but since dominance is a partial order (e.g. no dominance relation can be defined between A and C), dominance, per se, is not sufficient to disambiguate the processing sequence (e.g. choosing between processing phase A or phase C first is totally arbitrary). While in (17) *precedence* is sufficient to unambiguously order the four phases, in (17') this relation is less clear: certain lexical items of phase B both precede (*probably*) and follow (*suddenly*, *saw*) certain lexical items of phase A (*John*); then, precedence among elements belonging to different phases can not be independently used to disambiguate the processing sequence.

Hence we need to specify a principled way to order our sequence. The recursive nature of certain tree branches require a careful examination: in the SD (17), if we assume that any phase/subtree can be recursive, the only way to obtain an effective phase numeration would be to proceed top-down: this is because the Single Root Condition[38] which guarantees an unique (unambiguous) starting point[39]; then, ply by ply we can enumerate the phases to be expanded at the next step. Notice that since phases are finite domains, at every level the number of phases to be processed are at worst $c \cdot p$ (where c is the maximum finite number of elements that can be expanded within a phase, p is the number of phases at the processed ply). On the other hand, it is impossible to enumerate phases from-bottom-to-top since the n^{th} ply, which would be our starting point from bottom to top, is arbitrarily large, potentially infinite: hence we cannot guarantee that the numeration will ever start to produce elements at level *n-1*. On the other hand, there are no formal arguments to prefer Left-to-Right or Right-to-Left directionality, there is however an empirical one[40]: center embeddings, or better, counting/mirror recursion (corresponding to recursion in the position of the phase *A* in (17)) is highly marginal in natural language[41]; for instance, recursive complementation/relativization is highly restricted within a subject phase[42] but quite free within the last selected object[43]:

[38] In every well-formed constituent structure tree there is exactly one node that dominates every node (Partee and al. 1993:439).
[39] Cf. Kayne 1994.
[40] Besides the intuitive one: humans produce and parse strings sequentially from Left-to-Right.
[41] Even more restricted/marginal than identity recursion (i.e. cross-serial dependencies), which require more powerful grammars such as context sensitive ones (counting/mirror recursion simply needs context-free grammars). See Christiansen & Charter (1999) for discussion of these data.
[42] This is true for VO languages. Something needs to be said for OV languages, but the discussion of this point goes behind the goal of this paper.
[43] It may be objected that the contrast between (18.a) and (18.b) is merely matter of performance. A full discussion of the distinction between competence and performance would exceed the limits of this short paper (cf. Chesi 2004). This has nothing to do with relativization preferences discussed in Keenan 1975 (see Hawkins 1994 for some relevant data on extraposition and heavy NP-shift supporting this point).

(18) a. ??[venne l'acqua [che il fuoco [che il bastone [che il cane [che il gatto [che
it came the water, that the fire, that the stick, that the dog, that the cat, that
il topo [...]] mangiò] morse] picchiò] bruciò] spense]
the rat ate, bit, hit, burnt, extinguished

b. ... [venne l'acqua [che spense il fuoco [che bruciò il bastone [che picchiò il cane [che morse il gatto [che si mangiò il topo [che ...]]]]]]
"it came the water, that extinguished the fire, that burnt the stick, that hit the dog that bit the cat that ate the rat that..."

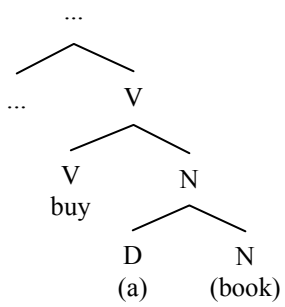
This would suggest that (at least in SVO languages) the last object is a privileged branch in recursive terms. Then, as a first approximation, we will formalize a very restrictive grammar which only allows for right-recursion (i.e. within the *C* / *D* phases of (17)). This would allow us to capture only right-recursive languages (i.e. languages that are generable by simple Regular Grammars, Chomsky 1956). In order to process recursive nodes at last, enumerating phases from Left-to-Right is the easiest way to reconcile precedence and dominance in a consistent and complete phase-based processing strategy.

Once we have fixed the "horizontal" dimension of phase processing, we can encode in our grammar the intuition that phases are generated/processed top-down by exploiting the selection properties of lexical items (discussed in (2)) as follows:

(19) **Phase Projection** (definition)
a lexical head projects the minimal SD (that is the minimal set of necessary immediate dominance relations) based on its *select features*.

For instance, as soon as we process a verbal head with a select feature requiring a determined nominal (i.e. [+D N]), we add to the SD the minimal set of dominance relations that satisfy this requirement, e.g. V $\lhd$ N, N $\lhd$ D, or, more explicitly[44]:

(20)



[44] Notice that specifying the set of precedence relation would be redundant, since this is implied by LP and by (15).

The notion of phase in (16) can be formally refined as follows:

(21) **Phase** (second preliminary definition)
a phase is a complete derivation involving:
- a *Phase Projection* of a potential SD,
- a set $\boldsymbol{i_p}$ of lexical/functional items such that any item is either the head of the phase, a complement or a functional specification of the head of the phase.

How this set of items is actually chosen is an independent problem which I cannot discuss here[45]. I would like to discuss instead some interesting consequences of the definition in (21):

(22) Consequences of the phase definition:
a. **completeness**
a phase is complete if the head of the phase is saturated, namely if all the mandatory selectional requirements (agent, patient etc.) are satisfied/ projected. The last Phase Projection closes the phase generating this expectation;
b. **phase selection/licensing requirement**
a phase is always licensed/selected (exceptional default selection licenses the first projected phase); unselected phases can be introduced in the computation before their selection point to comply with a functional specification of the superordinate phase (according to LP and to 15).

(22.a) tells us when the processing procedure can abandon, once and for all, the processed phase[46], obtaining the computational advantages discussed in §3.1: once the algorithm has processed the head (N or V) of the phase, it will process the ordered list of selectional requirements which are projected as independent phases. We should stress that the last selected complement projection is the last operation triggered by the previous phase; then its processing can be considered a sequential phase[47]. On the other hand (22.b) guarantees that any phase is part of the tree and that it is connected to a previously processed phase by a dominance relation that complies with either a functional or a selectional requirement.

3.3. Including phases in a Minimalist Grammar

We can now include the phase idea in our grammar[48]. As we saw in §3.2, there are at least two things we need to guarantee: first, the number of elements within a phase should be finite; second, we need to encode in a precise way the ordered set of functional elements. One way to avoid problems with the arbitrary introduction of (unordered, infinite) functional elements within the phrase structure would be to

[45] Exactly the same "problem" underlies the numeration idea (Chomsky 1995).
[46] Complying with the no tampering condition (Chomsky 1995).
[47] See Bianchi and Chesi (2005) for a discussion of right hand modifiers, and Chesi (in progress) for a discussion of extraposition and heavy NP-shift.
[48] I will not discuss in these pages the similarity of this proposal with respect to other influential frameworks (such as Dynamic Syntax, Cann and al. 2005) nor the resemblance of the notion of Phase Projection to the elementary trees discussed in TAG approach (Joshi 1985, Frank 2002); notice however that these assumptions are not completely original. Refer to Chesi ed. (in progress) for a comparison of these frameworks.

import into Stabler's formalization a stipulation on the universal order of functional projections, as discussed in (15). This specification can easily be incorporated by imposing an order on the *licensor* features (notice that this order is implicitly both linear and hierarchical); the *licensors* subset (which I will call *functional*, following the standard minimalist terminology) can be defined as an ordered set as follows:

(23) *licensors/functional features:*

V^{49}: <Force, Top, Int, Top, Focus, Mod, Top, Fin, Mood $_{speech\ act}$, Mood $_{evaluative}$, Mood $_{evidential}$, Mood $_{epistemic}$, T $_{past}$, T $_{future}$, Mood $_{irrealis}$, Mood $_{necessity}$, Mood $_{possibility}$, Mood $_{volitional}$, Mood $_{obligation}$, Mood $_{permission}$, Asp $_{habitual}$, Asp $_{repetitive\ I}$, Asp $_{frequentative\ I}$, Asp $_{celerative\ I}$, Asp $_{perfect}$, Asp $_{retrospective\ I}$, Asp $_{proximatives}$, Asp $_{durative}$, Asp $_{generic/progressive}$, Asp $_{prospective}$, Asp $_{sg\ completive\ I}$, Asp $_{pl\ completive\ I}$, Voice, Asp $_{celerative\ II}$, Asp $_{repetitive\ II}$, Asp $_{completive\ II}$, T $_{anterior}$, Asp $_{terminative}$, Asp $_{continuative}$, Asp $_{frequentative\ II}$>

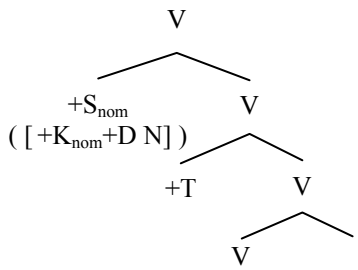
N^{50}: <K(case), D, ordinal number, cardinal number, subjective comment, ?evidential, size, length, height, speed, ?depth, width, weight, temperature, wetness, age, shape, colour, nationality/origin, material, compound element>

Given this restrictive structural skeleton, the most natural way to accommodate functional projections and the phase environment as part of the grammatical formalism is to define phases as structure building operations **F**: *Phase Projection* can be thought of as a finite set of partial functions from tuples of expressions to expressions such that for any select feature inspected, they add the minimal (pre-compiled) SD to comply with the relevant selectional requirements. Selection requirements can be expressed simply in terms of dominance relations as follows:

(24) *declarative*:
$\{V \lhd S(ubject)_{nom(inative)}, V \lhd T\}$

(S_{nom} = N-phase of the kind: $\{N \lhd D, N \lhd K_{nom}\}$; subscripts ($S_{nom}$, K_{nom}) express feature value/subclass)

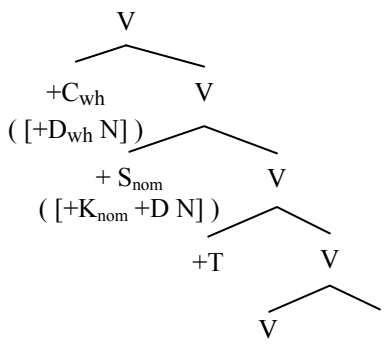
minimal SD projected:



Wh- question:
$\{V \lhd C_{wh}, V \lhd S_{nom}, V \lhd T\}$

(as before, S and C are N-phases; subscripts (C_{wh}, D_{wh}) express feature value/subclass)

minimal SD projected:



[49] Feature hierarchy based on Rizzi (1997, 2004b) and on Cinque (1999). See Cardinaletti (2002) on *subject* positions and Zanuttini (1997) on *negation* positions.
[50] Based on Scott (1998).

For every phase, every node will be processed/expanded from Left-to-Right in compliance with the LP and the order imposed to licensors/functional projections by means of the other structure building operations Merge and Move, which are defined in the following sections.

4. Merge as unification

A standard assumption within the minimalist mainstream is that *merge* is the first source of recursion in natural language:

(25) merge (Y , merge (X, ... merge (C, merge (A, B))...))

A, B, C, X and Y are either lexical elements or the results of other *merge* operations; in the second case, *merge* is applying in a recursive fashion:

(25') merge (*John* , merge(*said*, ... merge(*Mary*,
merge(*bought*, merge(*the*, *book*)))...))

Notice that in the previous paragraph we have already introduced a similar concatenating operation: Phase Projection allows for a comparable structural composition, though constrained by selectional features. Notice also, that Stabler (1997) assumed a more constrained version of *merge* than the standard minimalist one: in fact, it would be extremely inefficient to include in our grammar an operation that freely creates structures that have then to be filtered out[51]; in order to avoid filtering, structure building operations in (2) delete information (*destructive feature checking*, Stabler 1997) by feature checking/pairing as exemplified in (4). This has two interesting effects: first, the very same feature (e.g. N in [$_{N}$ dog]) can not enter more than one *merge* relation; this is potentially a welcome result since, for instance, we can prevent the very same element from receiving more than one thematic role. Second, a given feature can not select different categories. This solution is indeed not completely problem free: first, as discussed in Stabler (1997), successive cyclic A-movement (e.g. *John*$_i$ appears t_i to seem t_i ... to be t_i happy) is problematic from this perspective, since base features should not be deleted in these cases; second, rigid selection causes easily detectable problems as shown by the lexicon fragment reported below:

(26) **lex** = [$_{=Asp\ \ Mood}$ probably], [$_{=V\ \ Asp}$ suddenly], [$_{N}$ Napoleon], [$_{=N\ V}$ died]

With such a lexicon we could easily generate the sentence (26.a), but not (26.b):

(26) a. Probably suddenly Napoleon died
b. *Probably Napoleon died

This is because the derivation fails as soon as we merge the V complex [$_{V}$ [$_{N}$ Napoleon] died] with [$_{=Asp}$ probably]: *merge([$_{=Asp\ Mood}$ probably], [$_{V}$ [$_{N}$ Napoleon] died]) is undefined since the selectional requirement =Asp cannot be satisfied. Neither

[51] See Fong 1991 for a discussion of the inefficiencies related to the free application of structural generators.

changing the =Asp requirement of [Probably] to =V would solve the problem (this would make sentence (26.a) underivable), nor it does to change the [$_{=N\ V}$ died] entry with [$_{=N\ =Mood\ V}$ died]; in fact, this would yield an incorrect prediction on the grammaticality of sentences without adverbials:

(26) c. *[$_{=Mood\ V}$ [Napoleon] died] (=Mood would not be satisfied)

A solution for this puzzle (which is adopted, for instance, in some lexicon fragments in HPSG) would be to multiply lexical entries:

(27) *probably*: [$_{=Tpast\ Mood}$ *probably*], [$_{=T\_future}$ Mood *probably*],
[$_{=Asp\_completive\ Mood}$ *probably*]
[$_{=V\ Mood}$ *probably*], [$_{=T\_past\ =Asp\_completive\ Mood}$ *probably*],
[$_{=Tpast\ =Asp\_completive\ =V\ Mood}$ *probably*] ...

But this solution would produce *n!* lexical entries for *n* possible selecting features. This introduces a high level of unwanted ambiguity in our lexicon.

Another example of "inadequacy" of *merge* comes from agreement facts:

(28) *il gatti (Italian)
the$_{sing}$ cats $_{pl}$

The simplest hypothesis to explain the agreement requirement between nouns and determiners (and every adjectival form within the nominal domain) in Italian (as in many other languages) is to pose an agreement constraint on the success of a *merge* operation. This way the "simple" function that "takes two elements α, β already constructed and creates a new one consisting of the two" (Chomsky 2001:6) would become more complex, but definitely more constrained from an empirical (and computational) point of view. There is, in fact, a well known information combining operation that seems to do the wanted job: this is *unification*[52].

To sum up, it is inefficient to freely build structures that then have to be filtered out, so we need more specific devices to avoid building inconsistent SDs. Marking explicitly lexical items for *merge* by simply using select features, as Stabler does, is an interesting solution, but it is insufficient to discard some unwanted structures; moreover, it carries a big computational cost as soon as we try to capture optional functional selection of the type exemplified in (26)-(27).

4.2. Including unification in a Phase-based Minimalist Grammar

We have just shown that the simple idea of linking the *merge* operation to a single selecting mechanism (=f) turns out to be empirically problematic if we assume that any selecting/selected pair of features has to be deleted for convergence. In (23) I suggested that functional features should not be included in the *base* set but in the *functional* one (following Grimshaw's extended projection intuition). This, however, does not automatically solve the problem: there are functional elements that are optionally selected (for instance, adverbs) while others seem to be mandatory (like

[52] Up to now, the expressions in *Lex* are considered "flat" just because we do not need any more complex devices to describe the basic operations within this formalism introduction. A translation of these expressions in complete features structures such as *Attribute-Value Matrices* is however possible and desirable. This could lead to include in our grammar an *unification algorithm* similar to the one used in G/H-PSG (cf. Shieber 1986) that I can not discuss here.

determiners in some contexts/languages). As we saw in the preceding section, this asymmetry can be captured, within the Extended Projection assumption, if we assume a projection operation that includes some sort of subcategorization option: in this way, the *select* subclass is not just the mirror of *base*, but it is a set built from {*base* ∪ *functional*}. For instance =[+D N] projects a noun phrase where the functional *D* feature will be satisfied either via movement or by determiner insertion. I will procrastinate the discussion of the movement operation up to next section and I will now concentrate on lexical insertion: let us restrict the *merge* definition simply to the lexical insertion scenario:

(29) **Merge** (definition)
a partial function that unifies a lexical item with a (set of) *functional*/*base* features already present in the SD.

The function is partial since it fails (or it is undefined) for certain inputs (e.g. *merge*(D[number=plural], N[number=singular])). Going back to the previous example, if a V-head selects an N-phase with a D functional specification (i.e. $[_{=[+D\ N]}$ V *run*]) once we process this phase, from Left-to-Right, we have the following viable options:

(30) a. merging a proper noun: e.g. *merge*([+D N], $[_{+D\ N}$ John]) = $[_{+D\ N}$ John]; this would result in a fully lexicalized feature structure where both features are satisfied by a single lexical entry (this simply ends the derivation of the nominal phase);
b. merging a determiner, e.g. *merge*([+D N], $[_{D}$ the]) = $[_{+D}[_{D}$ the] $_{N}$], lexicalizing the D feature, then processing the next projected feature (i.e. N)
c. merging a functional element that is compatible with a functional specification of the related phase head (e.g. *merge*($[_{+D}[_{D}$ the] $_{N}$], $[_{+ADJ}$ nice]) = $[_{+D}[_{D}$ the] $_{+ADJ}[_{+ADJ}$ nice] $_{N}$])

It is easy to prove that merging an incompatible functional feature with a given phase-head or two features differing in values is excluded. In this sense we can think of Merge as a way to lexicalize a categorial feature strictly from Left-to-Right. [53]

5. Movement as a Left-to-Right dependency

Another source of recursion in natural language is successive cyclic movement[54]:

(31) merge (move (A), ... merge (move (A), ... merge (A, B)...)...))

A classical example is successive cyclic *wh*-movement in head-initial languages:

(31') *Who*$_i$ do you believe [t_i that Mary think [t_i that ...
[t_i that everybody admires t_i]...]] ?

[53] Recall that the linearization of functional features is deterministically predicted by the imposed ordered list among features and by LP.
[54] This is true at least for head-initial languages. Something has to be said for head-final languages but for sake of compactness I can not address the issue here.

Independently of the specific devices used to account for displacement (probe-goal, edge features etc.), in a bottom-to-top derivation, every intermediate step, required to escape the phase impenetrability condition (or subjacency or barrier-hood, depending on the framework), has to be triggered by some feature: in the classical minimalist understanding of *wh*-movement (Chomsky 1995), for instance, the displacement operation could not alter the relevant *wh*- feature that triggers the last step of the *wh*-chain. If this were allowed, either we would have a non-deterministic operation that could at any step leave the element undisplaced or move it at will (leading, in any case, to a grammatical result) or else prevent the element from moving further after the first step (possibly stranding it in a ungrammatical position). A relevant (and recurring) question, then, is what triggers intermediate steps in successive cyclic movement.

5.1. On intermediate steps

Within the minimalist mainstream, two influential solutions have been proposed in order to make successive cyclic A'-movement possible:

i. Formal Features (FFs), e.g. Probe-Goal approach described in Chomsky (2000:135);
ii. Edge Features (EFs), Chomsky (2005).

The nature of the two solutions differs substantially in terms of paired deletion: while FFs are selected both on the probe and on the goal and delete against one another, causing a freezing effect ((32.a-a'); Rizzi 2004a), EFs are present only on the phase head and force *internal merge* as shown in (32.b-b'):

(32) a. ... [$_{+FF}$ C] everybody admires [$_{-FF\ -WH}$ who]?[55]
a'. ... [~~$_{+FF}$~~ [~~$_{-FF}$~~ $_{WH}$ who] C] everybody admires <who> ?
b. ... [$_{EF}$ C] everybody admires [$_{WH}$ who]?
b'. ... [~~$_{EF}$~~ [$_{WH}$ who] C] everybody admires <who> ?

FFs describe movement strictly in terms of uninterpretable-interpretable feature checking, while EFs charge the entire burden of the movement operation on the phase-head which forces *internal merge* to comply with scope/discourse-related "edge requirements".

Both solutions leave some theoretical problems unanswered. As for the first solution, the use of FFs would predict any single step to be triggered by a single (different) feature. Unless we assume that A'-move is indeed a non-recursive operation, successive cyclicity would need a potentially infinite number of formal features on the *wh*-element to be moved[56]:

(33) a. [$_{+WH}$ C] do you think [$_{+FF}$ C] Mary said ... [$_{+FF}$ C] everybody admired [$_{-FF\ -FF\ ....\ -FF\ -WH}$ who]?
a'. [~~$_{+wh}$~~ [$_{-WH}$ who] C] do you think [~~$_{+FF}$~~ <~~$_{-FF}$~~ $_{-WH}$ who> C] Mary said ... [~~$_{+FF}$~~ <~~$_{-FF}$~~ $_{-FF\ ....\ -FF\ -WH}$ who> C] everybody admired <$_{-FF\ -FF\ ....\ -FF\ -WH}$ who> ?

[55] The formalism used here (+/-FF) is inspired by Stabler's (1997) proposal.
[56] For sake of simplicity I have not introduced +FF on the edge of *vP*. This would have been obviously unproblematic.

This is in contrast with the finitary nature of the lexicon[57].

On the other hand (Chomsky 2005:5,17), EF on a phase head would require movement (*re-merge* or *internal merge*) of the *wh*-element to the (relevant) position in the edge of the phase. This would allow successive cyclicity without requiring an infinite number of features on the *wh*-phrase:

(34) a. [$_{EF}$ C] do you think [$_{EF}$ C] Mary said ... [$_{EF}$ C] everybody admired [who]?
a'. [who ~~$_{EF}$~~ C] do you think [<who> ~~$_{EF}$~~ C] Mary said ... [<who> ~~$_{EF}$~~ C] everybody admired <who> ?

However, a problem with this device arises from the definition of *internal merge*, which is a genuine instance of *merge*: it is not clear (at least to me) how EFs should force *internal* Vs. *external merge* if the element to be displaced is not marked in any way. Why could EF on C not be satisfied by an expletive or, in general, by *external merge*? Another problem comes from the fact that we should expect a minimal variation of (34) to be grammatical, contrary to the facts:

(34') a. John thinks [$_{EF}$ C] Mary said ... [$_{EF}$ C] everybody admired [who]?
a'. *John thinks [who ~~$_{EF}$~~ C] Mary said ... [<who> ~~$_{EF}$~~ C] everybody admired <who> ?

The intimate relation between the *wh*-element in its verbal selected position and the very same element in its scopal position (the dual semantics property of the conceptual-intentional system, Chomsky 2005:7) is lost if we do not have any featural mark-up which identifies both the *wh*-element and the proper scope ("criterial") position.

It is worth to mention a variant of the first mechanism, which is suggested by Luigi Rizzi (2004): Rizzi proposes to eliminate the +/- distinction, marking both the *wh*-element and the attracting complementizer head with the very same feature *F*. In order to account for successive cyclicity, Rizzi includes formal features *f* (crucially different from the one that triggers the last step in the bottom-to-top derivation and creates the freezing effect) in the sense of McCloskey (2002). Following this idea, the derivation can be rephrased as follows:

(35) a. [$_{F}$ C] do you think [$_{f}$ C] Mary said ... [$_{f}$ C] everybody admired [$_{F}$ who]?
a'. [~~$_{F}$~~ [~~$_{F}$~~ who] C] do you think [~~$_{f}$~~ <$_{F}$ who> C] Mary said ...
[~~$_{f}$~~ <$_{F}$ who> C] everybody admired <$_{F}$ who> ?

This solution removes three important problems: first, it removes the inconvenient asymmetry between interpretable and uninterpretable features (as Rizzi noticed, we can hardly believe that question-related criterial positions and/or *wh*-elements involve morphologically overt uninterpretable features); second, it does not violate the finitary nature of the lexicon (since the *wh*-element only bears one relevant *F* feature); third, it is compatible with the freezing effect and it can circumvent the expletive-strategy (when the *wh*-element is marked with *F*, if an expletive satisfies the complementizer *F* requirement by matching, then the *F* features on the *wh*-element would stay unmatched; if we assume that *F* requires a matching in a criterial position, sentences like (34') are correctly predicted to be ungrammatical).

[57] Notice that this problem affects Stabler's definition of movement as well.

There is however one last standing problem: why are formal features *f* inserted phase-by-phase in the computation? This is nothing but a teleological device, in order to allow the *wh*-phrase to reach the criterial position. This can hardly be considered an explanatorily adequate answer.

5.2. Relativized Minimality

Another empirical problem is how to capture *relativized minimality* effects. The definition of movement in (3) is not sufficient to predict the correct constraints: as argued by Rizzi (1990), intervention effects in movement are not just a matter of identity of feature, but rather of classes of features. For instance, the movement of a *wh*- element is blocked not only by the intervention of another *wh*- element, as in (36.a).

(36) a. *how_i do you wonder <u>why</u> I should cook this stuff t_i?

but also by an intervening negation (36.b), a focalized element (36.c), or a quantificational adverbial (36.d) (Starke 2001:5):

(36) b. *how_i don't you think that I should cook this stuff t_i
b'. *how_i do you think that I should<u>n't</u> cook this stuff t_i
c. *how_i do you think that, <u>THIS STUFF</u>, I should cook t_i,
(not those eggplants over there)
d. ?*how_i should I <u>often</u> cook this stuff t_i?

It is possible to define a natural class of functional features relevant to minimality:

(37) Q(uantificational) {wh, negation, focus, quantificational-adverbs}

All elements bearing one of these features are potentially deadly interveners for movement of an element bearing a feature in the same set. Following Rizzi (2004), the classical A' class, can be articulated in the following categories:

a. Topic {topic}
b. Q {Wh, Neg, measure, focus, ... }
c. Mod(ifier) {evaluative, epistemic, Neg, frequentative, celerative, measure, ... }

Argumental features represent distinct class of interveners (A-chains/movement):

(38) A(rgumental) {person, number, gender, case}

Another refinement of the movement operation is then required to capture intervention effects.

5.3. Implementing movement

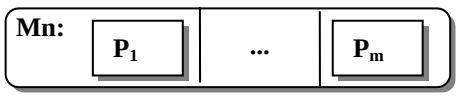
In the preceding sections I have pointed out two problems for the standard view of *move*: first (§5.1), the teleological nature of the mechanisms triggering intermediate movement steps; second (§5.2) the fact that intervention effects cannot be expressed in terms of identity of features.

A solution for both problems emerges if we adopt a mechanism that is widely adopted in the literature on parsing, namely, memory buffered long distance dependencies. Minimally speaking, in order to implement a memory-based dependency we need:

a. a definition of the memory buffer;
b. a deterministic procedure to move elements into the memory buffer;
c. a deterministic procedure to reintegrate moved elements (from the memory buffer) in the structure;
d. a success condition that allows us to determine, depending on the content of the memory buffer, whether the sentence is grammatical or not.

a. As for the structure of the memory buffer, let us assume a standard Last In First Out (LIFO) memory (used for instance in push-down automata[58]): the last element moved in the memory buffer is the first one to be re-inserted in the structure. This would allow us to easily capture nested dependencies[59]. Henceforth I will use a graphical representation (rounded boxes) to schematically represent the memory buffers:

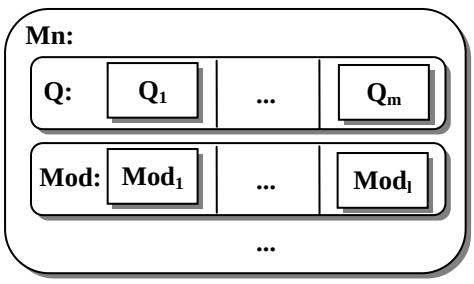
(39)



M stands for M(emory)-buffer and will be followed by an identification number/letter *n* which will relate it, univocally, to a specific syntactic object (i.e. a phase in a top-down derivation). *P1*... *Pm* are elements stored in the memory buffer: *P1* is the first inserted element, *Pm* the last inserted one. Due to the LIFO structure of the memory buffer, *Pm* will be retrieved and reintegrated in the phrase structure before *P1*.

We can impose a multi-layered structure to the memory buffer so as to deal with relativized minimality effects: introducing a set of classes of functional specifications, we can predict selective intervention effects as discussed in §5.2. This is the schematic representation of a multi layered (LIFO) memory buffer:

(39')



b. c. In order to regulate storage of an element within the memory buffer and the unloading procedure, we use the LP, repeated below:

[58] This is not a dangerous assumption in computational terms: our grammar would be no more powerful than context-free grammars. See Hopcroft and al. (1990) for a discussion of this point.
[59] Probably, this assumption will turn out to be too strong and it could be somehow weakened in order to capture some relevant empirical phenomena (cross-serial dependencies, apparent freedom in word order for topicalized elements). See §6 for discussion of these points.

(40) **Linearization Principle** (**LP**)
if A *dominates* B, then either
a. <A, B> if B is a *complement* of A (that is, A selects B), or
b. <B, A> if B is a *functional projection* of A

According to the LP, an argument is inserted in the memory buffer if it is placed to the left of the phase-head: *Wh*-elements in the left periphery, topicalized arguments and the subject in SVO languages, for instance, will be inserted in the appropriate layer of the memory buffer and discharged in the argumental positions as soon as they can be selected. This leads to a theory in which *move* preempts *merge*. This non-conventional assumption seems however to be empirically tenable, as discussed by Richards (1999).

d. Lastly, we need to specify a clear relation between the content of the memory buffer and the (un)grammaticality of the processed sentence. The most natural assumption is that the memory buffer be empty at the end of the derivation: if not so, we would have an unselected element within our structural description and that would violate the principle of *Full Interpretation* (Chomsky 1986). To guarantee a full interpretation we need to state explicitly this success condition:

(41) **Success condition**
(in order for the sentence to be grammatical) the memory buffer(s) must be empty at the end of the derivation.

To summarize, a Left-to-Right movement operation stores in a memory buffer all the elements which are not properly selected (according to LP) and must discharge them in appropriately selected positions by the end of the derivation.

Crucially, I propose that the memory buffers are local to computational phases:

(42) **Phase** (final definition)
a phase is a complete derivation involving:
- a *Phase Projection* of a potential SD structure,
- a set $\boldsymbol{i_p}$ of lexical items such that any item is either the head of the phase, a functional specification or a complement of the head of the phase,
- a memory buffer M, to store unselected items and retrieve selected ones.

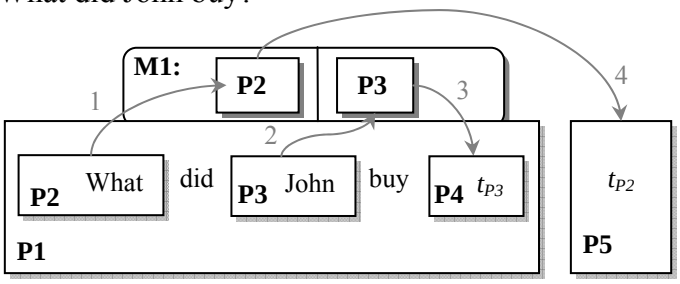
Notice that this definition allows us to process phases either in parallel or sequentially (cf. §3.1): in the first case, a superordinate phase is still open when we process another phase (we will call this phase *nested* from now on); in the second case, we close the previous phase before processing the sequential one; this happens essentially when we project the last selected complement of a phase (hereafter this last projected phase will be the *sequential selected phase*).

As for the content of the memory buffer, let us assume that it is initialized as empty, when the processing begins, and that at the end of every phase, if the memory buffer contains some element, these elements will be inherited to the (empty) memory buffer of the following phase. This inherited element will be part of $\boldsymbol{i_p}$ and will be licensed within the phase by being merged in the relevant position of the left-periphery, *edge* of the phase, as phonologically null element, unless specific

parameterized options hold. An interesting formal constraint we could impose on the inheritance mechanism, in order to satisfy the success condition (41) is the following one:

(43) **Constraints on memory buffer inheritance**
at the end of the phase, items that are still in the memory buffer, can be discharged only into the memory buffer of the sequential selected phase.

Let us now consider an example of this novel top-down implementation of movement[60] (as discussed in §3.2, we assume that phase heads are N and V[61]):
(44) What did John buy?



The steps of the derivations are as follows:

1. a default verbal phase is projected (P1):
   phase_projection(*Wh- question*) = [$_{+C\_wh\ +T\ +S\_nom\ V}$][62]
2. since the verbal phase is interrogative, this functional feature has to be explicitly marked; in English this can be done by merging the relevant *wh*-element within the specific "criterial" position. That is how [$_{+D\_wh\ N}$ what], phase P2 (computed as a trivial nested phase), is introduced in the derivation:
   merge([$_{+C\_wh\ +T\ +S\_nom\ V}$], [$_{+D\_wh\ N}$ what]) = [$_{+C\_wh}$[$_{+D\_wh\ N}$ what] $_{+T\ +S\_nom\ V}$]
3. since it is unselected (by LP) it is inserted (step 1) in the memory buffer (M1) of the matrix V-phase (P1):
   move([$_{+D\_wh\ N}$ what]) = M1[Q[$_{+D\_wh\ N}$ what]]
4. *did* is compatible with a tense functional specification of the matrix V-phase, then licensed in this position:
   merge([$_{+C\_wh}$ [$_{+D\_wh}$ what] $_{+T\ +S\_nom\ V}$], [$_{+T}$ did]) =
   [$_{+C\_wh}$ [$_{+D\_wh\ N}$ what] $_{+T}$[did] $_{+S\_nom\ V}$]
5. [$_{+D\ N}$ John] (phase P3, again computed as a (trivial) nested phase) is introduced to satisfy a subject-criterial (in the sense of Rizzi 2004a) requirement (functional specification of P1) and moved in the memory buffer since it is unselected (step 2):
   merge([$_{+C\_wh}$[$_{+D\_wh\ N}$ what] $_{+T}$[did] $_{+S\_nom\ V}$], [$_{+K\_nom\ +D\ N}$ John]) =
   [$_{+C\_wh}$[$_{+D\_wh\ N}$ what] $_{+T}$[did] $_{+S\_nom}$[$_{+K\_nom\ N}$ John] $_{V}$]
   move([$_{+K\_nom\ +D\ N}$ John]) = M1[Q[$_{+D\_wh\ N}$ what], A[$_{+K\_nom\ +D\ N}$ John]]

[60] Thanks to Valentina Bianchi for the "box-notation" that allow us to keep track of the derivation in a compact and meaningful way.
[61] This will roughly corresponds, respectively, to DP and CP phases in a bottom-to-top derivation. As for *vP* phase, it is not completely clear to me what relevant empirical data could be captured with this phase level in a top-down grammar: notice that reconstruction effects could be predicted by using VP-shells and intermediate traces along the lines of Barker (2007).
[62] It is fair to assume that aux-subject inversion is decided (as parameterized option) at this level.

6. then [$_{=[+D\ N]\ =[+D\ N]\ V}$ buy] is processed as the head of the matrix V-phase (P1). Since it has two selection requirements to be satisfied (an agent and a patient, both N-phases), these select features will project two phases, P4 e P5.
   merge([$_{+C\_wh}$[$_{+D\_wh\ N}$ what] $_{+T}$[did] $_{+S\_nom}$[$_{+K\_nom\ +D\ N}$ John] $_{V}$],
   [$_{=[+D\ N]\ =[+D\ N]\ V}$ buy]) =
   [$_{+C\_wh}$[$_{+D\_wh\ N}$ what] $_{+T}$[did] $_{+S\_nom}$[$_{+K\_nom\ +D\ N}$ John] $_{V}$[$_{=[+D\ N]\ =[+D\ N]\ V}$ buy]]
   phase_projection([+D N]**)** (per two) =
   [$_{+C\_wh}$ [$_{+D\_wh\ N}$ what] $_{+T}$[did] $_{+S\_nom}$[$_{+K\_nom\ +D\ N}$ John] $_{V}$[[buy [$_{+D\ N}$]] [$_{+D\ N}$]]]
7. P4 is a nested phase and will be unified by (re-)merging P3 (the first accessible element in M1);
   merge([... $_{V}$[[buy [$_{+D\ N}$]] [$_{+D\ N}$]]], M1[A:J.]) = [... $_{V}$[[buy [$_{+K\_nom\ +D\ N}$ J.]] [$_{+D\ N}$]]]
8. P5 is the last Phase Projection: it is a selected/sequential phase where the last selectional requirement of the previous phase will be lexicalized by merging P2.
   merge([$_{+D\ N}$], M1[Q:what]) = [$_{+D\_wh\ N}$ what]

I would like to stress that, despite its apparent similarity with a parsing algorithm, this is not a mere parsing strategy but it is the formal implementation of the move operation. The elements entering the derivation can be thought of as selected by using the "standard" (sub)numeration(s) device. Full Interpretation and the rigidly ordered cartographic structural positions force Left-to-Right linearization in a deterministic way. From a chain-driven perspective, this solution simply shifts the focus from the tail-position (selected position, always lexically derivable), to the head-position of a chain (scope position, dynamically determined by the speaker in a multiple clausals context). This shift is necessary in order to make the movement operation deterministic without any look-ahead device for intermediate steps[63], as shown in the following subsection.

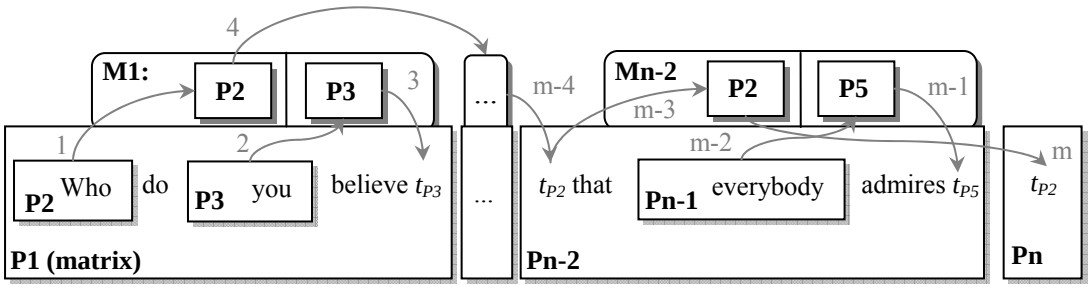
5.4. Successive cyclic Left-to-Right movement

We are now ready to account for successive cyclicity: the definition of movement predicts that an element can be released only in a selected (i.e. post-head) position or else it is transmitted to the selected sequential phase: the element is discharged in this phase and, since it is not properly selected (the Phase Projection added the minimal set of dominance relation so as to expect a V-phase, not a N-phase), it gets reloaded in the memory buffer of the sequential phase. And so on, until a compatible selected position is found where the moved element can be remerged/discharged:

(45) Who do you believe (that Mary said that ...) that everybody admires?

(46) [$_{CP}$ [$_{DP}$Who]$_{i}$ do you believe ... [$_{CP}$ t_i [that everybody admires t_i]]?

[63] This is because the scope in A'-movement is always marked essentially by the surface position of the A'-moved element. The selected position is indeed deterministically identifiable given the definition of phase and the inheritance mechanism of the memory buffer in (42).



The algorithm initializes a V(erbal) phase (the CP in (45), P1 in (46)). Then it computes the *wh*-phrase, which constitutes a separate N(ominal) phase (P2). Since this *wh*-phrase is not selected, it is stored in the local memory buffer (M1) of P1 by Move (step 1). Then, the computation of P1 proceeds by integrating the auxiliary as a tense specification and it stores the subject (nested) N-phase P3, which will be retrieved as soon as the head of the matrix V-phase P1, *believe*, is processed. At this point (step 3) P3 is discharged in the subject position and P1 is closed, since we have reached the last selected complement. The *wh*-phrase (P2) in the memory buffer M1 is then discharged (step 4) in the memory buffer of the selected-sequential phase (this is in order to comply with the success condition (41)) and so recursively. In the end, P2 will be remerged (step m-4) in the left periphery of the complement CP (Pn-2); then, since this position is unselected, the wh-phrase is re-stored in the local memory buffer of Pn-2 (step m-3). As a result, this "inheritance" mechanism leaves an intermediate copy/trace in the edge of the complement CP phase.[64] This way the computation can proceeds cyclically, phase-by-phase through selected phases, without using any formal/edge feature. Eventually the *wh*-phrase P2 will be discharged (step m), from the local memory buffer of P4, in the object position of a verb (phase Pn, selected by the V head *admires*): the Success Condition is thus satisfied at the end of the computation.

5.5. Understanding Strong Islands from a Top-Down, Left-to-Right perspective

In SVO languages, sentential subjects of (di)transitive verbs behave like (strong, in the sense of Cinque 1990) islands (47); on the other hand, subjects of unacussatives and passives (48) seem to allow for smoother extractions (Chomsky 2005b):

(47) a. *Who did [[close friends of *t*] become famous]?
b. *I wonder what [[reading *t*] would be boring]

(48) a. [Of which car] was [the driver t_i] awarded a prize?
b. It was the CAR (and not the TRUCK) of which$_i$ [DP the driver t_i] was found.

On a par with the first class of subjects, “true” adjuncts too, (49) Vs. (50), are considered to be strong islands for extraction:

(49) a. *Which concert did you sleep [during *t*]
b. *How did you leave [before fixing the car *t*]

[64] As pointed out by Luigi Rizzi, p.c., although this assumption is not strictly necessary for the algorithm to work, it seems fairly natural and it allows us to capture various successive cyclicity effects, like e.g. Irish complementizer alternations (McCloskey 2002).

(50) a. What did John arrive [whistling *t*]? (Borgonovo and Neeleman 2000:200)
b. What did John drive Mary crazy [trying to *t*]?

Looking at cross-linguistic variation, adjuncts (51.a) but not subjects (51.c)[65] behave as islands in head-final languages such as Japanese (Saito & Fukui 1998):

(51) a. ??Nani-o$_i$ [John-ga [$_{PP}$ Mary-ga *t*i katta kara] okotteru] no.
what-ACC John-NOM Mary-NOM bought since angry Q
'What$_i$, John is angry [because Mary bought t_i].'

b. ?Nani-o$_i$ [John-ga [$_{NP}$ [$_{IP}$ Mary-ga t_i katta] koto]-o mondai-ni siteru] no.
what-$_{ACC}$ John-$_{NOM}$ Mary-$_{NOM}$ bought fact-$_{ACC}$ problem-into making Q
'What$_i$, John is making an issue out of [the fact that Mary bought t_i].'

c. ?Nani-o$_i$ [John-ga [$_{CP}$ [$_{NP}$ [$_{IP}$ Mary-ga t_i katta] koto]-ga mondai-da to]
omotteru] no.
what-$_{ACC}$ John-$_{NOM}$ Mary-$_{NOM}$ bought fact-$_{NOM}$ problem-is that think Q
'What$_i$, John thinks that [the fact that Mary bought *t*i] is a problem.'

The classical solution to account for the islandhood of both subjects and adjuncts is Huang's (1982) *CED*:

(52) **Condition on Extraction Domains (CED)**
Extraction is only possible out of phrases which are *properly governed*, where a node A is taken to properly govern a node B iff
i. A c-commands B and no major category boundary intervenes between A and B,
ii. B is contained within a maximal projection of A, and
iii. B is assigned its thematic role by A

This condition essentially predicts extractions from objects but it leaves the data in (51) unaccounted for.

Within the more recent minimalist framework, Uriagereka (1999) suggests that cyclic spell-out could explain islandhood if we take into account economy conditions on linearization: in fact, it could be simpler for the linearization system (e.g. some implementation of Kayne's LCA) to apply to smaller chunks rather than to the whole set of terminal nodes in the structure[66]. Subjects and adjuncts can be shipped to spell-out as independent workspaces within which linearization can target only the relevant subset of elements. The matrix sentence and its object(s?) are instead linearized/spelt-out within the same workspace. Assuming that elements included in distinct derivational workspaces cannot freely enter the linearization of other workspaces, this proposal elegantly accounts for islandhood effects.

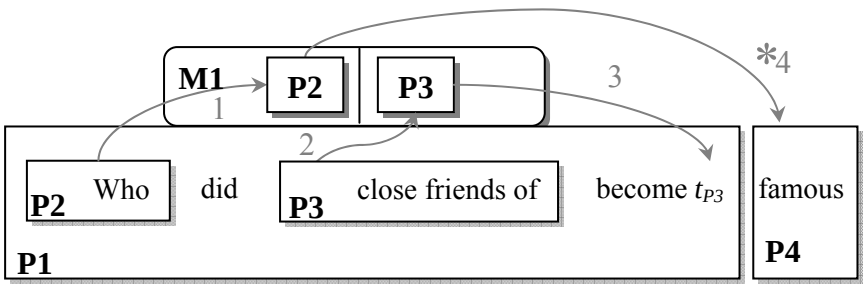
Both solutions require non trivial modifications so as to accommodate the contrast (47) Vs. (48) and (49) Vs. (50). Also the cross-linguistic variation pointed out in (51) seems to be unexplained. More precisely, we could force the application of

[65] The sentence is degraded since it involve an extraction from a complex NP, but crucially does not contrast with (51.b) (Saito & Fukui 1998).
[66] Remember that a similar intuition justified the formalization of the notion of phase in §3.

CED at a specific derivational point in order to allow for extraction from passives/unaccusatives (then adapting this condition to a derivational framework), so as to accommodate (47) Vs. (48); moreover, we could explain the transparent status of some "lower" adjuncts (such as *without*- or *after*-clauses, (50)) by assuming that these adjuncts have to be somehow thematically related to the predicated event structure (on the line of Borgonovo and Neeleman 2000; see also Truswell 2006). The Japanese data, however, remain unaccounted for. The same is true of a multiple spell-out approach (where, in fact, it would be even more problematic to explain why the subjects of unaccusatives/passives should be linearized within the matrix clause and not be independently shipped out to distinct derivational workspaces as other surface preverbal subjects).

The top-down perspective on strong islands is radically different from the standard view: as shown in (53), the ungrammaticality of this sentence, repeated below, does not result from the impossibility of extracting an argument from the island, but rather from the impossibility of integrating the unselected element within the phase in which it has been introduced or within another phase, that is sequential to the originating one:

(53) *Who did [close friends of _] become famous?



While the islandhood of subjects of (di)transitive verb follows directly, the transparency of passives/unaccusatives subjects needs a very strict application of the memory buffer inheritance mechanism; the definition is repeated below:

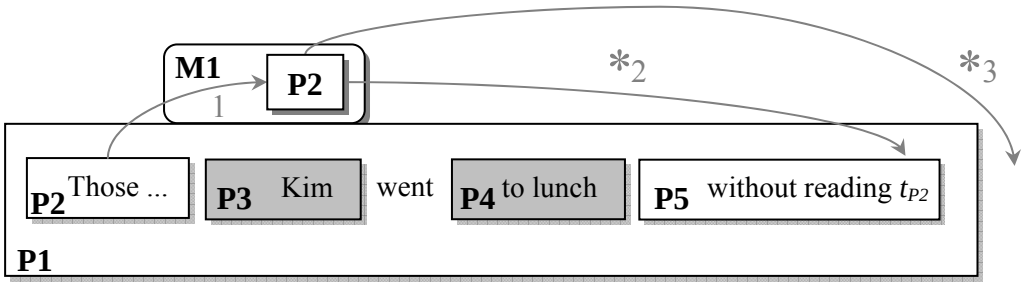
(42) **Constraints on memory buffer inheritance**
Items in the memory buffer, at the end of the phase, can be transferred only to the memory buffer of the last selected phase (if any).

We can successfully derive the extraction in (48) by first discharging the preverbal subject in the thematic position projected[67] by the unaccusative/passive phase-head; then, since this is the last selected argument, we should discharge within this projection the *wh*- element previously inserted in the memory buffer.

The contrast in (49) Vs. (50) requires us to first discuss the problem of right hand adjuncts. First observe that the LP in (11) does not allow an adjunct to be linearized to the right of the lexical head which it is a functional specification of[68]. Notice now that the adjuncts, being non-selected (since attached to a "functional specification", in the sense of (11)-(15) have to be considered genuine nested phases. Consider for instance the derivation of (54):

[67] In the sense of (19); not to be confused with the standard X' notion of projection.
[68] This is a standard prediction in any theory that assume some version of Kayne's LCA.

(54) ??[Those boring old reports]$_i$, Kim went to lunch [without reading e_i] .



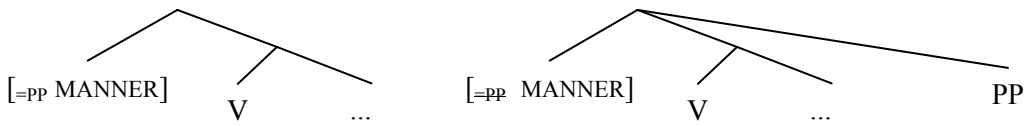
Ignoring for simplicity the phases P3 and P4, the relevant part of the derivation is that the topicalized ph(r)ase P2 is stored in the memory buffer since it is unselected (*step 1*), but it can be discharged neither in P5 (*step 2*) which is nested/unselected, nor in a sequential phase, since *went* does not select any other phase. Thus the islandhood of adjuncts too is a consequence of computational nesting.

In order to explain how a functional-related constituent can be linearized to the right, against LP, to the right of the modified head, we must allow for the possibility to delays Phase Projection and Merge:

(55) **Heaviness**
When a constituent licensed in a functional position is a (nested) complex phase, namely when it bears select features, it would rather be processed in a phase-peripheral position (i.e. on the right).[69]

The intuitive motivation of this definition is to reduce complexity (cf. §3), by marginalizing nesting. At this point it is unclear whether this is just a preference or a constraint[70]. Following Bianchi and Chesi (2005), I will tentatively assume that the nested phase-head bears some select feature (e.g. a *manner* phase-head as in (56.a) selects a PP as an argument) and, coherently with *Phase Projection* (19), this feature introduces the required SD at the end of the matrix phase (keeping however its nested relationship with this superordinate phase as in (56.b)[71]:

(56) a. b.

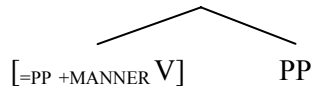


Since [manner] is a functional specification of the verbal phase, it has to be computed while this superordinate phase is still open (thus the adjunct is a genuinely nested phase, as expected). Only the special (complex) structure of this functional projection is responsible for the right-hand position of the selected PP.

[69] A little modification of this definition (i.e. the removal of any reference to the licensed/functional position) would predict heavy NP-shift as well, but this can not be discussed here (see Chesi in progress).
[70] Notice that some important constraints (e.g. right-roof constraint) would be readily predictable by (55), as discussed in Chesi (in progress).
[71] This is necessary in order to prevents the computation of the matrix phase from being interrupted. We have to assume that the top-down projection of this functional specification will follow the top-down projection of the head of the matrix clause.

Note that the right-hand modifier is *not* selected by the lexical head (unlike Larson 1988)[72]; hence, it is not a sequential phase but a nested one. On these lines, it is possible to capture the unexpected behaviour of some right-hand modifiers which seem to be transparent for extraction (e.g. (50)) by assuming a minimal difference between (56.b), where the island PP is "projected" by the functional specification, and (56.c), where the PP is a sequential phase because the select feature is specified on the verbal head:[73]

(56) c.



Finally, we need to account for the cross-linguistic variation of strong islands (51). An interesting unconventional solution is suggested by Choi & Yoon (2006): while in English predicates select their arguments (*P(redicate)-centered* language), in Japanese arguments select their predicates (*A(rgument)-centered* language). The idea of inverse selection is very powerful (and, potentially, very restrictive) but I do not have enough space here to explore many possible implications. Let me simply highlight that in order to accommodate this parameterization within the present model we simply need to allow nominal phases, by means of their case-marking, to intersectively project their top-down requirements: that is, a nominative and an accusative nominal phase would conspire so as to project a transitive verbal phase (or rather, the minimal set of verbal shells constituting a sequential-selected verbal phase). From this perspective, case-marked nominals are not strong islands. On the other hand, adjuncts do not select any verbal phase, but they simply license a functional specification of a verbal-phase (in accordance with LP, they lay on the left of the verbal-head). Then we expect adjuncts but not case-marked arguments to behave as nested phases, namely as strong islands.

6. Conclusion: how powerful is Phase-based Minimalist Grammar?

To summarize, a Phase-based Minimalist Grammar is an explicit grammatical formalism that incorporates various important insights of the Minimalist Framework; in particular, the notion of phase has been shown to constrain the complexity of the move operation; so as to achieve this result, however, the phase must be defined as a finite environment where structure building procedures operate strictly Top-Down and from Left-to-Right. This directionality shift with respect to the standard theory has some interesting empirical consequences, in particular in the domain of successive cyclic movement.

In more formal terms, PMGs can be expressed by 5-tuples {**V**, **Cat**, **Lex**, **F, M**} such that:

[72] This way the modifier is structurally superior to the VP-internal constituents; this avoids a number of problems with a generalized Larsonian "adjunct as complement" analysis (see Bianchi 2001, among others, for discussion).

[73] The idea that a selectional specification for a manner PP can be associated directly to a lexical head is made plausible by the existence of "selected adjuncts" (cf. Rizzi 1990): e.g., a verb like *behave* requires a manner specification; a verb like *weight* requires a measure specification of a certain kind; a verb like *be born* requires a locative or temporal specification, etc.

(57) Phase-based Minimalist Grammar **(**PMG**)**

V is a finite set of non-syntactic features, (P ∪ I) where
P are phonetic features and *I* are semantic ones;
Cat is a finite set of syntactic features, *Cat* = (*base* ∪ *select* ∪ *licensors* ∪ *interveners*) where
base are main categories: {*V(erb)*, *N(oun)*};
licensors/functional_specifications are ordered sets of functional features associated to V and N: {V:<Force, Top, Int, ... T, Mood, ... Asp ...>, N:<K(case), D, ... size, length, ...> };
select specify selection requirements {=*x* | *x* ∈ *base* ∪ *licensors* };
interveners are classes of elements that behaves uniformly w.r.t. minimality effects (i.e. *F*{*move*}): {A, Q, Mod, Topic};
Lex is a finite set of expressions built from *V* and *Cat* (the lexicon);
F is a set of the three partial functions from tuples of expressions to expressions{*merge*, *move, phase_projection*};
M is a set of LIFO memory registers associated to each phase as defined in *F*{*phase_projection*} according to *Cat*{*interveners*}.

I would like to conclude this outline, by showing that this grammatical formalism is more powerful than CFGs and TAGs, namely there are (artificial) languages that CFGs and TAGs cannot capture and that PMGs does. Let us start by considering how PMGs can capture simple counting recursion (a^nb^n, cf. Chomsky 1957); using the grammar/lexicon in (58) we can demonstrate (by construction) that the proposition in (59) is true:

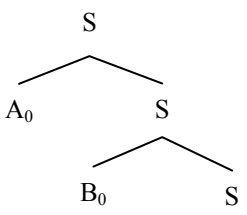
(58) PMG_1:

phase heads: *S(tart), R(ecursion), A, B, a, b*

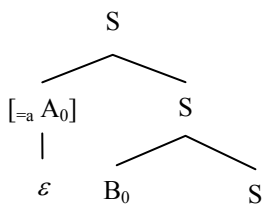
Lex: [a *a*], [b *b*], [$_{=a}$ A_0 ε], [$_{=b}$ B_0 ε], [$_{=a\ =A}$ A ε], [$_{=b\ =B}$ B ε],
[$_{+A0\ +B0\ =[+A\ +B\ R]}$ S ε], [$_{+A\ +B\ =[+A\ +B\ R]}$ R ε], [$_{=A\ =B}$ R ε]

(59) *strings*(PGM_1) = {a^nb^n *: n* ≥ 0}

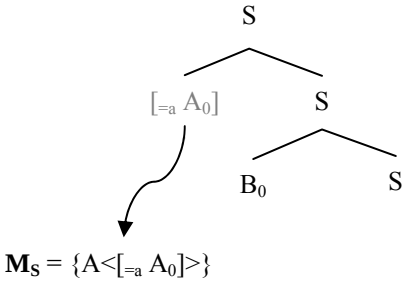
Sample derivation:



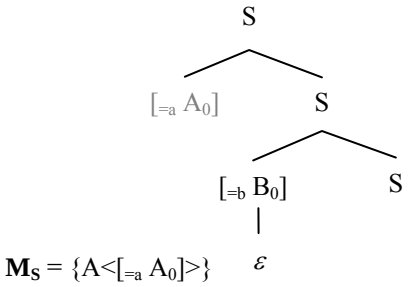
1. *Phase Projection* ([+A_0 +B_0 S])



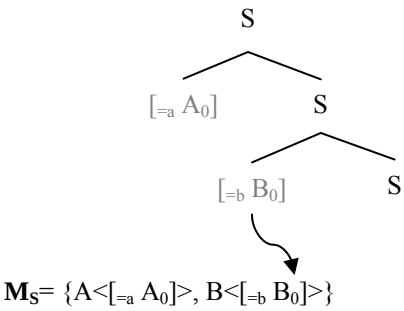
2. *merge* (A_0, [$_{=A}$ A_0 ε])



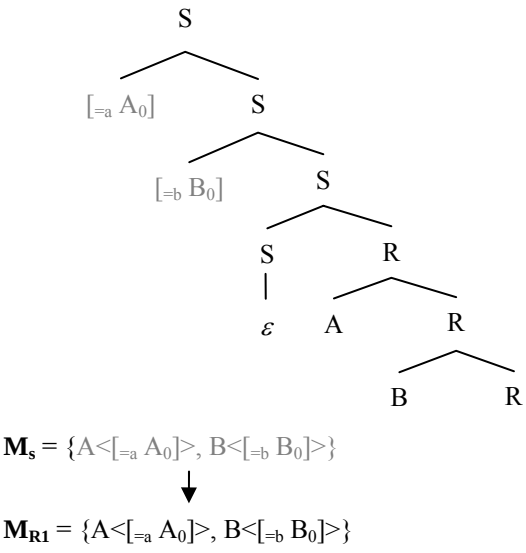
3. *delay Phase Projection* (=a),
4. *move*(A_0) (in the A-slot)



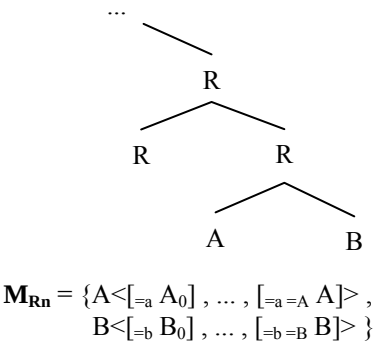
5. *merge* (B_0, [$_{=b}$ B_0 ε])



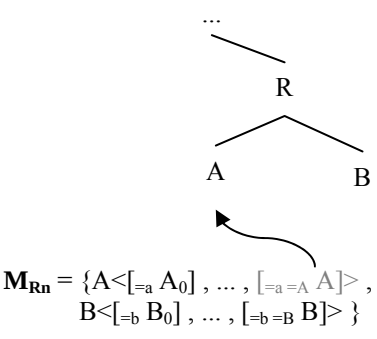
6. *delay Phase Projection* (=b),
7. *move*(B_0) (in the B-slot)



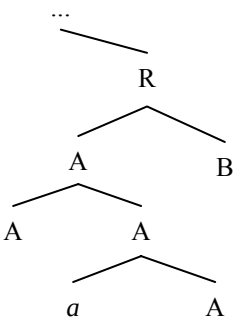
8. *merge*([$_{+A0\ +B0\ =[+A\ +B\ R]}$ S ε])
9. *Phase Projection*([+A +B R])
10. close the S-phase
11. discharge M_S-buffer in M_{R1}
12. mutatis mutandis (merge A, not A_0 and B, not B_0) repeat 2-11 as you like



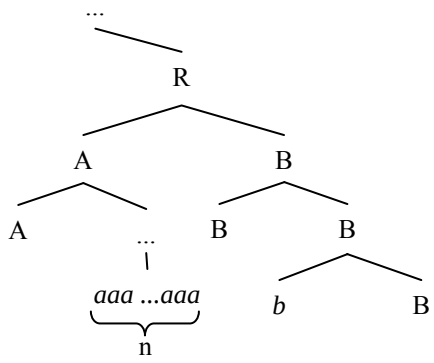
13. *merge* ([$_{=A\ =B}$ R ε]) (to exit the loop)



14. *merge*(M(A<*last*>))

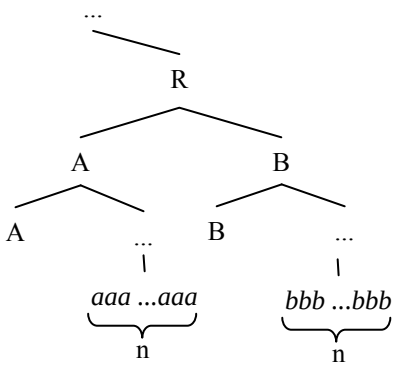


$\mathbf{M_{Rn}}$ = {A<[$_{=a}$ A_0] , ... , [$_{=a\ =A}$ A]> ,
B<[$_{=b}$ B_0] , ... , [$_{=b\ =B}$ B]> }



$\mathbf{M_{Rn}}$ = {A<[$_{=a}$ A_0] , ... , [$_{=a\ =A}$ A]> ,
B<[$_{=b}$ B_0] , ... , [$_{=b\ =B}$ B]> }

15. *Phase Projection*(=a =A)
16. *merge*(a)
17. repeat 14-16 until [$_{=a}$ A_0] is reached
18. close the last A-phase
19. *merge*(M(B<*last*>)) (i.e. [$_{=b\ =B}$ B])



$\mathbf{M_{Rn}}$ = {A<[$_{=a}$ A_0] , ... , [$_{=a\ =A}$ A]> ,
B<[$_{=b}$ B_0] , ... , [$_{=b\ =B}$ B]> }

20. repeat 18-19 until [$_{=b}$ B_0] is reached, then *merge*(b) and end the derivation.

Taking advantage of this very same recursive mechanism, it is easy to show that a minimal modification of this grammar, as shown in (60), is sufficient to capture also five counting dependencies (three counting dependencies are out of the power of any CFG, Partee and al. 1993; more than four counting dependencies can not be captured with TAGs, Stabler 1997):

(60) PMG_2:

phase heads: *S(tart), R(ecursion), A, B, C, D, E, a, b, c, d, e*

Lex: [a *a*], [b *b*], [$_{=a}$ A_0 ε], [$_{=b}$ B_0 ε], [$_{=a\ =A}$ A ε], [$_{=b\ =B}$ B ε],
[c *c*], [d *d*], [$_{=c}$ C_0 ε], [$_{=d}$ D_0 ε], [$_{=c\ =C}$ C ε], [$_{=d\ =D}$ D ε],
[e *e*], [$_{=e}$ E_0 ε], [$_{=e\ =E}$ E ε],
[$_{+A0\ +B0\ +C0\ +D0\ +E0\ =[+A\ +B\ +C\ +D\ +E\ R]}$ S ε],
[$_{+A\ +B\ +C\ +D\ +E\ =[+A\ +B\ +C\ +D\ +E\ R]}$ R ε], [$_{=A\ =B\ =C\ =D\ =E}$ R ε]

(61) $a^n b^n c^n d^n e^n : n \geq 0 \in strings(\mathrm{PGM}_2)$

In general PMGs can capture any finite number *k* of counting dependencies (where *k* is the maximum number of functional features within a phase). This guarantees that PMGs are at least mildly-context sensitive and as powerful as MGs, so they are sufficient to capture any relevant property of the natural languages.

References

Abney S. P. (1987) *The English noun phrase in its sentential aspect.* Ph.D. Thesis: MIT.

Aho, A. V., and Ullman, J. D. (1972) The theory of parsing, translation, and compiling. *ACM Classic Books Series* 2:1050.

Barker C. (2007) *Reconstruction as delayed evaluation*. Ms. Los Angeles.

Belletti A. ed. (2002) S*tructures and Beyond. The Cartography of Syntactic Structures*, vol. 3, OUP.

Belletti A., Rizzi L. (1996) *Parameters and functional heads: Essays in comparative syntax*. New York: OUP

Berwick, R. C., and Weinberg, A. S. (1986) *The grammatical basis of linguistic performance: language use and acquisition*: MIT Press Cambridge, MA, USA.

Bianchi V. (2001) Antisymmetry and the leftness condition: Leftness as anti-c-command. *Studia Linguistica* 55**,** 1-38.

Bianchi, V. and Chesi, C. (2005) "Phases, strong islands, and computational nesting", *Ms., University of Siena [presented at GLOW 28, 2005, Geneva]*.

Boeckx C., N. Hornstein (2004) *Movement under Control*. Linguistic Inquiry, Volume 35, Number 3, 431–452

Borgonovo, C. and Neeleman, A. (2000) Transparent adjuncts. *The Canadian journal of linguistics* 45**,** 199-224.

Cann, R., Kempson, R. and Marten, L. (2005) *The Dynamics of Language: an introduction.* Oxford: Elsevier.

Cardinaletti A. (2002) *Towards a cartography of subject positions*. In Rizzi L. ed. (2002) *The Structure of CP and IP*, *The Cartography of Syntactic Structures*, vol.2, OUP.

Chesi C. (2004) *Phases and Cartography in linguistic computation: toward a cognitively motivated computational model of the human linguistic competence*. Ph.D. Thesis. University of Siena

Chesi C. (in progress) *Rightward movement from a top-down perspective*. Ms. University of Siena

Chesi C. ed. (in progress) *Directions in derivations*. under review.

Choi, Y., Yoon, J. (2006) *Argument Cluster Coordination and Constituency Test (Non)-Conflicts*, NELS 37**,** University of Illinois at Urbana-Champaign.

Chomsky N. (1956) Three models for the description of language. *IRE transactions on information theory* IT-2**,** 113-124.

Chomsky N. (1963) *Formal Properties of Grammars*. In R.D. Luce, R.R. Bush, and E. Galanter, ed, Handbook of Mathematical Psychology. Wiley, New York

Chomsky N. (1981) *Lectures on Government and Binding: The Pisa Lectures*. Holland: Foris Publications

Chomsky N. (1995) *The Minimalist Program*, MIT Press, Cambridge, MA.

Chomsky N. (1999) *Derivation by Phase*. MIT Occasional Papers in Linguistics, no. 18. Cambridge, MA.

Chomsky N. (2000) *Minimalist Inquiries: The Framework*. In Step by Step: Essays in Minimalist Syntax in Honor of Howard Lasnik, edited by Robert Martin, David Michaels and Juan Uriagereka, 89-155. Cambridge, MA: The MIT Press, 2000.

Chomsky N. (2001) *Beyond explanatory adequacy*. MIT Occasional Papers in Linguistics, no. 20. Cambridge, MA

Chomsky, N. (1957) *Syntactic structures*. Mouton de Gruyter Berlin, New York.

Chomsky, N. (1986) *Knowledge of language: its nature, origin, and use,* Praeger.

Chomsky, N. (2005) "On Phases", *Manuscript, MIT*.

Christiansen M. H. & Charter N. (1999) *Toward a Connectionist Model of Recursion in Human Linguistic Performance*. Cognitive Science 23(2):157-205

Cinque G. (1999) *Adverbs and Functional Heads: A Cross-linguistic Perspective*, OUP.

Cinque G. ed. (2002) *The Structure of DP and IP. The Cartography of Syntactic Structures*, vol. 1, OUP.
Collins C. (2001) Eliminating labels. *MIT Occasional Papers in Linguistics* 20**,** 1-25.
Fong S. (1991) *Computational Properties of Principle-Based Grammatical Theories*. MIT: Ph.D. Thesis.
Frank R. (2002) *Phrase Structure Composition and Syntactic Dependencies,* Bradford Books.
Grimshaw J. (1991) *Extended projection*. In P. Coopmans, M. Everaert & J. Grimshaw ed. Lexical specification and insertion. The Hague: Holland Academic Graphics, pp. 115-134.
Grohmann K. (2003) *Prolific Domains: On the Anti-Locality of Movement Dependencies,* John Benjamins Publishing Company.
Harkema H. (2001) *Parsing Minimalist Languages*. Ph.D. Thesis. UCLA
Hawkins, J. A. (1994) *A Performance Theory of Order and Constituency*: Cambridge University Press.
Hopcroft, J. E. and Ullman, J. D. (1990) *Introduction To Automata Theory, Languages, And Computation,* Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.
Hornstein N. (1999) Movement and control. *Linguistic Inquiry 30(1)*, 69–96.
Huang J. (1982) Logical relations in Chinese and the theory of grammar. Ph.D. Thesis. MIT
Joshi, A. K. (1985) "Tree Adjoining Grammars: How much context sensitivity is required to provide a reasonable structural description" In *Natural Language Parsing*, Vol. (Ed, Karttunen, I., Dowty, D., Zwicky, A.) Cambridge University Press, Cambridge, U.K., pp. 206-250.
Kayne R. (1994) *The antisimmetry of syntax*. MIT Press, Cambridge, MA.
Kayne R. (2002) *Pronouns and their antecedents*. In S. Epstein & D. Seely. ed. Derivation and Explanation in the Minimalist Program. Oxford: Blackwell
Keenan, E. L. (1975) Variation in Universal Grammar. *Analyzing variation in language*:138–148.
Laenzlinger, C. (2005) Some Notes on DP-internal movement. *GG@G* 4**,** 227–260.
Landau I. (2003) Movement out of control. *Linguistic Inquiry* 34:471-498.
Larson R. (1988) On the double object construction in English. *Linguistic Inquiry* 19**,** 589–637.
Leonard L. B. (1998) *Children with Specific Language Impairment*. MIT Press, Cambridge, MA.
Matushansky O. (2005) Going through a phase, *Perspectives on phases***,** 157–181.
McCawley, J. D. (1968) *Grammar and Meaning*. Academic Press, New York, NY.
McCloskey J. (2002) Resumption, successive cyclicity, and the locality of operations. *Derivation and Explanation in the Minimalist Program***,** 184–226.
Michaelis J. (1998) *Derivational Minimalism is Mildly Context-Sensitive*. In M. Moortgat ed. (LACL '98), volume 2014 of Lecture Notes in Artificial Intelligence. Berlin: Springer Verlag.
Miller G. A. (1956) *The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information*. The Psychological Review, 1956, vol. 63, pp. 81-97
Ogawa Y. (2001) *A Unified Theory of Verbal and Nominal Projections.* Oxford University Press.
Partee, B., H.A. ter Meulen, and R. E. Wall (1993) *Mathematical Methods in Linguistics,* Dordrecht: Kluwer.
Pesetsky D. (1982) *Paths and Categories*. Ph.D. Thesis. MIT
Phillips C. (1996) *Order and Structure*. Ph.D. Thesis. MIT
Phillips C. (2003) Linear order and constituency. *Linguistic Inquiry* 34**,** 37-90.
Richards N. (1999) Dependency formation and directionality of tree construction. *MIT Working Papers in Linguistics* 34**,** 67-105.
Rizzi L. (1990) *Relativized Minimality*, MIT Press, Cambridge, Mass.
Rizzi L. (1997) *The Fine Structure of the Left Periphery*. in L. Haegeman (ed.), "Elements of Grammar". Kluwer, Dordrecht.
Rizzi L. (2002) *Locality and Left Periphery*. In Belletti, A. ed. (2002) S*tructures and Beyond. The Cartography of Syntactic Structures*, vol. 3, OUP
Rizzi L. (2004a) *On the Form of Chains: Criterial Positions and ECP Effects.*, Ms. University of Siena.
Rizzi L. ed. (2004b) *The Cartography of Syntactic Structures*, Oxford University Press.
Saito, M. and Fukui, N. (1998) Order in Phrase Structure and Movement. *Linguistic Inquiry* 29**,** 439-474.
Scott G. J. (1998) stacked adjectival modification and the stracture of nominal phrases. SOAS Working Papers in Linguistics and Phonetics. 8:59-89
Shieber S.M. (1986) *An Introduction to Unification-Based Approaches to Grammar*. Stanford, CA: CSLI Lecture Notes.
Shieber S.M., Y. Schabes, and F.C.N. Pereira. (1995) *Principles and Implementation of Deductive Parsing*. Journal of Logic Programming, 24
Sportiche D. (1988) A theory of floating quantifiers and its corollaries for constituent structure. *Linguistic Inquiry* 19**,** 425-449.

Stabler E. (1997) *Derivational minimalism*. in Retoré, ed. Logical Aspects of Computational Linguistics. Springer
Starke M. (2001) *Move Dissolves into Merge*, Ph.D. Thesis, University of Geneva.
Starke M. (2002) Against Specifiers. *Abstract for TILT 2002*.
Uriagereka J. (1999) Multiple Spell-out. In *Working Minimalism*, Vol. (Ed, Epstein, S., Hornstein, N.) MIT Press, Cambridge (Mass.).
Truswell, R. (2007) Extraction from Adjuncts and the Structure of Events. *Lingua* 117, 1355-1377.
Zanuttini R. (1997) *Negation and Clausal Structure*. New York: OU