PHASES AND CARTOGRAPHY IN LINGUISTIC COMPUTATION

TOWARD A COGNITIVELY MOTIVATED COMPUTATIONAL MODEL OF LINGUISTIC COMPETENCE

by

CRISTIANO CHESI

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN COGNITIVE SCIENCE
DECEMBER 2004

Major Advisor:	Prof. Luigi Rizzi
Associate Advisor:	Prof. Marco Gori
External Advisor:	Prof. Robert C. Berwick (MIT)

Committee:
Prof. Andrea Bonomi (University of Milan)
Prof. Leonardo Savoia (University of Florence)

UNIVERSITY OF SIENA - PH.D. IN COGNITIVE SCIENCE
(MAJORS IN LINGUISTICS AND ARTIFICIAL INTELLIGENCE)

ACKNOWLEDGMENTS

Even though many words have been written within this thesis in order to justify a symmetric access to (linguistic) *knowledge* both from a *production* and from a *comprehension* perspective, the story of these pages should point out that there could be much more work from the *comprehension* side than from the *production* one.

I suppose this to be true because of two (*minimalist*, cf. 2.2.2) fundamental points:

1. we should *produce* something ONLY when we are sure that we *comprehended* our knowledge fragment, that is, we tried to evaluate if it is *coherent*, *robust*, *complete* and *interesting* and therefore worth *producing*;
2. this is because the *complexity* of the world is so high (due to the huge "space of the problem", cf. §1.4), that *producing* something runs the risk of simply being ignored (best case) or making others waste their precious time trying to comprehend our (incomplete) knowledge fragment (worst case);

Following these rules of thumb, I spent almost-four-years just *comprehending* the "knowledge fragment" that is the topic of my dissertation, and, surely, this has not been enough.

I wish to thank Valentina Bianchi and Luigi Rizzi who have tried hard to help me depart from "point 1" and "force" me to *produce* something (these pages are mainly a proof of their invaluable efforts), moreover pointing out that *producing* something, even incomplete, (and getting feedback) is better than keeping silent and it often causes a *comprehension* boost instead. Without their precious feedback/support, this work would still be an underground work.

With respect to "fact 2", even though much of my *production* fell in the "best case" (it has been ignored) I wish to thank those who fell into the second (unlucky) case and had

the patience to try to *comprehend* the little I ever *produced*: without Klaus Abels, Pranav Anand, Asaf Bachrach, Robert Berwick, Noam Chomsky, Andrew Nevins, Claudia Felser, Justin Fitzpatrick, Sandiway Fong, Danny Fox, Morris Halle, Michela Ippolito, Roni Katzir, Ivona Kucerova, Giorgio Magri, Paola Merlo, Tomaso Poggio, Linnaea Stockall, Michael Wagner and Bencie Woll's comments/suggestions/feedback (and their patience with my English[1]), this "knowledge fragment" would have been much less clear to me now.

Last but not least, I wish to thank also the Centro Interdipartimentale di Studi Cognitivi sul Linguaggio (University of Siena), the MIT community (especially the MIT Libraries, the CogNet/VERA databases and the Stellar System), the City University of London, the Ecole Normale Supérieure of Paris and the European Commission (for the Marie Curie Fellowship HPMT-CT-2000-00208) who provided me with knowledge and "bread" despite the political/economical aberrations of these days.

[1] In fact, quoting Cristiano Castelfranchi p.c., Ancient Romans did not pretend people in their colonies would speak perfect Latin.

CONTENT

INTRODUCTION

In these pages I will define and evaluate from a cognitive perspective some fundamental properties of a computational model of *linguistic competence*, that is the *intensional (mechanic) procedure* that characterizes an infinite set of well formed sentences[2], namely a *language*.

Following Chomsky, *linguistic competence* is the "speaker/hearer's knowledge of his/her language" (Chomsky 1965:4) and it can be expressed by a *generative grammar*: a formal description of any possible sentence of a given language in terms of at least a single *Structural Description* (*SD*) expressing how this sentence is understood/produced by the (ideal) hearer/speaker.
Even though a *generative grammar* can be completely abstract, that is, independent from how the linguistic *competence* is really used by the (actual) hearer/speaker (cf. Chomsky 1995), it is plausible wondering, both from a cognitive and from a computational perspective, whether or not this knowledge representation (or at least part of it) is suitable to be used both in *comprehension*, that is, when we perceive a sentence, and in *production*, namely when we generate it.
This naive condition is however far from reachable in most of the current generative frameworks (e.g. the *Minimalist Program*, Chomsky 1993-2001 and related work, §2.2.2).

[2] To be intended as opposite to *extensional procedure*, namely as opposed to the exhaustive list of elements (i.e. well formed sentences) of this set.

I will refer to this property as the *flexibility* of the grammatical formalisms[3]:

a. **flexibility** (definition)
   the same *grammatical knowledge* should be usable both in *comprehension* and in *production*.

This is an important hypothesis necessary not only in order to develop linguistic resources (such as *lexica* and *grammars*) that can be effectively put in use in *performance* tasks, but also to underline aspects that seem to be cognitively motivated.
Beyond *flexibility*, three more properties confer to the grammar a *cognitive plausibility*:

b. **universality** (or **descriptive adequacy** from a cross-linguistic perspective)
   a grammar should capture linguistic generalizations on empirical ground, cross-linguistic similarities and account for variations;

c. **explanatory adequacy**
   a grammar should be *learnable* by a speaker/hearer**;**

d. **realism**
   once established what *performance* is about, *grammar* + *performance* should faithfully reproduce productive phenomena involved in *comprehension* and *production*; the computational model that describes this facts should be tractable.

These properties cause many difficulties at the formal level when we try to define precise algorithms that will take them all into account: to my knowledge no linguistic theory satisfies all these properties at the same time. Most of the frameworks, in fact, remain agnostic on some of these issues, causing difficulties for implementing a computational model that aims to be *flexible*, *universal*, *learnable* and *realistic*.

For instance, much work has been done with respect to the *universality* and *learnability* issues within the "generative tradition" (Chomsky 1957-2001, cf. §2.2). Even thought many (radically) different perspectives have been explored in the last fifty years, *flexibility* and *realism* have always been rare topics.

[3] Do not confuse *flexible* with *reversible* (e.g. *reversible natural language grammars*, Neumann 1994): *reversibility* is a mathematical property not necessarily implied by this informal notion of *flexibility*.

Notably, the *Minimalist Program* represents an interesting attempt to reach an adequate *descriptive adequacy* dealing with complexity issues and phrase structure building procedures in a *derivational* (that is mainly *incremental*) way. Even though the early warning on "abstractness" is still valid (Chomsky 1995), this can be considered as a legitimate move toward a cognitively more plausible model of grammatical description.
In particular, this program tried to reduce significantly the core grammatical system to simple (generalized) operations that concatenate linguistic elements: namely *merge* and *move. Merge* is the fundamental operation for building objects: it takes two adjacent elements *A* and *B* and combines them together creating a new object *C* minimally of the form {*A*, *B*}; *move* expresses the relation between two usually not adjacent objects (e.g. *A* and an empty position t_A (a *trace*) to be interpreted as an exact copy of *A*). As noted by Starke (Starke 2001) these two operations are both subject to strict constraints, also known as *locality conditions*: *merge* requires strict adjacency (an element can only *merge* with another element that is next to it), while *move* manages long distance relationships, unless an intervening element of the same structural type (in a sense that will be explored in detail in this thesis) is present between the object and the invisible/incomplete copy of the object.
While *minimalism* explores the potentialities of *merge* and *move*, it appears to be not as predictive as other frameworks in terms of *locality conditions*: the *Cartographic Approach* (Belletti, Ed. 2002, Cinque 1999, Ed. 2002, Rizzi 1997, Ed. 2002, 2004 and related work), from this perspective is much more heuristic, since its main goal is to define a syntactic map as detailed as possible in terms of structural asymmetries. This complementary nature could invite us to use both frameworks to implement a computational model that would be at least *robust* (that is, empirically adequate, thus meeting the *universality* requirement). Unfortunately, this solution is not readily viable; in fact, different linguistic frameworks cannot be easily integrated and implemented within the same computational model for the following reasons:

1. **underspecification of many essential primitives** - e.g. the *Minimalist Program*, as Chomsky has presented it (Chomsky 1995-2001) leaves many essential notions largely underspecified to make the picture cleaner, and accessible from a

wider perspective, but this is a limitation in terms of implementability; among these primitives, the *Minimalist Program* has left underspecified:

- the organization of the lexicon;
- the specification of interface conditions;
- the nature of the parameterization;
- many conditions on feature checking;

2. **internal coherence** - in order to make a model *consistent*, the principles/rules used should be non-contradictory. Often, when the set of "axioms" is huge (e.g. Extended Standard Theory, §2.2.1) and extracted from different analyses which solve local empirical problems, contradiction is an emergent property, hence difficult to be detected. These analyses are clearly useful because of the effort in taking into account empirical problems, but we should always evaluate if an explanation is "genuine", consistent with other analyses, or is just a theoretic artifact[4];
3. **complexity and tractability** - building a manageable model requires keeping in mind that the most *economical* assumptions are better candidates, where *economical* has to be intended both in a non-technical sense (simpler solutions have to be considered first, since they can be better evaluated/implemented; in this sense "reductionism" without loss of *explanatory adequacy* is an intriguing and necessary exercise we should deal with) and in a computational sense: some solutions are less expensive than others in terms of use of computational resources (basically *time* and *memory*, cf. Barton and al. 1987); *minimalism* does not present in any evident way a formalization of the devices proposed in order to reduce the overall complexity of the system even though this is a crucial point (for instance there is no accord on the notion of *phase*, cf.§2.2.2, §3.4).

These are the rationales behind the choice of many scholars who have preferred adopting less powerful grammars (such as simple *Regular Expressions* or *Context Free Grammars*) for which we know efficient algorithms (for instance in CFG parsing:

[4] This should be a leitmotif of the Minimalist Program (Chomsky 1993), unfortunately forgotten sometimes.

Earley, Earley 1970, and *CYK algorithm*, Kasami 1965, Younger 1967), completely redefining linguistic frameworks, formalizing/implementing new grammars that better fit with these "special" requirements (*precise formalization*, *flexibility* and *realism*; cf. *Head-driven Phrase Structure Grammars*, Pollard and Sag 1994, *Tree Adjoining Grammars*, Joshi 1985, *Lexical Functional Grammars*, Bresnan 2001).

In this dissertation, I will propose that most of these difficulties can be overcome and that a computational description of some recent linguistic intuitions is indeed pursuable in order to reconcile empirical adequacy (*universality* and *explanatory adequacy*) with a wider cognitive perspective (*flexibility* and *realism*).

With the purpose of doing that, in the **first chapter** of this dissertation I will introduce the linguistic concepts that are used in most frameworks, seeking to highlight their cognitive nature (essentially comparing processing at other cognitive levels, such as *vision*); in particular, consideration will be given to defining what *features* are (§1.1), how they *combine* (§1.2) and which *relations* shall be defined among them (§1.3); additionally some essential concepts will be provided in order to understand cost functions (that are measures of the *complexity*) of the proposed devices (§1.4).

Generative frameworks (briefly mentioned in this introduction) will be systematically reviewed in the first part of **chapter 2** (*Principle and Parameters* and the *Extended Standard Theory*, §2.2.1, the *Minimalist Program*, §2.2.2, and a "unusual" processing perspective for this framework, that is Phillip's (1996) *left-to-right* processing model, §2.2.3; finally the *Cartographical Approach* will be explored in §2.2.4). The second part of this chapter will show some (partially successful) attempts to implement/formalize these linguistic frameworks (the *Principle-based parsing* approach, §2.3.1, Stabler's *minimalist grammar* formalization, §2.3.2, and Fong's minimalist parser, §2.3.3).

The rest of the thesis will try to solve many standing problematic issues: in order to provide a precise context, in **chapter 3** I will firstly formalize the idea of *structural description* (§3.1.1) then the (*performance*) tasks that must access the grammar

(essentially *parsing* and *generation,* which are specific cases of *comprehension* and *production* respectively, §3.1.2). Afterwards, the most problematic aspects of the language (*ambiguities*, §3.2.1, and *long distance dependencies*, §3.2.2) will be formalized with respect to the proposed *performance* tasks. Finally the formalization of a *minimalist grammar* (inspired by Stabler's 1997 work and the *Minimalist Program*) enriched with considerations on the articulate geometry of the *functional feature structures* proposed within the *Cartographic Approach* will be provided (§3.3).

From the theoretical perspective, the standard minimalist approach sketches a model that does not fit in a clear way with specific *performance* algorithms such as *parsing* or *generation* even though much emphasis is put on "interface properties". The grammar formalized in §3.3 considers some important constraints posed both by the *generation* and *parsing* problems, in the end defining a model that is:

1. usable both in *parsing* and in *generation*;
2. cognitively motivated;
3. tractable;
4. as much as possible *deterministic*.

To achieve these results, I will focus on three important properties that are not only cognitively plausible, but also formally and computationally advantageous (§3.4):

1. *structure building operations* can be embedded within the grammar if they apply either *top-down* (in *generation*) or *from-left-to-right* (in *parsing*);
2. using a *Linearization Principle* (inspired by Kayne's 1994 LCA) and fixing the functional structure by mean of a universal hierarchy (*Cartographic Approach*) makes the algorithm mostly *deterministic*;
3. formalizing phases (cf. Chomsky 1999) helps us to make the algorithm (in relation to dealing with *ambiguities* and *long distance relations*) tractable.

The last chapter, **chapter 4**, will provide a full specification of the algorithm/grammar used, showing which empirical coverage this model can reach. In particular cases of *Argumental movement* (§4.3.1), *Criterial movement* (§4.3.2), *Locality effects* (§4.3.3), *Strong Island conditions* (§4.3.4), *cross-serial dependencies* (§4.3.5), *covert movement* (§4.3.6) and *parameter setting* (§4.3.7) will be explored.

CHAPTER 1
THE LINGUISTIC COMPUTATION FROM A COGNITIVE PERSPECTIVE

Any scientific framework, at some point, requires a definition of its *primitive components*: minimal assumptions that any model has to postulate in order to be first *descriptive* and then *predictive* in terms of generalizations on novel (acceptable) *input data*.

Coherently with this view, the main goal of any *generative* (that is formal) *linguistic theory* is to define the primitives that allow us to describe the empirical nature of the linguistic objects. This chapter is intended to provide a general introduction to these linguistic entities from a cognitive perspective, essentially describing related phenomena that are also relevant in other cognitive domains such as *vision*. The aim is to justify the primitives that will be used in this dissertation in a precise, non-redundant and coherent (at least in cognitive terms) way.

The first primitive I will introduce in this chapter is the entity bearing the smallest possible quantity of information: the *feature*, namely the minimal expression of the properties of any linguistic object. In §1.1 I will explore how these primitives can be related to the idea of *information compression* and to the *generalization* we are able to make on new patterns. Then *typologies* (§1.1.1, §1.1.4), *sources of information* (§1.1.2), *representations* (§1.1.3) and the *coexistence* of *features* on the same (lexical) object will be evaluated, eventually ending up with an abstract definition of what a

feature structure is (§1.1.5) and how much *variation* is allowed on this structures among different languages (§1.1.6).

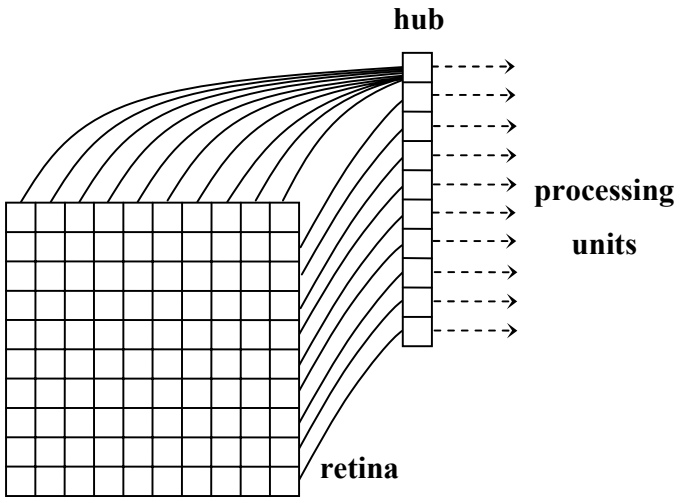
The second part of this chapter (§1.2) will investigate how *feature structures* can combine in order to build bigger meaningful units (§1.2.1), which operations are allowed (§1.2.1, §1.2.2) and in which domain they can apply. The computation described in these pages will have a strong *derivational* flavor, that is, essentially, each relation established among elements will be relevant only at a precise stage of the computation and, after that, no further operation will be able to access this relation anymore. I will however show how even in such a restrictive framework many relevant relations, essentially *C-command* and *Long Distance Dependencies*, can still be captured (§1.3).

In the final part, I will explain how the operations on *feature structures* proposed in §1.2 turn out to be extremely "complex" if they are not "bounded", where *complexity* and *bounding* have a precise meaning expressible in terms of *memory load* and *combinatorial possibilities* with respect to the states that the computation has to evaluate in order to store/retrieve/elaborate (linguistic) information; these concepts will be introduced in the last paragraph in order to understand how we could reduce the complexity of many problems by *chunking* the computation in different "phases" (§1.4).

1.1 FEATURES

To understand why the notion of *feature* is related to *information compression*, let us start with a simple example. Suppose we have a retina with 100 input units (a 10x10 matrix) to encode distal stimuli, then only 10 channels to switch the signal received from the retina toward other processing units as shown in (1):

(1) hypothetical retina (10x10 matrix) encoded by a 10 channel hub



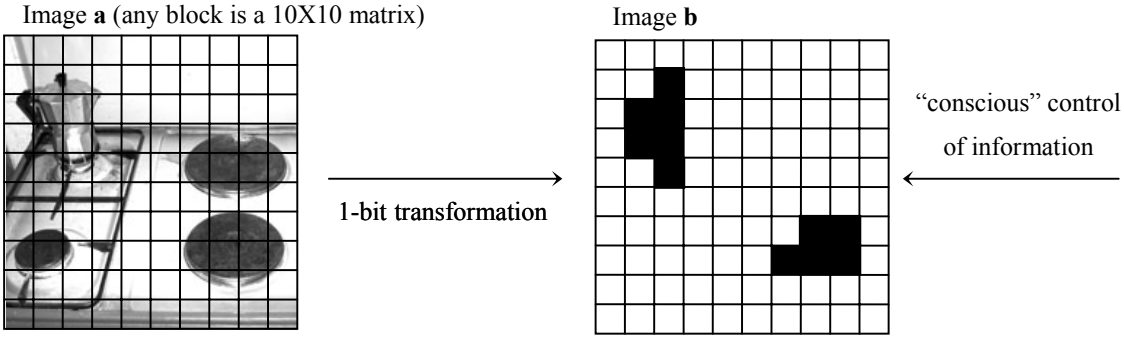
Assume that the input units (the retina) can behave analogically with respect to any distal stimulation (e.g. images), while the 10 channels are only capable of a digital modulation (namely these switches make the signal pass in a discrete quantity only if a certain amount of energy is received). Therefore, we would have ten bits of information to codify an infinite number of potential analogical inputs.

Note that for a wide range of stimuli, accurately perceived by the input units in their diversity, the transmitters will behave in the very same way, namely, they will *loose information* by flattening the variety received in input on only 2^{10} classes of answers (that could be potentially retransmitted to the inner processing units).

This structure could seem pretty inefficient, but answering the same way to different stimuli could be advantageous in terms of the *complexity* of the decision to make at further stages of processing.

Imagine, for instance, another analogical retina, composed by a 100X100 matrix of units, perfectly interfaced with an "infrared filter", that is, a 10X10 matrix capable of bringing only one bit of information per pixel (every 10X10 pixels block of the retina is linked to a different unique pixel of the filter). Tuning finely the sensitivity of the filter, we could classify every group of pixels as potentially dangerous (too hot for human contact) or safe for touching. Such a radical classification will avoid a lot of painful contacts, without requiring any further elaboration of the information received from the filter. Then, crucially, our "conscious" behavior could have access only to the 2-bits-color-filtered information in order to act properly, *loosing any previous feature* that leaded to the 2-bits-color-image:

(2) analogical retina (100x100 matrix) encoded by a 10x10 matrix (1bit x pixel)



This is an extreme simplification of the kind of categorization that the process of *digitalization* has to deal with and it is related to the notion of *feature* because, in fact, any pixel of the filter can be intended as a *feature detector* expressing a specific (binary) value. Then *features* are the minimal information units detectable at a specific level of computation that can be *elaborated*, that means *re-transmitted* as meaningful inputs to the inner processing units (then to further stages of the computation).

From a computational point of view, *information compression* is useful because it reduces the processing complexity both in terms of *memory load* (having 100 units to "control" is better than having 10.000) and in terms of *combinatorial possibilities* (patterns on a bigger matrix are more numerous than on a smaller one), biasing the representation of the information on the basis of only a relevant subset of information detected in the input (*features*). From this perspective, once used by the immediately subsequent stage of the computation, this information becomes inaccessible to further inner elaboration units.

We should notice that, in order to be computationally interesting in terms of complexity, *features* have to be *finite* in number and *discrete* in value[5].
In *linguistics* (but also in many other *cognitive sciences* such as *vision studies*) these properties are usually associated to the *categorization* (and *sub-categorization*) idea in *syntax*, *semantics* and *phonology*: the process of creating *classes* seems, in fact, to be triggered by *feature detection procedures*, fundamental to activate further elaborations/ categorizations, proceeding then, in a quasi-*deterministic* way (that is, extremely fast), to the construction of *complex objects* implicitly very informative even if poorer of accessible ("conscious") properties[6]. For these reasons, the notion of *feature* seems to have a privileged status that makes this "primitive" unit worth to be deeply explored in its theoretical and empirical nature.

Most of the current generative frameworks express crucial properties of the linguistic objects by using *features*: *features* determine the position that an object can occupy by triggering its *lexical insertion* (a transitive verb selects an argument, (3.a) Vs. (3.a')) and/or *displacement* (focalized elements stand usually at the beginning of the sentence rather than right after the verb that requires them as arguments, (3.b)); *features* provide essential *instructions* for the *performance* systems, namely, the *motor-perceptive*

[5] This does not imply they have to be binary.
[6] For instance, non-linguist speakers can hardly have conscious access, while speaking, to the notion of phrase structure or case information, but all of them have access to the thematic structure and to the event specification for any grammatical sentence produced in their language.

system and the *intentional-conceptual* system ([Mary] bears *phonetic features* that can be spelled out and heard, and it has *semantic features* that allow us to interpret this word as a [proper name] of an [animate] [person], (3.c)).

(3) a. John [$_{\text{transitive\_verb}}$ kisses [$_{\text{argument}}$ Mary]]

a'. *John [$_{\text{transitive\_verb}}$ kisses [$_{\text{determiner}}$ the]]

b. [$_{\text{focalization}}$ MARY] John [kisses [_ $_{\text{Mary}}$]]

c. Mary = {phon = /m æ r i/; sem = [[proper name], [animate], [person] ...]}

This is evident in many important generative frameworks such as the *Government and Binding* approach (to be presented in §2.2.1) and the *Minimalist Program* (§2.2.2): linguistic symbols are considered as *feature sets* composed by *phonetic*, *semantic* and *formal* (*selectional*, *categorial, case* etc.) *features* (Chomsky 1995:21-22). This common assumption brings up a first question:

why should we look for (natural) classes of *features*?

From a theoretical point of view, the main problem with unstructured *feature* bunches is the option of having an unpredictable interaction among linguistic properties: this would yield to an enormous computational burden, selecting, moreover, possible grammars in a too permissive way. Identifying *macro-classes* (or *typologies*) of *features* represents an elegant solution to reduce the potential interaction among properties, distinguishing levels of productive combination and predicting inert coexistences (§1.1.1).

There are however reasons to believe that these coarse-grained *typologies* can be further finely explored: for instance, much work in *phonology* (Clements 1985, Halle and al. 2000) led to a more natural and heuristic *hierarchy of features* (then properties suitable for hierarchies such as *inheritance*, *sisterhood*, *motherhood* etc. can be defined among *features*) instead of *flat* sets (§1.1.3). This idea has been inspiring also for some *morphosyntactic feature* analysis (Harley and Ritter 2002).

The bridge between *feature typologies* and *feature hierarchies* will be explored by investigating how and when *features* are introduced in the (linguistic) computation: in §1.1.2 I will briefly review Chomsky's idea on *inclusiveness* (any *feature* entering the computation should be projected from the *lexicon*, Chomsky 1995:231) and the distinction between *intrinsic features*, explicitly *valued* within the lexical representation of the linguistic object (for example *categorial features* in regularly suffixed adverbials), and *optional features*, which are underspecified within the lexical representation of the object (like *case* in English nouns, or *number*/*gender* in the English determiner "the"). This difference aims at distinguishing between *feature* values that are fully *listed* in the lexical entries (thus being a proper part of the *knowledge representation* of the cognitive object) and *feature* values that are *compatible* with the lexical entry but not directly specified in it (that is, they are not present in the mental representation of this object taken "in isolation", but associated to it *during* the computation).

The last three paragraphs will deal with three other relevant theoretical issues: the equivalence between *features* and *categories* within a set-theoretic framework (§1.1.4), the specification of some relevant patterns of distribution (§1.1.5) and, finally, some theoretical ranges of parameterization/variation on *feature structures* (§1.1.6).

1.1.1 TYPOLOGIES OF FEATURES

Following standard conventions, a *lexical entry*, that is the smallest meaningful informational cluster within our linguistic knowledge representation, should bear essentially three kinds of *features*: *phonetic*, *semantic* and *formal*.

This tripartite typology is justified by a *modular* view of the linguistic cognitive component: different principles/properties may hold at different *levels* of processing. Roughly speaking, the first two classes include *features* that are *instructions* for the *performance systems*, namely the *motor-perceptive* system (*phonetic features*) and the *intentional-conceptual* one (*semantic* ones), while the *formal features* are related to the *morpho-syntactic* processing unit.

This tripartite distinction is intuitively plausible since, for instance, *phonetic features* seem to have nothing to do in terms of structural combination with *semantic* ones: *features spreading* or *vowel harmony* (Clements 1985) are productive phenomena in phonological terms, but they are completely irrelevant for the meaning; the same is true for *mass/countable* or *animate/inanimate* distinction: they are intrinsically semantic relations, with related effects on the phrase structure, but with no specific effects on the phonological envelope of the words in most of the natural languages. An easy way to deal with these phenomena would be to assume independent *phonetic*, *semantic* and maybe *formal* (syntactic) representations related to each other by means of correspondence rules (cf. Jackendoff 1997).

From the narrowest perspective (namely the *Minimalist Program*, that will be presented in §2.2.2), the *performance* systems are external to the core linguistic (that is *syntactic*) processing module. This implies that these *performance* systems can respond to independent generative procedures and to different *feature* organizations.
More precisely, the *phonological* component interprets a sequence of sensory-motor instructions that have to be provided by *phonetic features* (e.g. [coronal], [voiced] etc.); the (structured) set of feature reaching the phonological module is called *Phonetic Form* (or simply *PF*).
Note that PF is characterized by a strict *linearization* requirement[7] imposed on these *features*. This property is often considered insignificant at other levels of processing (surely at the *semantic* level but, maybe, also at the *syntactic* one).

On the other hand, the *semantic* component, which receives in input a *Logical Form* (for short, *LF*), operates on *conceptual* primitives (i.e. [singular], [definite], [referential], [past], [epistemic modality] etc.) that directly affect the *meaning* of the expression. Among these *features*, no *linear order* is needed for interpretive purposes: by the *principle of compositionality* (Montague 1974) the meaning is largely

[7] Cf. Fox & Pesetsky 2004. We could predict a certain degree of specificity on this point, related to the phono-acustic modality; the story could be somehow different for other modalities (cf. Sign Languages).

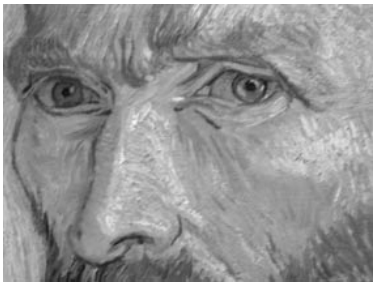
determined by the *structural configuration* of *conceptual-intentional features*. A plausible hypothesis[8] is that only *dominance* relations should be defined among *semantic features*.

Formal features represent the third (blurry) typological class, which groups *categorial*, *case* and *selectional features*, among others. The ambiguity of this class derives from the fact that most of these *features* can hardly have an independent nature with respect to the *semantic* and the *phonological* component: this is clear for *number* and *person φ-features* on DPs, but even *case* seems to have semantic correlates (at least it productively restricts the ability of any object bearing case markers to receive any semantic interpretation[9]); on the other hand, *selectional features* are restricted by semantic properties (the verb "die" takes as subject an *animate* object), while *focus features* have evident suprasegmental (i.e. phonological) and semantic correlates. Moreover, *dominance relations*, but maybe not *linear order*, seem to play an important role in the structural description of the *formal feature* organization of a well formed sentence.

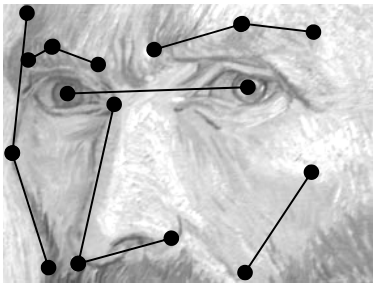
From a wider cognitive perspective, both *linearization* and *dominance* relations are relevant in many other processes such as *vision*, *motor control*, *tactile perception* etc.
In *vision*, for instance, we can think of *linearization* as the topological distribution of *features* in a perceived scene (4.a), while the *dominance* relations express a *hierarchy of constituency*, where dominated elements are parts of the dominating ones (4.c):

[8] See chapter 3 for a (formal) discussion on this point.
[9] This is surely true for *inherent case* (that is dependent from the thematic role assignment); *structural case* (that is, conversely, structurally determined) would require a more detailed discussion.

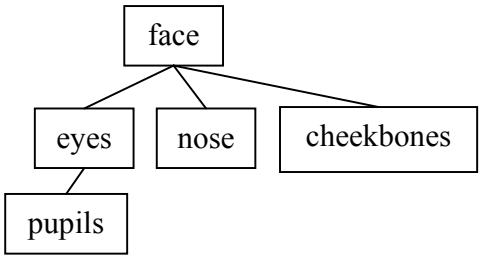
(4) a. original image



b. relevant topographic relations



c. constituent structure



Without pushing too far the parallelism between *language* and *vision*[10], we could notice that while *phonetic features* are clearly distinct from the topographical cues in a scene (points, edges, etc.), even if similar relations seem to hold in both domains, *semantic feature structures* could be fairly similar to the semantic representation of concepts used in language[11]. Moreover "formal" *features* hardly find a clear correlate in vision (color properties? movement?).

On the other hand, one property, quite accepted in vision science but mostly ignored in generative approaches to language, is the hierarchical structure of the processing (Marr 1982). On the perceptual side[12], vision starts from edges, bars, virtual lines and blobs detection (*primal sketch*), which, at a higher level, are combined using fading/shading information in order to lead to a description of the scene in terms shapes/surfaces orientation (*2½ Dimensional representation*). Then the process leads to a complete *3*

[10] That clearly are independent cognitive processes even if some underlying properties are significantly similar.

[11] Obviously the ontological status of this remark should be deeply evaluated, but this is out of the possibilities of this dissertation.

[12] We could imagine a "production" side of this process in terms of abilities of drawing/sculpting "grammatical" pictures/statues (Boccioni's ungrammaticality stands to "grammatical sculptures" as Marinetti's sonnets stand to "grammatical sentences").

Dimensional description of the perceived objects in terms of surfaces and volumetric primitives hierarchically arranged.

A similarity between Marr's and Chomsky's model is that both incorporate (implicitly or explicitly) a "safety device" that prevents the derivation from crashing: *features* do not enter levels of processing where they are *not interpretable*: uninterpretable φ-*features* (like *person* and *number*) are deleted from the verb before reaching LF in Chomsky's model[13], while *bars*, *virtual lines* and *blobs* do not directly enter the description of 3D objects in Marr's one.

In sum the *typologies of features* (such as *semantic*, *phonetic*, *formal*) are justified mainly with respect to the levels of processing at which they can be accessed and elaborated in a principled way (allegedly different from level to level).

[13] This happens essentially by checking, that is, by pairing uninterpretable features on the verb with their interpretable counterpart on the arguments. See §2.2.2 for more details.

1.1.2 FEATURES FROM THE LEXICON: THE INCLUSIVENESS CONDITION

In the previous paragraph, I referred several times to the notion of *lexical knowledge* without explicitly considering how this information could be represented. Following standard assumptions, the *lexicon* can be thought of a list of elements bearing the specification of *idiosyncratic* properties (that is *unpredictable* on the basis of *general principles* of linguistic processing). In fact, we could easily hop between static *knowledge* and general *processing instructions*, assuming *features* to be stored in the *lexicon* or derived by *general principles*; most of the time it will turn out to be equivalent, at least for computational purposes, as exemplified in (5):

(5) a. $\text{the}_{[\text{Determiner, selects-Noun}]}$ (lexical entry)

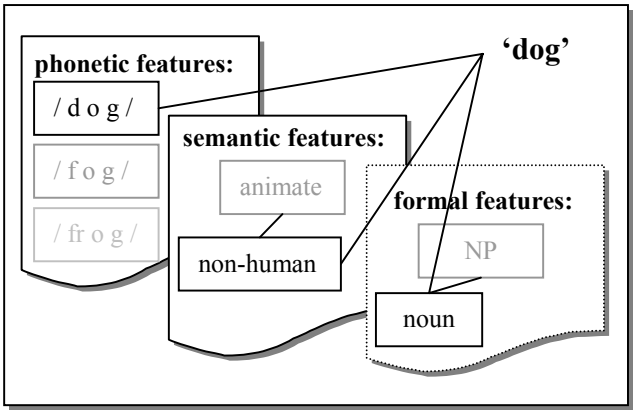
b. $\text{the}_{[\text{Det.}]}$ + Det. always selects Noun (lexical entry + principle)

A theory that assumes both the lexical entry in (5.a) and the principle in (5.b) at the same time will be redundant. We would avoid redundancy as far as we do not have empirical evidence of its utility[14]. A subtle difference between the two formalisms however exists and it lies on the fact that (5.a) does not allow for any generalization (namely, it is a language specific lexical entry), while (5.b) has a universal flavor and produces substantial limitations of the grammatical generative power: in this sense (5.b) seems the correct way of formalizing universal principles.

Going back to the lexical structure, in §1.1.1 it was explicitly assumed that the *lexicon* is the place where *features* are stored and, in addition, it has been suggested that it could be organized on multiple levels (*phonetic*, *semantic and formal*). Following these considerations, this could be a representation of the actual shape of a lexical entry:

(6) multiple levels of information in the lexical item “dog”:

[14] See learnability issues explored in Jackendoff 1997.



Within this perspective, the *lexicon* is a set of lexical elements *Lex* of the following form:

(7) *Lex*:{S,P,F} such that S, P and F are respectively finite sets of *semantic*, *phonetic* and *formal features*.

Therefore, this seems to be the locus of the (arbitrary) mapping among levels. To make a little step forward we could investigate the nature of the relationships among these levels of representation, firstly asking which *features* are introduced in the derivation and when it happens.

Chomsky attempts to answer this question by introducing a condition of inclusiveness (to be thoroughly discussed in §2.2.2):

(8) **Inclusiveness condition** (Chomsky 1995:228):
any structure formed by the computation is constituted of elements[15] already present in the lexical items selected for *N(umeration)*

Numeration is a one-time selection of items from the lexicon that will be available for the computation. For the time being, we are not interested in why this should be a one-time-for-all operation, but just to what extent it makes sense to think of the lexicon, structured as in (6), as the repository of the whole *feature* information needed to understand/produce a sentence.

From this perspective, selecting a lexical element would imply selecting the whole set of *features* associated to this element in the lexicon. Even if Chomsky explicitly avoids

[15] Namely *features*.

any *performance* implication for his abstract syntactic model, it is interesting to notice that the notion of *numeration* makes sense in *production*, but very little in *comprehension*. It is plausible, to some extent, that in order to produce a sentence the system has to select a set of lexical items with their complete featural make-up before starting to spell out words; but when we *perceive* a sentence, we *parse* it (that is, we assign it a *structure*) word by word, as soon as speech is perceived. This operation requires, first, that the lexicon be accessed more than once to retrieve items; second, some retrieved element could be "incomplete" or even wrong in featural make-up, as shown by many tricky phenomena such as the *garden path* data[16]:

(9) a. The horse raced past the barn fell (Bever 1970)

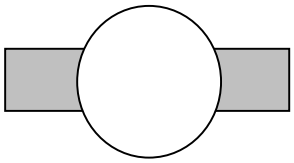
b. The horse (that was) raced past the barn fell (down)

In (9.a) the word "fell" produces a breakdown in *parsing*, since the reduced relative, disclosed in (9.b), is not recognized at the first reading.

This (*structural*) *ambiguity* shows that accessing the lexicon and retrieving the relevant (complete) set of *features* is not a trivial task. Then it seems plausible to assume that we drop unused *features* (or, better, we do not select them at all) according to a structural hypothesis that we are forced to make as soon as possible, integrating elements piecemeal in the sentence without waiting or requiring a complete numeration.

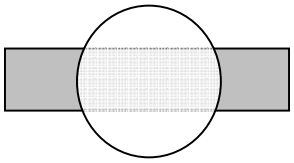
Something similar seems to happen also in vision, since we "interpret" pictures (that is, we assign *constituency* relations to the perceived elements) even if only a partial view is provided (10.a):

(10) a. input picture

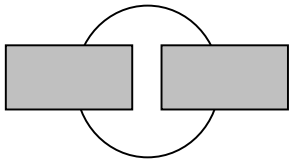


[16] In §2.2.3 more details will be provided on *Garden Path* and on *Incremental Parsing*.

b. interpreted/perceived objects (*constituency* relations based on detected edges)



c. real objects



Sometimes, other "perspectives" (10.c) force us to backtrack from early hypotheses (10.b).

From a speculative point of view, we could extend the same considerations to *production* as well: errors both in language (e.g. slips of tongue, false starts) and in vision (e.g. mis-drawing of lines, difficulty to complete a picture), suggest that even if the "semantic" content of the message is clear to the speaker/draughtsman, its practical realization could be less straightforward than we could think. This could be related to the fact that different classes of *features* are not always unambiguously associated to a given lexical item, hence they are not directly retrievable any time and at any level of the processing and, crucially, independently from the *performance* task.

A plausible hypothesis, that will be explored in this dissertation, is that *feature retrieval* is indeed part of our *competence* and that, depending on the *performance* perspective (*production* or *comprehension*), we access different classes of *features* trying to recover the whole set of features associated to the (lexical) element we need to parse/produce.

This hypothesis seems to be tenable at least within some generative linguistic frameworks such as *Distributed Morphology* (Halle and Marantz 1993): in this approach, lexical items are bunches of *phonetic features* inserted only at PF; before that level, the linguistic computation works only with de-lexicalized sets of *semantic-formal features*.

This asymmetric computation could suggest a distinction between *intrinsic features* that are associated to the elements directly in the lexicon (*off-line features assignment*) and *optional features* that are associated to these elements during the computation (*on-line features assignment*): for instance, the word "we" in English would bear the intrinsic *formal features* [first person], [plural], [nominative] and the *categorial* status of [pronoun]. These *features* are present even if the word is taken in isolation; this shows that the lexicon (or the "word shape" in *Lexical Functional Grammar* terms, following Bresnan 2001) provides enough information for *off-line features assignment* to this item.

On the other hand, consider words like "dog" ("do you *dog* me?" → [verb] Vs. "the *dog* chases the cat" → [noun]). This is a trivial case of *lexical ambiguity* (or *features underspecification*). In these cases, we have three logical possibilities:

a. both the *categorial features* [verb] and [noun] are selected, then one of them is dropped off during the computation;
b. the selection is procrastinated up to the point where we have more contextual information to identify the unique relevant *feature*;
c. only one *feature* is selected every time we try to parse a sentence. If the parse crashes then we backtrack and consider alternative *features*.

The last option is more attractive in terms of *computational complexity* (introducing extra *features* or delaying available choices, could make the processing time and the memory load grow pretty quickly).

This problem is also known as the "multiple tagging problem": it has been calculated (Derose 1988) that 40% of the words occurrences in the Brown Corpus is ambiguous, in terms of syntactic category, even if many of them are very easy to disambiguate: this is because of the different likelihood of the different possible tags.

We could describe the problem in terms of likelihood from two different points of view. First, there is a general likelihood for each item in the lexicon to receive specific *features* (e.g. *dog* is more likely to be a [noun] than a [verb]): this would be a case of *off-line feature assignment* (namely this information should be stored somewhere in our

lexical entry). Second, the likelihood of a *feature* is also dependent from other contextual *features* (what precedes/follows/dominate/is dominated by the element we would assign a tag to raises or lowers the chance of assigning a specific *feature* to this element). This seems to be a matter of *on-line feature assignment*, determined by the phrase structure model and probably deducible from an adequate knowledge structure: for example, in a classical rewriting-rules framework (e.g. Chomsky 1965), the *feature* for "dog" could be easily predictable following the only plausible *top-down expectation* as shown in (11):

(11) **grammar**: S > DP VP; DP > D N; VP > V DP; D > the; N > dog; N > cat; V > dog; V > chases

sentence: the dog chases the cat

tagging:

the > D	(by unambiguous bottom-up tagging)
dog > N/*V	(by top-down expectation: DP > D N)
chases > V	(by unambiguous bottom-up tagging)
the > D	(by unambiguous bottom-up tagging)
cat > N	(by unambiguous bottom-up tagging)

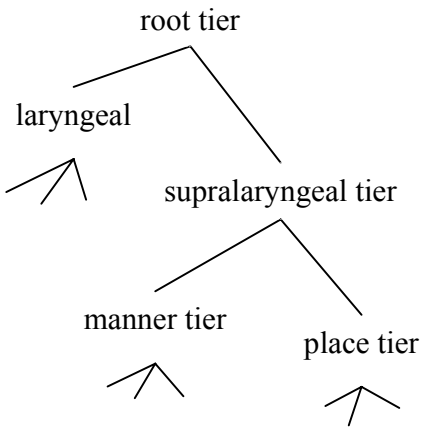
Summarizing the points discussed in this paragraph:

- lexical entries should be lists of *features* that are unpredictable from general principles;
- the lexicon could be organized on multiple, structurally independent, layers (crucially *semantic*, *phonetic* and, roughly speaking, *formal*) that could be accessed at different levels of processing;
- *features* are introduced in the derivation from the lexicon or assigned during the computation on the basis of the *feature structure* and by the application of specific (probably level dependent) principles;
- there could be (a)symmetries in the way *features* enter the computation depending on the *performance* tasks that access them (*comprehension* Vs. *production*).

1.1.3 FEATURE HIERARCHIES

An important (empirical) improvement in our *feature* knowledge representation could be achieved if instead of dealing with unstructured arrays of properties, we would explore hierarchical representations. For instance in *phonology*, a multi-tier hierarchical representation, as in (12), correctly characterizes sets of *features* that behave the same way with respect to *spreading* or *reduction* phenomena, for example, without affecting irrelevant *features* in different tiers (Clements 1985).

(12) *feature geometry* excerpt in phonology (Clements 1985)



Articulatory parameters (namely classes of *features*) sometimes present high degrees of independence (laryngeal configuration with respect to opening or closing the nasal cavity) while sometimes they are strictly constrained (spread glottal configuration determines non-voicing). Clements suggests that a hierarchical structure can be predictive of these phenomena: higher-branching categories tend to be more independent than lower ones (Clements 1985:230).

From a formal point of view, we should notice that this model expresses in a very elegant and compact way two important properties:

1. the **complementary distribution of features** (for instance *{[spread glottal], [voicing]});

2. the relevant **domain of application** of specific principles (*assimilation* works well only with *laryngeal features*);

Even from a syntactic perspective, some empirical data go in this direction: the *cartographic* frameworks (to be presented in more details in §2.2.4) for instance assume that the *functional* layers[17], classically identified as *Determiner Phrase*[18], *Inflectional Phrase*[19] and *Complementizer Phrase*[20], have been, during the last two decades, deeply explored and split up in subclasses to better express asymmetries in elements distribution. One of the best examples to explain this idea is Cinque's (1999) analysis of adverbials: Cinque suggests that the IP shell is more structured than we thought before and, crucially, this seems to be a universal generalization supported by robust empirical evidence.

Even in languages that do not have specific morphemes that realize Cinque's postulated functional heads, their existence is justified by the distribution of adverbial elements:

(13) a. Gianni *probabilmente spesso* mangia
G. *probably* *often* eats
b. *Gianni *spesso* *probabilmente* mangia

With a flat intonation, *probably* (in Italian as in English) has to take scope over *often*. The simplest way to express this intuition is to put the relevant *features* in a hierarchy that can be expressed by these three basic classes:

(14) *modals* > *temporal* > *aspectual*

This gross classification will be refined (following Cinque) in §2.2.4; for the time being, it is enough to highlight another important property that the *Cartographic Approach* suggests about hierarchies:

---

[17] Intend *functional* to be opposed to *lexical*: a *lexical* element bear argumental/eventive content into the sentence (*nouns* and *verbs*, for instance are lexical items); *functional* items (such as *prepositions*, *complementizers*, *determiners* etc.) modifies lexical items helping defining the *constituency* structure of the phrases, without affecting the lexical content.
[18] DP, e.g : $[_{DP}\ the_{D}\ dog_{N}]$
[19] IP, e.g : $[_{IP}\ does_{I}\ [_{VP}\ \ldots\ ]\ ]$
[20] CP, e.g : $[_{CP}\ that_{C}\ [_{VP}\ \ldots]\ ]$

3. *feature hierarchies* can be predictive of the **relative scope** that *features* have with respect to each other (and, maybe as epiphenomenon, on their linear order when linearization is required);

Turning finally to the semantic domain, here too *feature hierarchies* turn out to be relevant. Consider for instance the classic syllogism in (15.a) and the non-valid form of (15.b):

(15) a. Men are mortal, Socrates is a man, then Socrates is mortal
b. !Socrates is mortal, Socrates is a man, then men are mortal

This is just an example to point out how properties are inherited along structures: the validity of (15.a) follows from the relation of *hyponymy* between "Socrates" and "man": "Socrates" is an hyponym of "man", then it can inherit some properties form "man". Since mothers assign properties to daughters (15.a) but not vice versa (15.b):

4. we could predict **inheritance** relations of some *features* from hierarchies

Moreover, the meaning of a word can be predictive of its syntactic behavior: Levin (1993) tries, with interesting results, to derive classes of argumental selection from the semantics of the verbs (Levin 1993:2):

(16) a. *Gina *filled* lemonade into the pitcher
b. Gina *filled* the pitcher with lemonade

(17) a. Gina *poured* lemonade into the pitcher
b. *Gina *poured* the pitcher with lemonade

Fill and *pour* are subject to different constraints involving their arguments selection within the verbal phrase: the first verb is in the class that Levin calls *locative*[21]; according to Levin, crucial aspects of the meaning of the verb that determine its belonging to this class are the facts that it involves a change of location of substances/things and that it affects completely this substance/thing during the action. The syntactic specificity of this class is mainly expressed by the prepositional element

[21] Specifically in the *Spray/Load* subclass, Levin 1993:50.

that selects the *locatum argument* (substance or thing subject to the change of location), in this case constrained to be introduced by "with".

Pour, instead, is a member of the class called *benefactive verbs*: these verbs mainly express creation (or preparations as in this case, namely a process that can crucially be incremental). They require a double complement construction as the verb *fill*, but in this case the direct object has to be the *locatum argument*, and the location has to be introduced by a "to"-like locative preposition (*onto*, *into* etc.).

This is a good example of classes that do not describe hierarchies, but that suggest the plausibility of systematic relations among levels, in these cases between the *semantic* and the *syntactic* level:

5. **interface/mapping conditions** - the structure of the classes (maybe hierarchically organized) at one level (for instance syntactic alternations) could be determined[22] by the properties at other levels (i.e. semantic properties)

Summarizing, a *feature geometry* approach extended to all levels of linguistic analysis could be extremely useful to represent some important aspects of lexical knowledge. In particular, it allows for generalizations on:

1. distribution of *features*
2. domain of application of specific principles
3. relative scope (order) among *features*
4. properties inheritance
5. mapping relations among levels

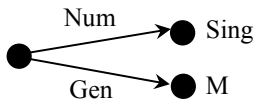
A unified hierarchy that represents at the same time *features* of any level is hardly conceivable. We rather need multiple hierarchies and a nearly *deterministic* way of mapping these structures across levels (Cf. Jackendoff 1997). A further theoretical possibility (that however will not be evaluated in these pages) is that *features* could be grouped in different ways once we refer to *comprehension* or to *production* (*feature geometry* in *phonology* is highly predictive in *production*, but maybe it could be less expressive than the standard theory of *distinctive features* in *comprehension*).

[22] or determine: the directionality of causality is not always unambiguous.

I will now introduce the formal devices that allow us to describe the proposed structures. The standard way of representing *features structures* (for instance in GPSG/HPSG) is by using *Attribute-Value Matrices* (18.a) or *Direct Graphs* (18.b), the two formalisms are equivalent as shown below:

(18) *Attribute-Value Matrices* *Direct Graphs*

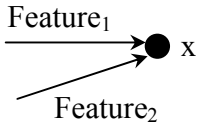
a. $\begin{bmatrix} \text{Num} = \text{Sing} \\ \text{Gen} = \text{M} \\ \dots \\ \text{Feature}_n = \text{Value}_n \end{bmatrix}$

b. 

Attribute-Value Matrices (henceforth AVM) are finite sets of symbols pairs: assuming F to be the set of *feature* sorts, and V the set of all possible values, an AVM is a function F → FxV. The value of a *feature* can be, in fact, either a (single) symbol of V (i.e. Num=Sing) or another symbol of F (Agr=Num,Gen).
It is theoretically possible (and acceptable in the grammar) to have more *features* types pointing to the same values as in (19.a) but not viceversa, as in (19.b):

(19) a. $\begin{bmatrix} \text{Feature}_1 = \text{Value}_1 \\ \text{Feature}_2 = \text{Value}_1 \end{bmatrix}$ b. * $\begin{bmatrix} \text{Feature}_1 = \text{Value}_1 \\ \text{Feature}_1 = \text{Value}_2 \end{bmatrix}$

(19.a) is a case of *structure sharing* (Pollard and Sag 1994): when *features* share the same value structure, we can use *pointers* (or *indices*) to relate the value structure:

(20) a. $\begin{bmatrix} \text{Feature}_1 = 1[x] \\ \text{Feature}_2 = 1 \end{bmatrix}$ b. 

The proposed *feature* geometries at different levels can be represented as in the following AVMs:

(21) a. phonology (based on Clements 1985):

$$\text{Root}\begin{bmatrix}\text{laryngeal}[\ldots]\\ \text{supralaryngeal}\begin{bmatrix}\text{manner}[\ldots]\\ \text{place}[\ldots]\end{bmatrix}\\ \ldots\end{bmatrix}$$

b. syntax (based on Cinque 1999):

$$\text{Verbal inflection}\begin{bmatrix}\text{mood}\begin{bmatrix}\text{act}[\ldots]\\ \text{valutative}[\ldots]\\ \text{probator}[\ldots]\\ \text{epistemic}[\ldots]\\ \ldots\\ \text{irrealis}[\ldots]\end{bmatrix}\\ \text{tense}\begin{bmatrix}\text{past}[\ldots]\\ \text{future}[\ldots]\\ \ldots\\ \text{anterior}[\ldots]\end{bmatrix}\\ \text{aspect}\begin{bmatrix}\text{habitual}[\ldots]\\ \text{ripetitive}[\ldots]\\ \text{frequenttative}[\ldots]\\ \ldots\\ \text{completive}[\ldots]\end{bmatrix}\\ \end{bmatrix}$$

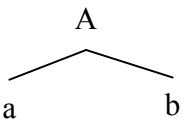
c. semantic (based on Levin 1993):

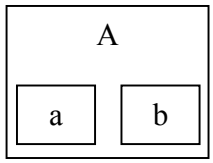
$$\text{Transitivity}\begin{bmatrix}\text{abj trans} = \text{subj intrans}\begin{bmatrix}\text{middle}[\ldots]\\ \text{causative}[\ldots]\\ \ldots\end{bmatrix}\\ \text{unexpressed obj}\begin{bmatrix}\text{unspec obj}[\ldots]\\ \text{body}-\text{part obj}[\ldots]\\ \ldots\end{bmatrix}\\ \text{conative}[\ldots]\\ \ldots\end{bmatrix}$$

1.1.4 CATEGORIAL FEATURES

In a *feature-hierarchical* approach, the notion of *category* is superfluous: the information that an element belongs to a category X can be expressed in terms of *features*, simply stating that this element has the (*categorial*) *feature* χ. This is a trivial observation from a formal point of view (as noted by Chomsky 1995:381 note 7), but it has strong theoretical implications for our model: i.e. we do not need different technologies in order to deal with *features*, *nodes* and *categories* (or *labels*, cf. §1.2.1, §2.2.2); operations on *features* (as the ones we will explore in §1.2) can apply to all these objects (then it is more accurate to refer with the term of *categorial feature* to what historically has been named *category*).

Intuitively, the concept of (*categorial*) *feature* is equivalent to the mathematical notion of *set*. In fact, when we refer to a grammatical category, for instance "noun", we refer to a homogeneous domain in which the included elements share some configurational, semantic or phonological properties. We usually use trees for representing *categorial* belonging but, in fact, the representations in (22) are equivalent:

(22) a. 

b. 

c. A[[a][b]]

We are used to consider the words "cat", "chase", "white" respectively as a *noun*, a *verb* and an *adjective*, because "cat" behaves pretty much the same, at least from a syntactic point of view, as words like "dog", "man" or "mouse", while "chase" behaves as "beat" or "push" and "white" as "sick" or "fat".

Roughly speaking, when two element can be substituted in a structure without affecting the grammaticality of the sentence, they belong in the same category:

(23) a. The cat chases the white mouse
b. The *dog* chases the white *man*
c. The cat *beats* the *fat* mouse

d. *The *push* chase the white mouse

e. *The cat *white* the fat mouse

f. *The cat chases the *drink* mouse

Computationally, this behavior has been classically caught, for example, by rewriting rules in *Context Free Grammars*:

(24) Determiner Phrase → D N;

D → the; D → a; D → this;

N → book; N → boy; N → dog;

The relation between the *categorial feature* D, or N, and the lexical entries is a one-to-many relation; this means that if the categorization is productive, it allows us to generalize a syntactic behavior, describing it in a compact way as shown in (24).

The advantages seem neat in terms of expressive power, but there are at least two tricky questions to be answered:

1. What allows us to associate a *feature* to an element?
2. How many *features* do we need in order to represent the lexicon?

Answering these questions is, again, mainly an empirical matter. To find an answer to the first question, let us concentrate on *comprehension*. We should observe that language, even if theoretically pervaded by ambiguities, is an efficient medium for information transmission; ambiguities are most of the time readily solved by the human interpreter (in fact, most of the time, the interpreter does not even perceive the "theoretical ambiguities" that the artificial parsers find[23]). What the human hearer captures as input is just a signal. The shape of this signal and the order of its subparts are the only possible clues for *feature retrieval*. This seems the general shape of the

[23] "I saw the man in the park with a binoculars" has three parses, only the first one is most of the time considered by humans:
a. I'm watching with a binoculars
b. I'm watching a man that has a binoculars
c. I'm watching a man in the park where there is a binocular.

problem at any level of processing: from a *phonetic* point of view, *features*[24] are detectable from relative changes in wave formants; this analysis requires accessing the "shape" of the signal and interpreting it in an ordered (temporal) way to appreciate productive differences. From a syntactic point of view, as LFG overtly assumes (Bresnan 2001), the dominance/scope relations among elements can be predicted either by the *element shape* (for instance, *case markers* are usually assumed to be the main source of structural information in non-configurational languages such as Warlpiri) or by the relative words position (as in English). From a semantic perspective, the word shape is a crucial link to the meaning, but the latter is often primed by the context (in a broad sense, what has already been presented/structured before the word we are analyzing). What we can say at this point is that, at any level, firstly the *shape* of the element then the *expectations* produced by the *context* help us in finding the (under)specified *features* to be associated to the element.

As for the second question (how many *features* the lexicon could be represented with?), it is easy to guess that the answer should be something like "as few as possible, avoiding overgeneralizations". No more than this could be said from a theoretical point of view. Assuming anyway a finite number of (*categorial*) *features*, a related problem is the nature of the sets generated by the choice of these *features*: many years ago, Ross addressed a similar issue: the "category squish" (Ross 1972). The empirical observation was that Nouns, Adjectives, Participials and Verbs constituted gross classes in which elements sometimes behave (with respect to their relative order, or to some specific requirement in terms of structural projection) on a border-line way; that makes it difficult to assign these elements to a unique category. Ross then suggested a continuum of the classical categories (N > Adj > Part > V) on which elements could be scattered. From a pure formal point of view, our framework does not allow for fuzzy categories (that is another way of expressing a *features continuum*): *features* are discrete entities; an element can bear one *feature* or not, namely, it can belong or not to a class without other possibilities.

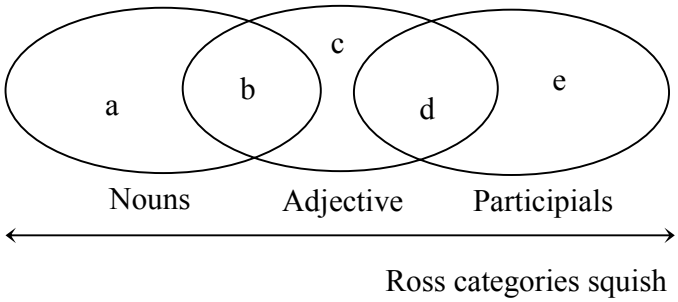
[24] Assume distinctive features standard theory.

Nonetheless, the phenomena discussed by Ross seem real and the problem must be accounted for in some way. Suppose we have six elements {a, b, c, d, e} with associated the set of features {α, β, γ} in the following way:

a → {α}; b → {α, β}; c → {β}; d → {β, γ}; e → {γ};

Every *feature* creates a set, for simplicity let us say that elements with feature α are *Nouns*, elements with feature β are *Adjectives* and elements with feature γ are *Verbs*. The graphic representation of these data is expressed below:

(25)



The elements *b* and *d* would be ambiguous if we should assign them a unique category; from this perspective, sets are clear-cut entities and the ambiguous behavior of some elements is due to the co-presence, within the same linguistic entity, of single features (or pattern of *features*) whose presence, per se, triggers a specific *categorial* status. Concretely, this suggests that the traditional "grammatical categories" might be epiphenomena determined by the interaction of more fine-grained *features*.

We could prevent that from happening in many ways, for example refining the sets (by a *subcategorization* operation) in the following way:

{α} → Nouns; {α, β} → Special_Adjective; {β} → Adjective ...

From a cross-linguistic perspective, we could observe, that *features* are assigned to lexical elements in a very diverse way among languages (as far as morphology and syntax are concerned). In this sense, the *scattering principle* proposed by Giorgi and Pianesi (Giorgi and Pianesi 1997) states that a bunch of *features* can be syncretically projected on the same head (that is, the same lexical item) or scattered on various (functional) heads as far as we get unique assignments: (functional)head$_{\{functional_feature\}}$.

Accepting a similar point of view, it would be possible to relegate this problem to a matter of parameterization. This seems to be a fairly suitable solution by now (that will be made more precise in §1.1.6).

1.1.5 FEATURES IN THE STRUCTURE: X-BAR THEORY AND FUNCTIONAL SEQUENCES

If we look at the pictures in (26) we should notice that even if there are *features* that can be *moved* or *modified*, changing productively the "semantic content" of the picture, this modifications can happen only according to some precise rules:

(26) a. b. c.* d.* e.*

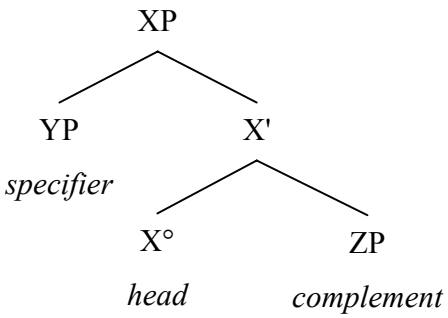
While a face with its components (eyes, mouth, hair... and expression) is easily recognized in (26.a-b), this is much harder when:

1. *features* are randomly *arranged* (26.c);
2. *features* do not *agree* (in (26.d), for instance, colors and edges do not agree in position);
3. something required is *missing* (26.e).

In this respect, language is quite alike vision and these constraints are incorporated in many generative frameworks.

An important constraint on *feature structures* is represented by the *X-bar theory* (Chomsky 1970, see §2.2.1 for a more thorough discussion on this point). This is the schema representing the generalizations on phrase structure to be captured:

(27)



Any well-formed phrase seems to conform to this pattern, where the *head* is the hub of the structure (a *noun*, a *verb* etc.), the *complement(s)* is the required element the head selects (e.g. any transitive verb selects a *direct object*) and the *specifier* expresses some further properties of the head, virtually optional unless specified (e.g. before Abney's DP hypothesis, Abney 1987, a *determiner*, like an article, specified the definiteness of the noun phrase). Any head seems to project a structure like the one expressed in (27) and some distributional constraints can be clearly stated (note the parallel with the constraints expressed in (26)):

1. the specifier is mostly (or always depending on the theory, cf. Kayne 1994) to the left of the head, while the complements position can vary (again, depending on the theory) across languages, even though it is stable within the same language (head-final languages are languages where complements precede their head, while in head-initial languages is the head that precedes its complements);
2. the specifier *agrees* (at least in person, number and gender, where these *features* are specified in the language) with their head;
3. complements, if *required* (that is, *selected*) by the head, have to be present in the structure. If not the sentence is ungrammatical.

Starke (Starke 2001:157) points out that *X-bar theory* is at least redundant, if not completely misleading, if we assume at the same time a highly articulated functional hierarchy (like Cinque's): the combination of these two theories would predict duplicated positions (then *features*) for every functional element (specifier position plus head position). It is in fact very hard, if they exist at all, to find languages that fill at the same time the specifier and the head position, doubly checking the same *feature* (*doubly-filled nothing*, Starke 2002).

In this sense, the sequence of functional elements (let us call it *Fseq* following Starke 2001) is a better starting point than *X-bar theory* in terms of constraints, but, in fact, it is not that efficient in terms of *cross-categorial* generalizations: what *X-bar theory* explicitly predicts and what *Fseq* has to postulate with independent hierarchies, are the similarities among phrase structures.

Recently, some similarity has been pushed quite far, so to describe interesting parallel (functional) structures both in nominal and in verbal domain (this would prevent the grammar from postulating independent *FSeqs*):

1. *nominalization*, the same morpheme can get both verbal ("destroy") or nominal ("destruction") morphology;
2. *adjectival counterpart of adverbial forms*, a great part of adverbial forms ("quickly") in the VP domain have equivalent adjectival forms ("quick") in NP domain;
3. *ontological similarities*, there are some ontological similarities between nominal and verbal properties (*mass/countable* and *atelic/telic* distinction, Verkuyl 1999, *tenses* and *pronouns*, Partee 1973, *modality* and *specificity*, Filip 2000), moreover these properties seem to interact in some way (*Specific Quantity Hypothesis*, Verkuyl 1999);
4. *delays in acquisition*, children show similar deficiencies in early stage of language acquisition in both domains (for example *optional infinitives* in verbal functional domain, Wexler 1994, *bare nouns* in nominal functional domain, Hoekstra and Hyams 1995).

Economical considerations suggest dealing with these facts exploiting as much as possible these productive similarities. Ogawa (2001), for instance, suggests that the functional shell above NP and VP is similar in many ways. Even though there are many theoretical advantages in accepting this intuition, this is not a problem-free solution and these problematic aspects cannot be sufficiently addressed within this dissertation. For the time being, I will then just try to describe some general regularities as follows without forcing any functional parallelism:

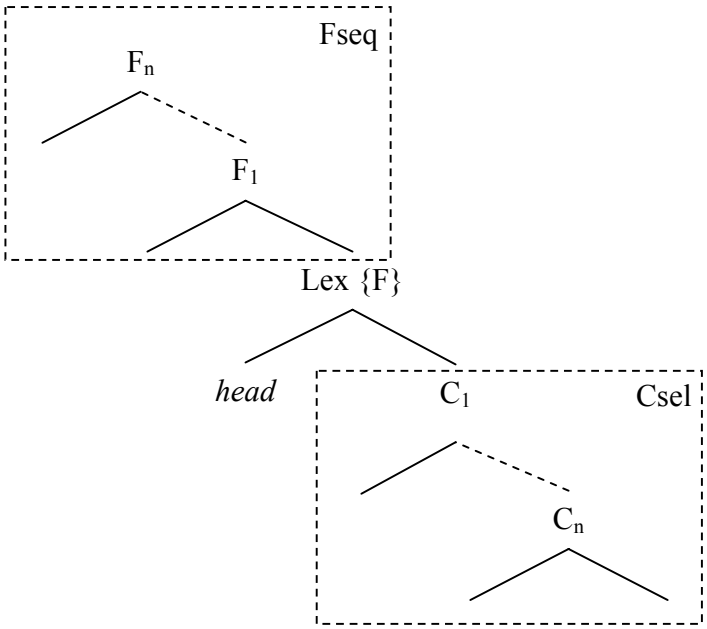
(28) *structural projection principle*

any lexical element (Lex), either a Noun or a Verb, selected to enter the computation, projects a structural skeleton SK consisting of a finite set of selectional constraints (Csel) and a finite set of functional specifications (Fseq) based on the *formal/semantic features* F present in Lex:

$SK(Lex\{F\}) \rightarrow Fseq + Csel$

This is a graphical representation of the principle (with no implication, for the moment, as to the actual position of the real elements):

(29)



Csel and Fseq should have asymmetric properties (in terms of scope) that I will try to describe in the rest of this dissertation.

1.1.6 PARAMETERIZATION ON FEATURES

The last issue I will discuss in this chapter is how *features* (*structures*) can vary across languages. It is a common assumption that any natural language selects a subset of *features* from a universal set of possibilities (Chomsky 2001:4). This is quite clear for the phonological component: there are data (Eimas 1975) showing that newborns are sensitive to phonetic distinctions that are not present in the mother tongue and that then they "lose"[25] this ability at later ages; this assumption is much more controversial for syntactic and *semantic features*, since there is no evidence, at least to my knowledge, showing any loss of discrimination between semantic/syntactic properties (for instance, *states* Vs *events* or *nouns* Vs. *verbs*) to which infants were sensitive. Definitely, that could be related to the difficulty of testing these properties with newborns, but, provided that Sapir-Worf hypothesis has been discredited (Miller & Johnson-Laird 1976) and translation among languages is hard but possible, we can assume at least that *semantic features* (and their organization) are pretty much the same across languages (cf. Chomsky 2001:4, but see Chierchia at al. 1998 for an alternative proposal).

Starting from Chomsky 1981 parametric approaches attained a very good *descriptive* power in terms of predicted cross-linguistic variations by using a relative little number of parameters: head-complement position, *pro-drop/non-pro-drop*[26], constituent *movements*. The leading idea was to consider parameters as related only to *formal features* (or, more restrictively, only to *formal features* of functional elements: Belletti and Rizzi 1988, Fukui 1986, Borer 1984), particularly in the mapping between these *features* and elements in the lexicon; recent evolutions of this framework accept the assumption that parameterization is restricted to the lexicon (Chomsky 2001:4). More generally, following the classical perspective, parameters look like variables playing a

[25] See Kuhl (1994) and related work for a more precise definition of the inaccurate idea of "loss" of discriminative ability.

[26] This parameter controls the optionality of the subject realization: in *pro-drop* languages the subject can be non-overtly realized (in this sense *pro* is the pronominal null subject, e.g. Italian), while in non-pro-drop languages this option is not available and even if semantically empty (expletives) it has to be present (e.g. English), Jaeglli & Safir 1989.

role in the application of universal principles, not as functions that directly select *features* or modify their structure.

From a formal perspective, this notion of parameterization can be described at least in three ways:

(30) three kind of parameterization on *features*:

a. **subset parameterization** - a proper Subset of Features (SF_{level}) is selected from the Universal Feature domain at the specific level (UF_{level}) by a language specific function (P_{subset}):
$P_{subset}(UF_{level}) \rightarrow SF_{level}$

(e.g. universal set of features: {A, B, C} → language selected features {A, B})

b. **structure parameterization** - a language specific function ($P_{structure}$) maps Universal Features Structures (UFS_{level}) to different ones at the same level of processing (SFS_{level}):
$P_{structure}(UFS_{level}) \rightarrow SFS_{level}$

(e.g. universal structure of features: $[_A\ [_B\ [_C\ ]]]$ →
language specific structure of features $[_B\ [_A\ [_C\ ]]]$)

c. **mapping parameterization** - a language specific function maps a set of Features at one level (F_{level}) to single elements at different levels (e_{new_evel}):
$P_{mapping}(F_{level}) \rightarrow e_{new\_level}$

(e.g. set of features at some level: {A, B, C} →
language specific mapping $\{[_{A, B}\ X], [_C\ Y]\}$)

(30.a) is the simplest case of phonetic parameterization, for instance:

$P_{subset\_chinese}(\{consonantic, sonorant, voice, continuous, coronal, anterior, lateral\}) \rightarrow$
{consonantic, sonorant, voice, continuous, coronal, anterior}

In this case, speakers of this hypothetical (Chinese-like) language would not perceive any distinction between /l/ and /r/.

Structural parameterization (30.b) is a tricky possibility: if we would have this option and any language would implement it, how could we assume that a *universal feature*

structure exists at all? Many empirical results actually show the opposite: even if from surface phenomena (for example word order) we could guess that *features structures* vary across-languages (e.g. the functional structure above VP), there are many reasons to believe that these are only apparent counterexamples to a *universal feature hierarchy* (Cinque 1999). I will assume that (30.b) is just a logical abstract possibility in fact not present in human languages[27].

On the contrary, (30.c) is, in my opinion, the most pervasive way of parameterizing *features*: the lexicon can be considered as an instantiation of functions like these ones; a lexical item can be seen as a set of functions mapping the "word level" to others levels roughly in the following way:

$P_{mapping}(</d/,/o/,/g/>_{phonology}) \rightarrow dog_{lexical\_root}$

$P_{mapping}(\{N...\}_{syntax}) \rightarrow dog_{lexical\_root}$

$P_{mapping}(\{animate, countable...\}_{semantics}) \rightarrow dog_{lexical\_root}$

The same is true for principles such as the *scattering principle* (Giorgi and Pianesi 1997, briefly introduced in §1.1.4), which map *functional features* to *categorial features* (namely to *features* at different hierarchical levels) in the following way:

$P_{mapping}(\{anterior, terminated\}_{syntactic\_features}) \rightarrow T_{syntactic\_feature\_at\_higher\_level}$

[27] But see Keenan and Stabler 2004.

1.2 BASIC OPERATIONS ON FEATURES

Finding features and classifying them is clearly not the unique task a cognitive process is responsible for: in order to capture the generative capacity of the human cognition, we should assume that novel patterns are not just recognized as bringing specific features, then belonging to specific classes, but even arranged in some structural scaffolding that supports an infinite (even if constrained) number of original, compositionally meaningful, combinations. To explain these facts, in this chapter, I will try to investigate how features can be recursively combined depending on their nature and their structural configuration.

In fact, as I presented them, features seem to be mostly inert entities, which do not specify, per se, how linguistic objects can be (un)built. Even though feature structures potentially represent powerful constraints on many possible operations, if we do not explicit the *dynamics* that allows them to be arranged in a precise structural way, these features would be completely useless.

We could call *structure building operations* the procedures that allow us to *combine* features and predict their *dynamics*. Following recent generative trends (c.f. §2.2) I will explore two operations (*merge* and *move*), trying moreover to afford a comparison with respect to vision in order to highlight some important cognitive generalizations:

(31) **Merge**
it is the core (recursive) building operation; when objects are combined (paired) together, this operation determines the feature structure of the resulting object;

(32) **Move**
it is an alternative building operation able to interpret an incomplete feature set (essentially in terms of π or σ) using features of a distant (that is, non-adjacent) object, then *merging* the interpreted element in the required position.

I will procrastinate up to the next chapter the discussion about the possibility to include these operations as a proper part of the grammar or as belonging to independent *performance*-related domains, while I will dedicate the next two paragraphs to the discussion of these operations. In the rest of the chapter I will introduce a principled distinction between *derivational* and *representational* approaches to the *competence* description (§1.3). Even if this discussion will be explored in detail in §2.2, I would address the issue concerning the fact that many geometrical relations definable on a whole tree *representation* (such as Fseq generalizations, *C-command* relations) can be reformulated in *derivational* terms without any loss of *descriptive adequacy*, moreover gaining in cognitive plausibility and in terms of computational complexity reduction (§1.4).

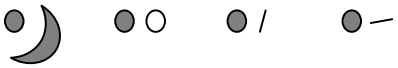
1.2.1 MERGE AS UNIFICATION

Before exploring linguistic data, let us start playing with pictures in order to highlight some interesting facts:

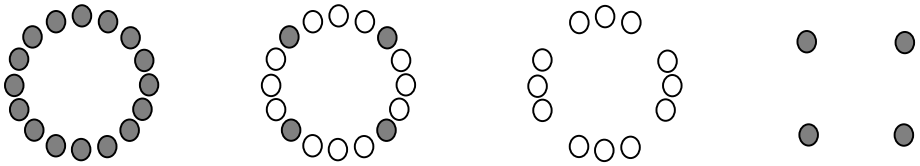
(33) a.

b.

c.

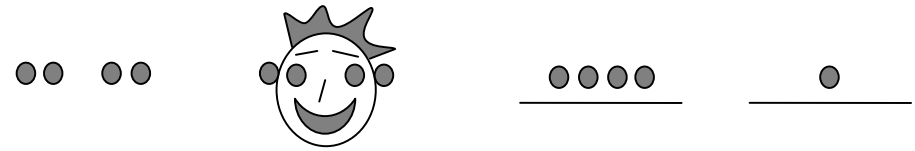


d.



i. ii. iii. iv.

e.



i. ii. iii. iv.

The examples in (33) tell us something about *grouping*. This "operation" is possible if the objects are "close enough" (like in (33.a) but not in (33.b)) and, in general, if they share some relevant features (33.a,c); which features can be shared in order to trigger grouping is ultimately an empirical matter: for instance, while the result of the grouping operation in (33.a) is pretty neat (the dots are combined in groups of two as result of their "compatible" positional features), the groups in (33.c) are much more obscure: *color*, *shape*, simple *proximity* seem to be able to trigger a group status but with different intensities[28].

[28] See Palmer 1999 for a detailed discussion on *grouping*.

On the other hand, (33.d) shows that two different features present in the same object (e.g. *position* and *luminosity*) can be used (separately) to create two different groups, namely a circle-group and a square-group. Both objects are visible in (33.d.ii), while decomposing the dots in (33.d.iii) and (33.d.iv), the former (33.d.iii) looses the ability to look like a clean circle-group as the one in (33.d.i) (this shows that the elements, with their positional features, belonging to the square-group are indeed crucial to complete the circle-group even if engaged in another *grouping*).
Finally (33.e) shows that groups can be overridden, depending on the context ((33.e.i) Vs. (33.e.ii)); moreover there are properties in the group (parallelism between the group of dots and a line (33.e.iii)) that are not present in their single components (a dot cannot be parallel to a line, (33.e.iv)).

These apparently unrelated phenomena show, indeed, properties that seem to be present also in language (with some fundamental differences):

(34) a. [[the dog] runs]

b. ?[[the dog [that Mary found in front of the door of her office when she was running out to say bye to Joan that was walking on the street at that time]] runs]

c. i. [the dog]

ii. *[the was]

iii. ?[probably yesterday]

iv. [probably yesterday [John [saw Mary]]]

d. *John [$_a$ kisses [$_b$ [Mary] $_a$] dances $_b$]

e. i. [John [saw Mary]]

ii. [[The father of John][saw Mary]] Vs.

The father of *[John [saw Mary]]

iii. *[[dog] runs] *[[the] runs]

The parallel with the data reported in (33) is not always direct, but, with some extent, we can say that words can be “grouped” if they are in a “local” structural configuration

((34.a) Vs. (34.b)) and if they bear compatible features (34.c); moreover groups are not simply defined only on the basis of the properties of their elements but they can be overridden depending on the context[29] ((34.c.iii-iv), (34.e.i-ii)). Eventually, groups show properties that do not belong to their single components ((34.a) Vs. (34.e.iii)).

Then some sort of "grouping" seems to happen even in language. In fact, there is a very precise way to build groups with "words": a principled way to describe the emergence of these structures has been (re)introduced in generative linguistics by the *Minimalist Program* (Chomsky 1995, to be discussed in §2.2.2) with the idea of *merge*; this operation is assumed to be the basic structure building component: it basically takes a pair of computationally well formed objects, A and B, and replaces them by a new object C computationally well formed as well.

As we mentioned before, there are important differences between *merge* in language and *grouping* in vision. At least two dissimilarities are worth to be mentioned here:

i. *merge*, in language, is assumed to be a *binary* function; this very same constraint is not required for *grouping* in vision (in (33.d.i) dots can enter the grouping relation all at once);
ii. in *vision*, but not in *language*, the very same object can enter *multiple grouping relations*[30] as shown by the opposition (33.d) Vs. (34.d)

Despite these idiosyncrasies[31], it seems possible to describe some common properties underlined by both processes; from this perspective, (a sort of) *merge* (both in *vision* and in *language*) happens if and only if:

1. a relevant *local configuration* is met among elements that are combined;
2. the feature structures of these elements is *compatible*.

[29] See *garden path* data in §2.2.3 for more details on this point.
[30] In language, this would imply having crossing branches in the structural description of a sentence.
[31] Probably accountable for in terms of parallel pathways for processing different features, Palmer 1999, Kandel and al. 2000.

Turning to the structure of the resulting linguistic object, standard assumptions (Chomsky 1995:243) suggest that this new entity C should have, minimally, the form $\gamma\{A, B\}$ where A and B are the *merged* constituents and γ is an identifier, called *label*, that would express the *category* which C belongs to.

Economy conditions lead one to assume that we should restrict as much as possible the searching space when we build new objects, then the optimal solution would be accessing only A and B in order to build C. Thinking recursively (that is, any *merge* is followed by another *merge*), after the *merge* operation, C should be (informationally) rich enough to dispense any further operation that applies to it, to retrieve its previous constituents. This is a radical assumption, hardly sustainable within the standard minimalist framework (as it will be clear later on[32]), but it seems the natural (null) hypothesis from a computational perspective which needs to be seriously evaluated.

For the time being, to understand what kind of information (namely which feature structure) should compose the object C, let us analyze four theoretical possibilities[33]:

(35) a. C is the intersection of A and B;
b. C is the union of A and B;
c. C is either A or B;

Logical considerations on possible outputs rule out both (35.a) (besides being null (36.a), the intersection between A and B could be irrelevant, (36.a')) and (35.b) (the union between α and β could be contradictory (36.b) or inconsistent (36.b')):

(36) a. $[_{\text{Adverbial}}\ A] \cap [_{\text{Verb}}\ B] = \emptyset$
a'. $[_{\text{D, specific, sing}}\ A] \cap [_{\text{N, animate, sing, masc}}\ B] = {}^{?}[_{\text{sing}}\ ]$
b. $[_{\text{D, sing}}\ A] \cup [_{\text{N, plur, masc}}\ B] = [_{\text{?(D,N), *(sing, plur), masc}}\ ]$
b'. $[_{\text{V, transitive}}\ A] \cup [_{\text{N, dative}}\ B] = [_{\text{?(V,N), \#(transitive, dative)}}\ ]$

[32] As Brody 2001 points out, any constituent, that is a result of a *merge* operation, has to be accessible in its subparts in order to account for *movement* (but this seems to happen only in a model that assumes a *bottom-to-top* perspective, cf. §2.2).
[33] This extends Chomsky's discussion, Chomsky 1995:244.

(35.c) is the solution provided by Chomsky, but his discussion is driven simply by the necessity to derive the *label*, rather than the whole set of features of C.
From the opposition (34.a) Vs. (34.e.iii), it is clear that some feature (e.g. DP) is not present in any of the components of the phrase (e.g. neither in [$_D$ the] nor in [$_N$ dog]), but, generally, in the ordered combination of the elements (i.e. <[$_D$ the], [$_N$ dog]>); in this sense (35.c), namely the selection of only one element, could not be the whole story and, definitely, it would not prevent further operations from accessing the constituent structure.

A related issue, which probably could help us understanding the featural nature of the *merge* outcome, is about what triggers the *merge* operation: the simplest hypothesis to be evaluated is that either A or B has at least one feature that requires *merge*. This is quite in line with what we usually call *selectional requirements* (standard *C(ategorial)-selection* but also *S(emantic)-selection*, Pesetsky 1982):

(37) a. *[John [kisses ∅]]
b. *[John [kisses girl]]
c. [John [kisses [the girl]]]
d. *[John [kiss [the girl]]]
e. [John often [kisses [the girl]]]
f. [[John [kisses [the girl]]][in the park]]

It is a property of the verb “kiss” to require two arguments, an agent and a patient (this expresses the verb *valence*, Pollard and Sag 1994), in order to meet grammaticality ((37.a) Vs. (37.c)). Going back to the phrase structure discussion (§1.1.5) we should notice that this kind of requirement was expressed exactly by the *C(omplement)Sel(elction)* domain. Then we could conclude that *CSel* features trigger a first kind of *merge*.
There are reasons (to be explored in chapter 3) to believe that this is not the only kind of *merge*-trigger. Many examples in (33) cannot be easily accounted for in terms of selectional requirement: *grouping* seems indeed *optional* in many cases, since the absence of part of the elements present in the pictures would not produce

"ungrammatical" images[34]. A parallel phenomenon in language is expressed by the "optionality" of elements such as *adverbials* in (34.c.iii-iv), (37.e) or *adjunct phrases* (37.f). Their presence is *compatible* with the *grouping*/*merge* options, but it cannot be accounted for simply in terms of selection. Then some "compatibility check" is required and it does not inspect only the lexical information, but rather the grammatical knowledge about larger phrase structure possibilities (e.g. *F(unctional)Seq(uences)*).
Note that a similar "compatibility check" should be present also to rule out merging operations such as the one presented in (37.b) (in this case the argument does not have the required *argumental* feature) and (37.d) (since functional position are involved, i.e. the subject position, *agreement* is required as postulated, even though not explained, in §1.1.5).
Both sorts of *merge* have in common the ability to prevent the same operation from being repeated more than once:

(37) g. *[John often [kisses [the girl] [Mary]]]
h. ??[John [[frequently] [kisses [the girl]] [with frequency]]]

The verb "kiss" once satisfied its requirements (once *saturated*, in Pollard and Sag's 1994 terminology) should not show any other selectional requirement (37.g), then we could simply assume its *selectional features* are "removed" once satisfied; on the other hand, even when a functional specification has been expressed, there is no more room for other elements bearing the very same feature (37.h). These facts are important cues that suggest us what survives after *merge*.

An important (computational) option to be seriously evaluated, since (partially) compatible with the previous observations, is that *merge* be nothing but a *unification algorithm*.

[34] Where "ungrammatical" means that the pictorial elements did not attained a specific "object status".

(38) **unification algorithm** (adapted from Shieber 1986)

given two sets of features, A and B, their unification C, expressed by C = A ∪ B, is the smallest possible set that comprehends all features present either in A or in B without duplications. The unification is undefined when features are incompatible (either they clash in value (36.b), or they are inconsistent (36.b')).

This seems to be a natural refinement of the option given in (35.b), but it would conflict with the (maybe apparent) evidence that, sometimes, features appear ([DP] from [[D][N]]), sometimes they disappear (for instance selectional requirements, when satisfied, should be removed from the resulting object).

In this sense, Chomsky's theory and the *unification algorithm* are somehow incompatible[35]. In the rest of this dissertation, I will try to pursue the idea that this incompatibility is only apparent and that the *unification algorithm* corresponds to the notion of *merge* if we accept few departures from the standard minimalist view, namely:

1. *CSel* are not simple features, but *expectations* on the phrase structure (then they do not need to be "deleted", since they simply express a required structural projection to be satisfied "after" the lexical head that have these selectional needs[36]);
2. $[_D\ ]$ and $[_N\ ]$ are not incompatible *labels* (then they are not different *values* for the same *Cat*(egorial) *feature*), but, indeed, $[_N\ ]$ is a *lexical feature* and $[_D\ ]$ a *functional* one, compatible, according to the FSeq projected by the lexical nominal head and potentially coexistent in phrase structures like the following ones: $[_{D\,N}\ [_D$ the $]\ [_D$ dog $]]$ or $[_{D\,N}$ John$]$.

We can now summarize the relevant points discussed in this paragraph: *merge* is assumed to be the basic (simplest) operation for (recursively) building linguistic objects. It seems plausible to assume that many properties of this operation are in common with other cognitive systems (e.g. vision), essentially:

---

[35] Jackendoff 1997:13 explicitly states that none of Chomsky's theories incorporates unification algorithms.

[36] This issue will be explored in §3.4.3.

a. it is an operation on *features*;
b. it *unifies* feature structures (checking their compatibility);
c. its scope is *strictly local*.

The simplest hypothesis is that what makes this *merge* operation special for the linguistic (Vs. visual) system is just the feature structure it applies to and the nature/number of pathways able to process these features (in parallel).

1.2.2 NON-LOCAL DEPENDENCIES: MOVE AND MERGE AGAIN

It is a matter of fact that language can express in a variety of ways non-local dependencies between elements:

(39) a. $[_X$ What $]_i$ do you think $[_Y$ _ $]_i$? (*movement*)

b. $[_X$ John $]_i$ gave $[_Y$ his $]_i$ picture to Mary (*pronominal binding*)

c. John $[_X$ bought the book$]_i$ and Mary did too $[_Y$ _ $]_i$ (*ellipsis*)

This (non exhaustive) list circumscribes a relevant set of phenomena such that an element X determines the "interpretation" of a distant (that is, *non-adjacent*) pronominal/empty object Y (the indices express this "binding" relation). While these linguistic phenomena will be explored in their varieties in the next chapters, we should note now that something fairly similar happens in *vision* too:

(40) a. a'. a''. ??

a'''. a'v. av.

b. b'. b''. b'''.

c. c'. c''. ??

An occulted corner (40.a) is interpreted as (40.a') rather than (40.a"); this fact could be accounted for simply in terms of "local coherence" (two straight convergent lines are

supposed to joint at some point forming the hidden corner[37]). It is however plausible to assume that not only local features contribute to the interpretation of a hidden element: (40.a''') can be easily mis-interpreted as (40.a^{iv}), while this is much more difficult in (40.a^{v}) where a single feature (an unambiguous straight corner) disambiguate all other hidden corners (even the opposite, non adjacent, one).
This "non-local" relation is maybe more clear in contexts where some sort of "pronominal binding" is realized: the convexity/concavity feature on the corner in (40.b) can be disambiguated depending on the "context" (the closure side determines the convexity). This is however a property of the whole figure and not only of the local geometry of the sides next to the corner as shown in (40.b'''). Finally, *simplicity* (e.g. *symmetry*, Palmer 1999) cannot be the whole story, since familiar shapes such as a Fiat 500, can hardly be interpreted as (40.c''') when a part of the picture is occulted (40.c).

The parallelism between *vision* and *language* resides on the fact that a hidden part of the object (that is, an *empty element* or *trace*) can be interpreted (or *reconstructed*) depending on the visible features (pronounced words) that enter in a relevant structural configuration with the hidden part (as in *movement* or *ellipsis*), triggering a clear *expectation* on the whole picture[38] (an *empty element* in a grammatical sentence).
On the other hand, a visible ambiguous part of an object (namely an element with some *underspecified* features like a *pronominal form*) can be interpreted differently (that is, these underspecified features can be *valued*) depending on the structural configuration of the features in the picture (pretty much like *pronominal binding*).

With respect to language, the nature/necessity of a *move* operation, (39.a), in a grammatical framework that already implements *merge* has recently been questioned (Kitahara 1994, Chomsky 1995-2001, Epstein and al. 1998, Starke 2001 among others). These discussions are partially justified by the observation that any *movement*

[37] This is a principle of *figural simplicity* or *Prägnanz*, Palmer 1999:288.
[38] A difference with respect to the language resides on the fact that an expectation in vision can be more easily deceived without necessarily lead to "ungrammaticality".

operation, indeed, should require a re-*merge* of the *moved* element in another (relatively distant) structural position.
Whether or not this "reduction" is a real option, we could look at this phenomenon from a wider perspective in order to capture some relevant properties that are not evident at all if we stick to the idea that a "chain" (the sequence of elements related by the *movement* operation) is a linguistic object per se, rather than an epiphenomenon of some deeper structure building process.

This second option is easily graspable in *vision*, where the interpretation of an incomplete element (either hidden or ambiguous with respect to some feature) is triggered essentially by two factors:

1. the perception of a missing/incomplete element/feature so to fulfil an "object" status;
2. the presence of an *active pattern* perceived as structurally compatible with the incomplete element.

We mentioned earlier that the perception of a missing/incomplete element within an object triggers the expectation for this part; once this expectation is expressed, the *active pattern(s)* can be inspected in order to fulfil this requirement.
Translating this analysis to language we can try to interpret *movement* in (39.a) in terms of *expectation triggers* and *active patterns*: as we pointed out in the previous paragraph, *CSel* "features" cause *expectations*, then if right after the lexical item bearing these *expectations* we do not find anything suitable to be integrated, these *expectations* should cause the inspection of the *active pattern(s)*. Both in *vision* and in *language*, we can think of *active patterns* as to elements present in a sort of *accessible memory* (maybe a short-term memory). Since only "structurally compatible" items can be used to fulfil a precise expectation, we can assume that only specific elements should be "moved" in this *accessible memory*. In (40.a) the compatibility with the square hypothesis "activates" the complete corner(s), while in (39.a) the compatibility with a

full sentence hypothesis activates "what" as a potential argument of a potential verbal phrase (not yet present in the structure[39]).
When the verbal head is found and a missing argument expected, the accessible memory can be inspected in order to complete the sentence. This last step requires a re-*merge* (to be intended as *unification* of the relevant feature structure) of the *active pattern* in memory.

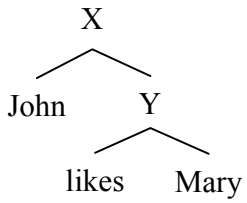
Many other points should be discussed in order to make this picture complete (e.g. a precise definition of the domain of the *movement* operation in terms of locality, but see chapter 2 on this point); for the time being, this is enough to get a glimpse of the cognitive insight (cf. Grossberg's *Adaptive Resonance Theory*, Grossberg 1976, Carpenter and Grossberg 1998) that justifies the formal analysis of *movement* that will be provided in chapter 3.

[39] Note that this is not a neutral assumption: the necessity to specify an order the elements should respect to enter the processing will be discussed in the next chapters.

1.3 RELEVANT RELATIONS AMONG ELEMENTS: INTRODUCTION TO DERIVATIONS

Thinking in terms of active patterns, it is necessary to explore which configurations among elements are required in order to create felicitous contexts (for instance, for movement to happen). It is trivial to note that the relevant relationships we can define among linguistic objects, in any structural description, are in fact very few with respect to the theoretical possibilities (Epstein and al. 1998); given a simplified structural description of a sentence like the one below, we can provide plenty of relations among elements that are completely irrelevant in terms of generalizations (e.g. John is *two-nodes-far* from Mary):

(41)



Among the significant ones, two are usually assumed to be necessary (and sufficient) in order to describe *tree-structures*: *precedence* and *dominance*[40].

Precedence is a *total* (that is, defined on all the elements of a given set) *strict* (*asymmetric, irreflexive* and *transitive*) *order* defined on the terminal nodes (leaves) of the tree:

{"John" *precedes* "likes", "likes" *precedes* "Mary", "John" *precedes* "Mary"}
or in a more compact way: <John, likes, Mary>;

Dominance is a *partial* (order even though "locally total", Kayne 1994), *asymmetric*, *transitive* and *reflexive* (if we assume that any node *dominates* itself) defined on the whole set of nodes of the tree:

---

[40] But see Frank and Vijay-Shanker 1999 for different primitives.

{"X" *dominates* "John", "X" *dominates* "Y", "X" *dominates* "likes", "X" *dominates* "Mary", "Y" *dominates* "likes", "Y" *dominates* "Mary", any-node *dominates* itself... }

otherwise: {X $\prec$ John, X $\prec$ Y, Y $\prec$ likes, Y $\prec$ Mary }.

Another important (*derived*, Gorrell 1995) relation involved in many descriptions of crucial linguistic phenomena is *C(onstituent)-Command*. This relation has been employed to define *government relations*, *binding conditions*, *scope constraints* (see next chapter for more details on these properties) and proposed in many flavors:

(42) a. **Reinhart 1979**

A *C-commands* B iff:

i. The first *branching node dominating* A *dominates* B,

ii. A does not *dominate* B,

iii. A $\neq$ B.

b. **Chomsky 1986b**

A *C-commands* B iff A does not *dominate* B and every C that *dominates* A *dominates* B.

c. **Kayne 1994**

A *C-commands* B iff A and B are *categories* and A *excludes* B and every category that *dominates* A *dominates* B.

d. **Chomsky 2000**

A *C-commands* B iff B is *contained in the sister* of A.

e. **Epstein and al. 1998**

A *derivationally C-commands* B when:

i. A and B are directly *paired/concatenated* by *merge* or by *move*,

ii. in this configuration even A and B *C-command* each other, and

iii. A *asymmetrically C-commands* all and only the terms of the category B with which A was paired/concatenated by *merge* or by *move* in the *course of the derivation*.

The non primitive nature of these relations seems quite evident: Reinhart uses the notion of *first branching node* and *dominance*, Chomsky refers either to *dominance* or to *sisterhood*, Kayne incorporates notions such as *exclusion* and *dominance*, while

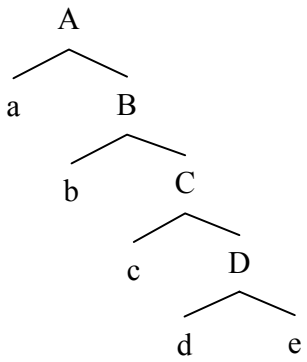
Epstein and al. employ the concepts of *merge*, *move* and the idea of *derivation*. Even though these definitions are not equivalent, they all capture the very same set of essential phenomena (*binding*, *scope asymmetries, movement and control* etc.).
It is however important to draw a clear distinction at least between (42.a-d) and (42.e) since the first four definitions give a view of the *C-command* relation that is *representational*, while the last one can be defined *derivational*. This opposition, sometimes considered simply ephemeral, has indeed crucial implications for a computational model as it will be explained in §2.2. For the time being, we should note that the main distinction lies on the assumption that *representational* relations can be defined among any node of the tree, simply depending on the overall geometry of the structure, while *derivational* relations are tightly related to the time the linguistic objects enter the computation. From a purely theoretical point of view, the derivational option is (potentially) more restrictive even thought the description of (*derivational*) *C-command* provided in (42.e) does not take advantage of this opportunity, since it can access any object introduced in the computation up to the moment this relation is established. This "transparency" of the whole structure makes quite useless the derivational machinery. I would suggest that this problem is partially related to the bottom-up perspective the derivation is assumed to follow and partially to the absence of *phases* (Chomsky 1999 or *barriers*, Chomsky 1986b).

Another important fact we should examine is that all the definitions of *C-command* associate the domain of the relation to the success of an immediate *constituency* formation: the *first branching node* is the immediate result of the concatenation operation, while *sisterhood* is the relation between the constituents that are concatenated; finally *merge/move* are directly concatenating operations. Going back to the first relations described in this paragraph, note that a narrower version of the *dominance* relation (e.g. *immediate dominance*, that is, dominance without transitivity) would exactly entails the relation among constituents and their mother. This is maybe not an accident.

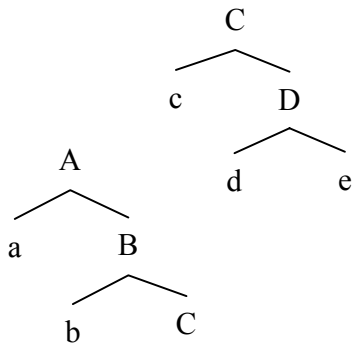
1.4 COMPLEXITY THEORY

In the last paragraph (§1.3) I pointed out that having a *transparent representation* of the phrase structure leads to the theoretical possibility of having a bigger number of possible relations among elements with respect to a *derivational* approach that postulates "opaque" structural descriptions. This can be explained with a simple example comparing two structural representations of the same sentence <a, b, c, d, e>:

(43) a.



b.



The difference between these representations can be accounted for in terms of number of *C-command* relations (assume, for instance, the definition given in (42.a)) we can define among elements: in (43.a) we can define eight relations taking *a* as argument (a *c-commands* B, B *c-commands* a, a *c-commands* b, a *c-commands* C, a *c-commands* c, a *c-commands* D, a *c-commands* d, a *c-commands* e) while in (43.b), only the first four relations (a *c-commands* B, B *c-commands* a, a *c-commands* b, a *c-commands* C) are directly derivable. In which sense this is empirically an interesting result will be evaluated in the next chapter (§2.2.2, §2.3.3 and §3.4), it is however important to note that this "impenetrability condition" has been obtained postulating a tree-splitting. This operation directly follows from the hypothesis that once processed, part of a tree is inaccessible to further elaborations (then also unavailable for the definition of new relations).

Which tree has to be processed before is by now an irrelevant issue, the important thing is that having fewer relations to be evaluated reduces the complexity of problem.

Even thought this fact seems pretty intuitive, the definition of *complexity of a problem* is indeed a precise, formal notion that requires an accurate definition: following Papadimitiou (1994), the *complexity of a problem* can be defined as a function of the resources, essentially *time* and *space*, needed to solve the problem, where *time* represents the *number of steps* needed to reach the solution and *space* is the *memory* required to store/retrieve the information to be elaborated.

We can say that the *complexity of a problem* is proportional to the *size* of this problem, which is determined essentially by three factors:

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

In order to predict the *tractability* of a problem (namely to guess whether or not this problem has a solution and how much it will take to discover it), we are interest in the *growing rate* of the complexity function, that is, roughly, how many more steps we shall make in order to find the solution, any time we add an extra item to the input. Without going into much detail, a problem with a complexity *linear function*[41] would be tractable since any extra item in input would require (at worst) an extra step to solve the problem; also *polynomial functions* are tractable[42]. Problems arise when the order of the function is *exponential* or *factorial* ($O(n^n)$, $O(n!)$): the growing rate of these functions is so fast that there are no possibilities to find a secure solution in any reasonable time[43]. In short, generally, we should avoid *exponential* or *factorial* complexity functions since fingerprints of *intractable* problems.

Within this context, it is however appropriate asking why we should calculate the complexity function in terms of input length when, in fact, a derivational perspective would allow us chunking this input in finite, manageable pieces. Actually, while for the

[41] e.g. $f = cn$, where c is a constant and n the length of the input would have a linear order of complexity, namely $O(n)$.
[42] e.g. $f = cn^2$, would have a complexity order = $O(n^2)$; then, assume c=1, with 5 items in input we should make at worst 25 steps; with 6 items, 36 steps and so on.
[43] See Barton and al. 1987 for more details.

moment there is little to say about the third point (the *algorithm* to be used will be presented in chapter 3), both the *length of the input* and the *space of the problem* are drastically reduced by the derivational trick presented in (43.b); this is essentially due to the combinatorial progression that regulates the growth of the number of relations definable on a given input set: the number of relations, such as *c-command*, has a growth that is polynomial with respect to the introduction of a new terminal item in a (*binary branching*) tree structure; given *n* terminals, in fact, the number of *c-command* relations is exactly n^2-n (e.g. with 3 terminals we can define 6 relations, with 4 terminals 12 relations, with 5 terminals 20 and so on). Then the tree-splitting proposed in (43.b) does not simply have the effect of reducing the number of relation involving the terminal *a*, but the overall number of relations from 20 (since we have 5 terminals) to 12 (two times 6 relations, that are the number of relations defined on a tree of 3 terminals). This is not an impressive result, since, as we mentioned before, problems with polynomial order of complexity would have been tractable, but it is however an interesting reduction. As it will be clear later on, better results will be obtained formalizing the idea of *cyclic movement* and the notion of *phase* (Chomsky 1999-2000) with respect to specific problems (i.e. *ambiguity* and *long distance dependencies*, cf. §3.4).

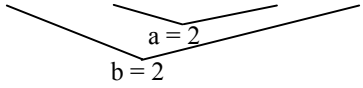
As Chomsky points out (Chomsky 2000:111), the fact that this computational complexity calculus should matter for a cognitive system is essentially an empirical matter. To my opinion, this would imply having a formal complexity measure of the linguistic processing that we could compare with genuine human processing data. Unfortunately, a complexity function expressed in terms of input length obfuscates an essential property of the linguistic processing system, namely its independence from the bare number of tokens to be processed, as expressed by the following contrast:

(44) a. #The reporter who the senator who John met attacked disliked the editor.
b. The editor was disliked by the reporter who was attacked by the senator who John met.

(44.b) is longer then (44.a) but "easier" to parse. Intuitively, in order to be "empirically adequate", a *complexity theory* should predict that what is "easier" for humans should be "easier" for the algorithm too, then the "simplicity" of a sentence should be productively expressible by an adequate complexity function.

Gibson (1998), among others, suggests that a "cost function" in *parsing* should be sensitive to the input length in a relative way: namely new words should have a cost of *structural integration* based on the actual distance, calculated in terms of some relevant elements (new *discourse referents*, in Gibson's theory), that intervene between the word and its integration point (namely where it is re-*merged*):

(44) a. # ... the senator who$_i$ John met t_i attacked ...



b. ... attacked by the senator who$_i$ John met t_i ...

In (44.a) both the "paths" *a* and *b* have a cost equals to 2 (because *John* and *met*, that, within Gibson's theory, are new *discourse referents*, are crossed), while in (44.b) only *a* has a cost. This should be enough to understand what causes the contrast.

In addition, Gibson suggests that another part of the function cost is determined by what he dubbed the *storage cost*, that is, every syntactic head, predicted at some point as the *minimum requirement* to complete in a grammatical way the sentence, has a cost:

(44) a. # ... the senator who John met attacked ...

2 1 3 2 1 1

b. ... attacked by the senator who John met ...

1 2 1 0 2 1 0

I think there are many reasons to believe that this is an insufficient solution for a realistic *cost function* (mainly in terms of *explanatory adequacy*), but there could be some truth in the way it decomposes the complexity function in two relevant

components (the *structural integration cost* and the *storage cost*) getting rid, in an important way, of the notion of *input length*.
For instance, recalling the discussion in §1.2, these two components could represent the cost of keeping an *active pattern* (that is, when an element that has to be *re-merged* somewhere should be *stored*, in the meanwhile, *in a memory buffer*; cf. *storage cost*) plus the cost of introducing new *top-down expectations* once processed a lexical head with *CSel* requirements, or a *functional element* before its head (that is a way to explain the *structural integration cost*).

Note that, theoretically, an interesting configuration could be obtained cyclically: namely, when all the *expectations* in the domain of a lexical head are satisfied; this should be the case of the thematic role saturation of the *verb* (and *noun*). I would like to suggest that these special moments of equilibrium correspond to *phase* boundaries in Chomsky's terms (Chomsky 1999, cf. §2.2.2, §3.4.3).

CHAPTER 2
LINGUISTIC AND COMPUTATIONAL MODELS

Formalizing a linguistic theory forces us to specify anything we (minimally) need in order to describe a language. This effort underlines an extremely fuzzy border between what has been usually considered *competence* and what has been dubbed *performance*; even the classical distinction (Chomsky 1965) between *grammaticality* and *acceptability* seems less neat than we were used to think (§2.1).

Moreover, when we choose a specific formalization, we must consider to what extent it encodes linguistic intuitions and at what computational cost it does. In order to evaluate these two aspects both *representational* (*Extended Standard Theory*, §2.2.1 and *Cartographic Approach*, §2.2.4) and *derivational* (*Minimalist Program* §2.2.2 and Phillips' *model* §2.2.3) linguistic theories will be reviewed (§2.2). Eventually their relative computational implementability will be evaluated (§2.3), especially the *principle-based* approach (§2.3.1) and some *minimalist* computational models (a full formalization of Chomsky 1995 due to Stabler 1997, §2.3.2, and an implementation of the derivation in terms of *phases*, *probes* and *goals* sketched in Chomsky 1999-2001, due to Fong 2004, §2.3.3).

2.1 BETWEEN COMPETENCE AND PERFORMANCE: PROCESSING MODELS

It is necessary for both generative and computational linguistics to define precisely the representation of *linguistic knowledge*, trying to take into account as much as possible those properties that make this representation cognitively plausible.

From a formal point of view, we can think of any language as an *infinite set of sentences*, each of them corresponding to at least one *Structural Description* (*SD*) that makes explicit some relevant relations among elements (such as *linear order*, *relative scope*, *thematic/eventive* structure etc.). From this perspective, the *competence* is the *generative* (that is *formal*) *procedure* that allows any native speaker to *recognize* and *produce* the whole infinite set of grammatical sentences which constitute his/her language.

The *competence* is an *intensional* procedure, rather than an *extensional* one: it would be impossible to store in our brain the infinite number of grammatical sentences (with their SDs), but, in fact, we definitely produce/understand completely original and grammatical linguistic expressions. In this sense, a *generative grammar* is a formal description of this intensional procedure (Chomsky 1986a).

The real use any speaker/hearer makes of this knowledge to produce/understand sentences seems to be a different matter: *memory limitation*, *restrictions in computational resources accessibility*, *rapid fading of signals* are only some of what have been usually considered *performance factors*. These "extra-linguistic" factors somehow limit the use of our linguistic knowledge and are responsible, following standard assumptions for instance, for the *low acceptability* (that is different from *ungrammaticality*) of *nested constructions* (45.a) or *self embeddings* (45.b), (Chomsky 1965:10):

(45) a. I called the man who wrote the book that you told me about up
(I called up the man who wrote the book that you told me about)

b. The man who the boy who the students recognized pointed out is a friend of mine

(The students recognized the boy who pointed out the man who is a friend of mine)

Therefore, *acceptability* is a matter of language use (namely *performance*) while *grammaticality* is well-formedness with respect to our grammatical knowledge (that is *competence*).

However, it is clear from this picture that the *competence* is inscrutable if not through *performance* data. An absolute distinction between *competence* and *performance* could be drawn only on the basis of a set of arbitrary stipulations concerning what *performance* factors are about:

(46) *competence + performance factors = linguistic behavior*

In fact, all of the following possibilities are equally plausible and essentially equivalent from a purely formal (that is, *generative*) point of view (c_n is a *competence* factor while p_n a *performance* one):

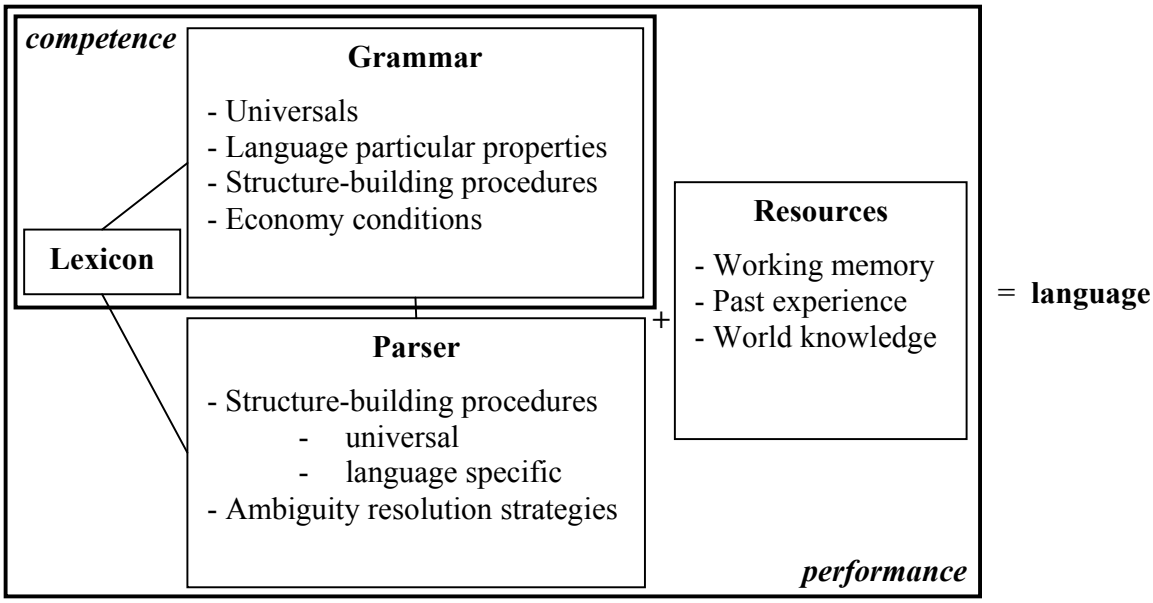
(47) a. *competence* $_{\{c1, c2 \ldots cn\}}$ + *performance factors* $_{\{p1, p2 \ldots pn\}}$ = *linguistic behavior*

b. *competence* $_{\{c2 \ldots cn\}}$ + *performance factors* $_{\{p1, p2 \ldots pn, c1\}}$ = *linguistic behavior*

c. *competence* $_{\{c1, c2 \ldots cn, p1\}}$ + *performance factors* $_{\{p2 \ldots pn\}}$ = *linguistic behavior*

This is pretty straightforward in a modular system (cf. §2.2.1, §2.3.1): for instance, *memory limitation* could be either a *performance factor* or a *competence filter* that imposes extra constraints on the generative power of the grammar in a way that is essentially similar to other *competence* principles such as conditions on *pronominal binding*. It has been usually assumed, at least before the beginning of the minimalist inquiry (§2.2.1), that a model like the one presented in (48) could be fairly representative of the *competence*/*performance* dualism:

(48) standard model (adapted from Phillips 1996:16)



Psycholinguistics is the research field that deals with *performance factors* and proposes plausible models for the "grammatical basis of linguistic *performance*" (Berwick and Weinberg 1986): the idea is to keep into account the whole *linguistic behavior* in order either to provide evidence in favor of generative theories about *competence*, or to disconfirm them, also supplying data in support of more adequate models. Besides native speakers' grammaticality judgments, two major complementary sources of information are *sentence processing* (real-time elaboration of structural descriptions during perception of visual or auditory linguistic inputs) and *language production* (the process involving a conversion from non-linguistic conceptual intentions to linguistic expressions). Assuming that *flexibility* and *realism*[44] are actual properties of our linguistic *competence*, these psycholinguistic investigations should converge toward the very same *competence* model, which would be interfaced with at least two *performance* systems: a *conceptual-intentional* one (which we know very little about) and a *sensory-motor* one.

[44] In the sense explored in the introduction: *flexibility* implies that the same grammatical knowledge should be used both in *parsing* and in *generation*; *realism* requires a faithful reproduction of productive phenomena involved in *comprehension* and *production* accounting for complexity issues.

In the early eighties, a similar problem was couched within the discussion about the relation between *human grammar* and *parser*: how could the parser use the grammar to perform human-like tasks in *sentence processing*? According to Berwick and Weinberg (1986) there are at least three relevant classes of relations between the *parser* and the *grammar*:

a. **token transparency** (Miller and Chomsky 1963) – every rule/principle postulated by the *grammar* is mirrored by a parsing step (null hypothesis);
b. **type transparency** (Bresnan 1978) – the parser essentially mimics the grammatical rules/principles but in a way that only preserves the typological nature of principles/rules (namely the goals are the same even if the means to attain them may vary), implementing the processing functions in a way that is (more) psycholinguistically plausible;
c. **covering grammar** (Berwick and Weinberg 1986) – the relation between *grammar* and *parser* is only expressed in terms of an equivalence relation with respect to the set of linguistic phenomena captured; this can be realized by completely independent procedures/principles/rules from the two different perspectives. This seemed to be the best way to keep separated *descriptive*/*explanatory adequacy* issues (domain of the *grammar*) from efficiency ones (domain of the *parser* implementation).

The third option clearly allows for more freedom and it has been probably the most successfully pursued at the time of *Government and Binding* (GB) approach (to be presented in §2.2.1): computational implementations that used this grammatical framework (for instance *Principle-Based Parsing*, §2.3.1) had to solve many "efficiency" problems due to various underspecified aspects of the grammar, like ordering principles/rules so as to limit the number of ill-formed structures to be examined (Fong 1991) or (partial)pre-compilation of the rules to speed-up the parser (Merlo 1996).

What is common among principles formalization using first order logic, efficient LR tables and human use of the linguistic *competence* could hardly be something more than a covering relation.

This problem slowly faded out during the last decade, essentially because the generative mainstream became more aware of some fundamental limitations of the GB framework, moving toward cognitively more plausible theories of *competence*: the *Minimalist Framework*, for instance, tried to consider some "*performance* issues" such as *interface conditions*, the notion of *spell-out*, the idea of *cyclic derivation* (by *phases*) during the phrase building operations (see §2.2.2). Some of these new devices try to account for problems which were classically considered "*performance* factors", like the cost of *accessing the lexicon* (which leaded to the concept of *numeration*), *global/local economy conditions* (any operation has a "cost" then, for instance, *merge* preempts *move*) or *memory limitation* (the elements are "active" in structure building only during a relevant period of time, the *phase*; after the *phase* is completed, its components are largely inaccessible to further operations).

These modifications make the original distinction between "declarative properties of the syntactic theory" and "procedural notions" (Crocker 1996) not easily deducible from the linguistic theory, and the classical *grammar-parser* opposition could much less easily be discriminated now. Generally, the *performance* side of the linguistic behavior became more penetrable to theoretical incursions.

Thus, with relatively few departures from "orthodox minimalism" (§2.2), the token transparency idea reappears as a real option. One of the clearest pictures from this perspective is Phillips' dissertation (1996). Phillips presents many empirical arguments supporting the assumption that the distinction between *grammar* and *parser* is unnecessary, ending up with the following model (Parser Is Grammar, PIG):

(49) PIG model (Phillips 1996:255)

Grammar		**Resources**	
Universals	+	Working memory	= **language**
Language specific properties		Past experience	
Lexicon		World knowledge	
Structure-building procedures		Attentional state	
Economy conditions			

This clashes with the standard model assumed within more classic frameworks such as (48), where *lexicon*, *grammar* and *parser* have been considered "modules", subject to independent principles/rules.

Although this approach is exactly an instantiation of the problem outlined in (47) (then it could easily turn out to be equivalent to a model that embeds in the same "grammatical box" resources such as *working memory*, *past experience* etc.[45]) what crucially represents a breakdown with the generative linguistic tradition is the assumption that classical *performance* phenomena (like *ambiguities resolutions*, §2.2.3) should be accounted for in terms of *competence factors*, namely they represent clues about how our linguistic knowledge is structured and not just about how it is used.

We should keep in mind this idea when we try to define the "virtual conceptual necessities" (Chomsky 2000:111) that any linguistic theory should postulate in order to be *flexible* and *realistic* (in addition to *explanatory* and *universal*).

In this dissertation, I will eventually argue (Ch. 3, Ch. 4) in favor of the formalization of a specific notion of *working memory* as an essential component of any *long distance relation* (notion of *move*, §3.3.3, §3.3.4), while I will account for *past experience* in terms of *parameters setting* (a fundamental part of any coherent *top-down expectation*, §3.4.3, §4.3.7).

[45] A cursory glance to this option could suggest that this unification has not been seriously attempted yet, just because the "modules" under the "resources" box are extremely difficult to formalize even if intuitively graspable.

2.2 DERIVATIONS OR REPRESENTATIONS? SOME LINGUISTIC MODELS

From a psycholinguistic perspective, it seems fairly accepted that *parsing* and *generating* sentences are incremental processes. Since building phrase structures piecemeal requires a precise sequence of steps, the dynamic nature of these processes justifies the idea of *derivation*, that is, the output of the computation is reached only after a sequence of successive, concatenated operations.

Generative grammars both historically (Chomsky 1957) and more recently (*Minimalist Program*, Chomsky 1993-2001, §2.2.2) adopted the idea of *derivation* as a way to describe sentence building.
Note, however, that this idea of derivation is often unlinked to the processing side (Phillips' model, §2.2.3, is a notable exception): Chomsky's *Minimalist Program* (Chomsky 1995:223,fn3), for instance, does not entail any real temporal sequence in the application of the operations: a *derivation* has to be intended as a sequence of purely formal, then abstract, successive transformations that operate on SDs. Following the considerations presented in the previous paragraphs, this is however an unwanted complication in computational/cognitive terms: processing inferences should be both reasonable and desirable[46].

On the other hand, there are linguistic frameworks (*Government and Binding Approach*, Chomsky 1981-1986b, §2.2.1, *Cartographic Approach*, Cinque 1999, Rizzi 1997 §2.2.4) where the derivation of a SD is completely irrelevant: any order of application of principles/rules would bring to the very same geometrical configuration among elements.

---

[46] Economy considerations and ambiguities resolution strategies has been shown to lead to a cognitively plausible (quasi-)*deterministic* processing system such as the *crash-proof syntax* model (Frampton and Gutmann 2002) where no attempt to build unused SDs is made; This is an interesting hypothesis to be evaluated in computational terms that would allow us to make an efficient use of computational resources (a similar attempt has been already explored in Marcus 1981, among others, but only from a *parsing* perspective).

Roughly speaking, while the first models (*derivational*) entail a sequence of discrete steps such that each step produces an output that is the input of the next step, the second ones (*representational*) only specify requisites of well-formedness, without any specification of what has to happen before and what after (Jackendoff 1997:12).

To better understand these different positions (which sometimes appear to be just dialectical) let us concentrate on a crucial point of discussion: the *chain* Vs. *movement* opposition (of interest for our purposes, since it represents the most classical case of *Long Distance Dependency*).

From a *derivational perspective*, *movement* really implies the displacement of an element from the base position (structurally the lowest position where the element occurs) higher up in the phrase structure[47], leaving behind a trace that is assumed to be a perfect copy, even though phonologically null, of the *moved* object (*copy theory of movement*, Chomsky 1995-2001). Following the minimalist trend (§2.2.2), any *movement* is triggered by feature requirements that have to be satisfied by successive cyclic operations targeting only the closest positions where these requirements can be satisfied (*shortest move*, Chomsky 1995).

Watching the very same phenomenon from a *representational perspective*, the *moved* element *x* and its traces *t* form a *chain* <x, tx_n, tx_{n-1} ... tx_0> for which specific conditions hold:

- *x C-commands* any *tx* in the chain and any tx_i *C-command* any tx_j such that $j<i$
- between *x* and tx_0 there are no intervening element *y* such that *x* and *y* belong to the same *class* of elements.

These two positions have been considered as incompatible options[48], illusory alternatives[49], or equivalent devices[50].

---

[47] Any *movement* is assumed to be toward *c-commanding* positions, but see Richards' (2004) analysis of *wh- movement* in Bulgarian or the classical analysis of Right Node Raising (Postal 1974).
[48] Epstein and al. 1998.

Instead of taking sides on this discussion, I think it is worth trying to go deeper in this distinction, defining some theoretical differences that might be useful not just to classify approaches, but, more important, to understand their essential properties and to evaluate seriously what can be *represented* and what should be *derived*.

(50) some differences between *representational* and *derivational* frameworks:

i. **completeness of the Structural Descriptions (SDs)**

- *representational* approaches keep unified and static the *SDs* of the linguistic objects at any level of representation (if multiple levels are considered as in standard *transformational grammars*); the whole computational process results in a set of *complete multiple SDs* expressing the whole structure of the sentence from different perspectives;

- *derivational* approaches transform *SDs* at every application of any principle/rule; the result of the computation is a *single representation*, if any, that is the history of the whole derivation. Otherwise during the computation any *SD* reflects only a *partial Structural Description* of the whole sentence.

ii. **ordering principle/rules**

- *representational* approaches make no explicit assumptions about the application of principles/rules. All combinations should result in the same SD.

- *derivational* approaches postulate an order (sometimes interpreted as the emergent property of a *cost function*, Collins 1997) in any operation that results in a grammatical (that is legal) transformation of the SD.

iii. **nature of the relation among elements**

- *representational* approaches define the relation among elements in a *static* way, using properties of complete SDs (e.g. *C-command* can be defined between any elements A and B present in the SDs at some level; this relation will be always valid within the same SD, e.g. elements in a chain);

- *derivational* approaches determine the relation among elements in a *dynamic*

[49] Pure derivational systems cannot exist, they need representations to work, then they should be defined at best *weakly derivational*, Brody 2002.

[50] For a formal discussion on the equivalence between *derivations* and *representation*, see Michaelis 1998.

way, crucially related to the time they enter the computation (e.g. A enters a relevant relation with B at the time τ); then the object (status) changes in the course of the derivation (e.g. the function *f* takes two object A and B and replaces them with C) so that the previous relation becomes inaccessible to the next steps.

iv. **nature of the constraints**

- *representational* approaches should postulate *filters*, namely procedures that select only grammatical SDs, once a set of correct SDs is produced by the indiscriminate application of principles/rules;

- *derivational* approaches can *constrain* the derivation, preventing it from producing ungrammatical outputs at any application of any principle/rule (e.g. by using *economy conditions*).

v. **processing implications**

- *representational* approaches makes no explicit assumption about how linguistic *competence* is put to use (e.g. they stay completely agnostic with respect to the *flexibility* and to the *realism* issue presented in the introduction);

- *derivational* approaches should directly entail processing *expectations* (either purely formal, Chomky 1995, or psychologically plausible, Phillips 1996), given that they make crucial assumptions about the time the elements enter the computation (so *flexibility* and *realism* could be at issue).

This distinction is probably too strong to be used for categorizing most of the main generative frameworks, but it has the advantage of making clear some crucial differences between the two hypothetical theories of *movement* as explained below:

	representational	**derivational**

i. completeness of SDs	*unique* and *complete*: all instances present in the sentence appear in the *chain*:
	
	<x, tx_0, tx_1 ... tx_{n-1}, tx_n>
		partial: only the relevant element (a segment, at best) of the "chain" is accessed at any step:
		
		step 1: x;
		step 2: tx_0 (x = tx_0);
		...
		step n: tx_n (tx_{n-1} = tx_n);

ii. principle/rules ordering	*irrelevant*: any order would postulate the same traces and discard ungrammatical options	*strictly defined*: unless we define extra backtracking options, postulating a wrong *movement* would prevent the derivation from retrieving correct SDs
iii. relation among elements	*absolute*: any relational property among elements in the chain is valid within a single SD	*relative*: any relational property is valid only within a relevant lapse of time τ_n (at τ_n: <tx_n, tx_{n-1}>), then further operation (valid at τ_{n+1}) would not have access anymore to the single constituents that established this relation
iv. nature of the constraints	*filters* on the unique resulting representation (e.g. *case filter*)	*constraints* on operation application (such as *shortest move*)
v. processing implications	*none*	*rigid* order predicted (potentially, this could have direct implications for processing)

As it will become clear in §3, these properties really make a difference in terms of computational complexity and the derivational perspective will be more suitable as grammar model.

Brody (2002), for example, points out that any derivational theory is at best *weakly representational*: he observes that in order to *move* an object from a constituent that contains it, this constituent has to be *transparent* for later operations (namely the representation uses some sort of SD as is was supposed to be the case only in representational approaches, cf. (50.iii)).
This argument, clearly pertinent from a general point of view, is however irrelevant under the derivational perspective in (50.iii), since constituents are not accessible anymore after they successfully built a phrase[51].

In the next paragraphs I will explore two paradigmatic examples of what we usually consider *representational* and *derivational* frameworks: the Extended Standard Theory (EST model, Chomsky 1973-1986a) and the Minimalist Framework (Chomsky 1993-2001). Then an even more radically derivational version of the minimalist model will be presented following Phillips' idea (§2.2.3). Finally, in §2.2.4 I will present a second (mainly representational) framework, the *Cartographic Approach*, useful to highlight an important complementary problem for any theory of *movement*: the definition of *locality constraints*.

[51] The solution proposed to account for *movement* without any necessity of inspecting constituents in a previously built phrase will be given in §3 and it resides on the directionality of the operation (*top-down*, *from left to right*, as I will assume in the next chapter, Vs. *bottom-to-top* as Chomsky and Brody do).

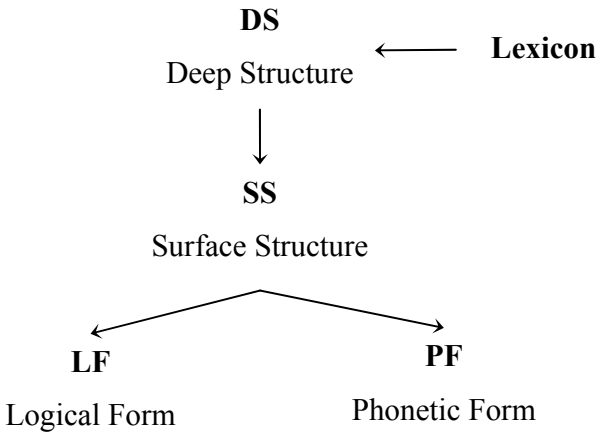
2.2.1 EXTENDED STANDARD THEORY (EST) AND GOVERNMENT AND BINDING (GB)

The *Extended Standard Theory* (EST, Chomsky 1973, 1981-1986a) is a theory of grammar pretty well representative of the representational class. It postulates five *levels of representation*: a *Lexicon*, a *Deep-Structure* (DS), a *Surface-Structure* (SS), a *Phonetic Form* (PF) and a *Logical Form* (LF). These are *levels of representation* in the following sense:

i. they are defined as *sets of symbols*;
ii. they describe their objects in terms of static *Structural Descriptions (SDs)*;
iii. they require *mapping algorithms* to translate one SD into another.

The relation among these levels can be described as follow:

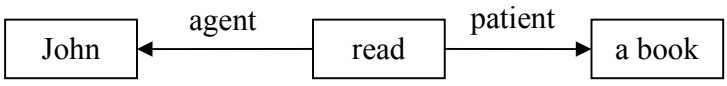
(51) Level structure in Extended Standard Theory (EST, Chomsky 1973-86)



The rationale behind such a multi-level representation was mainly empirical:

DS was intended to isolate the domain where thematic relations took place and where lexical items where inserted in the process:

(52) DS



SS was the locus of most of the *movement* operations and where case was assigned:

(53) DS: did John read what
SS: What$_i$ did John read t_i?

LF was where *covert movements*, relevant for interpretational purpose, took place:

(54) SS: Every boy likes a book
DS_1 (*surface scope*, "there is a different book for every boy"):
$\forall$ boy, $\exists$ book : likes(book, boy)
DS_2 (*inverse scope*, existential quantifier raised, "there is a single book for every boy"):
$\exists$ book, $\forall$ boy : likes(book, boy)

PF accounted for phonological phenomena like head incorporation or cliticization.

The *EST model* has been historically tied to the *Government and Binding* (*GB*) approach (Chomsky 1981-86). This framework tried to get rid of the numerous and complex rewriting rules that pervaded *transformational grammars* at that time, replacing them by a small set of universal principles. An adequate interaction among these principles plus a bunch of parameterized options should have been sufficient to account for many complex phenomena in a cross-linguistic perspective (*Principle and Parameters*, Chomsky 1981) that would have required hundreds of complex rules to be captured. For instance, before GB, *passivization* (55.b), *focalization* (55.c) and *binding* (55.d) had to be captured each by a specific rule that could interact in a hardly predictable way with other rules (55.e) nonetheless leaving many data unexplained (55.f). This quickly leaded to an extremely complex and language-specific grammar design. Then the idea was to postulate a compact set of general principles like *θ theory*, *move α*, *Case filter* and *Binding theory*, reported in (56), which capture the very same set of phenomena without any reference to special rules or to language specific properties other than a *lexicon* and a bunch of *parameters*.

(55) a. John called Mary

b. Mary was called by John (*passivization*)

NP_1 V+*tense* NP_2 → NP_2 be+*tense* V+*past_participle* (by NP_1)

c. MARY John called (*focalization*)

NP_1 V NP_2 → NP_2 NP_1 V

d. $John_i$ called $him_{*i/j}$ (*binding*)

NP V pro → NP_1 V pro_2

e. *JOHN Mary was called by (applying *passivization* + *focalization* to b.)

f. $He_?$ was called by $John_?$ (unpredictable by this set of rules)

(56) **θ theory**

i. every argument receives one and only one thematic role

ii. every thematic role is assigned to one and only one argument

Move α

a category α can be *moved* anytime anywhere

Free indexation

indices are freely assigned to categories in A(rgumental) position

Binding theory

condition A - An *anaphor* (e.g *himself*) is *bound* in its *binding domain*[52]

condition B - A *pronominal* (e.g *him*) is free in its *binding domain*

condition C - A *referential expression* (e.g *John*) is free

Case filter

any overt NP argument has to be case marked or associated with a case position

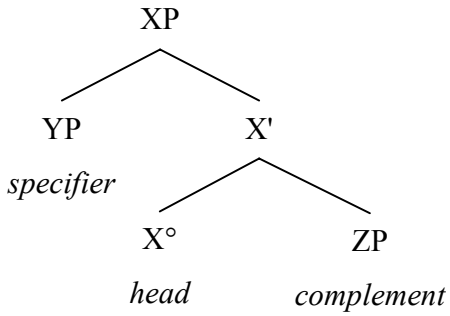
An interesting property we should notice is the highly *modular* design of this framework: every principle is, theoretically, independent from any other; their interaction is an emergent property.

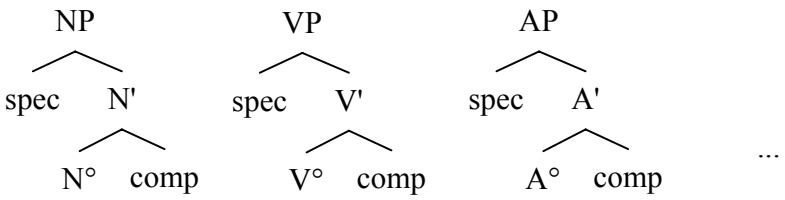
[52] The *binding domain* would be defined as the Minimal Governing Category (Chomsky 1981), namely the minimal phrasal projection containing the element, its governor and a subject accessible to this element. In order to approximate this complex definition, we could paraphrase "*binding domain*" as the "portion of the tree C-Commanded by a potential binder".

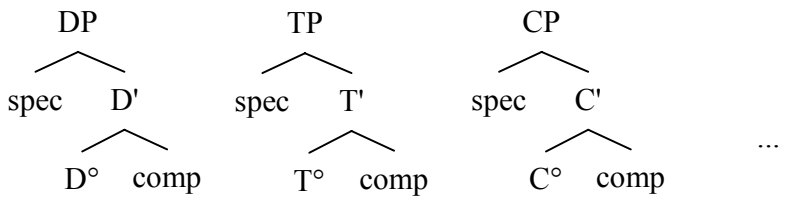
Taking a closer look at the principles structure, we should observe that while some of them *generate* more structures than what they get as input (for instance *Move α* and *Free indexation*), some other principles constrains this behavior by *filtering* out unwanted solutions (the *case filter*, for instance, rules out any structure where even a single NP is not marked for case).

An important *filter* of this kind, introduced in §1.1.5 and widely used within the GB framework, is the *X-bar theory* (Chomsky 1970), which captures a productive generalization about the internal structure of any syntactic constituent, namely that all *lexical categories* seem to project the very same structural skeleton once introduced in the phrase structure, as depicted in (57.a):

(57) a.



b. 

c. 

This seemed to be a tenable hypothesis not only for lexical elements (Nouns, Verbs, Adjectives (57.b)), but also for functional ones (Determiners, Tenses, Complementizers (57.c), cf. Chomsky 1986a).
X-bar structures definitely played a crucial role in identifying empirically productive relations expressing *cross-categorial* similarities (as we mentioned in §1.1.5): the *head-complement* relation was the locus of the phrase selection, while *spec-head* relation is the structural environment for agreement; moreover the head-complement order could be parameterized, accounting for cross-linguistic variation (*head-final* languages, like Japanese Vs. *head-initial* ones, like English).
From a computational point of view, the *X-bar* schema also limited the generative power of the grammar, *filtering* out incompatible structures.

Summarizing, the following properties of the GB framework are then directly relevant for the present discussion:

1. the grammatical *competence* is described by a small number of *universal principles* plus some *parameterized options* that map *lexical elements* through SDs at distinct *levels of representation*;
2. *principles* both *generate* and *filter* SDs;
3. the order of application of *generators* and *filters* is irrelevant for the final SD.

Despite its power in terms of *descriptive adequacy*, this framework revealed critical flaws during the last two decade. For instance, the theoretical desideratum of identifying at each level of representation characteristic operations not allowed at other levels quickly turned out to be unrealistic: *lexical insertion* could happen both at DS and at SS (as *tough-movement* shows (58), Chomsky 1981, 1993); reconstruction possibilities showed that the domain of *binding* cannot be limited only to SS: since in (59.a) the anaphor *himself* should be *C-commanded* by *Bill* to be coindexed with it, a possible solution, given the SS in (59.a), could be that *binding* takes place at DS, before the *wh-phrase* is *moved*, but this would leave unaccounted for the *Principle B* violation in (59.b).

(58) *John$_i$ is easy [$_{CP}$ PRO to please e$_i$]

(59) a. [which picture of *himself$_i$*] did *Bill$_i$* see *t*?
b. [which picture of *him$_{*i/j}$*] did *Bill$_i$* see *t*?

Moreover, two other critical problems are related to the free application of the principles: first of all, many emergent properties of this wild interaction turned out to be unpredictable and sometimes unwanted (cf. §2.3.1); second, *generators* like *Move α* or *Free indexation* appear to be computationally very expensive: if *filters* do not constrain as soon as possible *generators* output, SDs grow in number following easily a factorial progression, which is extremely hard to be computed real-time with finite resources as we have seen in §1.4.

2.2.2 THE MINIMALIST PROGRAM

The *Minimalist Program* (Chomsky 1993-2001) seriously tackles some of these problems, keeping the ideas of the Principles and Parameters approach. I wish to emphasize three aspects of this approach that are relevant for the computational model described in these pages:

1. the notion of *perfection*;
2. the simplification of the core computational system;
3. the *derivational* approach to the Structural Descriptions and the use of *economy* heuristics.

As explained in the previous paragraph, the proliferation of highly technical (often ad-hoc) solutions to limit the unwanted generalizations due to unconstrained principles application created an unmanageable tension between *descriptive* and *explanatory adequacy*. The solution proposed by Chomsky was then to remove unnecessary machinery, limiting the theoretical assumptions to the bare essential necessities. Following this logic, the null hypothesis was to consider the Faculty of Language (FL)

as a *perfect* organ[53], that is the best solution for bridging sounds and meanings (the two *performance* systems that FL necessarily interfaces with). *Perfection*, in this sense, essentially means avoiding any extra symbol in representation and extra step in derivation, beyond the minimal conceptual necessities.

The strongest version of this idea takes the shape of the *Strong Minimalist Thesis*:

(60) **Strong Minimalist Thesis** (Chomsky 2001:3, SMT)
FL should be defined only in terms of Interface Conditions and general conditions of computational efficiency

For instance, the distinction between *Surface* and *Deep Structure* has been discarded (because of its empirical inadequacy showed, for instance, by (58) and (59)) wheras the role of LF and PF was "reduced" from *levels of representation* to *interface conditions* (or *legibility conditions*, Chomsky 2000:94), namely precise requirements the linguistic representations have to conform to before being shipped out to the *performance systems*. Minimally, these *performance* systems are two: a *conceptual-intentional system* (reminiscent of the LF level) and a *sensorimotor system* (roughly speaking, the old PF). They are assumed to be *external* to FL (even if they actually impose constraints to FL objects) in a very important way, namely they could respect different principles/organization (Chomsky, for instance, assumes the *binding theory* presented in (56), namely the "module" of the grammar responsible for the interpretation of *pronouns*, *anaphoric* and *referential expression*, to be part of the conceptual-intentional system rather than of Narrow Syntax). Thus FL would produce expression of the form:

(61) $Exp = \{\pi, \sigma\}$

where π is a finite set of phonological[54] features arranged (within a SD) in a way that is interpretable for PF, and σ is a finite set of semantic features arranged so as to be interpretable for LF.

---

[53] "FL can be regarded as a *language organ*, in the informal sense in which the visual system, the immune system, and the circulatory system are commonly described as organs of the body: not objects that can be removed leaving the rest intact, but subsystems of a more complex structure that we hope to understand by investigating parts that have distinctive characteristics, and their interaction". Chomsky 2000:90.

[54] In the sense proposed in Chomsky 2001:5 fn.14

Even *X-bar structure*, as presented in (57), has been revisited under the light of some border-line phenomena, for example *clitic movement*: in their thematic position they behave as full XPs, but since they incorporate to verbal heads, they should be heads (Chomsky 1995:249). Another problem is related to asymmetries such as *agreement assignmen*t: *case* seems to be assigned to the subject by the verb, but it is the subject that determines agreement on the verb, not vice-versa; this asymmetric relation cannot be predicted simply in terms of *spec-head relation* (Chomsky 1995:258). Then *categories* within the *Minimalist Program*, are considered *elementary constructions* (*bare phrase structures*, Chomsky 2000:249), namely direct *projections* of the properties (*features*) of the *lexical items*, without any *bar-level* or lexical item/X°/X'/XP distinction. This would make it possible to avoid the introduction of extra symbols in the computation, meeting what has been called the *inclusiveness condition* (mentioned in §1.1.2 and here reported):

(62) **Inclusiveness condition** (Chomsky 1995:228):
any structure formed by the computation is constituted of elements already present in the lexical items selected for N[umeration]

As we have seen in §1.1.2, *numeration* refers to the operation of one-time selection of items from the lexicon that will then be available for the computation. Even though slightly revisited in the last decade, the main idea is still the same: accessing the lexicon has a *computational cost*; this cost would be too high if every time a new item should be picked up during the process; because of that, a fair assumption is to limit the access to the lexicon to only once, namely at the beginning of every relevant phase of processing[55]. Accepting (62), nothing but features from the lexicon will become part of the SD at the interface levels. Essentially these features should be either *semantic* or *phonological*; there is evidence, however, that there are more *abstract features*, which are interpretable neither at PF nor at LF (such as *case* on DPs, *agreement* ϕ-features[56] on verbs etc.). These features are dubbed *formal* (in some version of the *Minimalist*

[55] The term *phase* in this context is used in an informal way. See (68) for details on a more formal notion of *phase*
[56] Person and number.

Program even *uninterpretable*) and they must be removed by the computation before reaching the interfaces. This distinction is useful to understand the new conception of *movement*, since, following (62), GB *traces* have no room within this framework (*traces* are not present in the lexicon). Then a *trace* has to be redefined as a (perfect) copy of the *moved* element without its phonological features (Chomsky 2001:9):

(63) **Copy theory of movement**
traces are perfect copies of the moved element, with no phonological features.

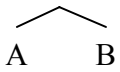
Moving toward an analysis of the (simplified) computational machinery, we should note that the *Minimalist program* tends to reduce the whole transformational apparatus to a single, simple and universal operation that builds structures recursively, also known as *merge* (cf. §1.2.1):

(64) **Merge** (adapted from Chomsky 2001:6)
is a no cost operation which takes two elements A and B (already constructed) and creates a new one consisting of the two: {A, B}

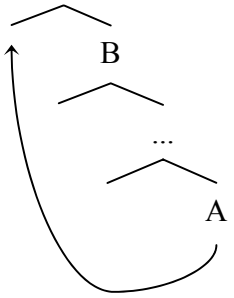
As we mentioned in §1.2.1, early minimalist versions suggested that the form of the new object built by *merge* should have been something like γ{A, B}, where γ is the *label* of the newly formed element. Given that the *label* should be easily predictable from the constituents that *merge*, either it is the product of a general rule (e.g. the head of the constituent is the label) or it is superfluous (Collins 2001); having something else would result in a violation of the Inclusiveness Condition.

This definition of *merge* is general enough to capture both simple combinations of elements that appear to be adjacent (64') and even displacement cases (64"); in fact, many scholars pointed out that the notion of *move* presents many similarities with that of *merge* (see Kitahara 1994 and Starke 2001). Chomsky's recent papers incorporate this insight, keeping, however, a distinction that minimally discerns these two relations (Chomsky 2001:8):

(64') **External merge**
A and B are separated objects



(64") **Internal merge** (old *move*)
A is a member of B



Chomsky (2001:9) notices that argument insertion (*base position*) is only fed by *external merge*, while "all the rest" (*derived positions*) can be satisfied by means of *internal merge* (e.g. EPP/OCC features[57], *left peripheral criteria*, see §2.2.4).

Before Chomsky 2000, it was usually assumed that *movement* is a "last resort" to eliminate uninterpretable features and thus to meet interface conditions: an element A (for instance a subject) had to rise to *merge* with B (an inflected *Tense* head) to check B *uninterpretable features* (*uninterpretable ϕ-features* on the verb). Later work inverts this perspective, looking at displacement as directly triggered by the element that bears uninterpretable features (the *probe*) as soon as it enters the computation: the dynamics of *displacement* is triggered by this *probe* which, because of some uninterpretable feature, searches in its domain a *goal*, namely an element with a complete set of features that can check its uninterpretable ones.

The relation that holds between the *probe* and the *goal* is an *agree* relation: these elements should match in features "under non-distinctness", that is, either the features of A and B have the same values or any unvalued feature F of A is interpreted as having the value of the same feature F of B. After a full checking operation, the *moved* element seems to be blocked in its landing site. To capture the freezing effect, Chomsky assumes that both the *probe* and the *goal* are active as far as they do not check their features

[57] EPP stands for *Extended Projection Principle*, the requirement expressed by any clause of having a subject. OCC(urrence) simply marks the requirement of a head to enter in a *merge* relation with some OCC of α, as in (64"), by *internal merge*, or by *expletive insertion*.

against each other. Then, a full check of all the uninterpretable features *inactivates* both the *probe* and the *goal*, as shown by the impossibility of checking multiple *case* features as in (65).

(65) a. *John*$_i$ said [$_{TP}$ *he*$_i$ [$_{VP}$ t_i went to bed very early]]

b **John* said [$_{TP}$ t_i [$_{VP}$ t_i went to bed very early]]

Apparently, *agree* can be a long distance relation (Chomsky 1999:12), since even a distant *goal* (*people*/*a man*) can satisfy a *probe* (an inflected Tense, *are*/*is*) *merged* to satisfy its OCC feature with a ϕ-incomplete expletive (*there*):

(66) a. *People*$_i$ *are* likely to be t_i in the room

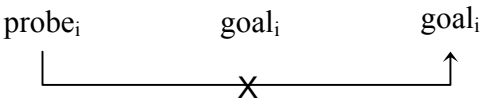
b. *There are* likely to be *people* in the room

c. *There is* likely to be *a man* in the room

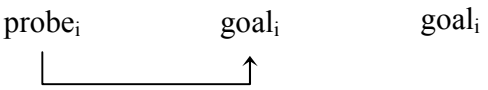
d. **There is* likely to be *people* in the room

The relation that holds between the *probe* and the *goal* is however *local* in the sense that no potential *goal* can intervene without checking the uninterpretable features of the *probe* as shown below:

(67) a.



b.



a'. * *There are* likely to be *a man* in the room with *the guests*

b'. *There is* likely to be *a man* in the room with *the guests*

The third (and last) minimalist assumption that I wish to discuss is the nature of the *derivation*. This point is relevant to the discussion begun in §2.2, because:

1. the resulting SD that has to meet Interface Conditions is "complete" (in the sense given in (50.i)) only if intended as the history of the whole derivation, otherwise it is built piecemeal from bottom (innermost VP shell) to top (most peripheral CP);

2. the operations apply *cyclically* and are ordered on the basis of the *bottom-to-top* structure of the process: in order to build phrases we start from the heads, first merging their complements, then their specifier(s) and so on; any *merge* takes place at specific time, so that it makes sense to think in terms of order of operations (e.g. thematic requirements are satisfied before OCCs needs);
3. any relation among elements is ephemeral, in the sense that it is present only during a relevant "phase" (a technical notion of *phase* will be provided in (68)), then later stages of structure building can not have access to the elements within this "phase";
4. the whole computation, driven by *feature checking*, is potentially *deterministic*, as it is driven by economy considerations (the actual "cost" of each operation): this is an important difference with respect to earlier versions of the *Minimalist Program*: there should be no parallel exploration of multiple SDs, then comparing them on the basis of global economy considerations (total length of the derivation, total computational resources used); the most recent minimalist idea of derivation is only driven by *local economy* consideration (Collins 1997, Chomsky 2000:99)

In this sense the Minimalist framework is mainly *derivational*, in the sense proposed in §2.2.

Summarizing these points, the *derivational* nature of this framework is justified by complexity considerations that deeply affect this approach (Chomsky 2000:104):

a. Simple operations preempts more complex ones[58];
b. the search space is limited (*locality conditions*, as in (67))
c. access to the features set *F* is restricted by *Numeration*
d. the computation is *locally determined* (no *look-ahead*)

[58] Merging *there* is "less expensive" than moving *a proof* :
i. $there_i$ is likely $[_{Tdef}\ t_i$ to be a proof discovered]
ii. *there is likely $[_{Tdef}$ a proof to be discovered]

Moreover, considerations on abstract "memory limitations" justify the assumption that as soon as an element is spelled out, there is no need to keep it in the working memory anymore. This is one of the leading reasons supporting the notion of *phase*:

(68) **Phase**

relevant unit during which the computation maps a subset LA' of elements selected in the Lexical Array (LA), to <π, σ>;

A *phase* PH has the form [α [H β]], where H is the head and α - H the *edge* of PH.

In this sense, a *phase* identifies an object "relatively independent in terms of interface properties" (Chomsky 2000:106); so *phases* are assumed to be CP (a complete discourse-related entity), vP (a complete predicative structure) and maybe also DP (a complete argument).

Rather than spelling out the whole *phase*, Chomsky (2001:5) proposes to spell out only β while keeping the edge available for *successive cyclic movement*. The memory limitations I pointed out above are captured by the *Phase Impenetrability Condition*:

(69) **Phase Impenetrability Condition** (Chomsky 2001:5 PIC)

The domain of H is not accessible to operations, but only the edge of PH.

Quasi-algorithmically speaking (indeed many points remains crucially underspecified yet), we could sum up the whole computational procedure that FL should perform as follows:

(70) Minimalist computation

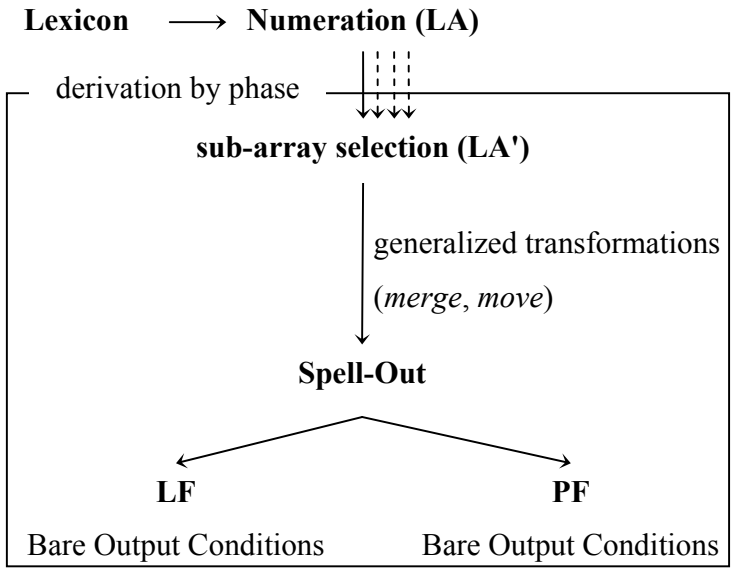
1. Select LA from the *lexicon*
2. Recursively maps LA to Exp by *phases* following this subroutine:
   i. select LA^i, such that LA^i is a proper subset of LA, and kept it active in memory;
   ii. take the most embedded verbal head and *merge* it with the adequate complements in LA^i;

iii. look for OCC features on the head, then check them by *internal merge* if no lexical items in LA^i can satisfy them (follows the economy preferences given in **Errore. L'origine riferimento non è stata trovata.**))

iv. proceed until LA^i is exhausted, then restart from i. until LA is exhausted too

This pseudo-algorithm can be illustrated by the following schema:

(71) Minimalist framework (*Minimalist "T-Model"*, Chomsky 1999-2001):



These assumptions largely unload on the *lexicon* the theoretical burden of accounting for cross-linguistic variation and apparent idiosyncrasies in phrase structures; because of that, Minimalism has been often considered a *lexicalist hypothesis*.

A first fault of this approach is however clear: the *lexicon* is highly underspecified within Chomsky's papers. Another point that has been criticized is the assumption that a perfect design would imply no redundancy at all: Jackendoff (1997) points out that redundancy could be useful for learning purposes, then highly welcome if present. Moreover, interface requirements (especially at LF) are quite underspecified, this only allows for highly arbitrary guesses concerning the status of *bare output conditions*.

2.2.3 LEFT TO RIGHT INCREMENTAL PROCESSING

Phillips (1996), in a pre-*phase* era, makes an important step forward as introduced in §2.1: he convincingly shows that, because of *constituency contrasts* and *ambiguities resolution* preferences, the distinction between the *parser* and the *grammar* is rather artificial; this suggests that the PIG model presented in (49) could be fairly adequate[59]. Consider the following data:

(72) a. John [[[gives candy] to *children*$_i$] in *their*$_i$ library]

a'. John intended to give candy to children in their library and [give candy to children]$_i$ he did t_i in their library

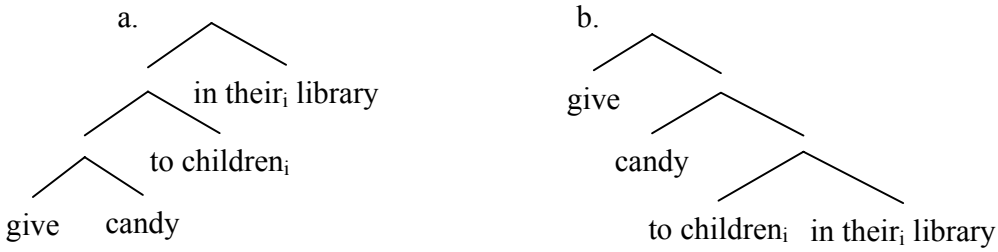
a". *John intended to give candy to children in their library and [to children in their library]$_i$ he did give candy t_i

b. John said Bill [left yesterday]

b'. !John [said [Bill left] yesterday]

(72.a-a") show that a normal *right-branching* structure (73.b) would not predict the contrast in VP-fronting between (72.a') and (72.a"); this rather supports a *left-branching* structure as the one proposed in (73.a). On the contrary, the *binding* option (following (56)) would require a *right-branching* structure of the sort exemplified in (73.b) where "children" *C-commands* "their".

(73)

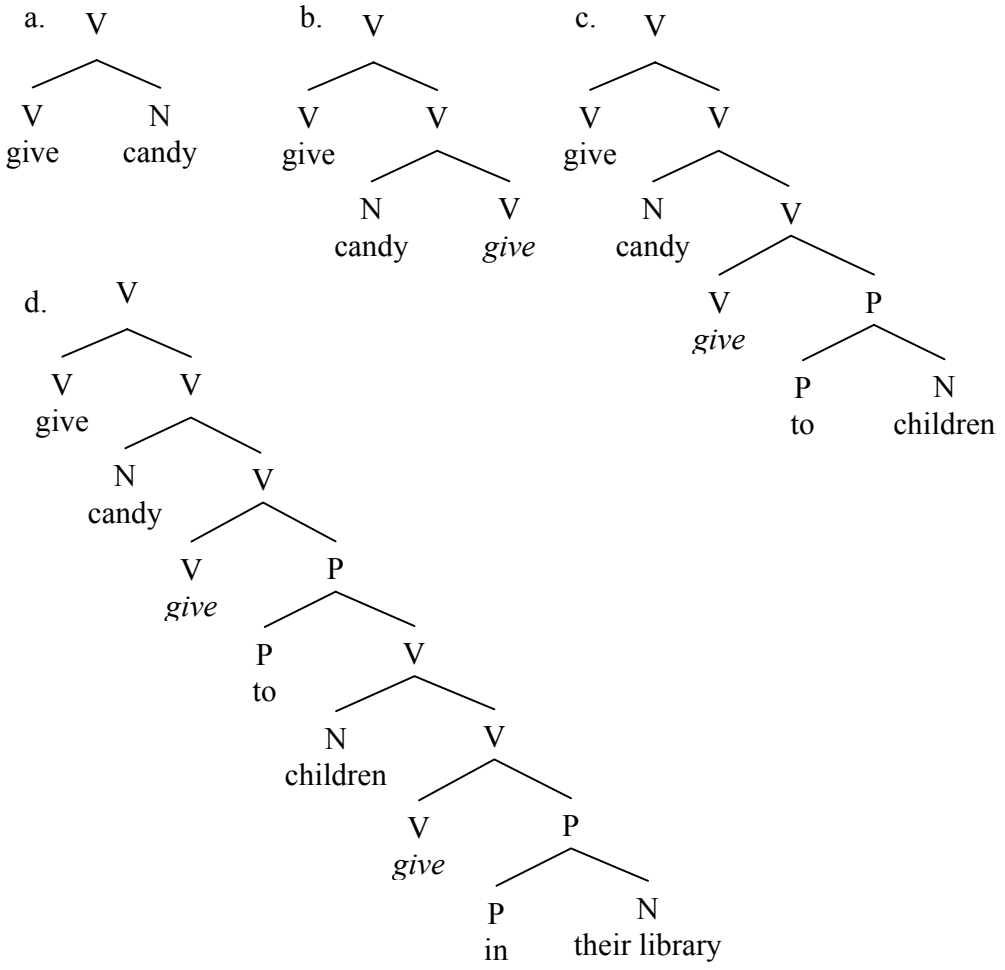


Pesetsky (1995) postulates both structures (the *layered structure* (73.a) as the locus where phrase extractions are computed, while the *cascade structure* (73.b) as the locus of *binding*, *negative polarity items* interpretation and so on). Phillips proposes that a slight modification of (73.b) would be sufficient to account for any apparent contrast

[59] Phillips, moreover, shows that this approach is compatible even with morphological/phonological-to-syntax interface phenomena (e.g. clitic placement).

that seems to require the *left-branching* structure proposed in (73.a). His solution is based on a *left-to-right* structure building derivation as illustrated below:

(74)

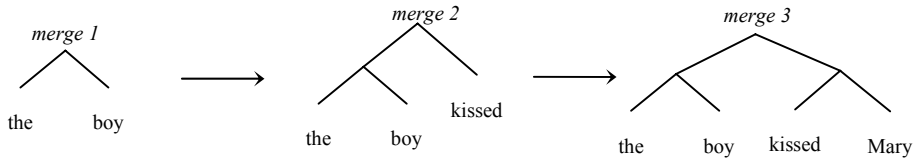


The derivation in (74) (reminiscent of Larson's (1988) *VP shells*) yields a solution to the *constituency problem*: in fact, both (74.b) and (74.c) represents *temporary constituents* (later destroyed by further *merge* operations) that are available for extraction at some specific point of the derivation. This is the cornerstone of the *left-to-right* structure building procedure Phillips proposes: *temporary constituency* is a property created during the derivation we could use to account for apparent paradoxes. Crucially, a *left-to*

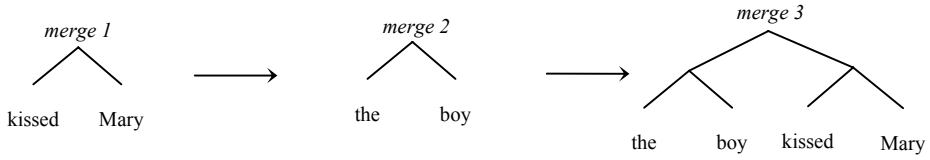
right processing procedure creates temporary constituents that are different from the standard *bottom-to-top*[60] structure building proposed by Chomsky (see §2.2.2):

(75) The boy kissed Mary

a. Phillips proposal:



b. Standard minimalist proposal:



For instance, only (75.a) creates the temporary constituent [the boy kissed].

Since these kinds of constituents are relevant for phenomena such as those described in (72.a-a"), Phillips takes this to be a proof of the fact that the grammar incorporates a simple structure-building procedure, *merge right*:

(76) **Merge right** (Phillips 1996:18)

new items must be introduced at the right edge of the structure

Even from a processing perspective, there seems to be a strong tendency to produce *right-branching* structures, privileging, moreover, the most local attachment, cf. (72.b-b'); this preference has been captured by the following principle:

(77) **Branch right** (Phillips 1996:19)

Metric: select the attachment that uses the shortest path(s) from the last item in the input to the current input item;

Reference set: all attachments of a new item that are compatible with a given interpretation

---

[60] That is quite different from *bottom-up*, that simply means starting from input data rather than from structure projections. On the other hand, *Bottom-to-top* means starting from the inner verbal shell, then adding higher and higher layers piecemeal.

What Phillips' model assumes is the incremental nature of the process without commitment to any *performance* modality. From a processing perspective, this seems to be fairly in line with psycholinguistic results:
garden path sentences are abused but neat examples on this point, showing that words in input are integrated in the structure as soon as they are heard and that most of the time only one structural solution is pursued. This leads to "reanalysis" when an unexpected word is found as the next token of the input:

(78) a. The horse raced past the barn fell (Bever 1970)
b. The horse (that was) raced past the barn fell (down)

The correct structural analysis requires the "reconstruction" of the missing words in (78.b), but at the first reading, any native speaker cannot avoid a parsing breakdown as soon as they reach the word *fell*, unexpected on the basis of the structural hypothesis adopted by the reader up to that point: "raced" is interpreted as the past tense of the verb "race" and "past" as the past tense of "pass", missing the reduced relative and the adverbial nature of "past" that, in fact, represents the only correct structural solution to integrate "fell" at the end of the sentence.
Garden path effects show that we do not pursue any possible structural hypothesis; rather, we make choices as soon as possible pursuing only one solution. The other options are retrieved only after a parsing breakdown, under the control of precise *top-down expectations*. This seems to be true not only of English (and in general of *head-initial* languages) but even of *head-final* languages like Japanese[61].

An essential difference between the standard minimalist model presented in the previous paragraph and Phillips' one is about the driving force of the process: Chomsky's vision of the grammar is *head-driven* (the computation first selects the inner heads, then *merges* them with complements, then with specifiers and so on, as external heads are introduced); Phillips' model, on the other hand, builds structures at any input token without any preference for complements or heads. Moreover, Chomsky's system is

[61] See Schneider 1999 for a review of the topic.

essentially compatible with head-final languages (even if hardly plausible as a processing model), while Phillips' one, if applied to head-final languages, seems less straightforward even though surely readily viable as a processing system.

2.2.4 THE CARTOGRAPHIC APPROACH AND LOCALITY CONSTRAINTS ON MOVEMENT

The *Minimalist Program* put much stress on the computational engine of the FL but very little is said about the phrase structure organization (apart from the critical inquiry of some standard *X-bar* assumptions). The *Cartographic Approach* (Belletti, Ed. 2002, Cinque 1999, Ed. 2002, Rizzi 1997, Ed. 2002, 2004 and related work), on the other hand, unleashes new hypotheses about the structural organization of the functional elements within the DP, IP and CP domain from a clean and consistent cross-linguistic perspective.

As I pointed out in §1.1.3, one of the clearest examples to grasp this idea is provided by Cinque's analysis of adverbial positions (Cinque 1999): his work suggests that the IP shell is more structured that usually thought and, crucially, this seems to be a universal generalization supported by robust empirical evidence. In fact, even if many languages do not have specific morphemes that realize Cinque's postulated functional heads, the existence of the latter is supported by the distribution of adverbial elements (allegedly occupying the specifier of these functional projections):

(79) a. John *probably* eats *a lot*
a'. *John *a lot* eats *probably*
b. *Necessarily* John *often* sees Mary
b'. **Often* John *necessarily* sees Mary

Probably and *necessarily*, in English as in many other languages, have to take scope respectively over *a lot* and *often*. Without trying to investigate the rationale behind this necessity, the best way to account for it is to assume, as we mentioned in §1.1.3, the existence of a hierarchy that we roughly expressed by these basic classes:

(80) *modals > temporal > aspectual*

This gross classification is refined in Cinque (1999) on the basis of distributional asymmetries such as the ones presented in (79). Being careful to discard apparent ordering counterexamples (adverbials that modify each other, *focalization* phenomena etc.) the resulting picture is quite articulated:

(81) Mood $_{\text{speech act}}$, Mood $_{\text{evaluative}}$, Mood $_{\text{evidential}}$, Mood $_{\text{epistemic}}$, T $_{\text{past}}$, T $_{\text{future}}$, Mood $_{\text{irrealis}}$, Mood $_{\text{necessity}}$, Mood $_{\text{possibility}}$, Mood $_{\text{volitional}}$, Mood $_{\text{obligation}}$, Mood $_{\text{permission}}$, Asp $_{\text{habitual}}$, Asp $_{\text{repetitive I}}$, Asp $_{\text{frequentative I}}$, Asp $_{\text{celerative I}}$, Asp $_{\text{perfect}}$, Asp $_{\text{retrospective I}}$, Asp $_{\text{proximatives}}$, Asp $_{\text{durative}}$, Asp $_{\text{generic/progressive}}$, Asp $_{\text{prospective}}$, Asp $_{\text{sg completive I}}$, Asp $_{\text{pl completive I}}$, Voice, Asp $_{\text{celerative II}}$, Asp $_{\text{repetitive II}}$, Asp $_{\text{completive II}}$, T $_{\text{anterior}}$, Asp $_{\text{terminative}}$, Asp $_{\text{continuative}}$, Asp $_{\text{frequentative II}}$

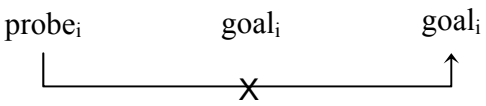
Similar conclusions have been obtained within other functional areas such as CP (Rizzi 1997, 2002-2004) and DP (Cinque 1994, Scott 1998). *Subject* (Cardinaletti 2002) and *negation* (Zanuttini 1997) positions have been investigated too with similar impressive results. The outcome of this research is then a detailed "map" of the functional phrase structure that would allow us to predict in a very precise way many universal distributional constraints.

This is an interesting result from the computational perspective adopted within this dissertation, since it emphasizes the "sensitivity" of the *movement* options: since we understand the nature of the operations involving linguistic elements (and their features) even from their domain of applicability, it is worth considering *locality conditions* as the set of principled constraints used to narrow the range of application of these operations. Influential work on this issue is Rizzi's *relativized minimality* (Rizzi 1990, 2002) and Starke's dissertation (Starke 2001) that I will quickly review in the rest of this paragraph.

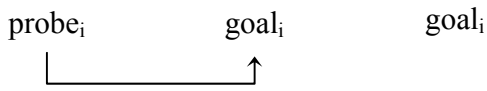
As we noted before (§2.2.2) the application of the *merge* operation has to be *local* in a very narrow sense, namely two elements that enter a *merge* relation have to be strictly adjacent. For *movement* the "local relation" has to be intended in a less restrictive way:

the object and its *deleted copy* (or *trace*) could be relatively distant, unless some feature of the same kind as the one searched by the *probe* on a *goal* intervenes between them. This is the insight sketched in (67) and repeated below:

(67) a.



b.



This simple picture, in fact, can predict in a very insightful way violations on extraction from *weak islands* (syntax), *assimilations* phenomena (phonology) and possibly even semantic and other perceptual (non-linguistic) facts (recall the discussion in §1.2).

Rizzi's and Starke's contribution to this idea has been to refine these *locality conditions*, relativizing them with respect to the classes of elements they apply to. We can draw some important generalizations from these works:

1. locality is sensitive not to single features but to classes of features (Rizzi 1990-2004);
2. locality conditions are better predictive if they apply to a hierarchy of classes rather than to unorganized subsets of features (Starke 2001).

The first point is well explained by *weak island* violations: the *movement* of a *wh*-element is blocked not only when it crosses another *wh*- element as shown in (82),

(82) a. **how*$_i$ do you wonder <u>why</u> I should cook this stuff t_i ?

but also when it crosses a negation (82.b), a focalized element (82.c), or a quantificational adverbial (82.d) (Starke 2001:5):

(82) b. **how*$_i$ don't you think that I should cook this stuff t_i ?
**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 ?

These facts circumscribe a natural class of sensitivity in terms of locality conditions:

(83) Q(uantificational) $\{_{\text{wh, negation, focus, quantificational-adverbs}}\}$

All elements bearing one of these features are potentially deadly *interveners* for *movement* (or *chain formation*) of another element bearing a feature in the same class.

Argumental features represent another well known class of *interveners* (*A-chains* or *A-movement*):

(84) A(rgumental) $\{_{\text{person, number, gender, case}}\}$

Starke suggests that these classes are not simply flat, but hierarchically organized: this is because hierarchical relations could be helpful to predict what has usually been caught as exception, but could, in fact, be perfectly integrated as a standard phenomenon given a richer feature structure. This fact can be represented by the example below:

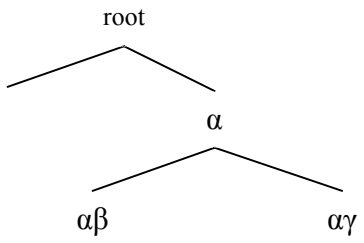
(85) [*which book*]$_i$ did *who* buy t_i ?

The chain *which book*$_i$ ... *who* ... t_i should represent a violation of the locality conditions, but, in fact, it is not: following standard interpretations, *which book* has "something more" with respect to other elements that usually cause a violation when extracted from a weak island. Even if the nature of the "extra property" of *which book* is not completely clear, the picture we can draw is pretty neat:

(86)

a.	*A_α ... B_α ... A_α	*what$_{\text{wh-}}$ did who$_{\text{wh-}}$ buy <what$_{\text{wh-}}$> ?)
b.	$A_{\alpha\beta}$ ... B_α ... $A_{\alpha\beta}$	[which book]$_{\text{wh-, d-linking}}$ did who$_{\text{wh-}}$ buy < [which book]$_{\text{wh-, d-linking}}$> ?
c.	*A_α ... $B_{\alpha\beta}$ ... A_α	*what$_{\text{wh-}}$ did [which student]$_{\text{wh-, d-linking}}$ buy <what$_{\text{wh-}}$> ?
d.	*$A_{\alpha\beta}$ ... $B_{\alpha\gamma}$ ... $A_{\alpha\beta}$	*who$_{\text{wh-, animate}}$ did [which student]$_{\text{wh-, d-linking}}$ see <who$_{\text{wh-, animate}}$> ?

e.



Given the hierarchy in (86.d) the feature αβ and αγ are *more specific* than α. We could assume, following Starke, that locality conditions, in this hierarchy, apply horizontally (features in the same class potentially block each other, (86.a,d)) and bottom-up (daughters are potential blockers for mothers, (86.c)) but not *top-down* (daughters cross mothers without any *locality* violations, (86.b)), i.e. a less specific feature cannot block a more specific one.

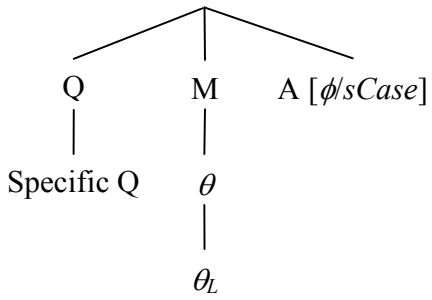
Then, following Rizzi (2004), the class of elements forming A' chains can be refined using at least three subclasses as shown in (87). Otherwise, following Starke (2001), it can be structured as a hierarchy as sketched in (88)[62]:

(87) a. Topic $\{_{\text{topic}}\}$

b. Q $\{_{\text{Wh, Neg, measure, focus, ...}}\}$

c. Mod(ifier) $\{_{\text{evaluative, epistemic, Neg, frequentative, celerative, measure, ...}}\}$

(88)



[62] This is the meaning of the symbols used: Q = Quantifiers; M = Modifiers, θ = theta-marked, θ_L = licensed theta-role; ϕ = phi-features; *sCase* = structural case.

Summarizing, in these pages I reviewed two types of approach (*derivational* Vs. *representational*), highlighting the differences that are most relevant from a computational perspective. I pointed out that two crucial improvements on the standard *Minimalist model* (moreover *consistent* with this framework) are Phillips' *left-to-right incremental processing* idea and the *cartographic* assumption about specific *functional features* realized in distinct projections and grouped in *macro-classes* for the purposes of *locality* constraints.

The next pages will evaluate the state of the art with respect to the implementation of these ideas within realistic computational/formal models.

2.3 COMPUTATIONAL MODELS

Linguistic frameworks presented in §2.2 mainly dealt with *descriptive/explanatory* adequacy, that is, relevant linguistic phenomena are captured using devices that are at least abstractly plausible. Implementing these intuitions within a computational model is unfortunately not readily possible. As I mentioned in the introduction, this is because:

1. theories are not precisely formalized, then many "technical" aspects are left underspecified and this turns out to be an heavy burden on the shoulders of the computational linguists since the choices they have to make to fill these blanks, often have unknown empirical consequences; moreover it is not always clear how to encode transparently linguistic intuitions even if they seem (intuitively) very precise;
2. once formalized, theories turns out to be unsound, that is, principles/rules are contradictory or incompatible then the system become inconsistent;
3. once formalized, models are impossible to be implemented since too complex, that is, requirements in terms of *space* and *time* are unsustainable (cf. §1.4).

In order to frame these problems, this paragraph will go through some models implementation/formalization that attained a significative level of transparency with respect to some relevant linguistic assumptions presented in §2.2. This would allow us to highlight where difficulties reside and why.

We should notice that, because of concreteness, many linguistic problems are often avoided or left underspecified in order to implement more light, manageable and tractable computational models.

2.3.1 PRINCIPLE-BASED PARSING

Early 90's are characterized by the attempt of building linguistic resources deeply inspired by prominent linguistic approach such as Principle and Parameters (mainly expressed within the *Government and Binding* framework, §2.2.1). An interesting research thread within this context has been classified as "Principle-based parsing" approach (Berwick and al.1991).

The main goal of this framework was defining a *parsing* model deeply inspired to the modularized idea proposed within the Principle and Parameters framework: components of the grammars, that is, both levels of representation such as *deep structure* or *surface structure* and *principles*, can be independently defined/formalized; their interaction would allow us to describe phrase structures (as shown in §2.2.1).

Fong's dissertation (Fong 1991) addresses many important issues about the difficulty of this approach, however providing important insights and a significant empirical coverage with his model implementation.

Notably he succeeded in:

1. defining any principle using a high level language specification (every principle has been formalized using a subset of the first order logic, Horn clause logic, following Prolog specifications);
2. identifying important inefficiency issues such as *principles ordering*, the logical inconsistency of part of the theory, the underspecification of many linguistic intuitions and the low efficiency of principle-based systems.

The final implementation of the system comprehends 25 principles plus two macro, used to expand automatically the principles application. As we noted in §2.2.1 the principles behavior, within GB approach, can be distinguished in two classes: *generator* (e.g. f*ree indexation*, *move α*) and *filters* (e.g. *case filter*); the predicted problem was that if *filters* do not constrain as soon as possible *generators* output, the number of hypothetical structures to be evaluated grows dangerously (even though any order would produce the same correct structural description, if any). Fong shows that this concern is quite realistic and that, apparently, there is not any a priori best ordering to be used to speed-up the parser in any context. In fact, depending on the sentence to be processed,

principle ordering can be finely tuned in order to generate the minimum problem space to be explored. Fong uses "cheap" cues within the sentence in order to propose a pre-ordering before start parsing (for instance *pronouns* suggest that the application of *binding principles filters* should happen as soon as possible).
Once ordered, principles and grammar are compiled in *L(eft-to-right)R(ightmost)* parsing tables[63]. The architecture used is in fact a slight modification of a LR(1) parser, consisting of a *Finite State Automaton* (using two tables, a *transition table* and an *action table*) and three *record stacks* (one to register the state of the FSA, one to store the partially built structural description, one to encode other contextual information).

An efficiency problem with this structure has been noted in Merlo 1996: once modules have a lot to communicate among them (that is, they are not *informationally encapsulated* in Fodor's 1983 terms), a fully modular system is inefficient; this inefficiency can be somewhat overcome using a *pre-compilation* of only part of the grammar (Merlo points out an interesting principled way to distinguish which part of the grammar to *compile off-line*, e.g. phrase structure information such as *X'-theory*, and which one do not compile, e.g. empty traces position assumptions).

Without exploring the details of this operation, the intuition behind this assumption should be clear: despite of the premises, the relation between *grammar* and *parser* can hardly be attained within this model; linguistic principles, such as the ones analyzed in Fong, are not *realistically* suitable for a parsing model (in the precise sense proposed in the introduction): for instance an *unbounded look-ahead*, required to retrieve long distance relations, can produce a perfect parse, at the first attempt, even for *garden path sentences*, but this is not a "cognitively plausible" behaviour (cf. §2.2.3).

[63] Aho and al. 1986.

2.3.2 MINIMALIST FORMALIZATION

Solving efficiency problems (both cognitively and computationally) was one of the main reasons for moving from *GB* to *Minimalism*. Unfortunately, implementing a computational system inspired to the *Minimalist Program* was not an easy task. The extremely dynamic nature of the inquiry produced many (often contradictory) ideas (cf. *earliness*, Pesetsky 1989, Chomsky 2000 Vs. *procrastinate*, Chomsky 1995). Actually, this extremely active environment is still evolving and it is hard to be crystallized in a consistent computational theory, moreover many crucial points are dramatically still underspecified. Within this perspective, Stabler's (1997) work is particularly valuable since it provides the cleanest (and coherent) formalization of the *Minimalist Program* (the version proposed in Chomsky 1995): following his intuitions, a *minimalist grammar* can be defined as a 4-tuple {*V*, *Cat*, *Lex*, *F*} such that:

(89) **Minimalist Grammar** (**MG**, definition 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 one of the three possible kinds of selection {=*x*, =*X*, *X*= | x ∈ *base*} where

=x means simple selection of an x phrase,

=X selects an X phrase, suffixing the selecting head with the phonetic features of the selected X phrase;

X= selects an X phrase, prefixing the selecting head with the phonetic features of the selected noun phrase;

licensees specify requirements forcing *phrasal movement* {-*wh*, -case ...}, -x triggers *covert movement*, while -X would trigger *overt movement*;

licensors are features that satisfy licensee requirements {+*wh*, +case ...}

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

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

The language defined by such a grammar would be the closure of the lexicon (*Lex*) under the *structure building operations* (*F*). (90) is a simple example of MG able to deal with simple *wh- movements*[64]:

(90) 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 = [-wh D what], [=V T did], [D you], [=D D= V see], [=T +wh C ∅]

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

merge (X, Y) = is a function taking two adjacent subtrees X and Y, outputting an unified structure Z of the form [$_X$ X Y] if and only if X has as first selecting feature (=*f*, =*F*, *F*=) and Y has the needed selected feature *F* as the first feature of the base set

move (X, Y) = if a function taking two subtrees [+*g* X] and [-*g* Y] such that <[+*g* X], W, [-*g* Y]> (where W can be any possible subtree, even null, but without any selecting/selector feature *g* in it) and produces Z of the form [[$_X$ Y X] W, t$_Y$]

Following Chomsky (§2.2.2), a derivation proceeds bottom to top and *licensees* trigger *movement* as shown in (91)[65]:

[64] For the sake of simplicity, let us assume that capital features directly select the position of the arguments without involving pre-/in-fixing (then, =X means that the argument X is selected to the right of the selecting head, while X= to the left). The very same result is however derivable by a combination of standard (non directional) selection plus a trigger for *movement* (for instance -*case*).

[65] This is a very simplified version of derivation, to be taken only as example. It would be clearly possible including *subject movement* too, but this would have been required extra steps in the derivation.

(91) 1. merge ([=D D= V see], [-wh D what]) → [$_{see}$ D= V see, -wh what]
2. merge ([D you], [D= V see, -wh what]) → [$_{see}$ you, [$_{see}$ V see, -wh what]]
3. merge ([=V T did], [$_{see}$ you, [$_{see}$ V see, -wh what]]) →
([$_{did}$ T did, [$_{see}$ you, [$_{see}$ see, -wh what]]]
4. merge ([=T +wh C ∅], [$_{did}$ T did, [$_{see}$ you, [$_{see}$ see, -wh what]]]) →
([$_{C}$ +wh C ∅, [$_{did}$ did, [$_{see}$ you, [$_{see}$ see, -wh what]]]])
5. move ([$_{C}$ +wh C ∅, [$_{did}$ did, [$_{see}$ you, [$_{see}$ see, -wh what]]]] →
[$_{C}$ What C ∅, [$_{did}$ did, [$_{see}$ you, [$_{see}$ see, t_{what}]]]]

Some interesting formal results show that there is a weakly equivalent *Multiple Context-Free Grammar* for any MG (then MG are included in the *Mildly Context-Sensitive* class of grammars, Michaelis 1998) and that a *recognizer algorithm* can be defined (both *top-down* and *bottom-up*) for MGs (Harkema 2001). However, it is difficult to draw any cognitive/computational (i.e. *realistic*) conclusion from these results, since, for instance, the *recognizer* is based on a *deductive parsing* perspective (Shieber and al. 1995) that is not a cognitively motivated procedure and the equivalence results are based on a *weak equivalence*: namely, other formalisms/derivations can produce the very same set of strings MGs will produce, but either they fail to associate the same structures to these strings or they encode *lexical items*, *features* and *structure building operations* in a less transparent way with respect to the linguistic intuitions that justified them. I believe that these two factors are indeed crucial parameters of evaluation, at least if the final goal of the formalization is to make clear what the computational/cognitive implications are either in *parsing* or in *generation*.
In this respect, very little can be said about **V** and **Lex**, largely underspecified within the *Minimalist Program*: Stabler's formalism on these points makes the simplest possible assumptions, worth to be kept the way he defined them.

Some important problems however arise with this formalization: the organization of **Cat** in four subclasses of features, for instance, is not completely satisfactory given the *cartographic* analysis provided in §2.2.4: *base* and *select*, the sets of standard categories, mesh together *functional* and *lexical features* (Tense, V, D, N...), even though there are

strong empirical reasons to believe that this distinction is both theoretically (Belletti, Rizzi 1996) and cognitively justified (Leonard 1998, see §3.3 on this point). Furthermore, such a simple categorization would not be able to predict the correct locality constraints on *movement*: in these terms, *locality* would be just a matter of features identity, although it has been shown that this cannot be empirically adequate (§2.2.4).

Another problem is related to the fact that this formalization is based on early version of the *Minimalist Program* (Chomsky 1995) then, for instance, the notion of *phase* (§2.2.2) is completely ignored. This allows the grammar to retrieve/produce arbitrarily distant relations, clearly not a welcome generalization (cf. §2.2.2, §2.2.4).

2.3.3 MINIMALIST PARSING

Efforts in implementing the minimalist framework have been made not only from the formalization side, but also from the parsing perspective: Fong (2004) explores recent minimalist intuitions (§2.2.2, §2.2.3) trying to build a *parsing* model that makes crucial use of technologies such as the *probe-goal* driven theory of *movement* and the idea of *derivation by phase*. His work is interesting since it implicitly shows how the *Minimalist framework* is not suitable for *parsing* using a *bottom-to-top* perspective. In fact, having a *left-to-right parsing* model (somehow similar to Phillips' model presented in §2.2.3) that aims to build structure incrementally, it is hard to make direct use of *merge* and *move*; for instance assuming that a *complement* is merged with its head before the *specifier*, from a *parsing* perspective, the *left-to-right* flow of information bears to the attention of the parser the *specifier* before the *head* or the *complement* (even in *head-final* languages like Japanese):

(92) a. $\text{Mary}_{\text{spec}}$ $\text{read}_{\text{head}}$ a $\text{book}_{\text{complement}}$

b. $\text{Mary-ga}_{\text{spec}}$ $\text{hon-o}_{\text{complement}}$ $\text{yonda}_{\text{head}}$ (Japanese)
Mary-nom book-acc read
'Mary read a book'

This would cause the parser to wait before merging the *specifier* with the *head* until the *head-complement* cluster is built. Having an item, the *specifier*, not yet integrated in a structure when the next words are parsed would easily cause backtracking and multiple choices options, that is, *non-determinism*.

Fong's incremental parser partially solves this problem assuming that the processing proceeds assembling *elementary trees* in a way that is reminiscent of the *Tree-Adjoining Grammars* (TAGs, Joshi 1985):

(93) **parsing algorithm**

step 1 – given a category *c*, pick an elementary tree headed by *c*

then start consuming input tokens:

step 2 – fill in the specifier (if one is predicted from the elementary tree selected)

step 3 – fill in the head

step 4 – fill in the complement(s) by recursively calling the parsing algorithm with *c′* where *c* has the lexical property of selecting *c′*

Once the right category *c* is selected, no *choice points* should be encountered during the parsing and the integration/rejection of the input tokens will be driven by simple *top-down expectations* (*elementary trees* can be considered nothing but that). As we have seen before, Chomsky's model avoids *ambiguities*/*non-determinism*, assuming a *numeration* process during which the elements that enter the computation are pre-selected and arranged in a *Lexical Array* (LA). Unfortunately, no LA is available for *on-line parsing* since any element that has to be integrated is revealed for the first time only when read from the input (§2.2.3). Then the critical point of the simple model sketched in (93) is located in the first step, namely on the procedure of picking up the "right" elementary tree[66]. Before addressing this issue, let us explore the other technologies Fong uses in his parser: *lexicon* and the *structure building devices*.

Fong's lexicon remains underspecified with respect to *phonological* and *semantic* features, mainly encoding syntactic properties in a way that can be rephrased as follows (note the similarity with Stabler's *Cat* set of features, §2.3.2):

(94) **Lexicon**

LI = {*Cat*, *P*, *uF, F*} where

Cat is a finite set of syntactic categories {$V_{transitive}$, $V_{unaccusative}$ ... *C*, C_{wh} ... $N_{referential}$, $N_{expletive}$ ...}

P is a finite set of properties {*select(x)*, *value(case(_value))* ... *spec(select(x))*}

uF is a finite set of uninterpretable features essentially of the form

uF = (*ϕ-features* ∪ *other*) where

ϕ-features = {*person, number, gender* ...}, *other* = {*EPP*...}

F is a finite set of interpretable features {*person, number* ...}

[66] Unfortunately Fong does not provide any effective cue to solve this issue.

This formalization can be represented by a table like the following one:

(95) **Lexicon fragment** (used in Fong 2004)

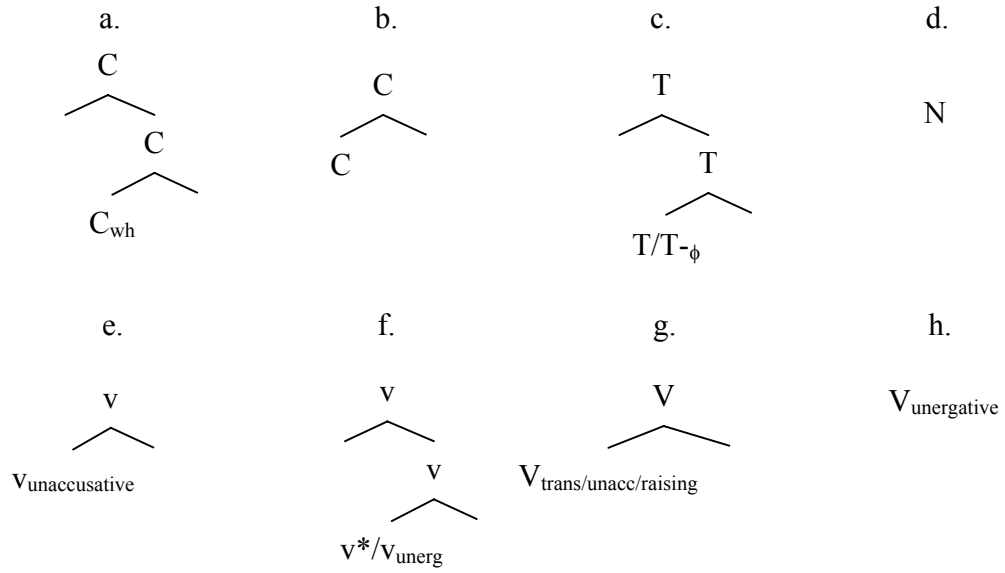
Main category	**Cat**	**P**	**uF**		**F**
			ϕ-features	*Others*	
v	$v^*_{transitive}$	select(V) spec(select(N)) value(case(acc))	per(P) num(N) gen(G)	epp	
	$v_{unaccusative}$	select(V)			
	$v_{unergative}$	select(V) spec(select(N))			
V	$V_{transitive,\ unaccusative}$	select(N)			
	$V_{unergative}$				
	$V_{raising}$	select(T-$_{\phi}$)			
T	T	select(v) value(case(nom))	per(P) num(N) gen(G)	epp	
	T-$_{\phi\ (infinitive)}$	select(v)		epp	
C	C	select(T)			
	C_{wh}	select(T)		epp	wh
N	$N_{referential}$			case(_)	
	N_{wh}		wh	case(_)	
	$N_{expletive}$		per(P)		

Where:

- *case(_)* instantiates an open slot (namely a *feature*), e.g. for case values;
- *select(x)* is a function that selects an element headed by x;
- *spec(select(x))* is a function that selects, in the *specifier* position, an element headed by x;
- *value(case(x))* assigns x case to an open slot;
- *epp* is an uninterpretable feature able to trigger *movement*; it legitimates a *specifier* position as landing for *movement*;

Then elementary trees can be drawn from the lexicon, projecting the basic *functional*/ *lexical heads* following their selectional properties:

(96) **elementary trees** (used in Fong 2004)



Fong suggests that in order to keep the parsing algorithm (locally) *deterministic*, we should underspecify the tree structure when more than one *elementary tree* is structurally compatible: e.g. once found an argumental NP, that is compatible with the *specifier* position both of a *transitive* and of an *unergative* v *elementary tree*, simply project the head and the structure common of both, keeping the rest of the non-matching structure underspecified; in this case, we could project the V *complement*, predicted both by the *elementary tree* headed by T and by the one headed by T-ϕ, but not the property *value(case(acc))*, present only in T.

On the other hand, this model incorporates devices able to deal with *Long Distance Dependencies*: the *move box* and the *probe box*. The *move box* is somehow reminiscent of the *HOLD* register of *Augmented Transition Networks* (*ATN*, Woods 1970): potential *trace* fillers are put in a sort of *short-term memory buffer*, the *move box*, in a principled manner and retrieved when necessary to fill the relevant empty positions in the sentence. Fong provides a precise algorithm that controls the behavior of this devices:

(97) **move box**

a. (*initial content*) at the beginning of any parse, initialize the *move box* as empty;

b. (*fill condition*) whenever an open position of an elementary tree is filled from the input, copy the filling element from the input to the *move box*;

c. (*preference rule*) whenever an open position of an elementary tree can be filled by the element in the *move box*, fill it before looking at the input,

d. if step 3 is successful, remove the element from the *move box*;

d'. (conditions on d, *empty condition for expletives*) if in the *move box* there is an expletive, then, after filling a position with it, *optionally* remove it from the *move box*.

Using the *move box* requires a refinement of the parsing algorithm given in (93) that can be done simply assuming that fillers can come both from the *input* and from the *move box* (the *preference rule* given in (97) avoids non-determinism that would be caused by the option of choosing a filler either from the *input* or from the *move box*).
This device is sufficient to deal with many simple *movement* phenomena:

(98) John saw Mary (assume *saw* → *past* + see)

step	partial phrase structure built	move-box	input	operation
1	[$_c$ _]	ø	J. saw M.	c:select(T)
2	[$_c$ [T _ [T T _]]]	ø	J. saw M.	fill spec-T
3	[$_c$ [T J. [T T _]]]	J.	saw M.	fill head-T
4	[$_c$ [T J. [T past _]]]	J.	see M.	T:select(v)
5	[$_c$ [T J. [T past [v _ [v v* _]]]]]	J.	see M.	fill spec-v with $t_{J.}$ from m.-box
5'		ø	see M.	free m.-box
6	[$_c$ [T J. [T past [v $t_{J.}$ [v v* _]]]]]	ø	see M.	v:select(V)
7	[$_c$ [T J. [T past [v $t_{J.}$ [v v* [V V _]]]]]]	ø	see M.	fill head-V
8	[$_c$ [T J. [T past [v $t_{J.}$ [v v* [V see _]]]]]]	ø	M.	V-select(N)
9	[$_c$ [T J. [T past [v $t_{J.}$ [v v* [V see N]]]]]]	ø	M.	fill head-N
10	[$_c$ [T J. [T past [v $t_{J.}$ [v v *[V see Mary]]]]]]	M.	ø	parse found!

From this example, it should be clear that the *fill condition* is actually applied only to *argumental* elements in order to deal with *A-movements*. In fact, the current implementation is able to deal, at best, with the simplest *locality conditions* (features identity) leaving unaccounted any *relativized minimality* effect (Rizzi 1990, §2.2.4).

Moreover, there is an extra price to be paid so as to capture *cyclic movement*: this is represented by the introduction of a choice option (namely a *non-deterministic* device) any time we fill a trace; for instance, with an *expletive*, the parser should choose whether to keep it in the *move box*, predicting a *cyclic movement*, or to remove it from the *move box* (remember that keeping it till the end of the sentence would cause the wrong behavior of inserting a "there-trace" instead of using "prizes" as shown in (99.b)):

(99) a. *there* are supposed t_{there} to be likely t_{there} to be prizes awarded
(free the *move-box* before "prizes")

b. *there* are supposed t_{there} to be likely t_{there} to be t_{there} *prizes
(keep the *expletive* till the end of the sentence)

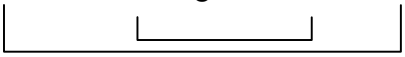
This *non-determinism* (as Fong notes) is present in any instance of *successive cyclic movement*:

(100) *Several prizes* are likely $t_{several\ prizes}$ to be awarded * $t_?$

After dropping the trace with $t_{several\ prizes}$, the *move box* is emptied because of (97.d). So we would need a more efficient way of controlling the lifespan of the elements in the *move box*.

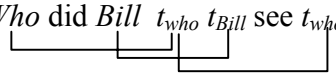
Moreover, it is possible to have multiple *phrasal movements* in the same sentence:

(101) *Who* was *a book* given $t_{a\ book}$ to t_{who}



Fong deals with this fact postulating *nested move boxes*; as the *HOLD register* in the *ATNs*, this memory has a LIFO structure (*Last In, First Out*): the parser can only see the last inserted element, namely the last created *move box* is the first available filler. But this would not predict the correct result, for instance, in the following case, namely when *movements* overlap:

(102) *Who* did *Bill* t_{who} t_{Bill} see t_{who}



Fong assumes that *overlapping* should require various *move boxes* and crucially a more powerful machinery than *nesting*, given that anytime we have multiple *move boxes* full of potential fillers we would have other choice points, namely a *non-deterministic* processing.

Last device Fong proposes is a *probe box* for encoding the essential *agreement* relation in the parser. As introduced in §2.2.2, *minimalist grammars* require that an *agreement* relation hold when a *probe merges* with a *goal*.
Once *agreement* holds, the *goal* values the unvalued features on the active *probe*. This is the proposed algorithm (*p* = *probe*, *g* = *goal*, *f* = *feature*, *α*, *β* = generic syntactic objects):

(103) **Agree(*p*, *g*)** if
- a. Match(*p*, *g*) holds, then
- b. Value(*p*, *g*) for matching features and
- c. Value(*p*, *g*) for property value(*f*)

(104) **Value(*α*, *β*)** holds if
- a. (Unification) Unify matching ϕ-features values of *α* and *β*
- b. (Assignment) If *α* has the property value(f), then *f* for *β* receives its value from *α*

The *probe box* has to be intended as a special slot of *short-term memory* where active elements (namely elements bearing *uninterpretable features*) are made available for checking purposes. The algorithm that controls this tool is similar to the one used for the *move box*:

(105) **probe box**
- a. (*initial content*) at the beginning of any parse, initialize the *probe box* as empty;

b. (*fill condition*) whenever an head position of an elementary tree is filled from the input with an element bearing one or more *uninterpretable features*, copy this element to the *probe box*;

c. (*unicity condition*) only one *probe* is allowed to stay in the *probe box*: new *probes* found in the input replace old ones

At this point, the parsing algorithm given in (93) has to be redefined as follows:

(93') **parsing algorithm** (refined)

step 1 – given a category *c*, pick an elementary tree headed by *c*

using elements from the *move box* or from the *input*:

step 2 – fill in the specifier *s* (if one is predicted from the elementary tree selected)

step 3 – run Agree(*p*, *s*) if *p* and *s* are non-empty

step 4 – fill in the head *h*

step 5 – run Agree(*p*, *h*) for ϕ-incomplete *h* and *p* non-empty

step 6 – copy *h* to *Probe Box p* if *h* is a *probe*

step 7 – fill in the complement(s) by recursively calling the parsing algorithm with *c′* where *c* has the lexical property of selecting *c′*

We can now see how the *probe box* behaves in the simple case presented in (98), reporting below the new operations required in order to deal with *agreement*:

(98) John saw Mary (assume *saw* → *past* + see)

step	**partial phrase structure built**	**probe-box**	
1	$[_c$ _]	ø	by initial condition
4	$[_c [_T$ J. $[_T$ *past* _]]]	*past*	*past* is a *probe* (it has uninterpretable *ϕ-features*, then copy it to *Probe Box*
6	$[_c [_T$ J. $[_T$ *past* $[_v$ $t_{J.}$ $[_v$ v* _]]]]]	*past*	run Agree(*past*, J.), value(*ϕ-John*, *ϕ*-T), value(T,case-John), v is a *probe*, then copy it to the *probe box* removing *past*
9	$[_c [_T$ J. $[_T$ *past* $[_v$ $t_{J.}$ $[_v$ v* $[_V$ see N]]]]]]	v*	run Agree(v*,Mary), value(*ϕ-Mary*, *ϕ*-v*), value(v*,case-Mary),

As Fong points out, we attain at least two good services from this device:

Phase Impenetrability Condition – a *probe* cannot penetrate into the domain of a lower *probe* since it is ousted by this lower *probe* from the *probe box*;

Valuation of ϕ-incomplete probes – unifying *probes* with *heads* as in (104.a) allows the unvalued *ϕ*-features of these heads to receive features; then if these heads become *probes* (this is the case of the defective T-ϕ) they will bear a complete featural make up so as to satisfy further *agreement* operations as for the following sentence:

(106) we expect there to arrive a man
we T expect [$_T$ there T-$_\phi$ arrive a man] (Agree (T,T$_\phi$))

moreover, this very same reason justifies the stay of a *probe* in the *probe box* also after having valued the *goal* (namely, when it becomes *inactive*): even if it cannot trigger further valuing procedures (e.g. *value(case(x))*), it can still unify its features with the following heads as for T-$_\phi$.

Summarizing, this paring strategy addresses some important issues:

1. it catches the ungrammaticality of the sentence as soon as an unpredicted element enters the computation;
2. it encodes *movement* (and residual properties of the *chains*) using a *memory buffer*, the *move box*;
3. it correctly predicts some case of *structural case assignment*, approximating, moreover, the strong *phase* boundaries using another *memory buffer*, the *probe box*;
4. it displays an *incremental* behavior, namely a *partial parse* is always available at any stage of the processing (this suggests us that some psycholinguistic phenomena could be captured);
5. it exhibits a *locally deterministic computation*, namely no choice points are present, once chosen the correct *category* (then the correct *elementary tree*);

6. it is an *on-line* procedure, that is, the lifespan of an element is restricted to the minimum necessary to be integrated in the structure, then, it is immediately removed unless saved in the *move/probe box*.

Despite these interesting results, the parsing model proposed by Fong is still imperfect from many points of view:

1. it is not clear how to pre-select the correct *category* or how to backtrack from a wrong choice;
2. neither *locality conditions* nor *cyclic movement* is easily catchable by the *move box*;
3. the current implementation of two *memory buffers* within the same model seems to be inefficient from several points of view:
   i. a redundancy seems to be present implementing both a *move box* and a *probe box*, since the first one, in fact, should be able to trigger *agreement* per se (any *A'-movement* is toward a *criterial functional position*, Rizzi 2004, then, by definition of functional position, cf. *FSeq* in §1.1.5, any moved element should *agree* somehow with the landing site *head*);
   ii. populating the *move box* is too easy and quite expensive in terms of memory resources (following the *fill condition* proposed in (97.b), any element is potentially copied in the *move box*; many times this turns out to be unnecessary, as in (98), step 10, where "Mary" is vacuously *moved* to the *move box*);
   iii. the *preference rule* for the *move box* sometimes seems to predict the wrong expectation (e.g. "who did Mary [read the book of t_{who}]?" Vs. "who did Mary [read t_{who}] the book of?" as wrongly predicted by the algorithm).

These inefficiencies will be tackled in the next chapter.

CHAPTER 3

FORMALIZING THE GRAMMATICAL COMPETENCE

According to what has been said in the previous chapters, we can now analyze the essential informational structure of the sentence in order to formalize the notion of *Structural Description*. I propose (§3.1.1) that this can be done simply in terms of *immediate relations* among adjacent elements (a simplified version of the notion of *dominance* and *precedence* presented in §1.3) so as to be accessible both from a *parsing* and from a *generation* perspective (§3.1.2): this will allow us to frame, in a very precise way, two problematic aspects of phrase structure: *ambiguity* (§3.2.1) and *long distance dependencies* (§3.2.2).

The third and fourth part of this chapter will deal with the *structure building operations*: *merge* (§3.3.2) and *move* (§3.3.4) will be redefined from a *top-down* perspective, in order to solve some empirical problems (§3.3.1, §3.3.3) emerging both at the theoretical (§2.2) and at the computational (§2.3) level. Finally, the notion of *phase* will be formalized (§3.4.3) in order to attain a satisfactory complexity reduction both in *parsing* and in *generation* with respect to *ambiguity* (§3.4.1) and *Long Distance Dependencies* (§3.4.2).

3.1 ASPECTS OF THE GRAMMAR TO BE FORMALIZED

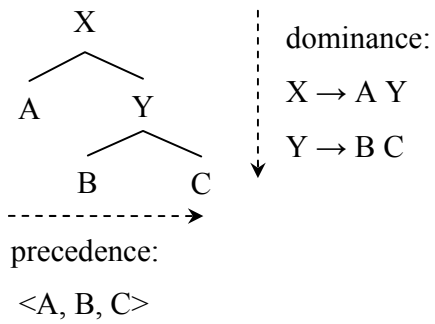
In order to identify the core characteristics that a minimal specification of the grammar should have so to map *features structures* both at the *sensory-motor* interface and at the *conceptual-intentional* one, we should define:

1. the relevant relations among features to be specified and a precise formalization of the notion of *structural description* capturing these properties in a way that should be readable from any *performance* perspective (§3.1.1; this refines the discussion presented in §1.3);
2. the exact specification of the *performance* problems, namely *parsing* and *generation*, which should have access to the grammar (§3.1.2);

3.1.1 THE BIDIMENTIONAL NATURE OF THE GRAMMAR

Considerations about the informational structure of the language explored so far, lead us to think of a sentence as a bidimensional entity bearing information on both *precedence* and *dominance* relations among lexical items, where *precedence* represents a total order among pronounced elements (namely *words*, that are sequences of phonological features) while *dominance* expresses the *constituency*/*dependency* relations among pronounced and other implied (*abstract*) elements (*semantic* and other *abstract features* like phrase identifiers). These two kinds of information have been usually encoded within tree-like structures such as the following one:

(107)



A *language*, as we defined it so far, is an infinite set of grammatical expressions each associated with at least one grammatical structural description, which productively restricts the theoretically possible precedence/dominance relations that can co-occur within the same sentence.

These properties are usually encoded in *Structural Descriptions* defined as 5-tuples such that (Stabler 1997):

(108) $\mathbf{SD_{standard}}$ = {**I**, **P**, **D** ,**V**, **A**} (definition)
where

I is a finite set of *identifiers* (e.g. {the, dog, is, black, DP, D', D°, N° ...})

P is a *precedence* order (a *total strict order*, that is a *binary*, *transitive* and *asymmetric* relation, defined on any element belonging to the subset I_T of I such that I_T is the set of terminal elements; e.g. {the, dog, is, black})

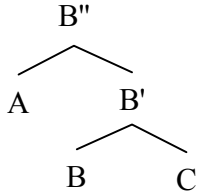
D is a set of *dominance* relations (a *partial strict order*, that is a *binary*, *transitive* and *asymmetric* relation, defined on some elements of I; e.g. "D" dominates "the", N dominated "dog", DP dominates "dog" etc.)

V is a finite set of *vertices* (the nodes in the tree)

A is an *assignment function* from V to I

This formalization allows us to build *tree* structures like the ones presented in (107) or in (109.a) (which is equivalent to the labeled bracketing (109.b); cf. §1.1.4):

(109) a.



b. $[_{B''}$ A $[_{B'}$ B C]]

It seems empirically relevant, given an ordered set of elements <A, B, C>, to distinguish the relation between A and B (*immediate precedence*, that is, no elements intervene between A and B; cf. *cliticization*, Zwicky (1977)) from the relation between A and C (*standard precedence*). The same seems to be true also in terms of dominance: given a dominance structure such as [*verb* [A_1 [A_2]]] (that is, the *verb* dominates A_1, A_1 dominates A_2 and the *verb* dominates A_2) a transitive verb seems to assign the patient thematic role only to the immediately dominated argument (A_1), not to an arbitrarily distant dominated one (e.g. A_2).

Then, leaving aside both V and A (maybe superfluous as it will be shown later on) I wish to redefine the first three elements in such a way that *precedence* and *dominance* only hold between *immediately adjacent* elements:

(110) (definition) $\mathbf{SD_{revisited}} = \{\mathbf{I}, \mathbf{P}, \mathbf{D}\}$ such that

I is a finite set of *identifiers* (let us consider only lexical items such as {the, dog, is, black...}, following the *inclusiveness condition*, cf. §2.2.2)

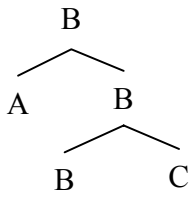
P is a finite set of *immediate precedence* relations (different from the classic *total strict order*, this relation is only defined between two adjacent items)
example (<A, B> means A precedes B):
P = {<A, B>, <B, C>, <C, D>} (simplified in <A, B, C>)

D is a finite set of *immediate dominance* relations (different from the classic *dominance relation*, this one is a *partial*, *binary*, *intransitive* and *asymmetric* relation)
example (A < B, means A dominates B):
D = {A < B, B < C, C < D}
(equivalent to $[_A$ A $[_B$ B $[_C$ C D]]])

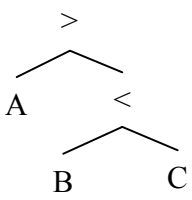
These restricted versions of *P*(recendence) and *D*(ominance), defined only between adjacent elements, will be useful to encode in an extremely transparent way some significant linguistic relations (both *merge* and *long distance dependencies*) as it will be clear in the rest of this chapter. In the next pages I will refer to *precedence* and *dominance* so as to mean *immediate precedence* and *immediate dominance*, unless specified otherwise.

The arboreal representations given in (111.a) and (111.b) are equivalent (adapting Stabler's 1997 convention, < means that the symbol on the edge of the arrow *immediately dominates* the symbol at the base of the arrow, then A < B means A dominates B):

(111) a.



b.



3.1.2 PERFORMANCE TASKS

Recall that the speaker's *competence* is the intensional procedure that characterizes an infinite set of sentences and that, formally speaking, this *competence* is represented by a grammar. The grammar includes, by standard hypothesis[67], at least a specification of a *lexicon* (a finite set of *words* built from an *alphabet* with associated specific *features*) plus some *universal properties* (usually encoded as *principles*) and *language specific options* (*parameters*) to derive the combinatorial potentiality of any given language.

As we saw in §2.1 and in §2.3.1, from this perspective, the specification and ordering of *Structure Building Operations* is controversial: namely, it could be superfluous (or impossible) and even illegitimate (if we consider them as part of the *parser* and not as a part of the *grammar*) to specify some precise algorithm that recursively defines the procedure for assembling bigger and bigger meaningful units, starting from the lexicon

[67] Let us assume by "standard hypothesis" the Principle and Parameter framework presented in Chomsky 1981 and quickly reviewed in §2.1.1.

and its properties. The minimalist framework, for instance, pays serious attention to this point, both providing interesting solutions (§2.2.2, §2.2.3, §2.3.2, §2.3.3) but also raising puzzling problems (as we will see in §3.3). As I pointed out in the previous chapters, I will try speculatively to consider, in line with the *Minimalist Program* (§2.2.2), Phillips' *left-to-right* model (§2.2.3) and Stabler's formalization (§2.3.2), the *Structure Building Operations* as a part of the grammar specification. Then I will explore, in order to understand some properties of these operations, two *performance* tasks from a formal perspective, eventually arguing in favor of the necessity of these operations within a minimal specification of the grammar.

The first task I will consider from this perspective is *parsing* a sentence: this means recovering a set of *dominance* relations from the *precedence* relations holding among a given set of lexical items, minimally in the form $\{\pi, \sigma\}$ where π is a set of phonetic features and σ a set of semantic features. This problem can be stated more formally as follows:

(112) **the parsing problem** (definition)

given a grammar *G*, a finite set of phonological features π (grouped by words) and a precedence total order among them, find the relevant set of lexical items *Lex*, compatible with π and the set of dominance relations *D* among σ features associated to π in *Lex*, if possible, if not reject the input.

The second task is "production", which means *generating* a sequence of lexical items, compatible with a given set of dominance relations defined on a set of semantic features; more precisely:

(113) **the generation problem** (definition)

given a grammar *G*, a finite set of semantic features σ and a finite set of dominance relations *D* among them, find the relevant set of lexical items *Lex* and the correct linearization among π features associated to σ in *Lex*, if possible, if not reject the input.

Let us now try to test these problems on some relevant linguistic phenomena.

3.2 TWO (COMPUTATIONALLY HARD) LINGUISTIC PHENOMENA: AMBIGUITY AND LONG DISTANCE DEPENDENCIES

At least two well known properties of any natural languages are extremely hard to be captured from a computational point of view, even though they apparently do not present any problem for a native speaker: the first property is that language is massively *ambiguous*, the second one is that the linguistic elements seem to be related among them in a way that is not always evident from the linear order in which they appear in the sentence.

While the second property, also know as *Long Distance Dependency*, has always been considered as irreducible core part of the grammar, the *ambiguity* issue has been often relegated to the domain of processing, that is, the "parser" side of the coin. A first step towards a proper understanding of their relation to grammatical knowledge requires first an empirical definition of these aspects (§3.2.1, §3.2.2) that will then lead to their formalization.

3.2.1 AMBIGUITY

Ambiguity arises any time an input can be analyzed in a *non-deterministic* way, that is, given a precise amount of information, the processor can make multiple choices that will lead to different outputs. For instance a word like "dog", taken in isolation, can be either a noun ("the *dog* is in the garden") or a verb ("did you *dog* me?"). Providing a bit more of context (e.g. some words preceding the target one) to the "analyzer" could help solving the problem. Thus *ambiguity* seems to be a property of both the input and the algorithm (based on the grammar) that drive the computation.

Among the levels of *ambiguity* present in any language, we can distinguish at least three macro classes: *lexical*, *syntactic* and *semantic* ambiguities.

A *lexical ambiguity* is present when multiple tags representing a *Part-of-Speech* (*PoS*) can be associated to a single word in a sentence:

(114) a. I read the [$_{\text{Noun}}$ *book*]
b. Did you [$_{\text{Verb}}$ *book*] the flight?

On the other hand, we have a *syntactic ambiguity* when more than one Structural Description for a given sentence is possible:

(115) a. Please, [put [the block] [in the box] [on the table]]
a'. Please, [put [[the block] in the box] [on the table]]
a". Please, [put [[the block] [in the box [on the table]]
b. [[Red roses] and [cherries]]
b'. [Red [[roses] and [cherries]]]

A third possibility is having multiple senses associated with the same word (same *PoS* and same π features), this is a case of *semantic ambiguity*:

(116) a. Tonight the *moon* is full (*moon* = the earth's unique satellite)
b. Jupiter has sixteen *moons* (*moon* = orbiting object)

Given a SD such as the one explored in (110), *lexical ambiguity* is expressed in terms of *abstract* (*categorial*) *features* opposition, so that it is invisible to *precedence* and perhaps even to *dominance*. This is predictable, since these two relations were defined respectively only on π features (*precedence*) and on σ features (*dominance*).

On the other hand, *syntactic ambiguity* is identified by the existence of more than one *dominance* set of relations associated to the same *precedence* group of relations.

Finally, *semantic ambiguity*, is not marked by different *precedence/dominance* combinations, but rather by different sets of σ features associated to the same lexical item.

3.2.2 TYPOLOGIES OF LONG DISTANCE DEPENDENCIES

As we have seen in §1.3, among the relations we could define on elements within a SD, other than *dominance* and *precedence*, only few are linguistically relevant:

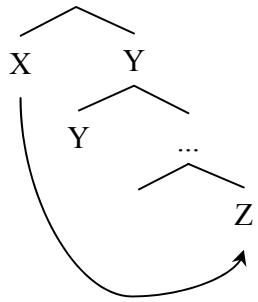
(117) **constituent formation**
a constituent is formed when two adjacent element are concatenated together (e.g. *B concatenated* with *C* forms the constituent *B′* in (109))

(118) **C(onstituent)-command** (Chomsky 2000)
A *C-commands* B iff B is contained in the sister of A
(e.g. A *C-Commands* B in (109))

Note that the relation in (117) is indeed perfectly equivalent to the *dominance* relation defined in (110). (118), on the other hand, can express a relation between arbitrarily distant elements and it is neither directly encoded within the *precedence* nor within the *dominance* definition, even though, in principle, it could be captured by adding *transitivity* to the dominance definition (§1.3), but this is not exactly what we would need empirically (see §3.3).

So to speak, any relation between two non-adjacent elements in a SD can be considered as an instance of *long distance relation* (this includes *C-command* as well). More precisely, this configuration should hold:

(119)



where X and Z are *identical* in some fundamental properties (e.g. formal and/or σ features) and Y intervenes between X and Z without blocking the relation as shown below:

(120) a. $[_X$ John $]_i$ $[_Y$ gave $]$ $[_Z$ his $]_i$ picture to Mary (*pronominal binding*)

b. $[_X$ What $]_i$ $[_Y$ do you think $[_Z$ _ $]_i$ $]$? (*movement*)

c. John $[_X$ bought the book$]_i$ $[_Y$ and Mary did too $[_Z$ _ $]_i$ $]$ (*ellipsis*)

The blocking nature of the Y category has been precisely defined in §2.2.4.

The *identity* relation between X and Z could be considered *directional*, in the sense that the structurally "higher" element provides the feature values for the lower (invisible/incomplete) one (this justifies the direction of the arrow). Notice that the nature of X, Y and Z can vary to some degree from a *full phrase*, as in (120.c), to a *simple head*, to a *pronominal form* (120.a), to an *unpronounced element* (120.b-c).

Given the formalization of SD in (110), we can define *discontinuous constituency dependencies* as follows:

(121) **Long Distance Relation** (definition)
two *non-empty*[68] elements enter a *long distance relation* (thus forming a *discontinuous constituency relation*) when a *dominance* relation but no *precedence* relation is defined between them.

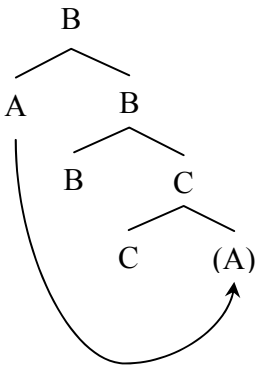
For instance, given the grammar in (122), A and C are subject to a *Long Distance Relation* since the dominance relation C<A exists but neither <A,C> nor <C,A> is present:

(122) I {A, B, C}
P {<A, B>, <B, C>}
D {B < A, B < C, C < A}

This can be represented by the tree in (123):

[68] Where *non-empty* means with at least some feature specified (either *formal* or σ features).

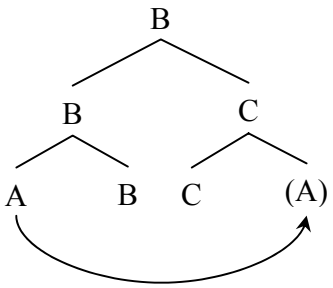
(123)



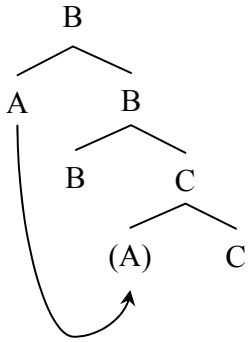
The *Long Distance Relation* in (122)-(123) is essentially an instance of *movement* (as discussed in §2.2), however it should be possible to extend this approach even to *pronominal binding* (along the lines of Kayne 2002) and to *control* (Hornstein 1999, Boeckx and Hornstein 2004). The relation between A and (A) is clearly identical to the relation between a *moved element* and its *trace* (again, the directionality of the arrow indicates that the highest element provides feature values for interpreting underspecified features of the lowest one).

Note that the information in (122) is ambiguous between the *right-branching* structure given in (123) and more complex structures such as the ones shown below:

(124) a.



b.



Without using standard SD devices (essentially the *assignment function* presented in (108)), we can rule out the unwanted representations by posing extra 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, would suggest a *deterministic* mapping between *asymmetric C-command* and *precedence* depending on the structural

function of the items involved in the SD. In a similar vein, I will adopt the following principle:

(125) **Linearization Principle** (inspired by **LCA**, Kayne 1994)
if A < 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*[69] of A

By now we do not need to discard (124.b) since within this formalism the linearization of a trace (π-features free element) is irrelevant[70]. Then let us tentatively assume that (124.b) and (123) are equivalent.

We can also discard (124.a) under the relatively standard constraint (126.a) on *movement*:

(126) **constraints on movement** (definition)
a. a moved element always *C-commands* its trace(s)

We are now ready to consider how these properties can be incorporated in a system that matches the (empirical) requirements of *flexibility* and *token transparency* discussed in §2.1.

[69] Assume *functional projection* to be synonyms of *specifier* following Starke 2001and §1.1.5.
[70] But see Nunes 2004 for a different perspective.

3.3 EMPIRICAL INADEQUACIES: RE-DEFINING *MERGE* AND *MOVE*

The Minimalist framework raises interesting questions on the nature/power of the grammatical computational engine, the core part of the human linguistic *competence* apt to build phrase structures. However, to what extent the proposed abstract grammar formalization is usable both in *parsing* and in *generation* remains largely an unexplored topic. On the other hand it is plausible, from a cognitive perspective, that the grammar we use to produce sentences is somehow used even to comprehend them (that is the *flexibility* property discussed in the introduction and in §2.1).

From this point of view, if we choose to formalize the *structure building operations* in a way that is compatible both with the *parsing* and with the *generation* problem presented in §3.1.2, we do not have any clear way to incorporate in our grammar operations like *movement* or *merge* that delete information or that apply *from-bottom-to-top*.

On the contrary, from a *top-down* perspective there are many viable options: in *generation*, for instance, given the set of *dominance* relations among lexical items[71] (127.b), it would be possible to recover (127.a) and then produce (127.c)[72], for instance, using the mechanical procedure outlined in (128):

---

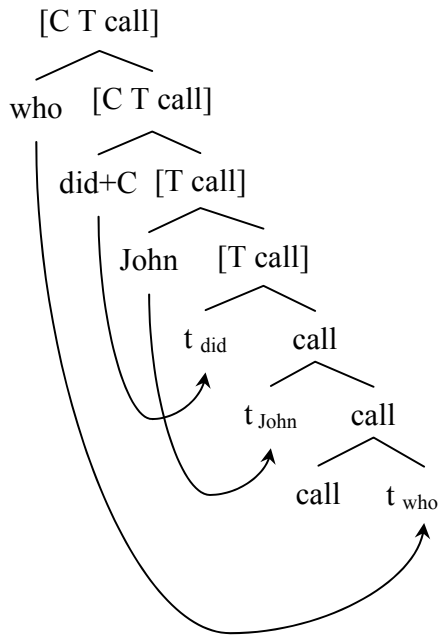
[71] More precisely, by the definition in (113), any *dominance* relation is among *semantic* (σ) *features* present in any lexical item (then, indirectly, among *formal features* too). For clarity, simple labels such as "who" stand for the whole set of σ features of the lexical items. In a parallel way the P set, following (112), expresses relations among π features, represented by the label of the lexical item (e.g. "who" stands for /wu/). Remember that the dominance relations "call < John" and "call < who" do not actually involve the very same "call" element: this is because the *direct argument*, in this case "who", has to be *merged* with "call" before "John": we could represent this asymmetry, necessary to distinguish the *subject* from the *object* relation, either rewriting the "call < John" relation as "[call who] < John" or providing the complete feature structure of call (then specifying the saturated status of "call" with respect to the *direct argument*), but that would have been hardly readable on a tree-like structure.

[72] The same clarification about notation reported in the note before is probably necessary to understand this tree-structure: e.g. the dominance relation "did < John" has been graphically presented as "call<John"; this is just a tree simplification: the actual structure should have been "[did call] < John", that is "[$_{call}$ T call] < John". The *dominance* relations reported simply express the relevant relation between the *functional elements* and the *dominated object*; the relation between these *functional elements* and their *heads* is trivially derivable from the grammar. The head-incorporation "did+C" would require an independent discussion (e.g Stabler's incorporation, =X/X=, should be adapted to FSeq).

(127) a. P = { <who, did>,
<did, John>,
<John, call> }

b. D = { C < who, C < did,
did < John, call < did,
call < C, call < John,
call < who }

c.



(128) **generation algorithm** (first sketch[73] of a *top-down* algorithm taking D as input and producing P and SD as output):

1. find the highest head in a given D set and put it in the rightmost position in SD (following the algorithm given in Appendix, (173)); (e.g. "C");
2. project the highest relation (R) headed by the head selected in 1 on SD, substituting this head in SD (e.g. "[call did + C call] < who"; SD = "[call did + C call] < who");
3. if the lexical elements do not dominate anything (they are *terminals* in the standard terminology) then select phonetic features π matching a single item in *Lex*[74] and put them in P according to (125) (e.g. P = <who, ...>), unless they are already linearized;
4. remove the projected relation from the D set (e.g. D = { "~~C < who~~", call < did, ... });
5. restart from 1.

[73] The procedure for controlling trace insertions is roughly outlined just to have a glimpse of the general idea: in fact, in order to verify whether or not an element has been already linearized we should postulate a memory register.

[74] In case of *ambiguity* (the relation from σ to π is a relation from one to many) use heuristics such as those proposed within the Distributed Morphology framework: e.g. select first more specific *Lex*(ical item)s.

This solution is possible, because we have all the *merge* operations already specified within the D set. We only need a simple rule that deletes phonological features from the lowest copies of the elements in the tree. In this sense, the choice of a *top-down* algorithm cause a *deterministic* behavior, provided that we could store in a memory register (cf. §2.3.3, §3.3.4) the linearized items in order to insert *traces* directly instead of full items when a dominance involving an item already linearized is inspected: "John", for instance, is present both in "did < John" and in "call < John"; following (128.1) "did < John" dominates "call < John". John has to be linearized as soon as "did < John" is inspected (since "John" is *terminal*, (128.3)). Inspecting the second relation, "call < John", we should know that John has already been spelled out, then we can insert a trace instead of the full lexical item. A *bottom-to-top* algorithm could produce the same P set, but it should backtrack from the choice to linearize <call, John> (from the relation "call < John"), when, later on, it will inspect the highest dominance relation "did < John" (that combined with "did<call" would produce <John, ..., call, John>).

Note that this algorithm does not directly refer to any notion of *structure building*, given that all the information is already implicitly encoded within the D set (that, this way, expresses any *merge* point). How the D set is built is another problem, to be explore maybe from different perspectives[75]. On the other hand, problems arise from the *parsing* perspective: crucially, we have less precedence relations than what we would need in order to recover the whole set of relevant dominance relations. The set P in (127.a) determines a linearization between elements (for instance *who* and *did*) that do not directly enter any simple dominance relation, since an empty head (C, in the previous case) mediates the relations between these elements. How this empty head can be introduced within the SD in parsing, starting from Stabler's formalization (or Chomsky's *minimalist grammar*) is not clear at all[76]. The same can be said about the lower copies of *moved* elements (e.g. *who*), invisible to any linear relation in P, but crucially present in D. The next pages will try to tackle these problems.

---

[75] Allegedly, this is not part of the grammar specification, as the procedure leading to the *Numeration* was not (for the time being, we could pin this problem on the *Conceptual-Intentional System*).
[76] The *numeration* idea, namely the pre-process of selection of the whole set of items will enter the computation, is clearly not viable in parsing.

3.3.1 CARTOGRAPHY AND EXTENDED PROJECTIONS

One possible way to avoid problems with the arbitrary introduction of empty functional elements within the phrase structure, would be to import into Stabler's formalization a stipulation on the universal order of functional projections (in the sense of §2.2.4). This specification can easily become part of Stabler's formalism by imposing an order on the features licensed by these functional heads (this order is both *linear* and *hierarchical* given the *linearization principle* in (125)). The *licensors* subset (§2.3.2) should be defined as an ordered set with the following shape (see §4.1 for the whole licensor set used):

(129) *licensors:*

CP^{77} = < Force, Top*, Int, Top*, Focus, Mod*, Top*, Fin>

IP^{78} = <Mood $_{speech\ act}$, Mood $_{evaluative}$ ... T $_{past}$, T $_{future}$, ... Asp $_{continuative}$ ...>

DP^{79} = <D, ordinal, cardinal, ... size, length ... colour, nationality, material>

These features (that in fact are *categorial features*) can be "selected" but can not be included as simple categories in the *base*/*select* subset; such a *move* would cause easily detectable troubles (note that, in the following examples, the star indicates the ungrammaticality of the sentences only with respect to the given grammar, in fact they are perfectly fine in English):

(130) problematic lexicon fragment where *functional features* are included in the *base*/*select* subcategories:

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

parsable/generable sentence:

a. Probably suddenly Napoleon died

[77] Based on Rizzi 1997, 2002.
[78] Based on Cinque 1997.
[79] Based on Scott 1998.

but not:

b. *Probably Napoleon died

here the derivation fails when we try to *merge* the V complex [$_{V}$ Napoleon died] with [=Asp Probably]:

1. merge ([N Napoleon], [N= V died]) → [$_{died}$ [Napoleon] V died]]
2. *merge([=Asp Mood Probably] [$_{died}$ [Napoleon] V died]]) →
   undef. (=Asp cannot be satisfied)

c. *Probably Napoleon died suddenly

changing the =Asp requirement of [Probably] to =V would not solve the problem;

1. merge ([N Napoleon], [N= V died]) → [$_{died}$ [Napoleon] V died]]
2. merge([=V Mood Probably], [$_{died}$ [Napoleon] V died]]) →
   [$_{probably}$ probably [$_{died}$ [Napoleon] V died]]]
3. *merge ([$_{probably}$ probably [$_{died}$ [Napoleon] V died]]], [V= Asp suddenly]) → *undef.*
   at this point the derivation fails because only one selectional =V feature can be satisfied by the complex [$_{died}$ [Napoleon] V died] (it can only bear one *categorial* V *feature*)
   (if [V= suddenly] were *merged* before, then [=V Probably] would not be satisfied)

d. *Napoleon died

changing the "died" entry in this way [N= Mood= V died] would create either an unjustified asymmetry in adverbial selection (why only Mood= and not even =Asp?) or an incorrect prediction on the grammaticality of sentences without adverbials:

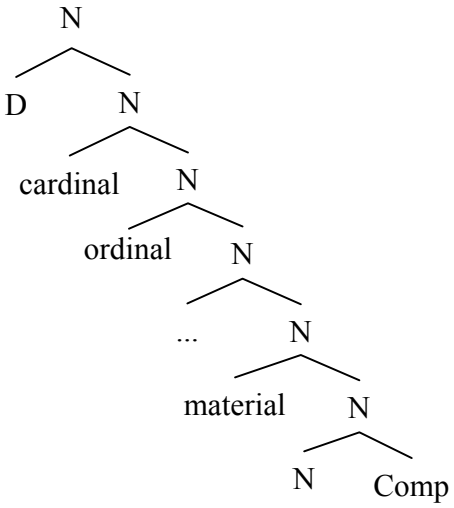
1. merge ([N Napoleon], [N= Mood= V died]) → [$_{died}$ [Napoleon] Mood= V died]]
2. *[$_{died}$ [Napoleon] Mood= V died] (Mood= is not satisfied)

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

(131) Probably → [=T_{past} Mood Probably], [=T_{future} Mood Probably], [=$Asp_{completive}$ Mood Probably] [=V Mood Probably], [=T_{past} =$Asp_{completive}$ Mood Probably], [=T_{past} =$Asp_{completive}$ =V Mood Probably] ...

But this solution would produce *n!* lexical entries for *n* possible selecting features. This would introduce a high level of unwanted *ambiguity*; therefore, I will pursue the option of considering functional elements as *licensors*. Then, the categories in the *base*/*select* subsets should be only *V* and *N*[80]. This has at least a theoretical consequence: since any verb should select as a complement either another verb or a noun (the only features still in the *base*/*select* class), these features have to be maximal projections, then, for instance, any functional projection (representing a single licensor feature) above a lexical head should be dominated by this head. This entails the structure (132), namely some sort of *Extended Projection* along the lines of Grimshaw (1991):

(132)



The assumption, however, requires a slight redefinition of the notion of *merge*.

[80] I will not consider in these pages any possible subcategorization either of V or of N; this option could however be pursued following Niyogi's suggestions (Niyogi 2001). Moreover, adverbial and adjectival items should be part of the licensors class, respectively within the V and the N domain. The same could be said for complementizers and prepositions. This (necessary) stipulation can have strong empirical implications that have to be evaluated yet.

3.3.2 MERGE

The simple idea of linking the *merge* operation to a single selecting mechanism (=f) has been shown to be empirically problematic if we assume that any selecting/selected pair of features has to be deleted for convergence reasons (see (130)). Introducing a distinction between *functional* and *lexical features*, however, does not solve the problem: in fact, there are functional elements that are optionally selected (for instance, adverbs) while others seem to be obligatory (e.g. determiners in some contexts/languages). This asymmetry can be captured, within the Extended Projection assumption, if we slightly modify the *merge* operation by introducing some sort of subcategorization: the *merge* operation should be able to "inspect" the *licensor* features within the lexical items (or well formed phrases) and to use them for selection purposes. This way, the *select* subclass is not just the mirror of *base*, but a set built from {*base* ∪ *licensors*}. For instance, =[d n] would select a noun phrase where the licensor *d* feature is satisfied either via *movement* (as for proper names, Longobardi 1994) or by determiner insertion, as shown below[81]:

(133) **Lex** = [d n John], [=[d n] =[d n] t v called], [d the], [n police]

1. merge ([d the], [n police]) → [$_{\text{police}}$ d n the police]
2. merge ([$_{\text{called}}$ =[d n] =[d n] t v called], [$_{\text{police}}$ d n the police]) →
   [$_{\text{called}}$ =[d n] t v called [$_{\text{police}}$ the police]]
3. merge ([$_{\text{called}}$ =[d n] t v called [$_{\text{police}}$ the police]], [d n John]) →
   [$_{\text{called}}$ [$_{\text{called}}$ t v called [$_{\text{police}}$ the police]] [John]]

Optional functional elements would not be selected by other elements, but simply *merged* within the relevant functional projections in compliance with the hierarchy in (129): for instance, *merge 1* in (133) is not triggered by any *=f* feature, but simply by the compatibility between the *functional feature* D and the *lexical feature* N given (129)[82].

---

[81] *Subject movement* is omitted in the example, just because it is irrelevant for understanding the *merge* operation.

[82] The (apparent) selection of D for N will be accounted for in terms of *top-down expectation*; see §3.4.3 and §4 for details. Note that the *functional* hierarchy proposed in (129) should be reinterpreted in terms

Another aspect of the *merge* operation we need to explore resides in its derivational nature: there are reasons to believe (§2.2.3) that, both in *parsing* and in *generation*, processing has to be somehow *incremental*, that is, the information is not assembled all at once, but piecemeal, following a temporal sequence. If we would seriously consider the *merge* operation from this *derivational* perspective, we should verify at least three possible hypotheses:

(134) a. the order of the *merge* operations is essential to derive the right structure;
b. *merge* is a binary function;
c. the order of the elements entering a *merge* operation is irrelevant: *merge*(A,B) ≡ *merge*(B,A);

(134.a) is easily provable assuming that at any application of *merge* a new constituent is built (maybe transitorily, in the sense of Phillips 1996). Using the example in (133), if the elements entering the operations 2 and 3 were inverted we would obtain the wrong *constituency* structure [[[John] called] [the police]] with the object *C-commanding* the subject.

(134.b) can be postulated either to get unambiguous structures (*paths*, in Kayne 1984) or by showing that any *n*-ary constituent (with *n* bigger than two) has indeed a binary equivalent (this is, basically, the idea behind *Chomsky's Normal Form*, Chomsky 1963). Nonetheless, I would like to suggest that binary *constituency* directly follows from the temporal order regulating *merge* operations (as suggested in §1.2.1): any time, a (unique) piece of information *merges* with a previously built (unique) constituent. This simply results in a binary function (e.g. A is the previously built constituent; B is a newly introduced piece of information that *merges* with A forming C; then D is newly introduced and it requires to be *merged* with C forming E... and so on[83]).

---

of the relative *lexical* heads, then DP licensors are *functional specification* of the N head, while both CP and IP are *functional specification* of V.

[83] As I have speculatively proposed in §1.2.1, this could be related to the fact that a unique neurological pathway is apt to process these features.

(134.c), namely the fact that *precedence* is irrelevant in *merge*, directly derives from the definition of this operation: provided that the lexical items are marked for selection and that selection is not directional[84], *merge*(A,B) should give the very same result as *merge*(B,A).

Let us now consider some success conditions on *merge*:

(135) *il gatti (Italian)

the $_{sing}$ cats $_{pl}$

The agreement requirement between nouns and determiners should help us defining this "simple" function that "takes two elements α, β already constructed and creates a new one consisting of the two" (Chomsky 2001:6). The first step that is missing in Stabler's formalism (and in Chomsky 1995 too) is a checking before merging, which, in fact, seems to be present in many languages. As mentioned in §1.2.1, I would like to suggest that the *merge* operation should be considered, indeed, as a sort of *unification* on feature structures[85]. Then mismatch in feature values would prevent the *merge* operation from succeeding in (135).

From this perspective a slightly more complex, even though empirically more adequate, information-combining operation among feature structures (similar to the *unification algorithm* used for instance in HPSG, cf. §1.2.1) is required to deal with these simple facts; the following formalization is sufficient for our purposes:

(136) **merge** (definition)

given a grammar **G** = {**V**, **Cat**, **Lex**, **F**},

given two non-null features structures α and β such that either α and $\beta \in$ ***Lex***

or α and β are built from **Lex** X **F**, and both are in the form [*licensors** *base*$^{\{0,1\}}$ *select** π σ],

[84] Incorporation cases proposed by Stabler with the capital features =X, X= could be accounted for by phonological properties and easily derivable by forcing *movement* without such a stipulation on directionality; Moreover the Linearization Principle, that would produce at any *merge* an ordered pair, is an independent constraint not a *merge* property.

[85] Up to now, the expressions in *Lex* are "flat" just because we do not need any more complex devices to describes the basic operations within this grammar formalism. A translation of these expressions in complete features structures such as Attribute-Value Matrices presented in §1.1.5 is however possible.

assuming α and β to be *lexical* if in their feature structure *base* is not null

assuming α and β to be *functional* if in their feature structure *base* is null and *licensors* contains at least one feature

merge(α, β) produces a unified feature structure γ such that:

- if α is *lexical* and β is *functional,* and β bears a *functional feature* not present in α but in the functional domain of α and all φ-features present in α and β are compatible (either underspecified or matching in values), then:
  $\gamma = [_{\alpha}\ \{licensors_{\alpha} \cup licensors_{\beta}\}\ base_{\alpha}\ select_{\alpha}\ \{\alpha, \beta\}\ ]$
- if both α and β are lexical and α selects β, then:
  $\gamma = [licensors_{\alpha}\ base_{\alpha}\ \neg\{select_{\alpha} \cap \{licensors_{\beta} \cup base_{\beta}\}\}\ select_{\beta}\ \{\alpha, \beta\}\ ]$
- if both α and β are lexical and β is compatible with the (underspecified) head [α H], then:
  $\gamma = [licensors_{\alpha}\ base_{\alpha}\ select_{\beta}\ \{\alpha, \beta\}\ ]$
- if both α and β are *functional* and belonging to the same functional projection of a lexical head *H* (even if underspecified for *licensors*, π and σ), if φ-features are compatible (either underspecified or matching in values), then:
  $\gamma = [_{H}\ \{licensors_{\alpha} \cup licensors_{\beta}\}\ \{\alpha, \beta\}\ H\ ]$
- any other case is undefined

Note that *merge* is defined only between elements that are not null. An element that *merges* with a null one trivially returns itself.

Two simple facts are worth noting before continuing:

1. the fact that the order of the merging elements is irrelevant for the result of the operation itself does not imply that the result is an unordered set of elements; in fact, if α is a pre-constructed element, namely an element built by previous *merge* applications and β a newly processed lexical item, this temporal sequence

results in an ordered set <α, β>[86]. This order could correspond not exactly to the one defined in P, but it would be highly desirable to find out a *deterministic* mapping between the two (both *LCA* and *Linearization Principle* go in this direction).

2. this formalization indirectly contains both a *agree* operation (somehow similar to the one proposed in Chomsky 2001:12 even though "local") and a *valuing* one (Fong 2004): *agree* is represented by the checking requirement between the φ-features present in the functional items and in their relative lexical (projecting) head[87]; *valuing* is indeed an unnecessary operation, given that the *unification algorithm* does the job of filling up the underspecified values of compatible features.

Summarizing I proposed that the *merge* operation can be formalized as a *binary* relation between two *features structures* that have to be *unified*; moreover, it is sensitive to *temporal order*.

3.3.3 DIRECTIONALITY OF MOVEMENT AND RELATIVIZED MINIMALITY

Let us now focus on an analysis of *licensees* and *licensors*. These symmetric classes formed by the very same set of features are parallel to the interpretable/uninterpretable distinction which Chomsky (1995) introduces to justify the fact that *uninterpretable features* have to be deleted before reaching *interface levels* (the same idea lies behind the *probe-goal* approach, Chomsky 2001). However, as Rizzi (2004) suggests, the only thing we minimally need to identify a *goal* (the element to be *moved* in a *probe* headed position) would be the feature itself; then we could get rid of the distinction *–f* / *+f*, replacing them just by *f*. Pushing forward this idea, we actually do not need the lexical item to be marked by the feature *f* in the lexicon in order to be found/selected by the

[86] This would be a simple version of the notion of "*cyclic spell-out*" even though completely unrelated to Chomsky's proposal.

[87] This is a way to keep into account the idea that the "spec-head" relation is the locus of agreement: we should look at the functional projections as instances of multiple (ordered) specifiers of the lexical head, rather than as alternative to the adjunction. Note that this notion of *agreement* differ significantly from Chomsky's 2001 *Agree* since this one does not entail any long distance relation.

probe to trigger *movement* (besides, this would cause lexical ambiguities that would raise problems like the ones illustrated in (130)).

The idea I will pursue is that the licensors hierarchy, per se, "triggers *movement*", in the sense a lexical element has to occupy the relevant *functional projection* in order to receive the related licensor feature (either it "lands" or it is generated there; note that, from this perspective, this distinction is invisible to any *precedence* relation).

Within the proposed formalism, a *top-down* (or *left-to-right*) algorithm does not need to use the *–f* / *+f* (uninterpretable/interpretable) technology, since the first position that is processed is the "optional" (functional) one, while the lowest position of the chain (to be "reconstructed") is the *selected* position; then the fact that the lexical item presents unselected features *f* will directly produce the "expectation" for this position. This way we could prevent the element from being overtly marked in the lexicon with the feature that will be added during the phrase structure building.

Another empirical problem at issue is how to capture *relativized minimality* effects: Stabler's definition of *movement*, provided in §2.3.2, is not sufficient to predict the correct constraints; remember that locality constraints in *movement* are not just a matter of identity of feature, but rather of classes of features (§2.2.4).

All elements belonging to a specific *intervener class* are potentially deadly *interveners* for *movement* or *chain* formation between elements belonging to the same class.
I will adopt (following Rizzi 2004, §2.2.4) a richer categorization than the classical A' class:

a. Topic $\{_{\text{topic}}\}$

b. Q $\{_{\text{Wh, Neg, measure, focus, ...}}\}$

c. Mod(ifier) $\{_{\text{evaluative, epistemic, Neg, frequentative, celerative, measure, ...}}\}$

Argumental features represent another distinct class of *interveners* (*A-chains* or *A-movement*):

(137) A(rgumental) $\{_{\text{person, number, gender, case}}\}$

These assumptions require a minimal refinement of the **Cat** set:

(138) modification of **Cat** (set of syntactic features) required by locality constraints on *movement*:

Cat = (*base* ∪ *select* ∪ *licensors* ∪ *interveners*) where

base are the categories { *verb*, *noun* },

select specify one of the three postulated kind of selection {=*x*, =*X*, *X*= | x ∈ *base*} where =x means simple selection of an x phrase,
=X selects an X phrase, suffixing the selecting head with the phonetic features of the selected X phrase;
X= selects an X phrase, prefixing the selecting head with the phonetic features of the selected noun phrase;

licensors are *functional features* organized in hierarchies of the following kind:
V = < Force, Top, Foc, ... C, $Mood_{speech\_act}$, $Mood_{evaluative}$, ... T_{past}, ... $Asp_{completive}$>,
N = <D, ordinal, cardinal, ... size, length, height ... colour, nationality, material>

interveners are classes of features hierarchically organized as follow:
A(rgumental)
Topic $\{_{topic}\}$
Q $\{_{Wh, Neg, measure, focus, ...}\}$
Mod(ifier) $\{_{evaluative, epistemic, Neg, frequentative, celerative, measure, ...}\}$

Note that *licensors* and *interveners* are composed by the very same set of feature, but, crucially, the set structure is completely different: *licensors* is an ordered pair of sets, while *interveners* are intersecting unordered partitions of the same set. It would be interesting to ask if this *hierarchy* or the *interveners classes* are somehow derivable from the internal properties of the features (and maybe if they are unifiable), but this is largely beyond the limits of this dissertation.

3.3.4 MOVE

Going back to Stabler's formalization, he notes a potential problem in his formalism related to the deletion of the *licensee* features when the element bringing them is subject to *successive cyclic movement*:

(139) John seems t_{John} to be t_{John} happy

In (139) the selected features on John (either [d n] or simply [d]) have to survive after *merge*. The proposed solution, namely that these features can optionally be deleted is neither satisfactory (it would cause non-determinism) nor explanatory (no generalization could be captured).

Another related inefficiency is directly associated to the mechanism driving *movement* that deletes features during the derivation by pairing *-f/+f* features: as we speculated before (§3.3.3), there are reasons to believe that this is not the most minimal hypothesis, then implausible as part of the minimal specification of the grammar.

In order to solve both problems, I would suggest that we should invert the directionality of the *move* operation: since both in *parsing* and in *generation* there are reasons to believe that structure building applies from left to right (*garden paths* in comprehension, *false starts* in production) let us seriously consider what the formal implications of this cognitively motivated preference of processing could be.

As informally anticipated in §3.3.3, processing a sentence from left to right would first bring to the hearer/speaker attention the element in its highest landing site position. In order to exactly determine this position the hearer should recover the potential *licensor* possessed by the item, either by inspecting morphological markers (that is the *word shape*, Bresnan 2001, e.g. case features and maybe prosodic cues) or/and inferring it from the adjacent elements (minimally by inspecting the next item). Once again, there are specific cases to consider:

1. when a *lexical* item *L* is found, then one of the following statement should hold:
   a. *L* is *selected* (given the *Linearization Principle* in (125)), a *selected* element can only be to the right of the *selecting head*;

b. *L* is not *selected*, then it must have "moved from" a lower position where it was *selected*

2. when a *functional* item *F* is found, then one of the following statement should hold:
   a. *F* is in its appropriate *functional* position (given the *Linearization Principle* in (125), then it is to the left of the selecting head, and the surrounding elements are compatible with the position of the element in one of the hierarchies proposed in (138);
   b. *F* is not in an appropriate position, then it has been "moved from" a lower position where it was legitimated (that is a particular form of *selection*) by the hierarchy proposed in (138).

More formally, we can define *move* as follows:

(140) **move** (definition)
given a grammar **G** = {**V**, **Cat**, **Lex**, **F**},
given a not null features structures α, such that either $\alpha \in$ ***Lex***
or α is built from **Lex** X **F**, in the form [*licensors* base*$^{\{0,1\}}$ *select** π σ],
assuming *M* to be a multidimensional memory buffer (as many dimensions as the *interveners* classes in *Cat*)
if α is not *selected*, then
move(α) = push the element α into *M[intervener_slot]* (where *intervener_slot* is chosen depending on α[*licensors* base*$^{\{0,1\}}$])

This definition of *movement* formalizes the idea that a "moved" element "stands in memory" as far as it is used to fill (interpret) the base position where it is selected (cf. *active pattern* proposed in §1.2.2); moreover, the memory structure is a sort of *stack* (*Last In First Out memory*) which can capture nested dependencies[88].

Note that the *intervener* slot where to put the moved element is determined by the features present on the element and by the structure of the *interveners* class in *Cat*. If a

[88] This kind of structure of memory is clearly not suitable for capturing cross-serial dependencies, but see §4.3.5 for details on how it could be possible to capture these structures.

feature belongs to more than one class (e.g. *negation*), the element is "virtually" copied in any relevant memory slot (e.g. [Q] and [Mod]): be virtually copied means that, in fact, all the copies behave as just one element, then any operation affecting an element in any slot, indeed, would affect all copies of this element.

To complete this device[89] we need at least two more conditions: a condition that triggers the *dropping* of the relevant element stored within the *M* register in the right selected position and a *success condition* that determines the final state of the *M register* in order for the derivation to converge.

(141) **dropping condition** (definition)

when a *lexical* element β in the form [$_{\beta}$ *licensors* base =x select** β] is introduced in the computation and the following conditions hold:

- $=x$ is the first of the *select* features of β that has to be satisfied,
- α is the last inserted element in the relevant *M* slot,
- α is of the form [α x ... α] where x is built from {*licensors* $\cup$ *base*}

then:

i. *merge* (β, α);

ii. remove α from *M* if i. is successful (unless β has some special feature that prevents this elimination from happen[90]);

(142) **success condition** (definition)

a sentence is grammatical only if at the end of the processing *M* is empty.

We should now ask ourselves if proceeding "from left to right" is plausible in *production*. I think the answer is yes if we interpret the *left-to-right* directionality of the derivation as an effect of a *top-down* processing: given the hierarchy in (129), *functional features dominate* each other in a very precise way then, the *directionality* of the process can be (*deterministically*) established even on a set of *dominance* relations.

[89] Somehow reminiscent of the *hold* register of the ATN, Woods 1970, its modifications apt to capture locality effects, as proposed in Merlo 1996 or the unification of *move/probe boxes* in Fong 2004.
[90] E.g. [*raising*] feature.

In order to account for cross-linguistic variation in terms of complements position, we could either assume a *parametric* option (*head-complement* parameter) or Kayne's approach (a *parametric variation* forces *complements* to "incorporate" to some *functional* position higher up in the structure in some languages but not in others).
Moreover, *false starts* and other *disfluencies* in speech could show the plausibility of this model from a cognitive perspective[91].

Nevertheless, an open problem still stands and it is related to the difficulty of producing very "large" *top-down expectations*:

(143) Who do you think Mary believes John saw?

In (143), for instance, the linearization of the elements would require a full *top-down expectation* for all the three sentences that will be produced (<you think>, <Mary believes>, <John saw who>); that could be cognitively (and computationally) implausible. The next section will tackle this problem using the notion of *phase*.

[91] Even though, as far as I know this has never been seriously tested.

3.4 COMPLEXITY ISSUES

As I mentioned in §1.4, the *complexity of a problem* is an expression of the resources (essentially *time* and *memory*) needed to solve the problem. More precisely, it is a function of the size of the problem, determined at least by three factors:

(144) **complexity factors**:

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

while (144.a) is largely independent from the grammar, (144.b) and (144.c) are strictly determined by it. From a *Minimalist* perspective, (144.b) is mostly determined by the *lexicon* and by the *structure building operations merge* and *move*; (144.c) is usually assumed to be a *bottom-to-top* algorithm (namely an algorithm that starts building structures from the inner most verbal shell, adding piecemeal more peripheral elements[92]) plus some economy conditions (*shortest move*/*attract closer*, *earliness* Vs. *procrastinate*, *merge preempts move* etc.).

The problems we wish to evaluate within this framework in terms of complexity are obviously the *parsing* and the *generation* ones (§3.1.2). Both problems have two distinct components to be explored: a part of *lexical ambiguity resolution* ("find the relevant set of lexical items *Lex*" compatible with π/σ) and a part of *structural mapping* (from *precedence* to *dominance* and vice versa). These problems are difficult to solve because of a theoretically non-univocal mapping between phonological features and words (*homo-phony/graphy*, *polysemy*, *synonymy* etc.) and because of discontinuous constituencies (*long distance dependencies*) that cause *structural ambiguity*. The goal of using *phases* is then on the one hand to reduce the *space of the problem*, on the other to make the complexity function independent from the *input length*.

[92] Cf. note 60.

3.4.1 THE COMPLEXITY OF AMBIGUITY

Considering both lexical and *semantic ambiguity* in *parsing*, the first (sub)problem can be stated as follows:

(145) **ambiguity problem in parsing** (definition)

> given an input ***i*** of length ***n***, composed by π features grouped by words, and a number ***c*** of possible *Part-of-Speech* (*PoS*), assign to each word from the input at best one *PoS*, if possible. If not reject the input. [93]

The complexity of the problem, considering a brute force algorithm to solve it, is at worst $\mathbf{O(c^n)}$ (assuming that any word could be ambiguous among all *PoS*).
A more realistic complexity order can be guessed by inspecting *dictionaries* or *corpora*. For instance, using Wordnet (Miller 1995), the *ambiguity factor*[94] would decrease down to about 1.44 (this factor seems to be fairly steady across languages such as English, Spanish and Italian), then we would obtain an order of complexity of $\mathbf{O(1,44^n)}$.
Slightly more optimistic results can be obtained with the Brown corpus: 40% of the word occurrences seem to be ambiguous and most of them are ambiguous between two *PoS*; then we could underestimate the *ambiguity factor* up to 40%, producing an order of complexity of $\mathbf{O(1,4^n)}$.
Hoverer, this is clearly not enough yet for the problem to be tractable: analyzing a text would imply processing hundreds of words[95]; the exponential combinatory of the problem still makes it impossible to find out a plausible solution in an acceptable time. Then, we should restrict the combinatorial domain across the input and/or use plausible clues to restrict the range of *ambiguity*.

One possible move is to consider the domain of *ambiguity resolution* to be restricted to a limited context: in fact, if the exponential *n* in the complexity function turns out to be a fixed number, the problem becomes tractable. But of course such a context cannot be

---

[93] *Categories* (or PoS) have to be intended as "indices" pointing to fully specified items in the lexicon. This way the very same problem can be stated from a *generation* perspective simply changing "π features" in "σ features".

[94] $\text{ambiguity factor} = \frac{\text{synsets}}{\text{lexical entries}}$

[95] With just 50 words to be disambiguated we could evaluate up to about 20 millions of possibilities.

arbitrarily fixed (e.g. *n-grams* approach): the length of a grammatical phrase containing ambiguities can be arbitrarily long (in principle it can be infinite, exploiting the complementation option), then fixing it once and for all would not be heuristic.
It is also implausible to reduce the "structurally defined" context to simple local selection as shown below:

(146) a. The [dogs] run ([$_{D}$ the] selects [$_{N}$ dogs])
b. Mary [dogs] me ([$_{DP}$ Mary] does not select any [$_{V}$ dogs])
c. Adverbials cannot select all possible lower adverbials and/or the verb (this would raise the problems presented in (130).

A more adequate solution is to define the *phase* as the "largest context" within which an *ambiguity* can be solved. Let us assume that each *phase* originates in a lexical head (N or V)[96], which then projects and dominates the functional licensors features, as shown in e.g. (132). Note that the *phase* turns out to be dimensionally bounded in length (maximum number of *precedence* relations) and in depth (maximum number of *dominance* relations) because of the following constraints:

- given a fixed hierarchy of licensors features (cf. §2.2.4), a *phase* contains at worst *k* functional elements;
- a *phase* contains exactly *one* projecting lexical element (by definition of *phase*);
- by assumption (Pesetsky 1982), each lexical head *selects* at most *three* ph(r)ases (subject, object and indirect object);

At this point, the complexity order within each *phase* would be $\mathbf{O(1,4^{k+4})}$ (where 1,4 is the most optimistic *ambiguity factor* based on dictionaries and on corpora inspection): the crucial point is that the exponent is, this time, a fixed number, virtually independent of the length *n* of the input.
Note that opening more than one *phase* (from this perspective, any complement represent a new *phase* opened within another *phase*) would produce an increase of the complexity order of $\mathbf{O(1,4^{p(k+4)})}$ where *p*, the number of open *phases*, could grow boundlessly in principle, leading quickly to intractability.

[96] These would roughly correspond to what orthodox minimalist approach (Chomsky 1999) considers CP and DP *phases*.

This is however a welcome result, since a graceful degradation of the human linguistic *performance* is reported in many processing experiments when the subjects have to *parse*/*generate* sentences with certain structures that clearly show a difficulty in "keeping unsolved ambiguities" (e.g. "Buffalo buffalo buffalo Buffalo buffalo" is a well-formed sentence in English but it is extremely difficult to parse unless we already know the structure: "bison from Buffalo (NY) confuse (other) bison from Buffalo"). Thus, the more *phases* we open (at the same time) the more difficult the problem will be.

In sum, the first desideratum of the notion of *phase* is to restrict the context where elaborations, such as *ambiguity resolution*, take place.

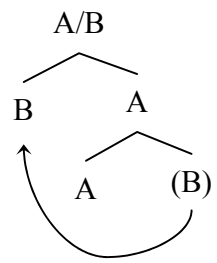
3.4.2 THE COMPLEXITY OF LONG DISTANCE DEPENDENCIES

From a *parsing* perspective, considering a brute force algorithm, finding out which dominance relations have to be associated to a given set of precedence relations given in input has, at least, the complexity order of $\mathbf{O(2^{n-1})}$, where *n* is the length of the input (namely the number of words in the sentence): among *n* items, in fact, we should define *n-1* (*immediate*) *dominance* relations (recall the definitions of *immediate dominance* given in (110)), at best (the minimum number of relations that would make a tree of *n* leafs, fully connected) and any of these relations can be ambiguous about the projecting head (A<B or B<A).

Complexity rises if we consider simple *movement* (let us put aside for the moment *cyclic movement*): at worst any item could have been *moved* out from a lower (*C-commanded*, *selected*) position, we could increase the complexity order of the problem up to $\mathbf{O(2^{(n^2-n)/2})}$: 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}... then this progression is represented by the function: (((n-1)+1)/2 * (n-1)/2).

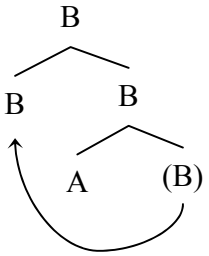
Complexity rises again (boundlessly this time), considering that empty (in terms of π features) heads can enter dominance relations and there is no way to determine how many empty heads can be present, in principle, in a sentence of length *n*. This is clearly not a satisfactory result, since the growing rate of any of these functions would make the problem quickly intractable. This difficulty does not seem to arise in *production*: starting from the complete set of dominance relations (see §2) we can easily follow a *deterministic* algorithm to drop phonological features, then reducing the number of precedence relations to the minimum requirement imposed by the phrase structure (for instance, any *C-commanded* copy of an element can be considered phonologically null, then invisible to linearization).

Notice, however, that there are many empirical constraints on *movement* (captured by the *standard* theory) that could significantly affect the space of the problem:

(126) b. 

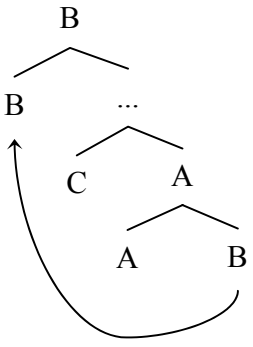
**re-merge*

*[$_{VP}$ Mary kisses t_{Mary}]

c. 

**self-merge*

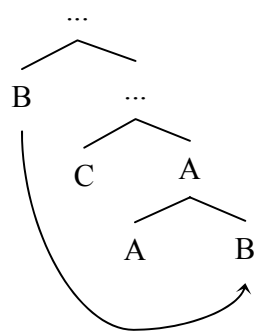
*kisses [$_{VP}$ Mary t_{kisses}]

d. 

**move and project*

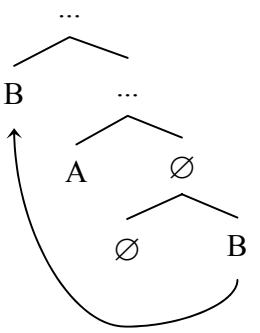
no landing site would be available before the *movement* if the *moved* element would project; moreover this option would violate the assumption that *movement* is triggered by uninterpretable features on a *probe* (Chomsky

1999); in this case *probe* and the *goal* would trivially coincide ("*target project condition*" Chomsky 1993:190, Brody 1998:391).

e. 

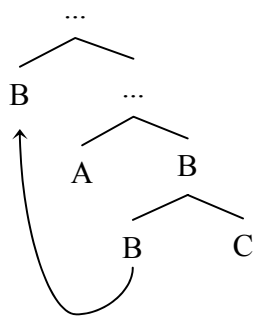
**lowering*

following Kayne (1994) any apparent instance of lowering (*Right Node Raising*, *Extraction Across the Board* etc.) could be accounted for in terms of *remnant movement* ((126.e) is however implied by (126.a))

f. 

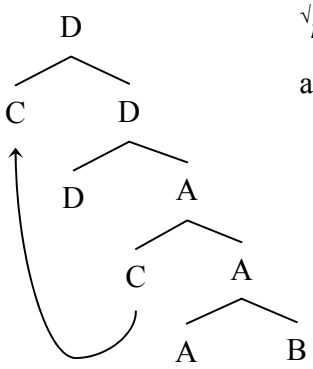
**movement from nothing*

the base position on a chain should be *selected*. An empty element cannot *select* anything

g. 

√*orphans formation, head movement*

verb *movement*

h. 

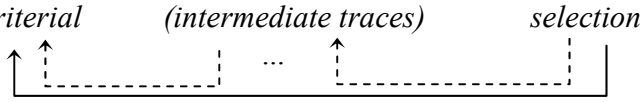
√*movement from functional positions*

adverbial *focalization*, *successive cyclic movement*

Even though these restrictions are defined in a *bottom-to-top* perspective, the very same constraints can be obtained also in a *top-down* framework. These restrictions unfortunately do not significantly reduce the growing rate of the function, since the only formal restriction that they predict is that an element cannot be *moved* from a position to the right of the next element (namely at least two elements should be interposed between the landing site position and the originating one).

The remaining intractability in *parsing* can however be improved upon by adopting the idea of *phase*: we have assumed before (§3.3.4) that *movement* can be detected by the presence of an element that is not *selected*[97]; in this case, this element would stand in "memory" as far as another element *selects* it. Following Chomsky (1998), let us assume that *movement* takes place within the *phase*: then, given a limited context corresponding to the *phase* boundaries, either we find the *selecting* element, so we can reconstruct the *moved* object in its base position, or else if we do not find any selecting element, the expectation to find a selecting sister for the unselected element will be projected on the lower *phase* (leaving an intermediate trace on the "left-periphery" or this lower phase). And so on.
This derivation can be abstractly represented by the chain idea (Rizzi 2004):

(147)



A *criterial* position is the position where the element is interpreted as (roughly) a scope-bearing element, via some interpretable licensor feature in the set defined on (138). Crucially an element cannot be *moved* from a criterial position: meeting a *criterion*, the element is *frozen in place* (Rizzi 2004:11) as shown in (148):

(148) a. *Who$_i$ do you wonder [$_{CP}$ t_i [$_{TP}$ t_i did [$_{VP}$ t_i call you]]]?
b. *Who$_i$ did t_i call you do you wonder t_i?

[97] Once and for all, remeber that *selection* (i.e. *CSel*, §1.1.5) means both *C(ategorial)-selection* (the selection apply only to *formal features*) and *S(emantic)-selection* (the selection apply also to σ *features*).

From our perspective, a *criterial* position is the (unique, excluding *raising* phenomena to be discussed in §4.3.1 and *head movement*) position where an element is inserted into the (local) memory buffer.

Selected positions are those positions where an element can be discharged from the memory buffer; this can happen either because of a felicitous *licensors* configuration (cf. §4.3.2) or simply because a lexical head *selects* an element that is not present in the input to be processed.

Between *criterial* and *selection* position, there could be *intermediate* positions where the element is copied before reaching its (*selected*) final destination. The properties of these intermediate traces are:

- they are not *selected* positions;
- they are available for *binding* (*reconstructions effects*);
- they are not triggered by any apparent satisfaction of semantic requirements (they are in fact accounted for, in a *bottom-to-top* perspective, by purely formal uninterpretable features, cf. §2.2.2).

These properties strengthen the idea that intermediate traces in the "left periphery" (*edge*, Chomsky 1999) of the *phase* serve as "memory refresh" (*phase balance*, Felser 2001) to make *long distance relations* possible. This empirical generalization produces a remarkable reduction of the complexity problem: in fact, for any *moved* element, we would have at most two possible landing site positions within a *phase* (a *left-edge* position, used as memory refresh, and one *selected* position). Then for any *phase* the number of dominance relations required would be, at worse, **2(k+3)** (just in case anything would have been *moved* to the *left periphery* of the *phase*[98]). Then the complexity order of the problem, considering any dominance as ambiguous, would be $\mathbf{O(2^{2(k+3)})}$. Opening a *phase* before having closed the previous one, again, leads to a duplication of the number of possible dominance relations and so on as far as we open new *phases*. The real order of the complexity of the problem is then $\mathbf{O(2^{p2(k+3)})}$ with *p*

[98] k is the number of functional features, 3, has to be intended as 3 possible selected complements plus 1 lexical head, minus 1, provide that for linking *n* elements we would need *n-1* relations.

representing the number of open *phases*. The relation between the length of the input (*n*) and this function is expressed in terms of *phases*, provided that any *phase* would represent a chunk of this input; in particular, the number of lexical items in *n* would determine the number of *phase* (at worse it could be equal to *n*).

We should observe that *discontinuous constituency relations* within a *phase* would not produce any remarkable effect on the complexity of the problem, while *discontinuous constituency relations* among *phases* would increase the complexity of the problem in a linear way if any *phase* is closed before the next one starts. On the other hand, there is an exponential increase of complexity any time we open a *phase* before having closed the previous one.
At this point, the intuition should be clear: the increase of the *complexity* is parallel to the degradation in processing, linked to the *movement* of elements (*storage*, or *active pattern* component of the cost function, §1.4) across open *phases* (*structural integration*, or *top-down expectation projection* cost component, §1.4, cf. *Strong Island Conditions*, §4.3.4).

3.4.3 PHASES

What has just been said informally, can be expressed more precisely as follows:

(149) **notions of phase** (definition)
given a grammar **G** = {**V**, **Cat**, **Lex**, **F**},
an input ***i*** to be processed, composed by elements belonging to ***Lex*** (even if specified only for π or σ features), consider

top-down expectation (definition)
the projection of an SD (based on the *select* features) optionally underspecified for π or σ features;

phase (definition)
a complete process (of *parsing* or *generation*) involving

- a *top-down expectation* about a potential SD structure,
- a proper subset $\boldsymbol{i_p}$ of ***i*** such that any item in $\boldsymbol{i_p}$ is a lexical element (N or V), the *head of the phase*, or it is a functional specification of the head of the *phase*,
- a memory buffer M to store unselected items and retrieve selected ones, initialized as empty, unless some items are inherited from the un-empty memory buffer of a *selecting* previous *phase* (this inherited element will be part of $\boldsymbol{i_p}$ and will be legitimated within the *phase* by merging it at the relevant position of the left-periphery, *edge* of the *phase*, as phonologically null element[99]);

completeness (definition)
a *phase* is complete iff the *head of the phase* is saturated, namely if any mandatory *selectional* requirement (direct object, indirect object etc.) is satisfied (a top down expectation can be considered a complete *phase* in order to close the *phase* generating this *top-down expectation*);

[99] Unless specific parameterization options hold.

complementation (definition)

a *phase* takes only complete *phases* as complements;

phase selection requirement (definition)

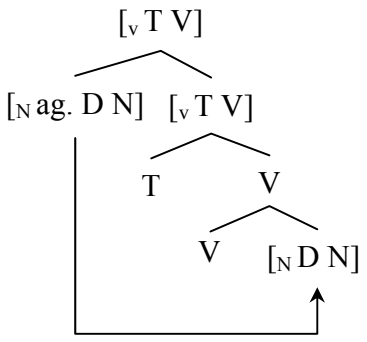
a *phase* has to be *selected*; items in the memory buffer, at the end of the *phase*, can be transferred only to the memory buffer of the *selected phase*(*s*).

Note that a lexical item can be introduced in a *phase* either by direct insertion from the lexicon (simple *merge*) or from the M (memory) buffer (*re-merge*, *internal merge* or *movement*). This second option seems to be available only if the current *phase* is *selected* from the previous *phase* with the non-empty M to be re-projected[100].

The most natural way to embed these notions within our grammar formalization is to consider *phases* as *structure building operations* **F**: a *phase*, then, is a sort of macro projecting *dominance* structures on the SD, on the basis of *selection* requirements. For instance, when we start parsing a sentence, we can assume we are *selecting* a well formed sentence structure (virtually deprived of any π feature[101]) such as *interrogative*, *declarative*, *imperative* etc. This is reminiscent of the TAG approach (Joshi 1985) and fairly in line with Fong's (2004) intuition (§2.3.3), since, in fact, we are projecting elementary trees underspecified for lexical items. Then the *selection* requirements can be expressed simply in terms of *dominance relations* as follows:

(150) *declarative*: minimal SD predicted:

{[v T V] < [$_N$ agentive_feature D N], V<T, V< [$_N$ D N], N<D}

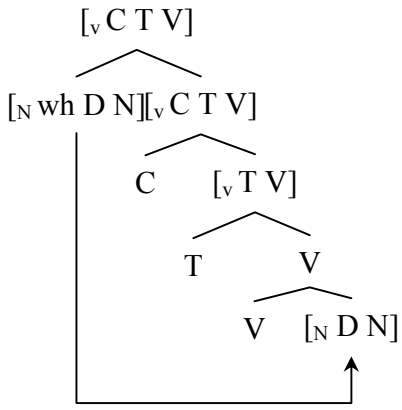


[100] This is the intuition beyond the account of both *subject* and *adjunct conditions* (Huang 1982, §4.3.4).
[101] But this is not a must; in fact we could expect even π feature to be predicted at some point in some cases: "the dog barks while the cat ...".

Wh- question:

{[v C T V]< [N wh D N], [v T V]<C,
V<T, V< [N D N]}

minimal SD predicted:



In order to be grammatical, a projected set of *dominance* relations has to be filled up with lexical items according to s*tructure building operations merge* and *move*. As we have seen before, the *linearization principle* force a precise expectation on the *linear order* the lexical items should conform to, based on the given set of *dominance* relations (cf. (128) and Appendix).

3.4.4 COMPUTATIONAL RESOURCES NEEDS AND COGNITIVE PLAUSIBILITY

In order to define empirically what is a "cognitively plausible" amount of resources, we can analyze data on slight degradation both in *comprehension* and in *production*. In this paragraph, I will just scratch the surface of the problem, providing some sample data based on a preliminary experiment on acceptability judgments.

Five (non-linguist) native Italian speakers have been asked to score from 1 to 5 (1 = lowest acceptability, 5 = highest acceptability) a set of 20 sentences, ten with adverbials modifying a verb and ten with adjectives modifying a noun, such that any sentence ranged between two and six modifiers. Even though much more data would be required in order to draw any kind of conclusion, some preliminary results are however interesting to be reported.

The first consideration comes from the observation that the number of *functional* specifications allowed within the same *phase* seems to be significatively bounded:

(151) a. *Probabilmente* Gianni va *spesso* a trovare Maria
Probably G. goes *often* visiting M.

b. *Prob.* Gianni va *spesso* *velocemente* a trovare Maria
Probably G. goes *often* *rapidly* visiting M.

c. ?*Prob.* Gianni *non* va *spesso* *velocemente* a trovare Maria
Probably G. do*n't* goes *often* *rapidly* visiting M.

d. ??*Francamente prob.* Gianni *non* va *spesso veloc.* a trovare Maria
Frankly probably G. do*n't* goes *often* *rapidly* visiting M.

e. ???*Franc. prob. improvvisamente* G. *non* va *spesso veloc.* a trov. M.
Frankly probably suddenly G. do*n't* goes *often* *rapidly* visiting M.

The difficulty reported in human processing seems proportional to the number of functional specifications associated to a lexical head: when the same verbal head bears between four and five full adverbial specifications, most native speakers report a remarkable difficulty in processing[102]. Comparable results (even though with less remarkable contrasts) can be obtained with adjectival modification too:

(152) a. il *bel* cane *grigio* mangia
the *beautiful* dog *grey* eats

b. il *primo* *bel* cane *grigio* mangia
the *first* *beautiful* dog *grey* eats

c. ?il *primo* *bel* cane *grigio* *italiano* mangia
the *first* *beautiful* dog *grey* *italian* eats

d. ??il *primo* *bel* cane *grande* *grigio* *italiano* mangia
the *first* *beautiful* dog *big* *grey* *italian* eats

These results could pose some empirical constraints on the size of the "*k* factor" (the maximum number of functional projections to be considered within a *phase*): since the

[102] Some speakers accepts more easily certain functional specification, such as *negation* and *lower adverbs*, moreover other functional specifications, such as tense and mood incorporated morphemes, seem to be integrated without any remarkable effort.

functional elements appear on the left of the "selecting" lexical head, this difficulty could be captured by assuming that these elements have to "wait" until the lexical head is integrated before being fully legitimated within the phrase structure[103]. This vague intuition can be made more precise, by assuming that functional heads, since they are not *selected*, are *moved* in an appropriate memory slot, M[mod], as long as the legitimating lexical head is met. Then the memory buffer is progressively unloaded (following the standard mirrored order: Last In, First Out) and *merged* with the verbal complex as *right adjuncts* (having these items in the memory buffer could prevent the *phase* from being closed even if no *selectional* features are present on its head, since these items could directly produce an expectation of their own "base position", that is, an ordered adjunction position on the right of the lexical head, Cinque Ed. 2002).

Another important issue related to the cognitive plausibility of the model is about closing a *phase*: I assumed that a *phase* is open until the saturation condition of its head is satisfied. Under this assumption, it will be easier to add an adjunct before opening of another *phase* and much more difficult (or impossible) to do it after the *phase* has been closed. The data reported below are coherent with these predictions:

(153) a. [$_{ph.\ 1}$The man with the pink scarf [$_{ph.\ 2}$ that Mary would meet]] left
b. ??[$_{ph.\ 1}$The man [$_{ph.\ 2}$ that Mary would meet] with the pink scarf] already left
c. ?[$_{ph.\ 1}$The house [$_{ph.\ 2}$ that Mary would buy] with the pink roof] has been sold

The semantic compatibility (man/Mary > scarf Vs. house/$^{??}$Mary > roof) seems to help us rejoining the interrupted *phase* but, by now, this seems to be out of the control of the proposed algorithm[104]; it is however clear that the preference of the grammar formalized is in line with Phillips' *branching right* hypothesis (Phillips 1996, §2.2.3). Then I will tentatively stick to the idea that a *phase* is closed as soon as the *top-down expectations* can be projected in order to satisfy the *selectional* requirement of the *phase* head.

---

[103] This is different from saying that no structure is built up to the *lexical head*. *Merge* has been explicitly defined between functional elements, then partial structures are built as soon as these objects are processed.

[104] Unless we would use some phonological cue in order to force the "correct" *top-down expectations*, a closed phase should be hardly re-opened after another phase has been processed.

CHAPTER 4
MODEL IMPLEMENTATION AND EMPIRICAL COVERAGE

So far, I pointed out that the model presented in this dissertation is both *cognitively plausible* (chapter 1) and *computationally efficient* (chapter 3). These results are greatly indebted to many decisive assumptions made by recent formal and computational linguistic theories (chapter 2) that have been incorporated, as transparently as possible, within the model sketched in these pages: for instance the notion of *derivation by phase* (*Minimalist Program*, §2.2.2) and the idea of a *functional universal hierarchy* (*Cartographic Approach*, §2.2.4) besides a precise formalization of *minimalist grammars* (Stabler 1997, Fong 2004).

Some important modifications of these ideas however has been show to be necessary in order to meet "interface conditions" or better to be used by *performance algorithms* (or *tasks*) such as *parsing* and *generation*. The *directionality* of the derivation from *left-to-right*, *top-down* (as firstly proposed in Phillips 1996) seems to be the only viable option we have. This modification however required to rethink, within this new perspective, the main *structure building operations* such as *merge*, *move* and the *phase* idea. While very little has been added to the proposal of *merge* suggested by Phillips (*merge right*)[105], both the concept of *move* (with the use of a multidimensional memory buffer helping building *long distance relations* from *left-to-right*) and the notion of *phase*

[105] In fact, the only relevant difference suggested is that *merge* should be sensitive to the distinction between *functional* and *lexical* elements.

(conceived as a *top-down expectation*, driven from a lexical head, complete in terms of selectional requirements) as been significantly modified (for instance with respect to other implementation attempts such as, notably, Fong's one §2.3.3). At this point, these modifications should be enquired at least in terms of *descriptive adequacy* (*universality* requirement, cf. introduction). In this chapter then I will summarize the precise specification of the grammar (§4.1) apt to be used both in *parsing* and in *generation* following a defined algorithm (§4.2). Then I will try to present some arguments showing that a wide empirical coverage can be attained using this model (§4.3). This last part requires a big effort in order to interpret any *long distance relation* simply in terms of *movement* and *phase expectations/selection* since only these two notions can guarantee a fully derivational system (dispensing the grammar from incorporating representational relations such as the classic *C-command* relation, §1.3, that would require inspections of already-built phrases, breaking up with the *Phase Impenetrability Condition* requirement, (69)).

4.1 THE GRAMMAR

Summarizing the problematic aspects of Stabler's formalization, highlighted and revised in the previous chapter, we can think of a (minimalist-cartographic) grammar **G** as a 4-tuple {*V*, *Cat*, *Lex*, *F*} such that:

V is a set of non-syntactic features, $(\pi \cup \sigma)$ where
π are phonetic features and σ semantic features;

Cat is a set of syntactic features, such as *Cat* =
(*base* ∪ *select* ∪ *licensors* ∪ *interveners*) where
base is a set of two lexical categories {*verb*, *noun*},
select specifies the selectional requirement of the head in terms of {*base* ∪ *licensors*}[106];
licensors is a pair of ordered sets of elements expressing the *functional features* associated any *lexical head* {V = < Force, Top, Foc, ... C, $Mood_{speech\_act}$, $Mood_{evaluative}$, ... T_{past}, ... $Asp_{completive}$>, N = <K, ... D, ordinal, cardinal, ... size, length, height ... colour, nationality, material>}
interveners is a set of four subsets of features organized as follows[107]:
Argumental {*ϕ, structural case...*}
Topic {*topic*}
Q {*Wh, Neg, measure, focus, ...* }
Mod(ifier) {*evaluative, epistemic, Neg ... celerative, measure ...* }

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

F is a set of three partial functions from tuples of expressions to expressions {*merge*, *move, phase*} where:

[106] For the time being, I will not use the incorporation device proposed by Stabler (=X/X=).
[107] For the time being, the hierarchy proposed in Starke 2001 will not be integrated within this formalization, since it would require a careful evaluation of its empirical/formal consequences.

merge takes a pair of well formed objects (*feature structures*) and produce a *unified* object according to (136);

move copies, in the relevant *M* slot, an unselected element from a position (where it is firstly successfully *merged*) and drops it (that is *re-merges* it) in a selected position or in a properly selected M buffer according to (140), (141) and (149);

phase is a pair of building tools {*M, phase projection*}associated to a (complete) *lexical projection* such that

M is a Memory buffer (Last In First Out), used by *move*, structured as the *intervener class* (that is, any *intervener class* has its own slot) where processed elements can be copied;

phase projection is a macro introducing empty (that is, non-lexicalized) structural descriptions, based on selectional requirements of a lexical head, driving further *merge* operations according to (149).

Then, in a compressed format, these are the fundamental components necessary and sufficient to describe (at least in formal terms) our linguistic *competence*:

(154) **G** [[**V** [[π] [σ]]]

[**Cat** [[Licensors [... see (129)...]]

[Select [[Licensors] ∪ [Base]]]

[Base [V, N]] (V/N = verbal/nominal heads)

[Interveners [[A], [Topic], [Q], [Mod]]]]]

[**Lex** [[Cat] [V]]]

[**F** [merge, move, phase [M [Interveners], P_projections [[Licensors]∪[Base]]]]]]

P (immediate precedence order): array of pairs P[Lex[V[π]],Lex[V[π]]]

D (immediate dominance order): array of pairs D[Lex[V[σ]],Lex[V[σ]]]

SD (structural description): *tree* (P ∪ D) SD[D, P]

4.2 PARSING AND GENERATION USING THE SAME GRAMMAR

We can now specify an algorithm in order to make explicit the integration among the *structure building operations* defined in the grammar just formalized and the underlined feature structures. In (155) the macrostructure of the algorithm is provided.

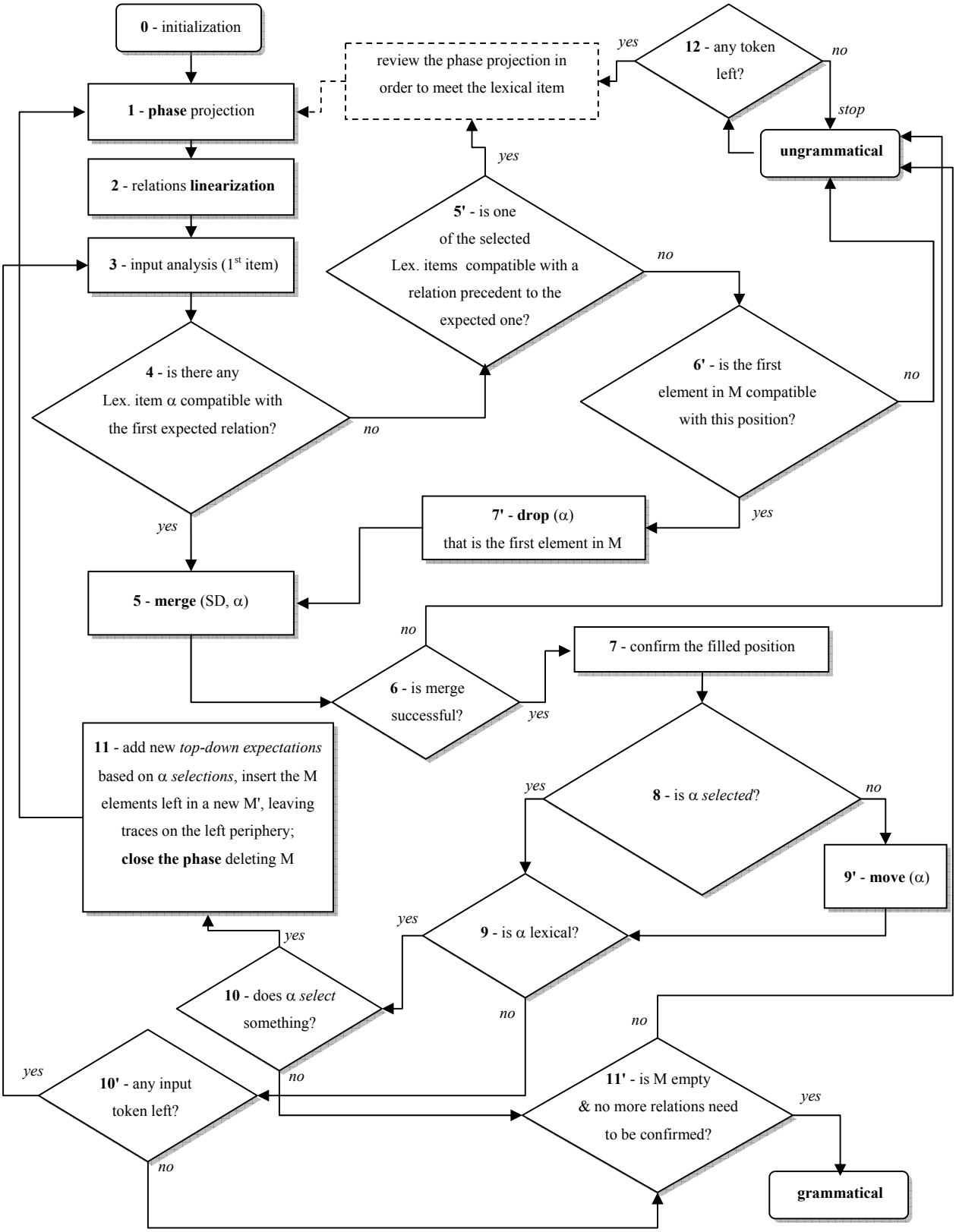
Despite this algorithm could seem too close to the *parsing* perspective to be able to capture even the *generation* problem, few notes have to be revealed in order to make the switch between *parsing* and *generation* suitable and effortless:

1. the linearization of the input relations (**box 2**) works both with π features (trivial) and with σ features (see Appendix); in *parsing* it applies to *precedence* relations, in *generation* it applies to *dominance* relations;
2. any application of *merge* should be trivially consistent with the *dominance* relation inspected (*agreement*, then, should be a result of lexical inspection, **box 3**, following a distributed-morphology-like approach as mentioned in §1.1.2)
3. the **check-box 4** has to be sensitive to the fact that an element is introduced in the phrase structure not directly from the input but from the memory buffer. If this is the case, the introduced element will be a trace, that is, without π features (thus the *drop α* operation, **box 7'**, should be interpreted as dropping a trace, instead of inserting the fully specified element present in the dominance relation inspected)
4. the ordering of elements in any *dominance* relation is directly derived by the hierarchical order imposed in (129) and by the Linearization Principle proposed in (125)
5. the input, in *generation*, is divided into *phases*, by assumption, the highest one is processed before lower ones.

Besides, note that this algorithm shows four sources of determinism worth to be reported:

1. the *Linearization Principle* (acting in **box 2**) predicts the absolute position of the functional elements, lexical heads and complements;
2. the *cartography* of *functional projections* (G[cat[licensors]]) determines univocally the licensors scope/precedence relations (**control box 4** and **5'**);
3. the *phase* projection (**box 1**) reduces the number of viable options both in retrieving the correct lexical item (a relevant subset of possibilities is focalized by the *top-down expectation*) and in solving adjunctions (forcing a *right-branching* structure, unless a different *top-down expectation* is proposed);
4. the underspecification of features promotes determinism (as shown in Fong 1991, 2004): *phase projections* (**box 1**, **G[F[phase]]**) should be general enough to provide only basic dominance relations; then **control box 4** and **5'** require *compatibility* rather than *identity*, that is, more (specific) features than what we would expect can be present in the selected items from the lexicon.

(155) Processing algorithm



4.3 EMPIRICAL COVERAGE

The idea of implementing a computational model inspired to recent *minimalist*/ *cartographic* research was justified by the necessity to met *universality* (that is a cross-linguistic version of *descriptive adequacy* as explained in the introduction) and *explanatory adequacy*. Despite radical redefinitions of the *structure building operations*, the very same set of relevant empirical data should be captured by this model. These last paragraphs show that this idea is plausible in many important ways. Then some relevant linguistic phenomena will be explored in order to verify whether or not this model is *descriptively adequate* and to refine, in case, some highlighted problematic aspects. In particular cases of *argumental movement* (§4.3.1), *criterial* (and *successive cyclic*) *movement* (§4.3.2) will be presented trying, moreover, to account for both *locality* (§4.3.3) and *strong island* constraints (§4.3.4). A cursory inspection of other phenomena (such a *binding*, *control* and *covert movement*) will give a hint of the potential domain of application of this model (that however would require a more detailed discussion).

In order to facilitate the reading of the following pages, below I provide some notes on the conventions used:

1. as introduced in note 71, according to (112) and (113) *precedence* relations are defined among π features, while dominance relations are defined among σ features (then, only indirectly, *precedence* and *dominance* are defined also on *cat* features through lexical items). For the sake of simplicity, labels (such as "John") will be used to refer to σ features in *dominance* relations (i.e. [$_{\text{D N animate person sing}}$ ... John]) while they will refer to π features (/dʒon/) when involved in *precedence* relations;

2. any single step of the process will be reported in the first examples of derivation using tables like the following one:

Task		**input**		
steps	**Structural Description**	**input analysis**	**M(emory)**	**Phase**
	phrases built up to this point (for the sake of clarity the whole structure will be reported; however the sole specification of the dominance/precedence relations, depending on the task, would have been sufficient)	actual token being processed by the algorithm (I will assume a morphological analyzer to retrieve some lexical complexes such as inflected verbs)	memory buffer, controlling *movement* (an element is *moved* here if it is not *selected*)	keep track of any open *phase*: an open *phase* acts as a *top-down expectation*, projecting a phonologically empty structure on SD (underspecified respect to some feature)
	relevant operations are reported here, the number of the involved box in the schema (155) is specified at the beginning			
output				

4.3.1 A(RGUMENTAL) MOVEMENT AND CONTROL

Let us begin by considering a simple case of *A movement* with unaccusative verbs:

(156) *John* [fell t_{John}] (*unaccusative/ergative*)

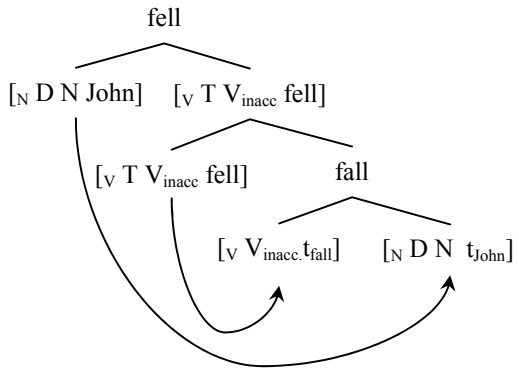
The analysis I assume for *unaccusative*/*ergative* verbs (Burzio 1981) implies that the subject of the verb receive the patient thematic role assigned to the direct object position of the verb (then the subject has to "move from" the direct object position). This analysis can be implemented by using a subcategorization feature (*ergative*) that marks this class of verbs in order to allow them to fill the thematic patient role by retrieving the argument from the Memory buffer instead of from the (external) input.

(156) Minimal SD retrieved:

P = { <John, fell> }

D = { T < John,
fall < T
fall < John }

SD =



These are the main steps the algorithm has to follow in order to retrieve the correct structural description:

Parsing	**input** = <John, fell>			
steps	***Structural Description***	***input analysis***	***M(emory)***	***Phase(s)***
init.	$[_V\ [_N$ D N] T V $t_N]$ **1** - *phase*(*s*) projection (tensed *phase* $[_V$ T V] with the required subject $[_N$ D N] specified)	∅	∅	V, N
step 1	$[_V\ [_N$ D $[_N$ J.]] T V $t_N]$ **4** - compatibility check: ok (J.'s features are compatible with the $[_N$ D N] projection) **5, 6, 7** - merge (∅, John): ok (J. is compatible with more than one V functional positions (*focus*, *topic*, *subject*), then *merge* it at the expected subject position, but remember that it could be potentially underspecified for other features) **9, 10, 11** - close the N *phase* since J. is lexical and it does not select anything	$[_{D\ N\ sing}$ John] **2** - <**John**, fell> **3** - input analysis: J. is an unambiguous N item present in *lex* as $[_{D\ N\ sing}$ John]	$[_A$ J.] **8, 9'** - move(J.) (J. is unselected, so put it in M as Argument)	V, ~~N~~ V is not yet completed, N is closed unless the next token forces the algorithm to reopen it (e.g. PP adjunction)
step 2	$[_V\ [_N$ D $[_N$ J.]] T fell $[_{v\ erg.}\ t_{fall}]\ t_N]$ **4** - compatibility check: ok ("fell" features are compatible with the $[_V$ T V] expectation) **5, 6, 7** - merge (J., fell): ok (J. is then the subject and it *agrees* with fell, then output **fall+T < J.**) **9, 10, 11** - project an argumental *phase* that has to be satisfied from the memory buffer: $[_N$ D N *t*]	$[_V$ T fell $[_{v\ erg.}\ t_{fall}]]$ **2 - fell** **3** - input analysis: (fall + T_{past}) (assume a morphological analyzer to recover the licensors features in this V complex, the required steps of *movement*, triggered by phonological weakness of T, are skipped here for space reasons, the output of the relation is **fall < T**)	$[_A$ J.]	V V is not closed since it projects a direct complement expectation that has to be satisfied by a trace (unaccusative verb)
step 3	$[_V\ [_N$ D $[_N$ J.]] T fell $[_{v\ erg.}\ t_{fall}$ $[_N$ D $[_N\ t_{J.}]\ ]\ ]\ ]$ **4, 5, 6, 7** - merge (fell, $[_A$ J.]): ok (J. receive then the patient role from *fall*, then output: **fall < J.**)	∅	~~$[_A$ J.]~~ **5', 6', 7'** - drop $[_A$ J.] in a *selected* position	~~V~~ V is now closed since the complement expectation is satisfied
end	**10', 11'** - grammatical sentence!	√ (no more tokens left)	√ (empty)	√ (completed)
output = {fall+T < J., fall < T, fall < J}				

An analysis parallel to the *unaccusative* construction can be pursued with passives too:

(157) *John* is kissed t_{John} (by Mary) (*passives*)

the subject has "to be moved from" the direct object position in order to receive the patient thematic role. The auxiliary "is" triggers the *passive* feature on the verb. This requires filling the thematic patient role by retrieving the argument from the Memory buffer rather than from the (external) input.

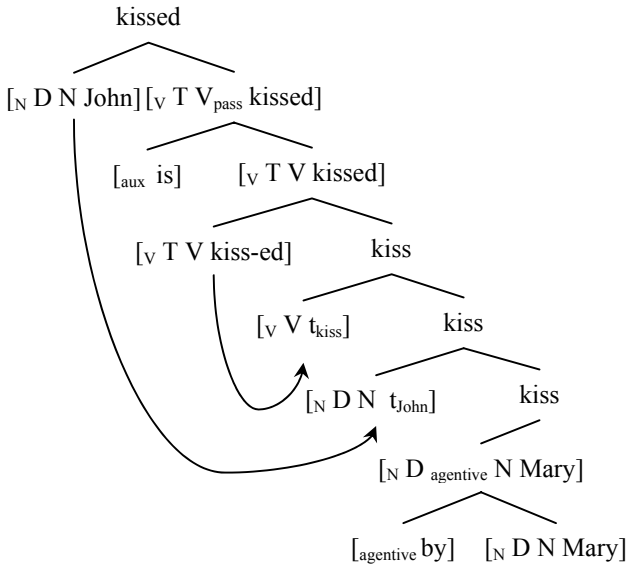
An analysis parallel to the *unaccusative* construction can be pursued with passives too:

(157) Minimal SD retrieved:

P = { <John, is>
<is, kissed>
<kissed, by>
<by, Mary> }

D = { aux + T < John,
kiss < aux,
kiss < -ed,
kiss < John,
kiss < Mary + by,
Mary < by }

SD =



These are the main steps the algorithm has to follow in order to retrieve the correct structural description:

Parsing	**input** = < John, is, kissed, by, Mary >			
steps	***Structural Description***	***input analysis***	***M(emory)***	***Phase(s)***
init.	$[_V\ [_N$ D N] T V]	∅	∅	V, N
	1 - same as (156)			
step 1	$[_V\ [_N$ D N J.] T V]	$[_{D\ N\ sing}$ John]	$[_A$ J.]	V, ~~N~~
	4 - same as (156) **5, 6, 7** - same as (156) **9, 10, 11** - same as (156)	**2** - <**John**, is, kissed, by, Mary> **3** - same as (156)	**8, 9'** - move(J.)	same as (156)
step 2	$[_V\ [_N$ D N J.] $[_T$ is] V t_{is}]]	$[_T$ is]	$[_A$ J.]	V
	4 - compatibility check: ok ("is" features are compatible at least with the [T] expectation) **5, 6, 7** - merge (J., is): ok (J. *agrees* with "is"). assume **is < J.**	**2** - < **is**, kissed, by, Mary> **3** - input analysis: (is + T_{pres}) ("is" is ambiguous among a tensed head of the *phase* and an auxiliary. Since the previous context does not provide enough info for disambiguation let us try the tensed-head way[108])		expect an adjectival phrase: $[_N$ A $t_N]$
step 3	*$[_V\ [_N$ D N J.] $[_{aux}$ is] $[_N$ A $t_N]$ kiss-ed [v t_{kiss}]]]	$[_{V\ T}$ kissed [v t_{kiss}]]	$[_A$ J.]	V
	4, 5' - compatibility check: no! (kissed features are incompatible with the $[_N$ A $t_N]$ *expectation*) then review the *phase* projection. This would imply removing the incorrect "**is < J.**" relation. Back to **step 2,** try $[_{aux}$ is]. This will integrate "kissed" adding: **kiss < is**, **kiss < J.**	**2** - < **kissed**, by, Mary> **3** - input analysis: (kiss + T) output: **kiss < -ed**		
step 4	$[_V\ [_N$ D N J.] $[_{aux}$ is] kiss-ed $[v_{passive}$ t_{kiss} $[t_{J.}]]]$	**2** - < **by**, Mary>	~~$[_A$ J.]~~	~~V~~
	4 - compatibility check: ok (kissed features are compatible with the $[_V$ T V] expectation) **5, 6, 7** - merge ($[_{v\ aux\ T}$ kiss], $[_A$ J.]): ok (J. receives the patient role from *kiss*, then output: **kiss < J.**)	**3** - input analysis: the passive force the algorithm to check the argumental slot in the M buffer.	**5', 6', 7'** - drop $[_A$ J.] in a *selected* position	V is now closed since the complement expectation is satisfied
step 5 & 6	$[_V\ [_N$ D N J.] $[_{aux}$ is] kiss-ed $[v_{passive}$ t_{kiss} $[t_{J.}]$ $[_{N\ agenr.}$ $[_{agent.\ P}$ by] M.]]]	**2** - < **by**, **Mary**>		~~V~~
	4 - compatibility check: ok **5, 6, 7** - merge ($[_{v\ aux\ T}$ kiss], $[_{N\ ag.\ D}$ by Mary]): ok then output: **kiss < Mary + by**	**3** - input analysis: $[_{N\ agent.\ D}$ $[_{P\ agentive}$ by] Mary] output: **Mary < by**		since there are tokens left re-open the *phase* projecting an agentive *phase*
end	**10', 11'** - grammatical sentence!	√ (no more tokens left)	√ (empty)	√ (completed)
output = { kiss + aux + T < John, kiss < aux, kiss < -ed, kiss < John, kiss < Mary + by, Mary < by }				

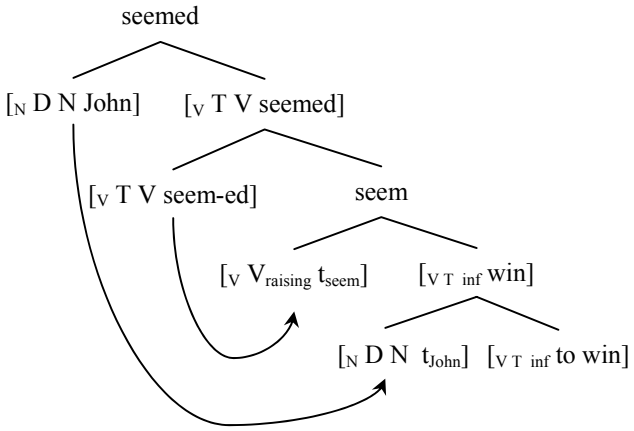
[108] This is clearly the wrong way, avoidable either assuming probabilistic models of disambiguation, Fong's intuition on underspecification or better following the idea that the highest possible expectation is firstly tried (*aux* is higher then *tense*). This choice has been expressly chosen only to show how the algorithm can backtrack.

Let us now turn on *raising* phenomena:

(158) *John* seemed [$t_{j.}$ to win] (*raising*)

Raising verbs have the ability to select a(n infinitival) verbal *phase* as complement; moreover, they do not an assign external θ-role (Chomsky 1981). John is "moved" in the memory buffer of the matrix verbal *phase* headed by "seem", for being discharged in a relevant *selected* position. When the (infinitival) verbal complement *phase* is projected (as a result of a *top-down expectation*), the matrix verbal *phase* is closed and the element present in the matrix memory buffer ([$_A$ John]) is discharged in the (*selected*) complement *phase* ([to win]). "John" is then *selected* by the verbal head "win" as argument

(158) derived SD:



An interesting case to analyze is the asymmetry between *subject* (159.a) and *object control* verbs (159.b) (Larson 1991)[109]:

(159) a. *John* promised Susan [$e_{J.}$ to leave] (*object control*)
a'. **John* [promised [$e_{J.}$ to leave] Susan] (Vs. $^{\surd}$*J.* [promised [$e_{j.}$to leave S.]]
b. John persuaded *Susan* [$e_{S.}$ to leave] (*subject control*)

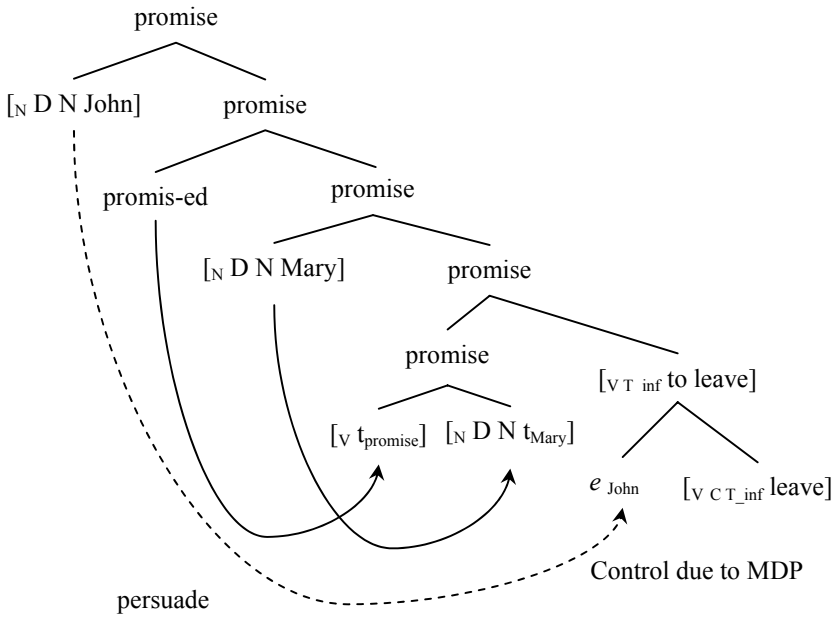
[109] empty elements in the examples are underspecified in terms of PRO or traces and expressed by an "*e*" since at least two analyses are compatible with this framework and no enough space is available in order to set up a proper discussion on these alternatives (that however are non-incompatible): *control as movement*, Hornstein 1999, and *binding as movement* (then PRO has to be interpreted as an anaphor), Kayne 2002

Following Larson's analysis, the control asymmetry is related to the verbal complex structure rather than to the *control theory*; assuming then a *double object construction* for "promise" as shown below in (159.a') and formulating Rosenbaum's idea on "Minimal Distance Principle" (Rosenbaum 1967) as follows, we can account for this asymmetry:

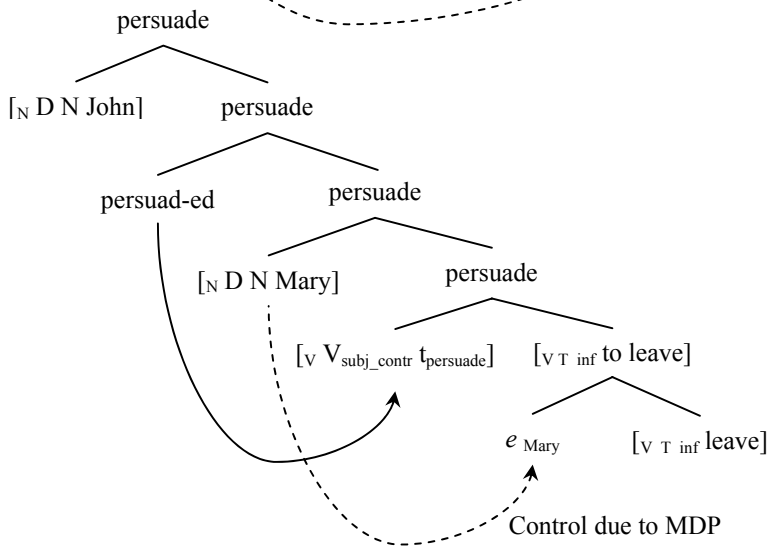
(160) **Minimal Distance Principle** (MDP)

An infinitive complement of a predicate P selects as its controller the minimal c-commanding noun phrase in the functional complex of P.

(159.a')



(159.b')



Translating this very same intuition within this framework, we need to:

1. assume that verbs such as "promise" are marked in the lexicon as *dative verbs*[110];
2. redefine the MDP condition in terms of *memory buffer* rather than using *C-command*;
3. linearize the proposed structure in terms of *merging points* and *selectional requirements*.

While the first point is a job for lexicon-builders, the second one should be a by-product of the correct linearization/selection prediction. I assume this to be the most transparent translation of Larson's analysis within this framework:

(159.a") [[[promise] Mary] to leave] John]

(159.b") [[[persuade] to leave] Mary] John]

In (159.a") since the order the elements enter the computation is <John, promise, Mary, to leave>, "John" is *moved* as soon as it is processed in the memory buffer of the "promise"-*phase*, Mary is integrated in the verb complex as a selected "direct object" (assume it is first *moved* in the memory buffer according to the analysis proposed in (159.a')) then the infinitive complement is selected and it can use (but not receive/discharge since it is not the last predicted *phase*) the memory buffer with the "John" argument inside. This would allow PRO to be interpreted as coreferential to John either assuming a "*control*-by-*movement*" analysis (Hornstein 1999) or a *PRO* as anaphor interpretation then assuming *binding* to be derived by re-*merge* of the *binder* with the *bindee* (Kayne 2002), cf. note 109.

On the other hand, (159.b) can be accounted for by assuming that Mary is "less internal" than the infinitival complement, then "Mary" is *moved* in the memory buffer right after "John" and it is firstly available as binder within the *infinitival phase*.

[110] This implies a double object construction involving passive-like thematic restructuring.

Last property I would highlight is that this perspective makes things a lot easier from a cross-linguistic perspective:

(161) Mary-ga hon-o yonda (Japanese)
Mary$_{nominative}$ book$_{accusative}$ read

"Head-final" languages, such as Japanese, do not pose any problem to this algorithm:

1. Mary → M[$_A$ Mary] (since unselected)
2. book → M[$_A$ Mary, book] (since unselected)
3. read t_{book} ← M[$_A$ Mary, ~~book~~]
4. read t_{book} t_{Mary} ← M[$_A$ ~~Mary~~]

Even though overt case markers represent a big help in terms of *top-down expectations* (we could imagine that they directly trigger the *expectations* for their *selected* merging site), Japanese could be processed without requiring any extra device since arguments, in their standard position, perfectly mirror the merging expectation driven by the verbal head[111].

[111] See languages such as Warlpiri, in order to appreciate the importance of overt case markers so as to project the correct *top-down expectation*.

4.3.2 CRITERIAL MOVEMENT

(162) a. *Who*$_i$ do you think [$_V$ t_i that *Mary*$_j$ will meet t_i t_j]]?(*Wh- object movement*)

b. **Who*$_i$ do you think [$_V$ t_i that will come t_i]]? (*Wh- subject movement*)

b' *Chi*$_i$ *pro* credi [$_V$ t_i che *pro*$_i$ verrà t_i]? (Italian)
Who$_i$ *pro*$_{you}$ think [$_V$ t_i that *pro*$_i$ will come t_i]?

b". *Who*$_i$ do you think [$_V$ t_i will come t_i]]?

b'". ?*What*$_i$ do you think [$_V$ t_i that there is t_i in the box]?

Wh- movement can be considered one of the best examples of *criterial movement* to the left periphery (recall Rizzi's 2004 analysis sketched in (147)). Many (tricky) asymmetries presented in (162) are worth to be noted.

Let us start with *object movement* (162.a); this structure involves a (*wh*-) dedicated left-peripheral position and it requires the algorithm to follow these main steps:

1. *move* the *Wh*- element in the memory buffer of the matrix *phase* since it is non-selected (M_{matrix} [[$_Q$ who]]);
2. interpret the phrasal complement introduced by "that" as the complement closing the matrix *phase*;
3. drop a trace of the *wh*- element in the edge of the complement verbal *phase*, as an effect of the (matrix) memory buffer discharge (*Who*$_i$ do you think [$_V$ t_i that ...]);
4. *move* "Mary" (the subject of "meet") in the memory buffer of the complement *phase* ($M_{complement}$ [[$_Q$ who][$_A$Mary]])

There are two crucial questions to be asked at this point:

a. speaking in terms of *interveners*, the memory slot occupied by "who" is not *A(rgumental)*, then how can a "non-argument" be dropped within an *A* position?

b. the *Who*-trace has to be dropped before the *Mary*-trace, in order for *who* to receive the direct argument thematic role from the verb "meet" how can this be the case, provided that either [$_Q$ who] and [$_A$Mary] are "unordered" (since both

first elements of independent slots) or [$_{Q}$ who] precedes [$_{A}$ Mary] (since "who" entered the memory buffer before, then "who" should be accessible only after "Mary" has been dropped in a selected position)?

I would like to answer the first question going back to Starke's intuition about the "intervention hierarchy": a *wh-* element such as "who" ("what" or "which" but not "how" and "why" for instance) has the properties of being a full DP then legible to be an *argument* because of these features. A *wh-* feature (or maybe more explicitly an "interrogative" feature), is "something more" these arguments have with respect to other arguments (in the sense of Starke, 2001). This is what allows these elements to cross other arguments escaping locality violations as explained in (87)-(88). Then an *argumental wh- element* has to be identifiable as a special case of *argument* and not only as a member of the Q *interveners class*; in this sense it has to be accessible in order to be dropped in an *argumental* position[112].

To answer the second question we should notice: first, that the *subject movement* to a wh- position is assumed not to be allowed (162.b) since when the *subject criterion* is met, the subject is frozen in place and can not *move* further (Rizzi 2004); second, the subject is usually the "most prominent" argument (in the sense it is usually pre-posed to the verbal complex or/and identified by case markers), in this sense being right before the verb (moreover agreeing with it) could be a relevant clue about the actual merging site of this element (that is, the position that allows it to receive the external theta role). These factors could trigger the correct *top-down expectations* about the phrase structure. The simplest effect of these *expectations* would be a (re-)ordering of the elements within the memory buffer to make [$_{Q\ A}$ who] accessible before [$_{A}$ Mary]. As it will be clear in the next pages (cf. cross-serial dependencies, §4.3.5) re-ordering of elements in the memory buffer should be a viable option[113].

With these premises, the following steps would be quite trivial:

---

[112] The same should be said for any *topicalized* or *focalized* argument.
[113] Then, the fact that top-down expectations can access the memory buffer and do some simple operations such as make some element more prominent then others.

5. drop [$_{Q\ wh-\ A}$ who] (as trace) in the selected internal object position of the verb "meet";
6. drop [$_{A}$ Mary] (as trace) in the external argument position of the verb "meet";
7. Memories are emptied, tokens are exhausted, the sentence is grammatical.

Going back to *subject movement* (162.b), the phenomenon is indeed far from being completely explored: there are in fact *escape hatches* that can be pursued in order to bring a question on the *wh-* subject: having a null *pro* makes the "who" extraction possible (162.b') and so it is deleting the complementizer (162.b") or (more marginally) using an overt expletive (162.b'"). In any case, the subject position seems to be skipped in order to make the *wh-* criterion satisfaction possible. Similar to the *pronominal binding* (see note 109), we could expect that the *pro*/*expletive* element has to be "controlled" by the *wh-* element. This can be done assuming that these elements "attract" the *wh-* object in the memory buffer and force a sort of *merge* operation with it (in order to get some features valued) as soon as the dummy subject is processed. The *expletives*, a rich verbal *agreement morphology* in pro-drop languages or the *absence of an overt complementizer* could be intended as functional place-holders able to meet the *subject criterion* that otherwise could have not been met by a bare *wh-* trace.

Another case of *A' movement* worth to be explored is the *modifiers focalization*:

(163) SUDDENLY Napoleon died (*modifier focalization*)

Following Rizzi (2004), a mod(ifier) position is reserved for adverbial *focalization* in the left periphery. Going through the algorithm proposed in (155) these steps are worth to be highlighted:

1. inspecting "suddenly" (box 3, in the flowchart (155)), recognized as an unambiguous aspectual adverb, force the default *top-down expectation* (declarative *phase* projection: [$_{V}$ [$_{N}$ D N] [T [V [t_N]]]]) to be reviewed at least as [$_{V}$ [$_{asp}$ suddenly] [$_{N}$ D N] T V t_N];
2. since English is a non-null-subject language and aspectual adverbs are lower than T (then lower than subject), the focalized position is the only one

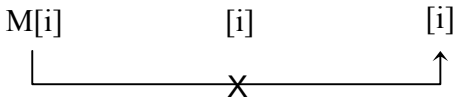
available[114]; then [$_V$ [$_{mod(focus)\ asp}$ suddenly] [[$_N$ D N] [T [$_{asp}$ t_{asp} [V [t_N]]]]]]) → M[$_{mod}$ suddenly]

3. when "died" is analyzed as [$_{V\ T}$ die-d [v t_{die}]] the M buffer is accessed in order to release the trace of the aspectual adverb:
   [$_V$ ... [$_T$ die-d [asp $t_{suddenly}$ [v t_{die} ...]]]] ← M[$_{mod}$ ~~suddenly~~]

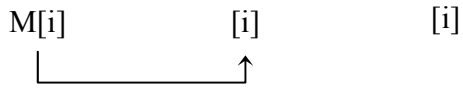
4.3.3 LOCALITY EFFECTS

As described in §2.2.4, we get a locality violation every time an element "overtakes" another element belonging to the same intervening class. We can adopt a *probe*/*goal*-like metaphor (Chomsky 1999, visualized in (67)) adapting it to the memory buffer idea:

(164) a.

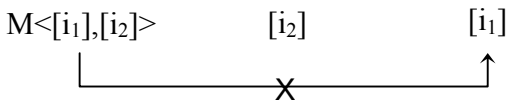


b.



From a purely theoretical point of view, this other option is equally plausible as locality violation:

(164) c.



(crossed dependency)

Following this assumption, I would like to interpret superiority effects as a case of crossed dependency:

(165) a. *what do you think who bought t_{what}? (superiority effects)
b. who do you think bought what?

[114] Unless we assume options such as heavy NP-shift; this would force the algorithm to look for extra features (like prosodic ones), since the (relative) position of the adverbial would be the same both in the *base* position and in the *focalized* one.

This phenomenon is evident when in a question with two *wh-* elements (assume we accept multiple *wh-* questions) the lowest one overtakes the highest one (we can assume this to be an effect either of *attract close* or *shortest move*). From the *top-down* memory-driven *movement* perspective, this effect can be accounted for simply because of the structure of the memory buffer: since, by default, it only allows to pile up (unselected) wh-elements in the same slot, the first one become inaccessible (Last In First Out) at the complement position. The re-ordering option seems not to be allowed this time, since it is not triggered by any particular property (both elements in the memory buffer are [$_{\text{Q(wh) argumental}}$], the head of the verbal *phase* "buy" does not express any preference between the two). On the other hand (165.b) is less problematic simply because the arguments are in the correct linear order with respect to the merging points after the verbal head.

4.3.4 STRONG ISLAND CONDITIONS

Another welcome empirical generalization we get assuming that the memory buffer can be discharged only on *selected phases* is a ban on extractions from strong islands:

(166) a. *I wonder [*who*$_i$ [books of t_i] are on the table] (*subject condition*)
b. *[[*which concert*]$_i$ did you sleep [during t_i]]? (*adjunct condition*)

both a *subject phase* and an *adjunct phase*, since *unselected*, are unable to free the memory buffer of the matrix clause. Hence the ungrammaticality of both (166.a) and (166.b) would derive not from the inability "to be extracted from" (that from this perspective would mean "interpreted within") *subject* or *adjunct islands*, but from the impossibility to legitimate these items by *selecting* them within the matrix clause.
This idea is supported by *parasitic gaps* (*p_gap*) constructions:

(167) a. [Which book]$_i$ did [you [file t_i]] [without reading *p_gap*$_i$]]?
b. *[Which book]$_i$ did [you [sleep]] [without reading *p_gap*$_i$]]?

Assuming that the memory buffer of the matrix clause can be copied without being discharged within an un-selected phase while the matrix phase is processed, this analysis accounts for *connectedness effects* at least for *subject* and *left* (with respect to

the verbal head) *adjunct islands* (see Bianchi Chesi, in preparation, for a discussion of these phenomena).

However, data such as (167.a) could be accounted for assuming the following structural hypothesis:

(167) a'. [Which book] $_i$ did [you [file [without reading p_gap_i] t_i]]?

Moreover, following the idea that "there is *binding* if there is a *binder* in an accessible memory buffer when the *bindee* is processed" the following examples show the "penetrability" of the *strong islands* since *possessives* (despite of the ungrammaticality of the sentence) can be interpreted as bounded to the fronted *wh-* element:

(166) a. $^{(*)}$[[*which politician*]$_i$ did you sleep [during *his*$_i$ talk]]?

b. $^{(*)}$I wonder [*who*$_i$ [pictures of t_i that were in *his*$_i$ closet] are on the table]

4.3.5 CROSS-SERIAL DEPENDENCIES

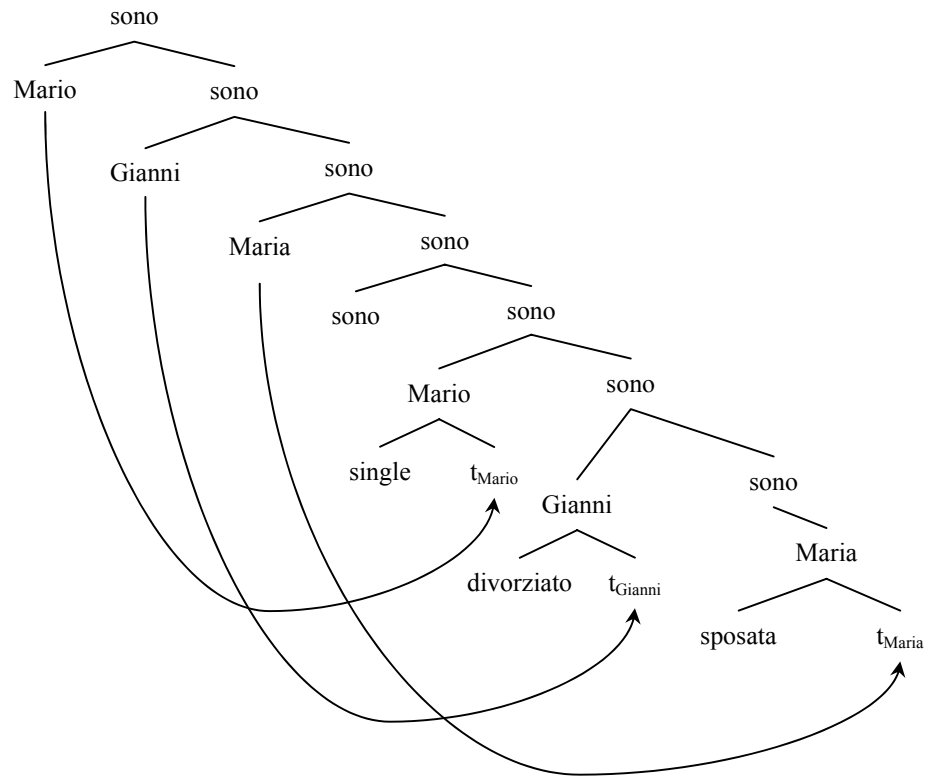
One of the properties of human languages that determined their mildly context sensitivity (Joshi 1985) is the fact that they show (limited) crossing dependencies as the one presented below:

(168) Mario, Gianni e Maria sono rispettivamente single, divorziato e sposata
M., G. and M. are resp. single, divorced$_{masc}$ and married$_{fem.}$

Gender agreement (in Italian) clearly marks the cross-serial relation between the list of elements on the left of the copula and the list of adjectival predicates on the right of the adverbial.

In order to capture this dependency, following the proposed model, we should postulate a merging point between the correct pairs of nouns and predicates.

The standard memory buffer would predict an embedded structure, allegedly wrong for the above sentence. As introduced before, a plausible idea would be inspecting the whole set of elements in the memory buffer and reorder them according to the more plausible *expectations*. Note that the adverbial form “respectively” can force an inspection of this kind and triggers either a *reordering* of the *memory buffer* or a set of *expectations* generating the correct phrase structure (for instance any adjectival form could be *moved* too in the *memory buffer* and then discharged following the logic Last In First Out in a position where any adjective is immediately followed by the correspondent nominal head).



4.3.6 COVERT MOVEMENT

In §2.2.1 I presented an example of *Quantifier Raising* (QR) that we could interestingly analyze from a *top-down* perspective:

(54) *Every* boy likes *a* book

reading 1: ∀ boy, ∃ book : likes(book, boy)

(surface scope; "there is a different book for every boy")

reading 2: ∃ book, ∀ boy : likes(book, boy)

(inverse scope; "there is a single book for every boy")

Following the assumption that the verbal subject is *merged* with the verb after the direct object, we would get the second reading (inverse scope) without assuming any raising operation:

(54) [[*Every* boy]$_i$ [[likes *a* book] t_i]]

(if interpreted "in situ": [[likes *a* book] *Every* boy], that is ∃ book, ∀ boy)

The question about when a reading or another (namely at which point the relative scope of *quantifiers* is computed) is available depending on the "presuppositions" (maybe in a non-technical sense) is still an open question (however unsolved even from a QR perspective).

4.3.7 (SIMPLE) PARAMETER SETTINGS

In §1.1.6 I presented some idea about how *parameters* could be related to the feature structure at different levels of processing formalizing in a more precise way the notion of "past experience" as related to the combination of *principles* (such as the *linearization* one) and *statistical patterns* detected in the input (somehow in line with Yang's 2004 work). The assumptions that leaded these pages on this point can be summarized as follows:

1. **subset parameterization** (a proper Subset of Features, SF_{level}, is selected from the Universal Feature domain at the specific level, UF_{level}, by a language specific function, P_{subset}) is present only at the phonological/phonetic level, that is, π set of features can vary across languages;
2. **structure parameterization** (a language specific function, $P_{structure}$, maps Universal Features Structures, UFS_{level}, to different ones at the same level of processing, SFS_{level}) is not a necessary assumption, then dropped as option;
3. **mapping parameterization** (a language specific function maps a set of Features at one level, F_{level}, to single elements at different levels, e_{new_evel}) is indeed the most suitable option to map π and σ features at the lexical level. Moreover this option has been implicitly exploited in *top-down expectations*: for instance, the structure projected for declarative sentences differs (minimally) from English to Italian because of the *null subject* option (in Italian) and (probably related) by the higher position checked by the verb (subject-related position). The same can be said for *Wh-* question where in English, but not forcedly in Italian and not at all in Japanese, the relevant *Wh-* position in the left periphery (spec-CP and/or related positions) has to be overtly realized.

4.4 CONCLUDING REMARKS: IMPLEMENTING STRUCTURE BUILDING OPERATIONS

Even though some empirical aspects are just touched on, the proposed model should be coherent in its totality: defining *structure building operations* means defining an effective way of putting together the building blocks of the linguistic system. Since this is assumed to be part of our *linguistic competence*, these *structure building operations* should be available for any processing task, crucially both for *parsing* and for *generation*.

Empirical and theoretical details were provided showing that, given a precise definition of the *parsing* and the *generation* problem, *structure building operations* such as *merge*, *move* (Chomsky 1995-2001) and the notion of *phase* (Chomsky 1999) are necessarily part of the grammar. Their effective applicability is however tied to the nature of the process (that, for *complexity* reasons, has been shown to be *derivational* and *top-down*, from *left to right*, as proposed by Phillips 1996). This "new" perspective has required a redefinition of these main components of the grammar: *merge* became a *unification algorithm* (cf. Shieber 1986), *movement* has been implemented (from left to right) through a *memory buffer*, *phases* are reinterpreted as complete *top-down expectations* provided with a single independent (from phase to phase) *memory buffer*. Moreover, these operations have to make an intense use of rich *feature structures* that have to be encoded within the grammar (cf. *Cartographic Approach*, Belletti, Ed. 2002, Cinque 1999, Ed. 2002, Rizzi 1997, Ed. 2002, 2004 and related work).

Even though the main aim of the thesis has been to define and evaluate the proposed solutions from a cognitive perspective, the whole model has been both formalized (see Appendix) and implemented for a small fragment of English and Italian grammar. The first results (both theoretical and empirical) are very encouraging even though much work has to be done (essentially in terms of lexical encoding) in order to have wider coverage and some competitive/comparable benchmark results.

APPENDIX
GRAMMAR FORMALIZATION AND ALGORITHMS

(169) shared variables and typologies

G (Grammar):

multidimensional Array G [[V [[π] [σ]]]

[Cat [[Licensors [... see (129)...]]

[Select [[Licensors] [Base]]]

[Base [V, N]] (V/N = verbal/nominal heads)

[Interveners [[A [A_Q][A_{topic}]]

[Q]

[Mod]]]]]

[Lex [[Cat] [V]]]

[F [merge, move, phase_projection]]]

P (immediate precedence):

array of pairs P[Lex[V[π]],Lex[V[π]]]

D (immediate dominance):

array of pairs D[Lex[V[σ]],Lex[V[σ]]]

SD (structural description):

tree (P ∪ D) SD[D, P]

M (memory buffer for long distance dependencies):

multidimensional array M[Interveners]

(170) lexicon fragment sample

John = *cat*[*licensors*[specific], *base*[N], *intervener*[A]], σ[ϕ[sing, masc], human], π[/John/];

fell = *cat*[*select*[[D N], [D N]] *base*[V]] σ[event] π[fell];

seem = *cat*[*select*[[inf V], [D N]] *base*[V]] σ[state, raising] π[seem];

-s = *cat*[*licensors*[tense]] σ[ϕ[3rd, sing]] π[-s];

to = *cat*[*licensors*[comp]] σ[infinitival] π[to];

be = *cat*[*select*[[adj], [D N]] *base*[V]] σ[passive] π[be];

likely = *cat*[*licensors*[adj qualificative] σ[qualificative] π[likely];

win = *cat*[*select*[[D N]([D N])] *base*[V]] σ[accomplishment] π[win];

(171) **Main functions**

```
function Parsing (P, G) returns D {
    SD = project_phase (G [ F [ phase_projection [ declarative_sentence ] ], G);
    Lex_selected = null;
        for each P[ 0 → P.length] {
        SD = project_phase (Lex_selected[CSel], G);
        Ordered_relations = linearization (P, G);
        Lex_selected = input_analysis(P[current_element]);
            if (compatible(SD, Lex_selected)) {          // (155.Choice box 4)
                partial_result = merge (SD, Lex_selected);
            } else if (compatible(<D, SD>, Lex_selected)) {
                                                         // (155.Choice box 5')
                review (SD, Lex_selected);
                re_add (P, P[current_element]);
                break "for each" loop;
            } else if (compatible(M[first], SD)) {       // (155.Choice box 6')
                partial_result = merge (SD, drop (M[first]));
            } else {
            review (SD, Lex_selected);
            re-add (P, P[current_element]);
            break "for each" loop;
            }
        if (partial_result) SD = partial_result + SD;
        if (!is_selected(Lex_selected)) move(Lex_selected);  // (155.Choice box 8)
        if (is_lexical(Lex_selected) & selects(Lex_selected)) {
                                                         // (155.Choice box 9, 10)
                add (SD, selects(Lex_selected));
                break "for each" loop;
            } else {
                if (!P & !M)     grammatical, returns SD;
                if (!P & M)      ungrammatical; review (SD, Lex_selected)
                                 returns false;
        }
}
```

```
function Generation (D, G) return P, SD {
    ...
    for each D[0 → P.length] {
        ...
        Ordered_relations = linearization (D, G);
        Lex_selected = input_analysis(D[current_element]);
            ...
        }
    }
```

(172) **Main grammatical functions**

(155.Box 1)

```
function project_phase (selection, G) returns SD {
SD = SD ∪ G [ F [ phase_projection [ selection ] ];
returns SD;
}
```

(155.Box 2)

```
function linearization (Set_relations, G) returns Set_relations {
if (Set_relations.is_precedence()) set_relation.sort_array();
if (Set_relations.is_dominance()) {
        while (Set_relations) {
        Dominating_heads = Set_relations[dominating heads];
        Dominating_heads = Dominating_heads - Set_relations[dominated
                                                                heads];
        Dominating_head = Dominating_heads[max(Csel and Fseq
                                                    satisfied)];
        Ordered_relations.push(Set_relations[Dominating_head < X]);
        Set_relations.remove([Dominating_head < X]);
        }
        Set_relations = Ordered_relations;
        }
returns Set_relations;      // either P or D
}
```

(155.Box 3) function **input_analysis** (*Lex[V[π][σ]]*, *SD, G[[Lex]]*) returns *Lex_selected[]* {

for each $G[[Lex[n_{\text{from } 0 \rightarrow Lex.length}]]]$ such that
((*Lex[V[[π]]] ≡ Lex[n][π]*) OR (*Lex[V[[σ]]] ≡ Lex[n][σ]*)) {
push Lex[n] in *Lex_selected[]*;
}
returns *Lex_selected[]*;
}

(155.Box 5) function **merge** (*A*, *B*, *G[licensors]*) return *D* {

if (A *selects* B) returns (A ∪ B, A < B)
else if (B[licensors] *compatible* A) returns (A ∪ B, A < B)
else if (B[licensors] *compatible* A[licensors]) returns (A ∪ B,
[[A [B [H_{empty}]]] or [[B [A [H_{empty}]]] depending on *G[licensors]*);
else returns false
}

(155.Box 9') function **move** (*Lex_item*) modifies *M* {

push *Lex_item* in *M[Lex_item[intervener]][Lex_item]*;
}

(155.Box 7') function **drop** (*selection*) returns *Lex_item* {

pop *Lex_item* from *M[selection[intervener]]*;
returns *Lex_item*;
}

(155.Box 9) function **is_lexical**(*Lex_item*) returns Boolean {

if (*Lex_item[[base]]*) returns true;
if not returns false;
}

(173) how to determine the **highest head** given a set of *dominance* relations D:

(e.g. "who said Susan to leave?": D: {[$_{C\ T}$ say] < who, say < who, say < susan, say < leave, say < to, leave < who})

1. select all dominating head from D (i.e., {[$_{C\ T}$ say], [say], [leave]})
2. delete from the selected set those heads entering at least one dominance relation without projecting (i.e. {[$_{C\ T}$ say], [say], ~~[leave]~~}, since "say < leave")
3. select the standing head, if exists and unique, from the set and choose the feature structure that subsumes the others, if any (i.e. select "say", choose [$_{C\ T}$ say]), if not fail.

Note: the **highest relation** is the one where the involved head has the majority of the selectional requirements satisfied and the biggest number of functional specifications in its featural make up (the unicity of this relation is guarantied by the fact that none of the dominating head can enter more than one relation with the same feature set, that would prevent the tree from being described simply in terms of *immediate dominance* and *immediate precedence*, requiring otherwise an *assignment function*).

REFERENCES

Abney, S. (1987). T*he English Noun Phrase in Its Sentential Aspect*. MIT: Ph.D. Thesis.

Aho, A., R. Sethi, & J. Ullman. (1986). *Compilers: Principles, Techniques, and Tools*. Reading, MA: Addison-Wesley.

Barton, G. E., R. C. Berwick, & E. S. Ristad. (1987). *Computational Complexity and Natural Language*. Cambridge, MA: MIT Press.

Belletti, A. (Eds.) (2002) S*tructures and Beyond. The Cartography of Syntactic Structures, Vol. 3*, Oxford: Oxford University Press.

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

Belletti, A., and Luigi R. (1988). Psych-verbs and theta-theory. *Natural Language and Linguistic Theory 6*, 291-352.

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

Berwick, R. C., S.P. Abney, & C. Tenny, (Eds.) (1991). *Principle-Based Parsing: Computation and Psycholinguistics*. Norwell, MA: Kluwer Academic Publishers.

Bever, T. G. (1970). The cognitive basis for linguistic structures. In J. R. Hayes (Ed.), *Cognition and the development of language* (pp. 279–362). New York, NY: Wiley.

Bianchi, V., & Chesi C. (in preparation) *Phases, strong islands, and computational nesting*. University of Siena: Ms.

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

Borer, H. (1984). *Parametric Syntax: Case Studies in Semitic and Romance Languages*. Dordrecht: Foris Publications.

Bresnan, J. (1978). A realistic transformational grammar. In M. Halle, J. Bresnan, & G. Miller (Eds.) *Linguistic Theory and Psychological Reality*, (pp. 1-59). Cambridge, MA: MIT Press.

Bresnan, J. (2001). *Lexical-Functional Syntax*. Malden, MA: Blackwell.

Brody, M. (1998). Projection and phrase structure. *Linguistic Inquiry 29(3)*, 367-398.

Brody, M. (2002). On the status of representations and derivations. In S. Epstein & D. Seely (Eds.) *Derivation and Explanation in the Minimalist Program*, (pp. 19-41). Oxford: Blackwell.

Burzio, L. (1981). *Intransitive Verbs and Italian Auxiliaries*. MIT: Ph.D. Thesis.

Carpenter G. A. and S. Grossberg (1998). Adaptive resonance theory (ART). In M. A. Arbib (ed.) *The handbook of brain theory and neural networks*, (pp. 79-82). Cambridge, MA: MIT Press.

Cardinaletti, A. (2002). Towards a cartography of subject positions. In Rizzi L. (Ed.) *The Structure of CP and IP*, *The Cartography of Syntactic Structures*, *vol.2*. Oxford: Oxford University Press.

Chomsky, N. (1957). *Syntactic Structures*. The Hague: Mouton.

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

Chomsky, N. (1965). *Aspects of the theory of Syntax*. Cambridge, MA: MIT Press.

Chomsky, N. (1970) Remarks on Nominalization. In R. Jacobs, & P. Rosenbaum (Eds.) *Readings in English Transformational Grammar*, (pp. 11-61). Waltham, MA: Blaisdell.

Chomsky, N. (1973). Conditions on transformations. In S. R. Anderson, & P. Kiparsky (Eds.) *A festschrift for Morris Halle*, (pp. 232–286). New York: Holt, Rinehart & Winston.

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

Chomsky, N. (1986a). *Knowledge of language: Its nature, origin, and use*. New York: Praeger.

Chomsky, N. (1986b). *Barriers*. Cambridge, MA: MIT Press.

Chomsky, N. (1993). A Minimalist Program for Linguistic Theory. In K. Hale and S. J. Keyser (Eds.), *The View from Building 20*. Cambridge, MA: MIT Press. Reprinted in Chomsky (1995) pp. 167–217.

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

Chomsky, N. (1999). Derivation by Phase. *MIT Occasional Papers in Linguistics,18*.

Chomsky, N. (2000). Minimalist Inquiries: The Framework. In R. Martin, D. Michaels, & J. Uriagereka , *Step by Step: Essays in Minimalist Syntax in Honor of Howard Lasnik* (pp. 89-155). Cambridge, MA: MIT Press.

Chomsky, N. (2001). Beyond explanatory adequacy. *MIT Occasional Papers in Linguistics, 20*.

Cinque G. (1994). On the Evidence for Partial N-movement in the Romance DP. In G. Cinque, J. Koster, J.-Y. Pollock, L. Rizzi, & R. Zanuttini (Eds.) *Paths Towards Universal Grammar. Studies in Honor of Richard S. Kayne*. Georgetown: Georgetown University Press.

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

Cinque, G. (Eds.) (2002). *The Structure of DP and IP. The Cartography of Syntactic Structures*, *Vol. 1*. Oxford: Oxford University Press.

Clements, G. N. (1985). The geometry of phonological features. *Phonology Yearbook 2*, 225–252.

Collins, C. (1997). *Local Economy*. Cambridge, MA: MIT Press.

Collins, C. (2001). Eliminating labels. *MIT Occasional Papers in Linguistics, 20*.

Crocker, M. (1996). *Computational Psycholinguistics: An Interdisciplinary Approach to the Study of Language*. Amsterdam: Kluwer Academic Press.

Derose, S. (1988). Grammatical category disambiguation by statistical optimization. *Computational Linguistics, 14*.

Earley, J. (1970). An efficient context-free parsing algorithm. *Comm. of the ACM, 13(2)*, 94-102.

Eimas, P. D. (1975). Auditory and phonetic coding of the cues for speech: discrimination of the [r-l] distinction by young infants. *Perception & Psychophysics, 18(5)*, 341–347.

Felser, C. (2001) Wh-Copying, Phases and Successive Cyclicity. *Essex Research Reports in Linguistics, 37*.

Filip, H. (2000). Nominal and Verbal Semantic Structure: Analogies and Interactions. in N. Gisborne (Eds.) *Elsevier Science, Ltd., a special issue of Language Sciences, 23*, 453-501.

Fodor, J. (1983). *The modularity of mind: An essay on faculty psychology*. Cambridge, MA: MIT Press.

Fong. S. (1991). *Computational Properties of Principle-Based Grammatical Theories*. MIT: Ph.D. Thesis.

Fong, S. (2004). Computation with Probes and Goals: A Parsing Perspective. In A. M. Di Sciullo, & R. Delmonte (Eds.) *UG and External Systems*. Amsterdam: John Benjamins.

Fox D., & D. Pesetsky (2004) Cyclic Linearization of Syntactic Structure. To appear in K. E. Kiss (Ed.) *Theoretical Linguistics, special issue on Object Shift in Scandinavian*.

Fukui, N. (1986). *A Theory of Category Projection and its Application*. MIT: PhD Thesis.

Gibson, E. (1998). Linguistic complexity: locality of syntactic dependencies. *Cognition, 68*, 1–76.

Giorgi, A. & Pianesi F. (1997). *Tense and Aspect: from Semantics to Morphosyntax*. New York: Oxford University Press.

Gorrell, P. (1995). *Syntax and Parsing*. Cambridge, UK: Cambridge University Press.

Grimshaw, J. (1991). Extended projection. In P. Coopmans, M. Everaert & J. Grimshaw (Eds.) *Lexical specification and insertion* (pp. 115-134). The Hague: Holland Academic Graphics.

Grossberg, S. (1976). Adaptive pattern classification and universal recoding: I. parallel development and coding of neural feature detectors. *Biological Cybernetics, 23*, 121-134.

Halle, M. & A. Marantz (1993). Distributed Morphology and the Pieces of Inflection. In K. Hale and S.J. Keyser, (Eds.) *The View From Building 20* (pp. 111-176). Cambridge, MA: MIT Press.

Harley, H., & E. Ritter (2002). Person and number in pronouns: a feature-geometric analysis. *Language 78*, 482-526.

Harkema, H. (2001). *Parsing Minimalist Languages*. UCLA: Ph.D. Thesis.

Hoekstra, T., & Hyams, N. (1995). Missing heads in child language. In C. Koster & F. Wijnen (Eds.) *Proceedings of the Groningen Assembly on Language Acquisition* (pp. 251-261). Groningen: Center for Language and Cognition.

Hornstein, N. (1999). Movement and control. *Linguistic Inquiry 30(1)*, 69–96.

Huang, J. (1982). Logical relations in Chinese and the theory of grammar. MIT: Ph.D. Thesis.

Jackendoff, R. (1997). *The Architecture of the Language Faculty*. Cambridge, MA: The MIT Press.

Jaeglli, O. and K. Safir (eds.) (1989). *The Null Subject Parameter,* Amsterdam: Kluwer Academic Press.

Joshi, Aravind K. (1985). Tree adjoining grammars: how much context-sensitivity is required to provide reasonable structural description. In D. R. Dowty, L. Karttunen, & A. M. Zwicky, (Eds.) *Natural Language Processing: psychological, computational, and theoretical perspectives*, Cambridge: Cambridge University Press.

Kandel, E. R., J. H. Schwartz, & T. M. Jessell (2000). *Principles of Neural Science, 4th Ed*. Appleton: McGraw-Hill.

Kayne, R. (1984). *Connectedness and binary branching*. Foris Publications.

Kayne, R. (1994). *The antisimmetry of syntax*. Cambridge, MA: MIT Press.

Kayne, R. (2002). Pronouns and their antecedents. In S. Epstein & D. Seely (Eds.) *Derivation and Explanation in the Minimalist Program*, (pp. 133-166). Oxford: Blackwell.

Kasami, T. (1965). *An efficient recognition and syntax algorithm for context-free languages*. Technical Report AF-CRL-65-758. Bedford, MA: Air Force Cambridge Research Laboratory.

Keenan, E. and E. Stabler (2004). *Bare grammar: A Study of Language Invariants*. CSLI-SML.

Kitahara, H.. (1994). *Target a: A Unified Theory of Movement and Structure-Building*. Harvard University: Ph.D. Thesis.

Kuhl, P. K. (1994). Learning and representation in speech and language. *Current Opinion in Neurobiology, 4*, 812–822.

Larson, R. K. (1988). On the Double Object Construction. *Linguistic Inquiry 19(3)*, 335-391.

Leonard, L. B. (1998). *Children with Specific Language Impairment*. MIT Press, Cambridge, MA.

Levin, B. (1993). *English verb classes and alternations: A preliminary investigation*. Chicago, IL: University of Chicago Press.

Longobardi, G. (1994). Reference and proper names. *Linguistic Inquiry, 25*, 609-665.

Marr, D. (1982). *Vision*. San Francisco, CA: Freeman.

Merlo, P. (1996). *Parsing with Principles and Classes of Information*. Amsterdam: Kluwer Academic Press.

Michaelis, J. (1998). *Derivational Minimalism is Mildly Context-Sensitive*. In M. Moortgat (Eds.) (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, 63*, 81-97.

Miller, G. A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM, 38(11)*.

Miller, G.A. & N. Chomsky, 1963. Finitary models of language users. In Luce, R.D., R.R. Bush , & E. Galanter (Eds.) *Handbook of Mathematical Psychology, Vol. 2,* (pp. 419–491). New York: Wiley.

Miller, G. A. & Johnson-Laird, P. N. (1976). *Language and perception*. Cambridge, MA: Harvard University Press.

Montague, R. (1974). *Formal philosophy: papers of Richard Montague*. Richmond H. Thomason. (Ed.) New Haven: Yale University Press.

Neumann, G. (1994). *A Uniform Computational Model for Natural Language Parsing and Generation*. University of Saarlandes: Ph.D. Thesis.

Niyogi, S. (2001). A Minimalist Implementation of Verb Subcategorization. *Proceedings of the 7th International Workshop on Parsing Technologies (IWPT-2001)*.

Nunes, J. (2004). *Linearization of Chains and Sideward Movement*. Cambridge, MA: MIT Press.

Ogawa, Y. (2001). *A unified Theory of Verbal and Nominal Projections*. Oxford: Oxford Studies in Comparative Syntax.

Palmer, S. E. (1999). *Vision science: Photons to phenomenology*. Cambridge, MA: MIT Press.

Papadimitriou, C. (1994). *Computational Complexity*. Reading, MA: Addison-Wesley.

Partee, B. (1973) The semantics of belief-sentences. in J. Hintikka & al. (Eds.) *Approaches to natural language*, (pp. 309-336). Dordrecht: Reidel Publishing.

Pesetsky, D. (1989). *Language-Particular Rules and the Earliness Principle*. MIT: Ms.

Pesetsky, D. (1982). *Paths and Categories*. MIT: Ph.D. Thesis.

Phillips, C. (1996). *Order and Structure*. MIT: Ph.D. Thesis.

Pollard, C. and I. Sag (1994). *Head-driven Phrase Structure Grammar*. Chicago: CSLI Publications.

Pullum, G. K. and G. Gazdar (1982). *Natural Languages and Context-Free Languages*. Linguistics and Philosophy, 4, 471–504.

Richards, n. (2004). Against Bans on Lowering. *Linguistic Inquiry, 35(3)* 453–463.

Rizzi, L. (1990). *Relativized Minimality*. Cambridge, MA: MIT Press.

Rizzi, L. (1997). The Fine Structure of the Left Periphery. in L. Haegeman (Eds.) *Elements of Grammar*. Dordrecht: Kluwer Academic Press.

Rizzi, L. (2002). *Locality and Left Periphery*. In Belletti, A. (Eds.) S*tructures and Beyond. The Cartography of Syntactic Structures*, vol. 3, Oxford: Oxford University Press.

Rizzi, L. (2004). *On the form of chains*. University of Siena: Ms.

Rizzi, L. (Eds.) (2002). *The Structure of CP and IP, The Cartography of Syntactic Structures, vol.2*, Oxford: Oxford University Press.

Rosenbaum, P. S. (1967). *The grammar of English predicate complement constructions*. Cambridge, MA: MIT Press.

Ross., J. R. (1972). The category squish: Endstation Hauptwort. In *Papers from the Eighth Regional Meeting. Chicago Linguistic Society*.

Scott, G. J. (1998). Stacked adjectival modification and the structure 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*.

Stabler E. (1997). Derivational minimalism. in Retoré (Ed.) *Logical Aspects of Computational Linguistics*. Springer.

Starke, M. (2001). *Move Dissolves into Merge*. University of Geneva: Ph.D. Thesis.

Starke, M. (2002). Against Specifiers. *Abstract for TILT 2002*.

Verkuyl, H. J. (1999). Aspectual Issues. Structuring Time and Quantity. *CSLI Lecture Notes*, 98, Stanford, California.

Wexler, K. (1994). Optional infinitives, head movement and the economy of derivation in child grammar. In N. Hornstein & D. Lightfoot (Eds.) *Verb Movement*. Cambridge, MA: Cambridge University Press.

Woods, W. A. (1970). Transition Network Grammars for Natural Language Analysis. *Communications of the ACM, 13(10)*, 591-605.

Yang, C. D. (2004). Universal Grammar, statistics or both? *Trends in Cognitive Sciences, 8(10)*.

Younger, D. H. (1967). Recognition and parsing of context-free languages in time n^3. *Information and Control, 10(2)*, 189–208.

Zanuttini, R. (1997). *Negation and Clausal Structure*. New York, NY: Oxford University Press.

Zwicky, A. (1977). *On clitics,* Bloomington: IULC.