INTRODUCTION

STEPS TOWARDS THE MINIMALIST PROGRAM

1. Aim of the course. Topics covered

This set of lectures offers a presentation of the *English complementation system*, within the general framework of Chomsky's *Minimalist Program* (Chomsky 1993, 1995, 1998, 1999).

From a descriptive point of view, the course covers an extensive fragment of the English complex sentence, namely, the domain of *complement* clauses. Complement clauses may informally be defined as subordinate clauses which function as *arguments of predicates* (subjects, objects). The domain of complement clauses includes *that*-clauses, infinitive clauses and *ing*-complements (gerunds and some participial constructions).

(1) a. He considered *that it was a mistake*.
b. He considered *it to be a mistake*.
c. He considered *accepting their offer*.

Complement clauses represent a distinct type of subordination, in terms of their structural properties. As known, subordinate clauses may be classified according to several criteria, such as, the structural criterion; the functional criterion, or the type of licensing involved (cf. Rothstein (1991)).

From a structural perspective, informally, what counts is the nature of the *introductory element*, whether it is a complementizer, a relative pronoun, a subordinative conjunction. More technically, admitting that (most) subordinate clauses are CPs, the structural criterion classifies clauses according to the type of constituents that fill the CP projection. Three types o subordinates may be identified:

a) *Complement clauses* are introduced by a complementizer (C°), an abstract element whose role is to partly nominalize a clause, turning it into an argument of a predicate. Thus, while sentence (2a) may be used to make an independent assertion, sentence (2b) cannot be used independently, but may be selected as an argument of a predicate, as in (2c) and (2d):

(2) a. It is spring.
b. ...that it is spring.
c. I can feel that it is spring.
d. Everybody is aware that it is spring.

Examples of complementizers are *that, for, whether*, a.o. Complementizers select particular types of IPs. *That* selects finite declarative clauses, *for* selects infinitives, etc.

We will roughly define complement clauses as clauses introduced by complementizers, which function as arguments of predicates. Predicates (verbs, adjectives, nouns) c-select and s-select complements, and their subcategorial properties are listed in the lexicon.

b) A second type of subordinates are *wh*-complements. These are introduced by *wh*-words (relative and interrogative pronouns and adverbs). More technically, for *wh*-complements, it is the specifier of the CP which is filled by a *wh*-phrase. The range of *wh*-complements includes relative clauses, indirect questions and cleft constructions, illustrated below.

(3) a. The man on whom the project depends is her brother.
b. I asked what type of book I should buy.
c. It is her brother who said that.

c) Finally, a subordinate clause may be introduced by a "subordinative conjunction", an introductory element which indicates the semantic interpretation of the clause (a time clause in (4a) a concessive clause in (4b), a comparative clause in (4c), etc.).

(4) a. He abandoned her *before* he could find out the truth.
b. He abandoned her *although* he had found out the truth.
c. He abandoned her, *as if* he had not found out the truth.

The second criterion mentioned above is *the syntactic function of the clause*. From this point of view, it is relevant to distinguish between *subject* clauses, *object* clauses and *adjunct* clauses (adverbial and attributive clauses). Thus, subject clauses and adjunct clauses pattern alike regarding certain phenomena, such as the possibility of extracting constituents out of them. Both subject and adjunct clauses are *islands* for extraction, differing from object clauses, which are transparent for extraction. For instance, in (5b), the interrogative constituent was pulled out of a deeply embedded object clause. But in (5d) and (5f) it is not possible to extract a constituent out of a subject clause or out of an adjunct clause.

(5) Object clauses
a. John thought that Pedro told him that the journal had published *the article* already.
b. *What* did John think that Peter told him that the journal had published t already?
Subject clauses:
c. [That Mary was going out with him] bothered you.
d. *Who did [that Mary was going out with t] bother you?
Adjunct clauses
e. Mary was bothered because Peter discussed *her past*.
f. *What was Mary bothered [because Peter discussed t]?

From other points of view, subjects and objects, which are arguments of predicates, have characteristic properties, not true of adjuncts. The Extraposition *+It* Insertion construction is characteristic of subjects and objects, not of adjuncts.

(6) a. It was suggested to them that they should sell the house as soon as possible.
b. He owes it to his father's influence that the committee appointed him to this position.

Syntactic functions therefore play a part in determining the syntactic properties of complement clauses.

The third criterion, the *type of licensing*, deals with the semantic integration of the subordinate within the main clause. The subordinate clause may be an argument of a predicate in the main clause. In this case, the subordinate clause is said to be *θ-licensed*. This is, by definition, true of complement clauses. Adverbial subordinates are also θ-licensed in as much as they are assigned a thematic interpretation by the conjunction which introduces them. For example, any clause starting with *although* is interpreted as concessive because of the θ-role assigned by the

conjunction *although*. A subordinate clause may also function as a *predicate* on an element of the main clause which function as the subject of predication. This is the case of relative clauses, for instance, which are predicates on their antecedents. In the example below, wearing the straw hat and looking exhausted are both properties of the (semantic) subject, *the man*.

(7) The man who was wearing the straw hat looked exhausted.

The perspective of this course is *structural*. The major sections of the course deal with three important types of complement clauses: *that* clauses, infinitive clauses, *ing*-complements.

The specific goal of the present account of English complementation is to be *explanatory*, and also *descriptively complete*, even if surely not exhaustive. We are trying to win a hard bet: that of being theoretically consistent and accurate without sacrificing or ignoring the complexity of the empirical data. The tacit belief is that a theoretically-oriented presentation of one natural language should be at least as comprehensive and empirically significant as a descriptive presentation. The hope is that the theory illuminates the data, reveals its systematicity, without detracting from its richness. Expectedly, stress is laid on the understanding of the syntactic mechanisms that characterise English complements. However, due attention is systematically paid to semantic and, sometimes, discourse considerations, always choosing the syntactic analysis which offers the best or at least a better syntax/semantics fit.

The choice of the Minimalist Program(MP) is natural if explanatory adequacy is in view. To a greater extent than before, in the MP, Grammar is viewed as integrated among other cognitive systems of the mind. *Its structure is determined by the manner in which language interacts with the other cognitive systems*. Grammar thus retains only those elements needed at the interface with other modules of the mind. This is a superior form of functionally motivating and thus of explaining why language is the way it is.

The next section is devoted to a brief presentation of some very general aspects of the minimalist program, the framework assumed in the presentation of English complementation. The presentation is succinct and necessarily incomplete, since many specific conceptual or technical aspects will be introduced in the chapters to come. General acquaintance with the GB framework, as defined in Chomsky (1981, 1982, 1986) would be welcome, but is not really a prerequisite.

2. Plato's problem and the GB program

One of the essential ideas lying at the basis of generative grammar is that language, a very complex object, is learnable in spite of the poverty of the stimulus with which the child is presented, and also without explicit instruction. This is, in essence, what Chomsky dubbed 'Plato's problem' (cf. Chomsky (1986)). The greatest virtue of the Principles and Parameters (PP) account of language, as embodied in the GB model, is that it provides a viable approach to Plato's problem in the domain of language. The idea is simple. Children are biologically equipped with a set of principles of grammar, that is, a Universal Grammar (UG). These principles have open parameters. Specific grammars arise once the open options are given fixed values, which are determined on the basis of primary linguistic data. The PP framework thus proposes the first plausible account of both crosslinguistic variation and language acquisition.

GB is the best-known version of a PP theory of UG. It has several distinctive features:

1. GB is modular: grammar is divided into various subcomponents, incorporating well-formedness requirements, and organized around distinctive principles, primitives, and rule formats or schemas. These modules are X'-Theory, θ-theory, Binding and Control, Case Theory, Move α, in addition to a few general notions that play the role of framework concepts, figuring in several theories and thus unifying them. Such key concepts are 'government', 'binding', 'c-command', etc.

2. Through its Move α module, GB contains a very unconstrained transformational component, because, in principle, Move α allows any category to move anywhere at any time. Overgeneration is a natural by-product, and the various modules, applied at various levels, filter out all and only the undesired structures.

3. GB has four levels of representation at which various conditions are applied to filter out illicit structures: D-Structure (DS), S -Structures (SS), Logical Form (LF), and Phonological Form (PF).

4. The central grammatical relation in GB is *government*. This relation is what lends formal unity to otherwise rather diverse subcomponents.

The shift from GB to MP is motivated by the same tension between descriptive and explanatory adequacy, which has always motivated the reshaping of generative grammar. Concern for descriptive and for explanatory adequacy is as old as the study of language. As soon as the two traditional goals were reformulated within modern generative grammar, serious tension arose between them: the search for descriptive adequacy seems to lead to ever greater complexity of rule systems, while search for explanatory adequacy leads to the conclusion that language structure is largely invariant. The PP conception of invariant principles determining parameters of variation and thus great epiphenomenal diversity has offered a way to resolve the tension between descriptive and explanatory adequacy, thus presenting some conception of the form that a genuine theory might take. As a research program, GB has mainly focused on finding a suitable answer to Plato's problem; consequently, its proposals have largely been evaluated by whether they succeeded in doing so or not, proposing plausible accounts of language variation and language acquisition. This is not to say that other methodological standards have been irrelevant. Simplicity and naturalness have also played a role in evaluating competing proposals. However, in practice these measures of theory evaluation have been swamped by the requirement of finding principles suitably parametrized to address Plato's problem.

The very success of the GB enterprise has considerably changed the methodological landscape. It has spawned a consensus that PP accounts may well answer Plato's problem in the domain of language. This general consensus allows the other sorts of measures of success to guide minimalist theory construction and self-evaluation. The issue now becomes, *which of the conceivable PP models is best*, and the issue is in part addressed, using conventional (not uniquely linguistic) criteria of theory evaluation.

3. General design of the MP

It therefore becomes possible to consider some new question about the faculty of language (FL). In particular, Chomsky considers the following questions: How well is FL designed? How close does language come to optimal design?

The MP is the attempt to formulate and study such questions. The program presupposes the common goal of all inquiry into language - to discover the right theory - and assumes that language is the way it is in order to be adequate and to 'function' properly. More narrowly, the MP seeks to discover to what extent *minimal* conditions of adequacy (= success at the interface) suffice to determine the nature of the right theory.

A first major point of simplification, springing from the idea of the adequate functioning of FL as a module of mind, lies in the conception regarding the levels of representation. The program addresses the question of what conditions are imposed on the linguistic system in virtue of its interaction with the *performance systems*.

Chomsky takes these performance systems to be the *Articulatory Perceptual System* (A-P) and the *Conceptual- Intentional System* (C-I). The A-P system can only use information presented in a certain form, with temporal order, prosodic and syllable structure, certain phonetic properties and relations. The systems of thought (C-I) require information about units they can

interpret (contentful) and the relations among them: certain arrays of semantic features, event and quantificational structure, and so on. In so far as we can discover the properties of these systems, we can ask how well the language organ satisfies the design specifications they impose, providing legible representations at the interface.

The MP explores the hypothesis that language design may really be optimal in some respects, approaching a perfect solution to minimal design specifications.

(8) Language is an optimal answer to legibility conditions. (cf. Chomsky 1998)

Particular phenomena should not be overdetermined by linguistic principles and the linguistic system should be subject to economy restrictions of a specific type.

Earlier versions of the PP approach hypothesised that the linguistic system has the four levels of representation mentioned above, all of them encoding systematic information about linguistic expressions. The MP restricts the class of levels of representation to those which are required by conceptual necessity, because they interface with the performance systems. To be "usable", language must meet certain "legibility conditions", called "bare output conditions" in MP; "output" because they are conditions on interface levels, hence "outputs" on a derivational approach; "bare" to distinguish them from "filters", ranked constraints, and other devices that are a part of the computational system itself. Thus, grammatical objects must be phonetically and conceptually legible by at least this pair of interfaces. An expression *converges* at an interface level, if it consists solely of elements that provide instructions to that external level, thus being legible for the respective external level. The presence of objects which are not interpretable at an interface causes a derivation to *crash*.

The linguistic levels which interface with A-P and C-I are PF and LF respectively. PF and LF can be conceived of as those parts of the linguistic system which provide instructions to the performance systems. A grammatical object is licit just in case it can be interpreted fully at the interfaces that meet LF and PF. All principles and parameters, as well as the properties of linguistic expressions must be described in a vocabulary suitable for PF and LF, and thus for A-P and C-I. *Linguistic expressions are then taken to be optimal realisations of interface conditions, where optimality is determined by economy conditions specified by UG.*

The form of the Grammar is thus drastically simplified, as can be seen by comparing the two models of grammar organisation in (9)-(10) below. There are only two levels of representation, LF and PF. The main argument in favour of eliminating SS and DS is conceptual. A successful theory must capture the fundamental property that grammars pair sound and meaning. If a theory is to posit levels at all, they must be levels that interface with the phonetic and conceptual systems. However, it is not clear that anything more is conceptually required, and parsimony counsels against proliferating levels. If this position is accepted, then the many empirical problems discussed as properties of DS and SS must be addressed without postulating any levels other than LF and PF. To give an example, the Case Filter was a well-formedness condition on S-structures, since it referred to the assignment of Case after A-movement had occurred. The effect of the Case Filter must be replaced, for instance, by a derivational system of checking case features, by one of the mechanisms made available by UG.

(9) *The organization of a GB Grammar*

Lexicon
#
D-Structure
#
Move α (Affect α)
#
S-Structure

2

Logical Form Phonological Form.

(10) *The organization of an MP Grammar*

Lexicon

#

(Spell-Out)

2

Logical Form Phonlogical Form

The organization of the MP Grammar is thus as in (10). Part of the derivation provides information relevant for both PF and LF. Given that the type of information used at the interface is different for LF and PF, respectively, from some point on, called Spell-Out, there is a bifurcation leading to the separate LF and PF representations.

Grammar still associates *structural descriptions* with each sentence /expression. But instead of associating a sentence/ expression with four representation (D-Structure, S-Structure, Logical Form, Phonological Form), the structural descriptions of a sentence / expression is now a pair of representations (π, λ). π is a *PF representation* interpreted at the articulatory perceptual (A-P) interface. λ is an *LF representation* interpreted at the conceptual-intentional (C-I) interface. A derivation *converges* if it yields such a pair (π, λ); otherwise it *crashes*.

To be a linguistic expression, a pair (π, λ) must be formed by a derivation that is not only convergent, but also optimal, where optimality is determined by economy principles.

Conclusions

1. The MP seeks a maximally simple design for language. Given this view, the linguistic levels are taken to be only those conceptually necessary -namely PF and LF - meaning that that there are no (intermediate) levels of D-Structure or S-Structure.

2. Each expression is associated with a structural representation, a pair (π, λ), where π is a PF representation, interpreted at the articulatory perceptual (A-P) interface, and λ is an LF representation, interpreted at the conceptual-intentional (C-I) interface.

4. Structure of an (I)-language: A lexicon and a computational procedure

4.1 The components of a language are a *lexicon* and a *computational procedure for human languages* $C_{HL,}$ that is, a procedure for constructing or generating *linguistic expressions* using the items in the lexicon. The lexicon lists the items which enter into the computational system with their idiosyncratic properties, excluding whatever is predictable by principles of UG.

The lexicon of any language is itself constructed on the basis of the set of features F available in UG, making use of a subset of these features. Thus, while previously, UG was modular, consisting of several sub-theories that checked the well-formedness of the representation associated with each expressions, now UG simply provides a set of features F (linguistic properties) and operations $C_{HL,}$ the computational (generative) procedures for human language, which access F to generate expressions. A particular language will then be specified (apart from parameter setting) starting from UG, using the following procedures (cf. Chomsky, 1998).

(11) (I) Select a *subset* [F] from the *universal set of features F*. Select those features which are relevant for the particular language L.

(II) Select a *lexicon*, assembling features from [F].

The procedures in (11) specify the lexicon of a language on the basis of the set of features available in UG. The computational procedure C_{HL} builds expressions starting from a *subset* of the lexicon, a lexical array or Numeration (see below). No further features may be introduced in the derivation. The following procedures are involved in building expressions:

(12) (III) Select lexical items from the lexicon.
(IV) Map lexical items to expressions, with no recourse to F for narrow syntax.

Statements (I) and (II) embody the old leibnizian idea that there is a universal alphabet of features F (=Universal Grammar) out of which each language constructs its lexical items, which represent a subset [F] of F. Linguistic items fall into two main categories: substantive (N, V, A, P) and functional (C, T D, etc.).

According to (III) and (IV), the derivation of a particular expression makes a one time selection of a lexical array from the lexicon, then *maps these lexical items to expressions*. The derivation actually amounts to the construction, by means of syntactic operations, of a syntactic object, without further access to the lexicon, and without introducing new features. As Chomsky (1988) puts it, "We may take C_{HL} to be a mapping of the LEX(icon) to the LFs and PFs of EXP(ressions)."

The computational system (narrow syntax) consists of a few trivial operations Select, Merge, Move, and (more recently) Agree.

Select is involved in the initial choice of the Numeration, as well as in providing pairs of objects that undergo Merge.

Merge operates on pairs of elements chosen by Select and maps them from a pair into a single element with a more complex structure. Merge is the basic combinatorial device for obtaining complex objects out of simpler or basic ones.

Move (Attract) applies to a single element of a complex structure and displaces it to another part of the structure, leaving a copy (trace) behind. Move always applies to establish a checking relationship, necessary to check (and erase) a morpho-syntactic feature (Tense, Case, etc.).

Agree establishes a checking configuration between a linguistic item α (a head), possessing a morpho-syntactic feature F, and a matching feature F on some other linguistic item situated in some restricted search space (the domain of α), allowing the checking and erasure of the morpho-syntactic feature F. Agree establishes a checking configuration between two items, without involving movement.

4.2 In order for the phonetic form π and logical form λ of some expression to be produced, an initial array of lexical items must be *selected* from the lexicon, out of which a syntactic object will be assembled. Chomsky (1995:225) proposes that such a lexical array is a *Numeration* N, that is, a set of pairs (LI, i) where LI is a *lexical item*, comprised of phonological, semantic, and formal features, and the index *i* indicates the number of times the particular lexical item is accessed (used in the derivation) by the operation Select, during the respective derivation. (The same item may occur several times in a sentence, as in, *The* boy kissed *the* girl, which is why an index is needed).

The computational system then maps N to pair of representations (π , λ), where π is a representation at the level of phonetic interpretation (PF) and λ is a representation at the level of semantic interpretation (LF). These levels interface with the articulatory-acoustic and conceptual-intentional systems of the mind, as explained. *Select* pulls out LIs from the Numeration, reduces their index by one and makes the lexical items available for further operations of the computational system. *Select* continues to apply to the lexical items in N, reducing the index of the element it applies to by 1, until all indices of all elements are reduced to 0.

We are now in a position to define a derivation:

(13) A *derivation* is a sequence of symbolic elements S mapped from a numeration N, such that the last member of S is a pair (π, λ) and N is reduced to zero.

Conclusions

1. The computational system arranges the LIs in N in a way to a form a pair (π, λ) where π is a PF object and λ is an LF object.

2. If π and λ are legitimate objects, that is, they satisfy Full Interpretation, the derivation is said to *converge* at LF and PF.

3. If either π or λ does not satisfy Full Interpretation, the derivation is said to *crash* at the relevant level.

5. Merge. From X'-Theory to Bare Phrase Structure

5.1 ***Defining Merge*** As no one can dispute that sentences are composed of words, an operation is conceptually required that can build sentences from words. The simplest operation for constructing complex units from words, as well as from larger units already formed, is Merge. Merge cannot apply to the internal structure of a previously constructed expression; it always applies to the root (highest) nodes of two structures

Merge takes two syntactic objects (α, β) and forms the new object (K (α, β)) from them.

(14)

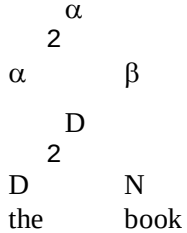
Input	Output
α, β	K

The output of Merge is the new object K; α and β are eliminated, in the sense that they are no longer visible for further combinations. The newly constructed object K is a set {γ {α, β}}, where α and β are constituents of K and γ identifies the category of K. *Merge* is the analogue of *concatenation,* the unique operation of phrase structure grammars, which, out of any two object α, β, produced the object α^β, or β^α. In fact, Merge forms the new object K by *concatenating* α and β and specifying the category of the derived object.

Since the possibility of Merge depends on the c-selectional/ s-selectional possibilities of the combining lexical items, Merge is obviously the analogue of X'-Theory.

The principle of *endocentricity* is still present in as much as, of the two items that combine, one, the *head*, is that which projects. The operation Merge that applies to two objects α and β is asymmetric, projecting either α or β. If α projects, then (the head of) α becomes the label of the newly formed object K. The fundamental idea of endocentricity represents a crucial similarity between X'-Theory and Merge. One example is provided in (15) below:

(15)



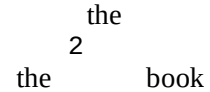
The relations of *head-complement* (sister) and *head-specifier* continue to be available and important. A head has one complement and one s-selected specifier. The theory of phrase structure proposed in the MP allows for multiple specifiers, possibly needed to check the several formal features of the head.

There are however certain differences between Merge and X'-Theory, which result from the same conception of economy of representation which is constitutive of the MP.

An important principle is that grammatical operations *cannot add features*. This principle, known as the *Inclusiveness Condition*, leads to a reanalysis of X-bar structure in more primitive terms. Bar levels will be conceived of as relational (contextual) properties of constituents, not primitives of the system. Bar levels are not inherent properties of lexical items, so what are called *the minimal* and *the maximal* projections are entirely determined by the configuration in which they appear: This view is apparent in the definitions of minimal and maximal projection below:

A category that does not project any further is a *maximal projection* XP, and a category that is not a projection at all is a *minimal projection Xmin*; any other category is an X', invisible at the interface and for computation (see Carnie (2000)).

At the same time, since categorial (non-terminal symbols) are not in the Numeration, one should not introduce them, but simply allow the head to label the newly formed complex object. Accordingly, in the example above, if Merge applies to the two objects *the* and *book*, it forms a projection of *the*.

(16)


This view of phrase structure, devoid of bar levels and categorial labels, is known as *bare phrase structure*.

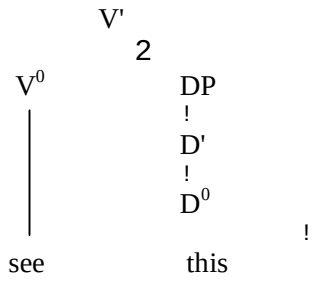
A second conceptual difference from X'-Theory is also noticeable. The structures of X' theory are usually considered to be trees, topologically defined as sets of nodes with certain relations holding among them (sister of, dominate, etc.). As apparent in the presentation of Merge above, Chomsky shows that these structures can also be thought of as *set-theoretic* entities, where the constituents are sets within a particular structure. When it applies to two lexical items, x and y, Merge produces the set {x, y}. Then the object {x, y} may be further input to Merge, etc. Using sets in this way means that no appeal is made to the notions of X' or XP as generalizations over structures. Once again, bar levels and category labels should be thought of as derived concepts, stemming from relationships in particular structures.

In the present volume, the more commonly used, labelled tree structures and X'-Theory notations will still be employed, as a matter of ease and familiarity of notation, though they contain by their very nature extra information which is actually inaccessible to the computational system. Nothing in the analysis, we hope, hinges on this notation.

5.2 ***Kayne's Linear Correspondence Axiom***. A proposal which is related to Chomsky's view of phrase structure, but predates it slightly, is Kayne's Antisymmetry Theory. Like Chomsky, Kayne (1994) claims that much of a X'-Theory is derivative. In his view, X'-notions can be obtained from a constraint on the proper linearization of hierarchical structure (trees), the

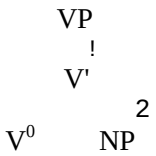
Linear Correspondence Axiom. Kayne's insight is to relate the left to right order observable at the level of the terminal elements (words) with the relation of *asymmetric c-command* that should hold between the corresponding non-terminal nodes. If word α precedes word β in linear word order, then the non terminal category symbol A which dominates word α should asymmetrically c-command the category symbol B which dominates word β. Thus, in the tree structure below, the node V^0 asymmetrically c-commands the node D^0, therefore *see*, the word under V^0, precedes *this*, the word under D^0.

(17)

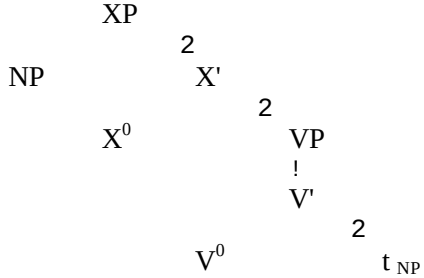


Kayne (1994) proposes that specifiers are always generated to the left of their heads, giving a universal base word order *specifier-head complement*. All languages are thus underlyingly SVO languages. This means that leftmost complements of heads (as in (18b)) must be viewed as derived structures, where a rightmost complement (cf. (18a)) has moved into a specifier position, as suggested in the phrase markers below.

(18) a.



(18) b.



The bare phrase structure system proposed in Chomsky (1995) also adopts a version of Kayne's (1994) *Linear Correspondence Axiom*, supporting the claim that a phrase marker's hierarchical structure determines the linear order of the constituents.

Conclusions

1. The bare phrase structure theory adopted by the MP is represented by the operation Merge. *Merge* takes two syntactic objects (α, β) and forms the new object K((α, β)) from them.
2. Endocentricity continues to function given that, of the two elements that merge, only one, namely the head projects.
3. The relations of head-complement and head-specifier are available, as before.
4. Bar levels and categorial symbols may not be introduced by Merge because of the Inclusiveness Condition, which forbids the introduction of novel features of symbols during the derivation.

6. Spell-Out

Elements interpretable at the A-P interface (e.g. phonologic features) are not interpretable at the C-I interface, and vice versa. At some point in the derivation, the computational system must then split into two parts, one forming π, and the other forming λ, which do not interact any further after the bifurcation. S-structure was the point of this split in pre-minimalist versions of the Principles and Parameters theory. From a minimalist perspective, the problem with there being a distinct level feeding PF and LF, such as S-structure, is that, since it does not interface with any performance system, it is not conceptually necessary. Thus every substantive property attributed to S-Structure should be restated within the minimalist framework in either LF or PF terms.

The only thing required under minimalist assumptions is a rule which splits the computation to form the distinct objects π and λ. Chomsky (1993:22) dubs this operation Spell-Out. Spell-Out is free to apply at any point in a given derivation. After Spell-Out the lexicon will no longer be accessed, and the items in the Numeration have been used up.

The computation from Spell-Out to PF is referred to as the *Phonological Component*. The computation from Spell-Out to LF is referred to as the *covert* component, and the computation that obtains before Spell-Out is referred to as *overt syntax*. PF contains, in addition to phonological rules proper, a *morphological subcomponent* and it also deals with linearization.

7. Types of features and feature checking

7.1 The items combined by Merge group features of different types: phonological, semantic, but also formal (grammatical) features, such as Person, Number, Gender, (= φ-features), Case, Tense, etc. The formal features of the lexical items must be checked during the derivation. Intuitively, one has to verify that each item is suitably placed in an expression. Thus **We goes to school* is ungrammatical because the Number feature on the subject does not match the Number feature of the verbal inflection. Feature checking is thus an essential aspect of a derivation. Several types of formal features have been described (cf. Chomsky 1993, 1995).

From the point of view of their effect at the interface, as described in Chomsky (1993, 1995), features can be *strong* or *weak*. Strong features are supposed to be *uninterpretable at PF*, so they must be eliminated in the overt component of Syntax by overt movement (see below). It is indeed conceptually necessary that a feature which will affect word order, therefore the phonological structure of the string, shoud be checked before PF.

A strong feature must enter into a checking relation *as soon as possible*, causing movement or insertion. More technically, we will say that strong features induce *strict cyclicity*. The idea is that "a strong feature merged at the root must be eliminated before it becomes part of a larger structure by further operations" (Chomsky 1995-234). Thus if F is a strong feature of a

functional category X, F must be checked while the only categories dominating X are projections of X. (see representations (19b, c) below). Suppose that T has a strong Case feature; according to strict cyclicity, T must have this feature checked by having a DP in its Spec, *before* TP merges with C, while the derivation still works on the T projection.

In the analysis of strong (formal) features, the intuition that a strong feature is checked immediately and that it has visible effects (displacement) has been constantly emphasized.

7.2 From a different point of view, features can be *interpretable or uninterpretable* (cf. Chomsky (1998) and later). Restricting ourselves to semantic considerations, we will say that a feature is interpretable if it is legible at LF. A feature like Case is always uninterpretable, features like Number or Gender may be interpretable or not, depending on the constituent on which they show up. For instance, the feature [+plural] number, in a sentence like *They entered*, is present on both the DP and the V. It is interpretable on the DP, since *they* designates a set of individuals, not an individual, but it is uninterpretable on the verb, since the sentence designates one event of entering the room, not several events.

Uninterpretable features like Case, etc., are not *legible* at LF, therefore they must be erased before LF. Following the same intuition, uninterpretable features must be eliminated as soon as possible, therefore they induce strict cyclicity. If uninterpretable features are "strong", they are checked by overt movement and erased after checking. (A slightly different description of strong uninterpretable features will be given in the next chapters). Interpretable features like Number on nouns survive to LF and may be used several times in a derivation

The analysis of "weak" uninterpretable features has undergone changes in the minimalist models presented so far. (Chomsky (1993, 1995), vs. Chomsky (1998, 1999)). In the 1993 version of the model, weak features are checked by *covert* movement, that is, movement at LF.

Later, Chomsky proposes another mechanism of feature checking, the operation Agree, which applies cyclically. In this case, an uninterpretable feature on an (head) item is simply deleted under identity with an identical feature on a sufficiently local item, without requiring any movement

Features are checked in specific configurations. Such are the specifier- head configuration or the head- head configuration. The idea is that the relevant locality notions are defined on phrase structure configurations.

7.3 Chomsky radically distinguishes θ-roles from morphological features like case. The domains of thematic assignment and morphological checking are disjoint. The former takes place in lexical domains. Moreover, θ-roles themselves are defined configurationally (as will be shown later). They are not features, hence cannot be checked, and cannot license movement. They contrast with morphological features like case and agreement, which are checked in functional domains and can license movement in the sense that Move is licit only if one of these features is checked.

Conclusions

From the point of view of their effect at the interface, features of lexical items may be "weak" or "strong", and also they may be interpretable or uninterpretable.

Strong features must be eliminated through overt movement before Spell-Out, since they are uninterpretable at PF. Weak features must be checked in the covert component by covert movement, or alternatively, they may be verified by the checking operation Agree.

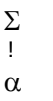
From the point of view of their legibility at the interfaces, features may also be [+Interpretable] and [-Interpretable]. [-Interpretable] features must be eliminated before they reach the interface or else the derivation crashes, since they are not legible at LF.

8. Move and Agree

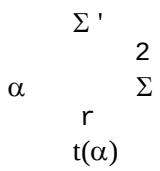
8.1 ***Move*.** In this section attention is paid to the checking operations Move and Agree. While, Merge forms a new object by concatenating two objects that are separate phrase markers, *Move* forms a new object by concatenating two objects that are in a *single* phrase marker. Move is defined as follows:

(19) a. *Move* (from Kitahara (1997))
Applied to the category Σ and α, Move forms Σ' by concatenating α and Σ.
Input: Σ containing α.
Concatenate α and Σ, forming Σ'.
Output: Σ'

b.



c.



Move (Attract) applies to a single element of a complex structure and displaces it to another part of the structure leaving a copy (trace) behind. Move (Attract) applies mainly to eliminate some feature F on the moved-to position, i.e., the attracting head Σ. Move establishes a checking agreement relation between Σ containing (uninterpretable) F and the feature F contained in α, by merging α and Σ. If α is a phrase it will move to the specifier position of Σ. If α is itself a head, it may move and adjoin to the head Σ, thus checking its feature.

Move always establishes a checking relationship, enabling a previously uncheckable feature to get checked. Move is a costly "last resort", chosen when nothing else is possible. In sum, Move is licit only if required to form a fully interpretable phrase marker. Chomsky (1998, 1999) finally argues for a version of Move in which, though feature checking is the only motivation for Move, the result of movement may be the elimination of a strong/uninterpretable feature on either the attracting head or the moving phrase (or head). This runs counter to the earlier view that a constituent moves only to check its own feature (Greed) or that only the attracting head checks and erases its uninterpretable features. This more permissive interpretation of the checking operations performed by Move is due to Lasnik (1995) and is dubbed the principle of Enlightened Self Interest.

Chomsky (1993) incorporates the *copy theory* of movement. According to the copy theory, a *trace* is a *copy* of the moved element which is deleted in the phonological component, but is available for interpretation at LF. A *chain* thus becomes a set of occurrences of a constituent in a constructed syntactic object.

Summarising, Move appears to be a complex operation comprised of copy, merge, chain formation, and finally, chain reduction. Chain reduction is the deletion at PF of all the copies in the chain, but the highest (the head of the chain). As shown by Nunes (1995) a multi-membered chain cannot be realized with all of its links overtly realized, because it cannot be linearized in accordance with the Linear Correspondence Axiom. This forces chain reduction.

The requirement that moved constituents should simply be copied springs from the same Inclusiveness Condition which forbids the introduction of new symbols like traces or indices. We will nevertheless write traces for convenience, remembering that they abbreviate copies of the moved constituents.

The GB typology of A(rgument) and A' (non-argument) movement is carried over. What matters is, as before, whether the head of the chain is in an argument or non-argument position. A-Movement is the movement of a phrase to a head that has strong φ- or Case features. Thus A-movement (e.g., in passive and middle constructions) remains related to the checking of Case. A' Movement occurs when the attracting head has what Chomsky (1998) calls peripheral-features (P-features. These are features of the peripheral system, such as the *wh*-feature, Focus, Topic features, etc. As known, there are important empirical properties that differentiate between A and A' movement, properties that will be used in the analysis of English complementation below.

Move observes the following two requirements:

(i) Constituents always move (to the left) to c-commanding positions.

(ii) The closest constituent that has the appropriate checkable feature is the one that moves

Both requirements are familiar from GB. Movement always takes place to a higher position, because traces must be c-commanded by their antecedents. The second requirement expresses the most important property of movement, *its locality*. Moves must be shortest. An important unifying principle which expressed this idea for the different types of movement was Rizzi's *relativized minimality*, based on the idea that *an antecedent must antecedent-govern its trace*. There are three types of antecedent-trace relations, according as there is head movement, A-movement or A'-Movement (*wh* Movement). Minimality can be stated as below (cf. Rizzi (1990)):

(20) X α-governs Y only if there is no Z such that,
(i) Z is a typical α-governor,
(ii) Z c-commands Y and does not c-command X.
(where α-governs ranges over A'-governs, A - governs or head-governs)

Here are examples. In (21) there are minimality effects for heads. The lower auxiliary *have* has skipped over the higher auxiliary *could*. The trace t cannot be head-governed by *have*, since there is a closer c-commanding antecedent, namely *could*. In (22), there is an example of minimality effect for A-movement, namely subject raising. The subject *John* has raised from the infinitive clause into the main clause, crossing over the subject *it*. Again, the trace left in the infinitive clause cannot be linked to the intended antecedent, since there is a closer potential antecedent, namely the pronoun *it*. Finally, for an A' example consider (23b). The constituent *which problem*, a potential antecedent intervenes between *how*$_j$, the intended antecedent and its trace t_j.

(21) a. Could they t have left?
b.*Have they could t left?

(22) a. John$_i$ seems [t_i to be likely t_i to win]].
b. *John seems [that it is likely [t to win]].

(23) a. Which problem$_i$ do you wonder [how$_j$ PRO to solve t_i t_j].
b *How$_j$ do you wonder [which problem$_i$ PRO to solve t_i t_j].

The MP no longer expresses locality as a filtering condition on movement. Locality becomes a built-in condition, stated as the *Minimal Link Condition* or the *Minimize Chain Links Condition*, which specifies that a constituent always travels the shortest possible distance, or equivalently, that if two candidates could check the same feature, it is the closest that actually

checks it. It is immediately apparent that this condition has the same effect as Rizzi's Minimality. Thus, in (21) the chain link (*could*, t) in (21a) is shorter than the chain link (*have*, t) in (21b), etc.

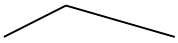
The notion of minimality is incorporated into the definition of Move itself. The Minimal Link Condition has become an intrinsic part of the definition of Move, rather than a constraint on movement. By definition, non-local moves are not moves at all.

8.2. ***Agree***

Agree is a relation between two items, *the probe*, which is the agreeing item and which is a head that possesses uninterpretable features and *the goal* a phrase or a head, possessed of a feature that matches the feature of the agreeing head. In the case of Agree, matching of the features of the probe under identity with features of the goal is sufficient to delete the strong uninterpretable features on the probe, rendering movement unnecessary. Agree allows the checking and erasure of an uninterpretable feature, by matching it with an identical feature of another item, in a sufficiently local domain. The conditions governing Agree are summarized below (cf. Carstens (2000:149)):

(24) *Agree* operates between a probe α and a goal β iff:
a. α has uninterpretable features.
b. β has identical interpretable features.
c. α c-commands β.
d. There is no closer potential goal γ such that α-commands γ and γ c-commands b.

(25)



α	β
[-interpretable]	[+interpreatble]
(probe)	(goal)

According to this conception, Agree is driven by uninterpetable features of the probe, which must be deleted for legibility. As with Move, one may adopt Lasnik's principle of Enlightened Self-Interest, allowing both the probe and the goal to delete matching uninterpretable features.

Parametrizing and feature types

A particular language is simply a setting of the property of strength on features of lexical items, where a strong uninterpretable feature is one that must be immediately eliminated in the derivation. The computational system is uniform across languages.

Conclusion

Move and Agree are alternative mechanisms of deleting uninterpretable features, so as to meet the legibility conditions of LF.

9. Economy Principles

The economy principles apply to both representations and derivations (Chomsky 1991, 1993, 1995).

Economy of representation is nothing other than the principle of Full Interpretation: every object at the interface must receive an "external" interpretation, i.e., must be interpretable by the external performance systems. In order to be interpreted by the A-P system and the C-I, each representation must consist of legitimate objects at the interface. If this condition is not satisfied, the

derivation crashes. Full Interpretation thus determines the set of convergent derivations for a a language.

Economy of derivation determines whether a convergent derivation D from a numeration N to a pair π, λ is *optimal*: the derivational economy principle holds among convergent derivations and determines the set of optimal convergent derivations. One principle of derivational economy is Procrastinate.

(26) *Procrastinate*
Minimize the number of overt operations necessary for convergent.

If such a principle is at work, it follows that derivations where there is little overt movement are more highly-valued than those where overt movement takes place. More generally derivations with fewer steps at any level are better. This is the Shortest derivation Condition:

(27) *Shortest derivation Condition*
Minimize the number of operations (overt or covert) necessary for convergence.

The thrust of these principles is that an operation applies only if that operation will bring the derivation closer to a successful completion (optimal convergence).

Among the operations of the computational system, some are costless (Select, Merge, Agree) and others appear to induce computational cost (Move, Delete).

Instead of Conclusions

The novelty of the MP lies in its addressing the question of the optimal design of language, the answer to which is the Strong Minimalist Thesis stated in (28):

(28) Language is an optimal answer to legibility conditions (cf. Chomsky (1998)).

Adopting the strong thesis has proved to have the following consequences:

1. The only linguistically significant levels are the interface levels (PF, LF)

2 The interpretability condition: linguistic items have no features other than those interpreted at the interface, properties of sound and meaning.

3. The inclusiveness condition: No new features or symbols are introduced by C_{HL}.

4. Relations that enter into C_{HL} either (i) are imposed by legibility conditions, or (ii) fall out in some natural way from the computational process.