1 *Rationale*

1.1 Introduction

While there has been a lot of research on asymmetry and antisymmetry in syntax, symmetry has been mostly ignored or claimed to be outright impossible (Kayne 1994, Di Sciullo 2002, 2005). This is somewhat surprising from a biolinguistic perspective, which seeks to integrate linguistics with the natural sciences, where symmetry is the normal state of affairs and asymmetry requires an explanation (as pointed out by Boeckx and Piattelli-Palmarini 2005, Brody 2006, Chomsky 2005, Jenkins 2000, among others). My main goal in this book is to remedy this gap by examining symmetric aspects of three fundamental syntactic mechanisms: the mechanism responsible for recursion, the mechanism responsible for displacement, and the mechanism responsible for determining the categories of syntactic objects. I look at these three mechanisms through the lens of Chomsky's minimalist program, which takes the mechanism responsible for recursion to be External Merge (often referred to simply as Merge), the mechanism responsible for displacement to be Internal Merge (often referred to simply as Move) and the mechanism responsible for determining categories of both Merge and Move structures to be Labeling. The standard minimalist assumption is that the structures created by Merge are asymmetric (because only such structures can be linearized), that Move is asymmetric (because it 'privileges' one of two potentially movable elements) and that labels are asymmetric (because they contain features of only one element). In the course of the book I will challenge these three assumptions and argue that Merge can also create symmetric structures, that Move can sometimes treat two elements in a symmetric fashion, and that labels can sometimes contain features of two objects undergoing Merge.

The rest of this introductory chapter serves three goals. First, it provides a general introduction to the concepts of symmetry, asymmetry and antisymmetry. It outlines what these concepts mean in general, as well as in more specific, linguistic terms. Second, it provides an overview of the theoretical framework assumed throughout the book, the minimalist program. The overview focuses on the workings of Merge, Move and Labeling, which are at the core of the claims I advance in the book. This chapter also explains why the empirical focus of the book is on symmetric aspects of these three mechanisms, as opposed to many other phenomena that the image of symmetry in syntax might conjure. And third, this introductory chapter provides an overview of the rest of the book.

1.2 Symmetry and asymmetry

The terms *symmetry* and *asymmetry* are used in two different ways in the literature. One is a fairly intuitive non-technical sense, and the other one is somewhat more technical and tends to vary from discipline to discipline.

In its non-technical sense, the term *symmetry* is used to refer to the similarities between two parts of an object (or two objects), and the term *asymmetry* to the differences between them. In a linguistic context, the objects in question could be syntactic features, categories or transformations. Let us first look at a couple of simple cases. For example, we know that arguments differ from adjuncts in that they are bearers of theta roles. Thus we might speak of the symmetric behavior of different types of arguments (i.e. subjects and objects) with respect to theta theory, and the asymmetric behavior of arguments and adjuncts in the same respect. Another well-studied example involves cross-categorial symmetry, such as the symmetry between noun phrases and clauses, which has been studied quite extensively at least since Chomsky's (1970) "Remarks on nominalization" (see Abney 1987, Douglas-Brown 1996 and Hiraiwa 2005, among others, for more recent ways to capture this symmetry). The data in (1a–b) illustrate the symmetric behavior of noun phrases and clauses with respect to theta role assignment.

(1) a. The Romans$_{Agent}$ destroyed the city$_{Theme}$
b. the Roman$_{Agent}$ destruction of the city$_{Theme}$

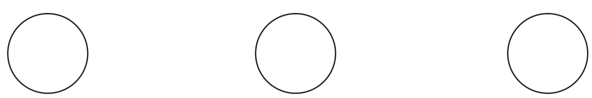
And the Hungarian data in (2a–b) illustrate the symmetric behavior of subjects and possessors with respect to case marking; both are marked with the same (nominative) case. Furthermore, the possessee in (2b) agrees with the possessor in a way that parallels subject–verb agreement.

(2) a. **Te** ve-tt-el egy kalap-ot.
2SG.NOM buy-PAST-SG.INDEF INDEF hat-ACC
'You bought a hat.'

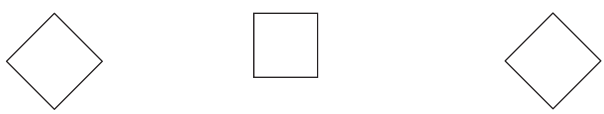
b. a **te** kalap-ja-i-d
D **2SG.NOM** hat-POSS.PL-2.SG
'your hats' (Hiraiwa 2005:19–20, citing Szabolcsi 1994:186)

In a more technical (not necessarily linguistic) sense, the terms symmetry and asymmetry are used to describe geometric patterns, or relationships between two elements in a set. In geometric terms, an object is symmetric if it can remain unchanged when a transformation applies to it. Geometric figures under rotation transformation provide a straightforward illustration. A circle, for example, is symmetric under any rotation; if we rotate it by any degree, the result is still going to be a circle, as shown in (3a–c). A diamond, on the other hand, is only sometimes symmetric, as shown in (4a–c). If we rotate it by 45 degrees, the result is a square. However, if we rotate it by 90 degrees, the result is a diamond again.

(3) a. b. 45° rotation c. 90° rotation

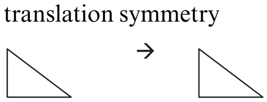


(4) a. b. 45° rotation c. 90° rotation



Mathematicians distinguish four types of symmetric transformations: reflection or mirror symmetry, rotation symmetry, translation symmetry and glide reflection symmetry (see Lee 2007 for an accessible overview). Rotation rotates an object (as we have just seen), translation shifts it (whilst preserving its orientation), reflection yields a mirror image of it, and glide reflection combines reflection and translation. As we will see shortly, the ones that apply most straightforwardly to linguistic patterns are translation and reflection symmetries, illustrated in (5a–b).

(5) a. translation symmetry



b. reflection or mirror symmetry

In set theory, the terms symmetry and asymmetry are used to refer to binary relationships between elements in a set.[1] This is by far the most common usage of the two terms in linguistics. A relationship between two elements in a set is *symmetric* if for every ordered pair $<x,y>$ in the set, the pair $<y,x>$ is also in that set. A good illustration comes from the domain of kinship terms; the relationship 'cousin of' is an example of a *symmetric* relationship. If John is Bill's cousin, Bill has to be John's cousin as well. A relationship between two elements is *asymmetric* if it is never the case that for any pair $<x,y>$ in the set, the pair $<y,x>$ is in the same set. The relation 'is older than' is asymmetric; if John is older than Bill, Bill cannot be older than John. A related concept is that of *antisymmetry*. A relationship between two elements in a set is *antisymmetric* if whenever both $<x,y>$ and $<y,x>$ are members of the set, x must be the same as y.

With this general background on symmetry (and asymmetry), we are almost ready to begin our examination of symmetry in syntax. First, however, let me briefly introduce the theoretical framework assumed in this book, the minimalist program. This is the topic of the next section.

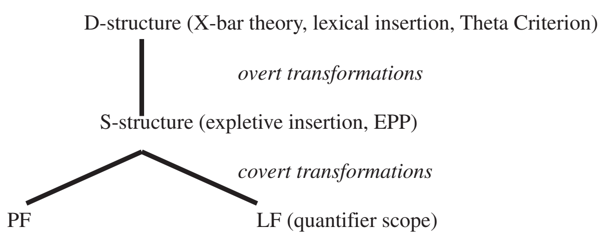
1.3 Theoretical framework

The general framework of this book is the minimalist program pioneered by Chomsky (1995), in particular the version of it laid out in Chomsky (2000, 2001) and subsequent works, often referred to as *Phase Theory*.[2] My goal in this section is not to provide a comprehensive overview of minimalism (or even a general introduction to it), but to give readers less familiar with it sufficient background to follow the rest of the book.[3] The minimalist program is couched within the biolinguistic tradition, which takes the language faculty to be a biological organ, a product of evolutionary processes and pressures. The shape of the language faculty is determined by the following three factors, with the third factor gaining more prominence in recent years.

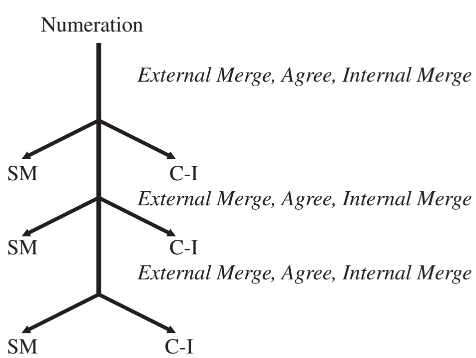
(6)
(i) external data;
(ii) genetic endowment (for language, the topic of UG);
(iii) principles of structural architecture and developmental constraints that are not specific to the organ under investigation, and may be organism independent. (Chomsky 2008:133)

At the core of the minimalist program is the so-called *Strong Minimalist Thesis* (SMT), which states that "language is an optimal solution to interface conditions" (Chomsky 2008:135).[4] The interface conditions are those imposed by the sensorimotor (SM) and conceptual-intentional (C-I) systems. The SMT thus significantly changes the general architecture of the grammar. Readers well versed in Government and Binding theory (and its predecessors) will recognize the Y model of the grammar given in (7a) below, with four distinct levels of representation; D-structure, S-structure, Phonetic Form (PF) and Logical Form (LF). Operations could happen en route to any of these four levels. Likewise, conditions, principles and filters could apply at any level. The "new" minimalist architecture is given in (7b); there are only two relevant levels, the interface levels. Thus, all the syntactic conditions and principles have to be (re-)stated as interface conditions; there is no S-structure or D-structure levels to appeal to.

(7) a.



b.



Each derivation starts with a Numeration: a set of lexical items (or features, to be more accurate) to be manipulated in the course of the derivation. Once the Numeration is exhausted, the derivation is complete.

Another crucial innovation in current minimalism is the idea that derivations proceed in chunks called *phases* and that transfer to the two interfaces can happen more than once per derivation. The terms *Phase Theory* or *Multiple Spell-Out Theory* reflect this aspect of the theory.[5] The points of transfer to the interfaces are determined by phase heads, which are taken to be (transitive) *v* and C heads (perhaps also D heads).[6] More specifically, every time one of these phase heads is merged, the complement of the lower phase head is transferred to the interfaces (and becomes inaccessible for further computation). For example, when C is merged, the complement of *v* is spelled out. This means that the *v* head itself, its specifiers and adjuncts (if any) remain accessible to the derivation. Otherwise, each derivation would stop with the first Transfer. The condition that ensures this is called the *Phase Impenetrability Condition*, one formulation of which is given in (8).

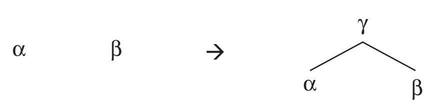
(8) *The Phase Impenetrability Condition (PIC)* (Chomsky 2004:108)
  a. PH = [α [H β]]
  b. The domain of H is not accessible to operations, but only the edge of HP.

As mentioned above, the three mechanisms at the center of this book are External Merge, Internal Merge and Labeling. Let me thus conclude this overview with a brief discussion of how they work.

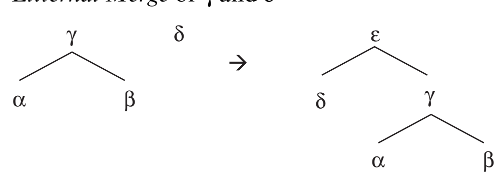
The issue of what kinds of structures Merge can generate is arguably one of the most fundamental issues in syntactic theory. It becomes particularly pressing in the context of recent work by Hauser, Chomsky and Fitch (2002) (see also Fitch, Hauser and Chomsky 2005), who propose that recursive Merge (or some mechanism akin to it) is the only uniquely human property of language, and, as such, it is what separates human language from the communication systems of other species.[7]

Merge comes in two guises, External Merge and Internal Merge. Simply put, External Merge takes two disjoint syntactic objects and combines them to form one larger syntactic object, as shown in (9a). One of these objects could itself be a result of a previous Merge operation (which is what captures recursion), as shown in (9b).

(9) a. *External Merge* of α and β

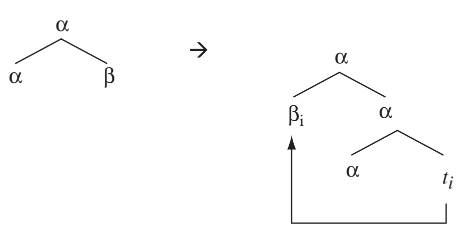


  b. *External Merge* of γ and δ

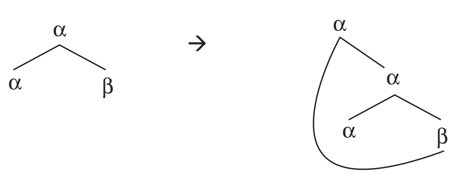


Internal Merge, often referred to simply as Move, is an operation "responsible" for displacement in the grammar; it captures the intuition that syntactic objects can be pronounced and interpreted in different positions. Internal Merge is like External Merge in that it also takes two objects and combines them into one bigger object. What differentiates it from External Merge is that one of these two objects is a part of the other. (10a–b) below illustrate Internal Merge of α and β; (10a) represents it as a standard Copy and Merge (and Delete) operation, whereas (10b) represents it as literal Internal Merge; the moved element β, instead of being copied into a new position, is simply merged again in its new position.[8] The issue of whether the choice between these two ways of conceptualizing movement is substantive or simply a matter of stylistic convention is not trivial but it is not an issue that is directly relevant for our purposes.

(10) a. *Internal Merge* of α and β



b. *Internal Merge* of α and β



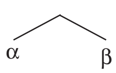
The version of minimalism assumed here maintains the early minimalist assumption that uninterpretable features play a crucial role in syntactic computation. Uninterpretable features are the features that enter the derivation unvalued (marked in what follows as [uF]) and receive values in the course of the derivation via an operation called *Agree*. This is a major departure from early minimalism, where feature checking (now conceived of as feature valuation) required movement to a licensing position, typically a specifier of an appropriate functional projection. Now, they can get valued via *Agree* between a *Probe* (an item whose feature provides a value) and a *Goal* (an item whose feature is in need of a value), as shown in (11a–b).

(11) a. $\text{Probe}_{\text{F:val}}$ > $\text{Goal}_{\text{uF:}}$
b. $\text{Probe}_{\text{F:val}}$ > $\text{Goal}_{\text{uF:val}}$

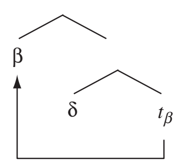
For Agree to take place, the following conditions have to be met. First, the Goal has to be active, where being active means having an unvalued feature. Second, the Goal has to be in the c-command domain of the Probe. And third, there can be no closer potential Goal. Crucially, in a system with Agree, there is no direct relationship between the need to value unvalued features and movement. Features can be valued in situ and what drives movement is the (generalized) EPP feature (or property) of the Probe, which requires it to have an overt specifier.

The last core concept that we will need is the concept of *labels*, whose existence has been implicit in the discussion of External Merge and Internal Merge, above. If External Merge were simply concatenation and Internal Merge displacement, we would expect their outputs to be (12a) and (12b), respectively.

(12) a. *External Merge* of α and β



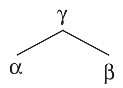
b. *Internal Merge* of α and β



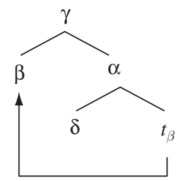
This, however, is not sufficient; the output of Merge also needs a label. The view of labels that I assume throughout this book is essentially that of Chomsky (1994), where features of one of the two objects undergoing Merge determine the label of the new complex object. The issue of whether Labeling is a separate mechanism (or part of Merge itself) is an interesting one, but it does not bear directly on the issue at stake here, which is the symmetric (or asymmetric) nature of labels. I will thus treat Labeling as distinct from Merge and Move (as argued recently by Hornstein 2009, for example), although nothing substantive hinges on this choice, and my conclusions hold irrespective of whether it is an independent mechanism or not.

The idea that labels are necessary (or desirable) in a minimalist architecture is by no means uncontroversial. Chomsky (2004:109) states: “A still more attractive outcome is that L [language, B.C] requires no labels at all” (see also Collins 2002 and Seely 2006, among others, for arguments that labels might be dispensable in a minimalist system). In Chapter 5 of this book, however, I argue against this view, and provide concrete arguments in favor of the existence of labels. For now, let me proceed on what I consider the standard minimalist assumption, which is that the grammar needs labels, and that the result of External Merge is (13a), not (12a), and the result of Internal Merge is (13b), not (12b).

(13) a. *External Merge* of α and β



b. *Internal Merge* of α and β



1.4 The proposal

Given that Merge, Move, and Labeling occupy a central spot in minimalist theorizing (for reasons outlined in the previous section), it seems natural to focus on them in our exploration into the locus and amount of symmetry in syntax. The three questions that I ask in the course of the book are:

(14) a. Does symmetric Merge exist?
b. Does symmetric Move exist?
c. Do symmetric labels exist?

The central claim of this book is that the answer to all three questions is "yes." In the next five chapters, I will provide both theoretical and empirical arguments to support this claim, and, indirectly, against the claims that asymmetric relations are the core relations of the language faculty, articulated in various forms by various researchers (for example, as the *Asymmetry Theory* of Di Sciullo 2005 or the *Antisymmetry Theory* of Kayne 1994).

1.5 Disclaimer: other sources of symmetry?

My goal in this book is to argue for symmetry in Merge, Move, and Labeling. Arguably, these are not the first (or most obvious) mechanisms that come to mind when we think of symmetry in syntax. This raises a natural question of why focus on these three, as opposed to the perhaps more apparent symmetric patterns. Part of the reason is purely practical; the issue of symmetry (or the lack thereof) in syntax is vast, and I hope to make it more manageable by narrowing down the domain of inquiry to these three mechanisms. Furthermore, it is not clear that the symmetry we see elsewhere is true (or only apparent). Let me nevertheless conclude this chapter with a brief discussion of other syntactic phenomena that could be (or have been) classified as symmetric and explain why this book is *not* about them.

Consider the schematic word order patterns given in (15a–b). If both exist, we have a case for the existence of mirror symmetry in the grammar; (15b) is a mirror image of (15a).[9]

(15) a. ABCD
b. DCBA

A fairly straightforward illustration of the two orders comes from the domain of adjective ordering, where both patterns are attested crosslinguistically (in addition to many others, not relevant here). For example, the ordering of French adjectives in (16a–b) is a mirror image of the ordering of their English counterparts in (17a–b). The factors that determine the ordering of the adjectives relative to each other are also not directly relevant here (for discussion and concrete proposals, see Bernstein 1993, Cinque 1994, Scott 2002, Sproat and Shih 1991, Svenonius 2008, among many others).

(16) a. une voiture *italienne* *magnifique* *French*
a car Italian beautiful
b. une fusée *américaine* *énorme*
a rocket American huge (Laenzlinger 2005:658)

(17) a. a *beautiful Italian* car
b. a *huge American* rocket

This is not the only possibility; in (18a), the ordering of postnominal adjectives matches the ordering of prenominal adjectives in English, and in (19a), some adjectives precede the noun and others follow it.

(18) a. une voiture *rouge* *française*
a car red French (Laenzlinger 2005:658)
b. a *red French* car

(19) a. un *joli* *gros* ballon *rouge*
a pretty big ball red (Cinque 1994:101)
b. a *pretty big red* ball

The range of logical possibilities for a sequence consisting of two adjectives and a noun is given in (20a–f). Since there are three elements (two adjectives and a noun), there are 6 (3!) possible ways to order them. (20b) is a mirror image of (20a). The orderings in (20c) and (20d) have been