Constructive Adpositional Grammars: Foundations of Constructive Linguistics

By

Federico Gobbo and Marco Benini

CAMBRIDGE SCHOLARS

PUBLISHING

Constructive Adpositional Grammars:
Foundations of Constructive Linguistics,
by Federico Gobbo and Marco Benini

This book first published 2011

Cambridge Scholars Publishing

12 Back Chapman Street, Newcastle upon Tyne, NE6 2XX, UK

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

ISBN (10): 1-4438-3007-0, ISBN (13): 978-1-4438-3007-2

Table of Contents

Chapter One

Introduction

This book presents a framework to understand how language is structured—in particular at a morphological and syntactic level—and how language takes its place in the general human cognition. [1]

We consider that the study of languages should be intertwined to the research in the field of cognitive sciences on one hand and with the expressive power of constructive mathematics on the other hand. In fact, constructive mathematics allows a full-blooded computational development of natural language grammar description, without invoking idealistic interpretations. In other words, every concept of our framework poses on a solid mathematical ground when interpreted in a cognitive way.

A preliminary clarification is needed. As the main area of investigation within this book is natural language grammar, we will use the term 'language' to refer to natural languages—such as English, Chinese, or Urdu—in general terms if the context is clear enough to avoid ambiguity. Otherwise, we will specify if we are talking about natural or formal (artificial) languages. Analogously, this kind of convention will be followed for other common terms which are used both by linguists and mathematicians in very different ways, such as 'grammar', 'syntax' or 'semantics'. [2]

This clarification is needed because this book is addressed both to people belonging to the humanities—such as most linguists are, but also pedagogists involved in language learning issues—and people belonging to hard sciences, above all, mathematicians, but also logicians, computer scientists and natural language engineers. We believe that the so-called 'two cultures'—even three, as recently posed by Kagan (2009)—should at least dialogue one with the other if not put in confluence for mutual improvement.

This is the main reason why we decided to put together concepts and results of our investigation, as they are at the same time linguistic and formal in nature. In other words, there is no dedicated chapter to the formal modelling [3] or to linguistic description, simply because the insights we have on one side have an immediate and direct effect to the other side, and writing in such a separate way would lead to unnecessary intricacies in order to let the reader understand our line of reasoning.

Of course, we do not pretend to reinvent the wheel; our work is well rooted both in the linguistic and mathematical traditions of their own. The sequel of this chapter is devoted to clarify what are our start points and how they are related (or not) with other works in the fields.

1.1 The adpositional paradigm

It is worth noticing that the term 'constructive' is used in different and complementary ways within the book. In this section, we explain 'constructive' referring to constructive mathematics. The fundamental concept of this book is the *adposition*, which is linguistically a generalisation of conjunctions, prepositions, postpositions, and so on—broadly speaking, linking words. The adpositional paradigm was the main result of the PhD dissertation by Gobbo (2009).

Since then, we put our work a step forward, and this book is the presentation of the results of that effort. In fact, now adpositions—the cornerstones of the adpositional paradigm—are described in the spirit of constructive mathematics, and that's why the framework is called 'constructive adpositional grammars' (Constructive AdGrams for short).

Constructive mathematics (Bridges and Richman, 1987) is, synthetically, a way to develop the mathematical thought that strictly preserves the information content of any statement. Precisely, disjunctive and existential statements are required to indicate witnesses for their truth: for example, $\exists x.P(x)$ can be proved in some theory T if and only if we can exhibit some value v such that $P(v)$ holds, i.e., it logically descends from T. So, some forms of logical thinking are not accepted, e.g., the Law of Excluded Middle, since they introduce unjustified information: in fact, if $P \vee \neg P$ is an axiom, it does not indicate whether the first or the second disjunct holds and there is no guarantee that such an information can be recovered from the theory T, and, in general, it cannot (Troelstra and Schwichtenberg, 2000; Troelstra, 1977).

The term 'grammars' refers to the fact that the mathematical structure is one, while the instantiations are many as natural languages are. We look at different shadows (i.e., natural languages), each one with its different shape, while the sun is always the same (the structure)—see Chapters 2 and 3 for details.

Before proceeding, it is important to explain how we deal with the 'shadows', i.e., the diversity of natural languages—in other words, the problem of linguistic universals and typology. From a phylogenic point of view, while every species seems to understand their members, only humans developed different languages—actually, more than 6,000. Furthermore, from an ontogenetic point of view, every child builds his own linguistic repertoire according to the environment he is grown up into, not his ethnic origin, and there are some regularities in the developing of language skills, from baby utterances to adult-like representations. According to Tomasello (2003), the most rigorous and plausible theory of language that characterises adult linguistic competence in child-friendly terms is rooted both in cognitive and social skills. From the cognitive side, the theory of mind has shown us that our ability to communicate is based on (a) intention-reading, i.e., the ability of inferring what the listener is expecting from us, beyond the literal meaning of what the speaker said; (b) pattern-finding, i.e., the ability to categorise our sensibilia in a mapping into the mind. From the social side, we learn the intentional actions of other humans by imitation.

These universals of learning (intention-reading, pattern-finding and imitation) applied to our innate linguistic ability are the fundamental elements of linguistic *constructions*—synthetically, patterns of usage of form-meaning correspondences that carry the messages beyond the gist of the words that took part in the construction itself. This fact leads us to another use of the term 'constructive' within this book.

Linguistic communication is symbolic, and symbols are "social conventions by means of which one individual attempts to share attention or mental state to something in the outside world" (Tomasello, 2003, 8). Symbols are used in patterned ways, and these patterns give form to linguistic *constructions*. Hence, constructions arise by two different forces: by the meaning of their parts, and by the frequency of use of the pattern itself. This species-unique process occurs over time and is called *grammaticalization*.[4]

The results of this cultural and historical process are natural language grammars, and that is why we have more than one. Therefore, grammars should be investigated as a result of the process of grammaticalization, not as static, monolithic entities. Most puzzling linguistic phenomena can be enlightened in terms of the grammaticalization process, as we will explain throughout this book. In fact, the frequency of pattern-usage leads to loose (hide) unnecessary information: that's why highly frequent words are usually short in terms of syllables.

Each grammar is made of constructions, providing us the linguistic data to be analysed. Constructions being central, our work can be put into relation with the early stages of transformational grammar (Chomsky, 1957, 1965) and to the works of cognitive linguists proposing constructional analyses, such as Tomasello (2003), Goldberg (1995, 2006), and Croft (2001).

Most of our linguistic observations derive from the works by cognitive linguists. The main limit of these analyses lies in the fact that they are *constructional* instead of constructive. That is, they take constructions as primitives, rejecting *in toto* any formalism, which is seen as inevitably 'Chomskyan' in nature. Matthews (1993) already observed:

> […] for more of the past fifteen years, despite occasional disparagement from one side or another, each school has in practice had little reason to refer to the other. It is worth noting, for example, that Croft's recent introduction to *Typology and Universals* (1990) cites no work by Chomsky. (Matthews, 1993, 45)

The situation is even getting worse, as exemplified by the most radical constructional approach, which states that there is no linguistic category that is both formal and universal, as asserted by (Croft, 2001, 4):

> Of course, abandoning universal categories and relations leads to a very different view of Universal Grammar. Under the alternative view, Universal Grammar does not consist of an inventory of universal categories and relations available to all speakers. […] The formal structures in grammars are language-particular, and universals of language must be sought elsewhere. […] In principle, that appears to be the direction that Chomskyan generative grammar has headed: general constraints on syntactic structure but a proliferation of syntactic categories. In practice, however, the syntactic categories are assumed to be cross-linguistically valid, and the same categories (or a subset thereof) are posited of every language. This practice also holds for other formal syntactic theories.

We claim that a formalism is possible without neglecting the usage-based results of cognitive linguists. The problem is in the general perspective. Most cognitive linguists are 'maximalists', i.e., they consider semantics and syntax indissociable, and therefore semantics is put at the centre of analysis, while syntax is put at the periphery—traditionally, most results in cognitive linguistics are devoted to the semantic level, in particular the analysis of metaphors (Lakoff, 1997, for example). Conversely, in whatever (linguistic) formalism, the algebraic rules are insensitive to the meanings of the element they algorithmically combine, and hence linguistic meaning is put at the periphery of the system, the (morphosyntactic) rules being the core.

Nonetheless, we take into account the critique of later Chomskyan development (Chomsky, 1981, 1992) that constructions cannot be disregarded as epiphenomena, while the earlier Chomskyan models showed that the formalisation of constructions is not only possible but also feasible (Goldberg, 1995, 1). After all, Chomsky is a leading figure in the Cognitive Revolution, along with George Miller, Marvin Minsky, Allen Newell and Herbert Simon [5]: we claim that at least part of Chomsky's work can be put in the stream of cognitive linguistics, if we do not reject formalisation as a whole. At the same time, we believe that another way—not based on constituents—to formalise grammars is possible and worth attention, particularly for cognitive linguistics. [6]

In sum, it's no more the time of the "linguistic wars" (Harris, 1995): we claim that both approaches are valid, as they talk about different aspects of the same phenomenon, i.e., natural language grammars: the Chomskyan approach is top-down, deductive, because it looks for regularities beyond the variety of languages, while cognitive linguists try to explain variety on a usage-based perspective, following a data-driven, inductive approach. Our aim is to take the best practices from both approaches, without adhering to any 'Church'. Here we offer a strong, general, explicative formalism with a lot of examples taken from different languages of the world, with a special regard to English.

Our formalism takes into account the results of linguistic typology. Are there any language universals? The cross-analyses of grammars made by typologists showed us that grammar categories, arisen in Greek and Roman context for educational purposes—i.e., teaching and learning of Greek and Latin as written languages—cannot be forced as such into native languages of most part of Africa, Southeast Asia, the Americas and Australia. Should we look through the Procrustean lens of Standard Average European to those languages? Certainly not. Tomasello (2003) takes a radical conclusion, claiming that no universals of form exist, in particular no linguistic symbols, no syntactic constructions and no grammatical categories. If linguistic symbols highly depend on the socio-cultural context within they emerge (but what about the word *mama*?) and typological studies have shown that the variety of syntactic constructions is really impressive, nevertheless this variety *should* have a common, cognitive basis—and Greenberg's results in syntactic typology cannot be disregarded so easily. Tomasello claims that the only language universals that exist are the ones of communication and cognition, being in particular the presence of expressions of reference and the presence of predication. Moreover, they have no counterpart in terms of linguistic universals of form. We claim exactly the opposite: the mechanisms underlying the constructions are the linguistic counterparts of the cognitive ability of pattern-finding and intention-reading, which let the speaker and the listener—in the simpler, default case—to build up the linguistic representation within their joint attentional frame. The best account of the attention focus as a cognitive linguistic ability we have found in literature is the trajectory/landmark asymmetry by Langacker (1987, 1990, 1991), of which we will give a formal interpretation in terms of information prominence—see Chapter 2.

Attention should be directed somewhere in order to function properly: we claim that four grammar characters are universal, being the expressions of reference, predication, and their respective modifiers—respectively, adjuncts and circumstantials. Whorf (1945) and Tesnière (1959) did come to the same conclusion, even if starting from different bases. In particular, the work by Tesnière (1959) was the start of the so-called Dependency Grammars, which was rarely put into relation with cognitive linguistics.[7] We deeply analysed that classic work through the lens of modern formalisms and we have found that most, if not all, derived works have taken the concepts of dependency and verbal valency from Tèsniere, but, at the same time, completely disregarding his use of grammar characters.[8]

Moreover, we retain the concept of 'prepositional system', borrowed from Pennacchietti's analysis of Brøndal (1940): each natural language grammar has a relatively small set of prepositions (or other kind of adpositions) whose function is determined by the result of the opposition with the other prepositions in the grammar itself. In fact, even genetically close languages—such as French and Italian—show a considerable difference in the use of their prepositions because they belong to different prepositional systems (Pennacchietti, 2009, 2006).

The central role given to adpositions—broadly, prepositions, postpositions or in-positions, depending on the specific language—lead to another fundamental grammar character: adpositions. This grammar character collects the 'structural morphemes' of a language. Because of its somewhat technical nature, it has been neglected for a long time. The name 'adpositional paradigm' followed naturally.

The adpositional paradigm was first developed in Gobbo (2009), which extends Pennacchietti's work in an original way. Adpositions become a more abstract, general element in order to understand grammar structures. In particular, zero-marked adpositions—signed through an epsilon (ϵ)—put word order phenomena in the same realm of morphology. Thus morphology and syntax are clarified through a unified model and a unique mechanism—with some special features of their own. Semantics and pragmatics also have their place in the model, while phonetics and phonology have not.[9]

A remark made by an anonymous reviewer on the linguistic foundations of the adpositional paradigm was that the model was tested only into nominative-accusative grammar cases. We have taken this remark very seriously, and the present book fills this gap, providing a new constructive adpositional grammar model which takes into account both nominative-accusative and ergativeabsolutive grammars. Our reference for ergativity under a theoretical point of view is the work by Dixon (1994), who notes in the appendix that, surprisingly, there are few theoretical models which gives an explanation of ergativity, because the theory “would have to recognise that there are three basic syntactic-semantic primitives (A, S and O) rather than just two (‘subject’ and ‘object’) however these are defined”, see (Dixon, 1994, 236).

Constructive adpositional grammars solve this problem in terms of information prominence, as explained in the next chapters. Furthermore, we adhere to the unaccusative hypothesis, originally proposed by Perlmutter (1978), which gives a clear and convincing linguistic account of the problem, even if the Relational Grammar (pseudo)formalism is not convincing. [10]

Finally, no agreement is possible on a finite, fixed list of the universal types of semantic roles—called in a Chomskyan perspective ‘theta-roles’—simply because they depend on the constructions they belong to. In other words, semantic roles are construction-dependent and hence language-defined, and so their list is open and undefined, at least for the purposes of the present book. [11]

For example, if a construction deals with food, such as *X eat Y* it has much more sense to call *X* Eater instead of Agent, following the tenets of cognitive linguistics, as the semantic roles belonging to the semantic frame of food activated by a constructions such as *X eat Y* show distinctive characteristics in the distributional analysis of the corpus in different languages—e.g., English, German, and Bengali (Croft, 2009)— which cannot be explained with a suitable degree of precision by the more used dichotomy Agent *vs.* Patient.

For example, the English construction *Y-eating X*, exemplified by *oil-eating microbes*, states that *Y* is the Food and *X* is the Non-human eater, prototypically having the sememe animal, so *oil-eating robot* is also acceptable, while *fish-eating child* is not, since *child* is not a Non-human eater. [12]

What it is important to underline here is that the *productivity* of constructions—i.e, how much *X*s and *Y*s can vary within the scope of that particular construction—is bound to the construction itself.

The aim of semantic roles is exactly to represent this degree of freedom, which can be zero at the limit, as in the case of completely grammaticalized idiomatic expressions—such as *kick the bucket*, where each lexical item cannot be moved paradigmatically—or very high, such as *Let’s X!*, where the only constraint applied to *X* is its grammar character, in particular the need to belong to the class of English verbs.

1.2 The formal model

In the fields of mathematical and computational linguistics there are many natural language grammar formalisms currently under investigation. In particular, our formalism can be put into the realm of the so-called 'categorial grammars'—i.e., representations of natural language grammars in terms of categories. This line of research is far from being new, being rooted in the works by Ajdukiewicz (1935), Church (1940), Bar-Hillel (1953), and Lambek (1958). [13]

Our formal model is intended as a guiding reference for the development of linguistic concepts. In this sense, it should be understood as a 'weak' model which provides enough insight to understand the complexities of natural languages but not yet enough power to treat them in a computational fashion. The full details of the formal model are presented in mathematical terms in Appendix B: the reader is referred to that part for the formal details.

When the formal model is applied to any natural language, a parser, i.e., a computer program which allows to recognise the expressions belonging to the grammar, can be naturally derived, due to the constructive nature of the formal representation. [14]

The starting point is to consider any linguistic expression as a formal object, represented as an adpositional tree (adtree, for short) on morphemes. The precise structure of such trees is explained in the following chapters; for the moment being, we just need to know that each of them represent a unique piece of text in some natural language. Moreover, a tree represents a 'unique interpretation' of the corresponding text, i.e., a unique way to understand the grammar characters of each element in the text (the concept of 'grammar character' will be discussed and explained in detail in Chapter 3). For example, let's take the Italian sentence:

(1-1.) La vecchia porta la sbarra.

Example (1-1) can be understood in two completely different ways: *the old lady brings the bar* or *the old door bars her*. In the first interpretation, *la vecchia* (the old lady) counts as a noun and plays the role of the first actant, *porta* (brings) counts as the bivalent verb, *la sbarra* (the bar) again counts as a noun, playing the role of the second actant. In the second interpretation, *la vecchia porta* (the old door) is grouped as the first actant, while *sbarra* (bars) acts as the bivalent verb. Finally, the pronoun *la* (her) is a still unresolved anaphoric place-marker of a noun, playing the role of the second actant.

The two interpretations correspond to two similar but distinct adtrees [15], as shown in Figure 1.1. Both interpretations are fully acceptable as the ambiguity

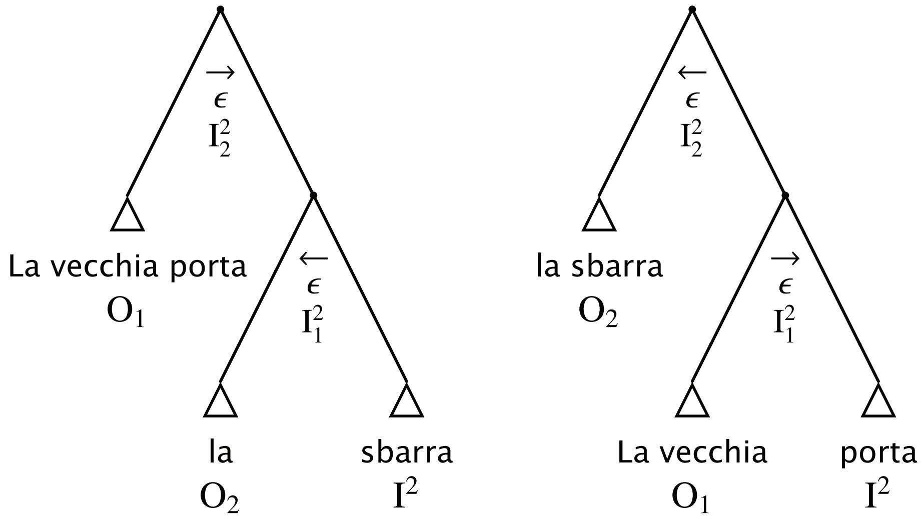

Figure 1.1: The two interpretations of *la vecchia porta la sbarra*

stands in the parser of the Italian language. In fact, two chains of constructions equally valid and semantically plausible can generate the respective adpositional trees. Also in English there are some known examples of sentence where there is more than one valid syntactic relationship, for example:

(1-2.) Time flies like an arrow and fruit flies like a banana.

where 'flies' counts as a verb in the first occurrence and as part of the noun in the second one.

By contrast, there are cases where there is only one adpositional tree but there is more than one semantic interpretation.

(1-3.) È la rossa di Maranello.

Example (1-3) can be interpreted as *She's the red-haired girl coming from Maranello*, an Italian town in the Emilia-Romagna region, as well as *it's the Ferrari car*, meaning the famous sport car produced in Maranello.

In this case, we have two different meanings for the same text, under the same syntactic interpretation: the adtree will be the same (Figure 1.2).

Adtrees are built from some basic bricks via a few standard constructions. Correctly identifying the bricks and the basic constructions is the main objective of this book. But, let's suppose to have them: then, we can generate the whole language and, vice versa, we can check if a piece of text is syntactically correct by asking if there is a corresponding adtree. Moreover, we would say that a piece of text is syntactically ambiguous if it admits more than one correct adtree. [16]

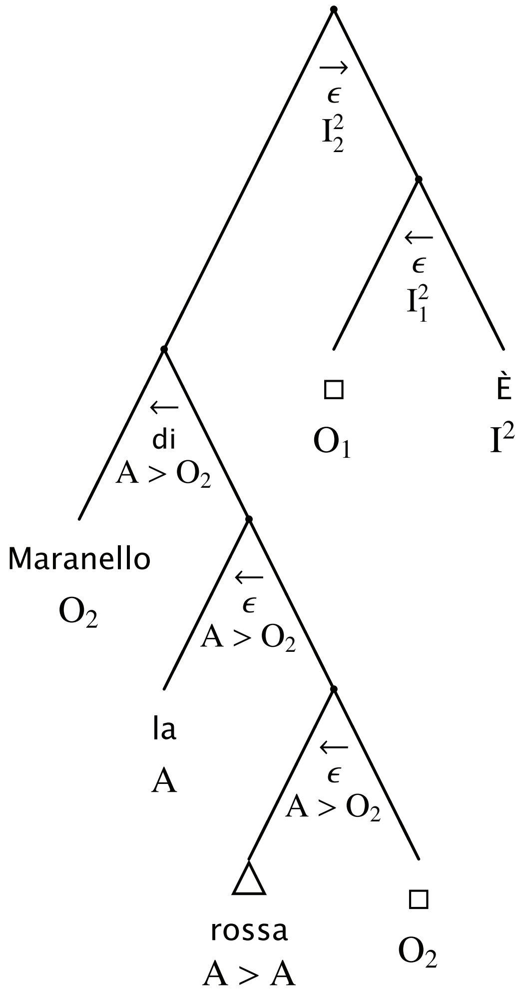

Figure 1.2: The adpositional tree of *È la rossa di Maranello*

In mathematical terms, adtrees and constructions between them form a structure **AdTree** which is a *category*[17], see Mac Lane (1998) and Borceux (1994). A mathematical category is an algebraic structure composed by two classes, the *objects* and the *arrows*; arrows lie between two objects, the *source* or *domain*, and the *target* or *codomain*. Also, a category states that there are distinct arrows, the *identities*, one for every object A and such that the source and the target are A. Moreover, a category is equipped with a partial operation allowing to compose two arrows whenever one has the domain which is the target of the other one. Composition is required to be associative and identities act as one expects with respect to composition.

Intuitively, there is an arrow f from A to B whenever we can construct the B tree starting from the A tree applying the construction f. We do allow complex constructions obtained by sequentially composing simpler ones; if f and g are constructions such that $f(A) = B$ and $g(B) = C$, that is, if f maps A into B, and

g constructs C from B, then $g \circ f$ is the construction which maps A into C by doing g after f.[18]

We observe that, calling M the free monoid over the alphabet of morphemes of some natural language, i.e., the set of all possible (finite) sequences of morphemes obtained by juxtaposition, the functions mapping the trees in **Adtree** into the sequences of M comprehend the textual renderings of adpositional trees. If we restrict our attention to *contravariant functors*, i.e., the functions preserving the identical transformation and the reverse composition of adpositional trees, we get a class of functions which is called *presheaves over M*. Requiring that a presheaf maps morphemes in the adtree into themselves in the monoid, we get exactly the lexicalizations of adtrees. In other words, there is a subclass of presheaves which directly corresponds to the texts the adtrees represent and which encodes the transformations that constitute the grammar. It is this space of presheaves which is generally understood as the subject of linguistics.

It is possible and fruitful to interpret the basic constructions of Category Theory, e.g., natural transformations, limits and colimits, etc., in the framework we have just introduced. But it requires a better understanding of the basic blocks of the **AdTree** category. So, we stop here in our analysis, at least until the following chapters will introduce the required elements.

As a side effect of this intended model of interpretation, it follows that whatever construction over adtrees which is built by combinatorially composing the fundamental constructions, is an arrow. Lifting the structure of the **AdTree** category into the spaces of presheaves, which is a category, we can reason in a larger and richer environment, where the full power of mathematical methods can be applied: in fact, the presheaves space is a *Grothendieck topos* (Mac Lane and Moerdijk, 1992; Johnstone, 2002), one of the richest mathematical structures we can deal with.[19]

As we have suggested, categorial mathematics give a set of elegant tools to build natural language grammars because it provides a transparent account of the correspondence between the syntactic and the semantic combinatorics—in other words, how meaning is reflected into collocation and word order phenomena, hypothesis put at first by Montague (1973) in Hintikka et al. (1974).

From a mathematical point of view, formalisms based on categories,[20] usually employ some variant of the Lambek calculus, because it corresponds to a non-commutative, substructural variant of linear logic. So, Lambek calculus, as well as derived formalisms, is based on a logical system where we have limited *resources*, allowing to model, e.g., that a phrase may contain at most one verb, and connectives are not commutative, modelling the ordering of elements in a phrase. An important result about pure categorial grammars is (Pentus, 1993) that shows how their generative power is that of context-free grammars and thus inadequate for theories of natural language syntax; this result applies only to categorial grammars which have a natural computational counterpart defined as an extension of the λ-calculus, i.e., where composition rules are functional applications. Unfortunately, leaving the simple and safe world of λ-abstraction and application in favour of more sophisticated rules allows for an explanation of more complex linguistic phenomena, but at the price of having models whose behaviour is far from being well-understood. In fact, they have been nearly abandoned for the automatic processing of natural languages.

A modern formalism currently under development is the Combinatory Categorial Grammar (CCG) by Steedman and Baldridge (2007), a lexicalized grammar formalism which can be parsed in non-deterministic polynomial time. It is especially used in the field of statistical parsing. Linguistic categories are assigned to words by the lexicon, e.g., an intransitive verb has the category $S\backslash NP$ while a transitive verb has the category $(S\backslash NP)/NP$. Under a theoretical linguistic point of view, a CCG is constituent-driven, i.e., it retains the advantages and limits of the Chomskyan syntactic perspective we put into question here, so we can't use CCGs as such. Under a mathematical point of view, a grammar is a set of inference rules controlled by the linguistic categories, interpreted as functional spaces, see van Benthem (1995); such rules can be inferred by suitable techniques from machine learning (Manning and Schütze, 1999), essentially using statistical measures. Also, the probabilistic approach is used to cope with the fact that non-deterministic polynomial problems cannot be efficiently solved, as far as modern mathematics knows. [21]

Ranta (2004) proposes a formal framework for writing grammars and linguistic theories—following the motto *sentences-as-proofs*. From a mathematical point of view, it is based on Martin Löf Type Theory, see (Martin Löf, 1984), using the Coquand's algorithm for type construction/partial proof derivation (Coquand, 1996). So, it is ultimately based on a dependent-type, predicative variant of the λ-calculus.

There are two advantages in this line of research: first, the Grammatical Framework is a piece of software which can be used freely and it has a growing up community (Ranta, 2009); second, it might be used for implementation regardless of the linguistic theory behind. However, software implementation is out of the scope of the present book, which is dedicated to linguistic analysis.

Dynamic Syntax is based on the program of Labelled Deductive Systems, which aims to bring semantics back into syntax (Kempson et al., 2001). Labelled formulae and deductive systems are connected with algebraic frames and type logical grammars. In fact, Labelled Deductive Systems have been first introduced by Gabbay (1996) as an algorithmic way to control proof development using labels from some simple algebra whose operations are used to model the potential application of inference rules.

Unlike Chomskyan grammars, in Dynamic Syntax the trees represent the structure of the sentences first as the result of a growth obtained by a parsing strings uttered in contexts—this is the sentence's 'dynamics'. As it will be clear in the sequel of this book, we follow the hypothesis posed by Dynamic Syntax that (morpho)syntax is the mechanism for constructing representations of content. However, there are important differences, namely the use of linguistic constructions—based on cognitive linguistics results—is formally resolved in a totally different way. Moreover, while Dynamic Syntax makes use of Hilbert's epsilon calculus in order to solve quantified noun phrases, our formal bases are completely different.

1.3 What is in this book?

The aim of the present work is foundational. In particular, we want to give the right hints to understand natural language grammars within the adpositional paradigm. For us, 'understanding' means both having insights about the functioning of real-world usage of a given natural language—in terms that are comparable with other languages as well—and having insights about what is structural in a given natural language, in constructive mathematical terms. In practice, there are no distinct "linguistic *vs*. mathematical" insights; rather, the joint linguistic and mathematical apparatus should be regarded as the two sides of the same coin.

Under a different perspective, this book is foundational as here the reader can acquire all the instruments necessary for the founding of Constructive Linguistics, i.e., an application of constructive mathematics to the realm of linguistics—broadly, the study of natural languages. When foundations are posed, many building details are still to be defined. In this book, we indicate which details, instantiations, specific phenomena are not covered, in order to help the reader see possible further directions of this work, standing on the solidity of the foundations posed here.

A crucial part of the foundations are the instruments from Category Theory, in particular, the intuitions about sheaves and Grothendieck topologies—the impatient reader is invited to read Appendix B at first.

The choice of the mathematical instruments apt to represent the results obtained within the adpositional paradigm was lead by two simple criteria: expressiveness and naturalness. From the point of view of expressiveness, it is easier to obtain more general result with an elegant formalism than narrower results with a less powerful formalism, and topos-theory is the strongest, general, elegant and broad formalism belonging to constructive mathematics we have nowadays.

On the other hand, the representation of linguistic phenomena in constructive mathematical terms should be natural. In other words, the mathematical constructs we use should have a direct and immediate counterpart in linguistic terms: in that way, readers interested only in the linguistic side can still follow the intuition of the model, without being forced to use abstract tools whose linguistic nature is not clear. Nevertheless, we invite even readers not used to mathematical formalisms to give a chance to our formal model, in reading the whole book in the order of presentation, even Appendix B. We are confident that—after the reading and understanding of the main text—the Appendix will be readable with profit and interest to all readers.

Appendix B

The Formal Model

The purpose of this appendix is to describe the formal model as a standalone mathematical entity.

The motivations behind its structure lie in linguistics and they have been described at wide and in depth in the main chapters of the present book. Here, we want to abstract over the linguistic context in order to show the formal model, which guided our investigations and enabled our findings, in its purest form.

The purpose of the formal model is to provide a natural framework to describe the basic structures of the grammars and the languages, focusing on the adpositional paradigm. The formal model has not been conceived to provide a parser for the language or an algorithm to generate all the well-formed productions of a language, although, to some extent, it can be used to obtain these results as side-effects.

In our experience, accumulated in the years devoted to study the adpositional approach and in writing this book, the formal model proved to be a formidable tool to clear thinking, as it often showed naturally and easily the correct way to interpret the most intricate and obscure linguistic phenomena.

Also, many unification results we have introduced in the text has been derived by firstly representing a phenomenon in the model and then, by observing that the obtained representation was wider and deeper than initially planned, eventually modelling other phenomena. So, we decided to include a complete mathematical presentation of the formal model, to provide the reader with the same instruments we used in pursuing our discoveries.

But we have to warn the reader that, in the following, deviating from the practice of previous chapters, we assume confidence with the fundamental concepts, definitions and results of basic Category Theory, and some degree of mathematical maturity.

Specifically, we assume as given the notions and results in Mac Lane (1998) and Borceux (1994), and we suggest some acquaintance with the notion of *sheaf* and *Grothendieck topology* (Johnstone, 2002; Mac Lane and Moerdijk, 1992) in order to track the inspiration and the possible future developments of the model.

B.1 Grammar categories

In the following we deal with general categories. As an intuitive guideline, the reader may think to objects in a category as adtrees and to arrows as constructions. This view, although inspiring, needs some work to be consistent, so we will introduce a series of definitions to fix the fundamental properties leading to the intended interpretation.

The fundamental property of grammar characters is that every object has exactly one grammar character. Also, we require that the set of grammar characters is meaningful, i.e., every grammar character contains at least one object.

Definition B.1.1 (Grammar character) *Given a small category $\mathbb{C}$, we say that G is a set of* grammar characters on $\mathbb{C}$ *if G is a partition on* Obj $\mathbb{C}$*, the set of objects in $\mathbb{C}$, such that no class is empty.*

Formally, $G = \{G_i\}_{i \in GC}$ such that $\bigcup_{i \in GC} G_i = \mathrm{Obj}\,\mathbb{C}$ and, for any $i, j \in GC$, if $i \neq j$ then $G_i \cap G_j = \emptyset$, and $G_i \neq \emptyset$ otherwise.

In order to exactly define what we intend for constructions, we need the technical concept of grammar product.

Definition B.1.2 (Grammar product) *Fixed a small category $\mathbb{C}$, a set $G = \{G_i\}_{i \in GC}$ of grammar characters with $W \in G$, we call* grammar product on $\mathbb{C}$, G, W *any pair $\langle Y_1 \times \cdots \times Y_n, (z_1, \ldots, z_n)\rangle$ where n is a natural number called the* degree *of the product, $\times$ is the Cartesian product of sets and, for every i in $1 \ldots n$, $Y_i \in G \setminus \{W\}$, and $z_i \in W$.*

The reader is invited to notice that grammar products are not products in the category $\mathbb{C}$—we do not even require that $\mathbb{C}$ has products. Instead, they should be thought of as finite products of grammar characters, where the i-th projector is named by an element of W, which denotes the class of adpositions.

Constructions are families of arrows with the same domain and whose codomain lies in a single grammar character which is not the one of adpositions. These arrows are indexed by a grammar product.

Definition B.1.3 (Grammar construction) *Given a small category $\mathbb{C}$, a set $G = \{G_i\}_{i \in GC}$ of grammar characters with $W \in G$, a* grammar construction $\eta_{P,x}$ over P from x *is an indexed collection $\left\{f_j\right\}_{j \in P}$, where*

- *$P = \langle P', a \rangle$ is a grammar product on $\mathbb{C}$, G, W;*
- *x is an object of $\mathbb{C}$;*
- *there is $V \in G \setminus \{W\}$ such that, for every $j \in P'$, $f_j \in \mathrm{Hom}_{\mathbb{C}}(x, y)$ where $y \in V$.*

We denote the set V by $\mathrm{cod}\,\eta_{P,x}$*, the codomain of the grammar construction. Similarly,* $\mathrm{dom}\,\eta_{P,x} = x$ *and we call P the product of $\eta_{P,x}$. Sometimes, we will write $\eta_{j,x}$ for f_j.*

A family $\{\eta_{P,x}\}_{x \in U}$ of grammar constructions over P from x varying in the U grammar character forms a grammar construction $\eta_{P,U}$ over P from U *when all its elements share the same codomain. We denote the common codomain with* $\mathrm{cod}\,\eta_{P,U}$*. Also,* $\mathrm{dom}\,\eta_{P,U} = U$ *and P is the product of the construction. Again, we will write $\eta_{j,x}$ to indicate the j-th component of the x-th component of $\eta_{P,U}$.*

We say that the construction η over $P = \langle Y_1 \times \cdots \times Y_n, (z_1, \ldots, z_n) \rangle$ on the object x, has x as *governor* and its *dependents* are the elements of $Y_1, \ldots, Y_n$; its adpositions are $z_1, \ldots, z_n$. So, its product P describes the adpositional structure that η imposes over the governor.

For symmetry, given a construction η as above, it is useful to consider the constructions which differ from η by exchanging a dependent with the governor.

Definition B.1.4 (Conjugate construction) *Fixed a small category $\mathbb{C}$, a set $G = \{G_i\}_{i \in GC}$ of grammar characters with $W \in G$, and a grammar construction*

$$\eta = \left\{ \left\{ f_j \colon x \to c(x, j) \right\}_{j \in P'} \right\}_{x \in V}$$

over the product $P = \langle P' = P_1 \times \cdots \times P_n, (a_1, \ldots, a_n) \rangle$ from the V grammar character, a grammar construction θ over the product Q from the grammar character U is said to be conjugate to η *if, for some k in* $1 \ldots n$*, $U = P_k$ and*

$$Q = \langle P_1 \times \cdots \times P_{k-1} \times V \times P_{k+1} \times \cdots P_n, (a_k, a_2 \ldots, a_{k-1}, a_1, a_{k+1} \ldots, a_n) \rangle$$

and, for all $(x, j_1, \ldots, j_n) \in C \times P'$, $\mathrm{cod}\,\eta_{(j_1,\ldots,j_n),x} = \mathrm{cod}\,\theta_{(j_1,\ldots,j_{k-1},x,j_{k+1},\ldots,j_n),j_k}$.

We notice that, in general, a construction may have any number of conjugates for each elements in its product, even zero.

Constructions from grammar categories can be composed, to obtain more complex constructions.

Definition B.1.5 (Composition of constructions) *Given a small category $\mathbb{C}$, a set $G = \{G_i\}_{i\in GC}$ of grammar characters with $W \in G$, and grammar constructions*

$$\eta = \left\{ \left\{ f_{(y_1,\ldots,y_n)} \colon x \to c(x, y_1, \ldots, y_n) \right\}_{y_1\in P_1,\ldots,y_n\in P_n} \right\}_{x\in V}$$

over the product $P = \langle P_1 \times \cdots \times P_n, (a_1, \ldots, a_n)\rangle$ from the V grammar character and

$$\theta = \left\{ \{ g_{(z_1,\ldots z_m)} \colon x \to d(x, z_1, \ldots, z_m) \}_{z_1\in Q_1,\ldots,z_m\in Q_m} \right\}_{x\in U}$$

over the product $Q = \langle Q_1 \times \cdots \times Q_m, (b_1, \ldots, b_m)\rangle$ from the U grammar character, the composition $\theta \circ \eta$ *is defined when* $\operatorname{dom}\theta = U = \operatorname{cod}\eta$. *In this case,*

$$\theta \circ \eta = \left\{ \left\{ g_{(z_1,\ldots z_m)} \circ f_{(y_1,\ldots y_n)} \right\}_{y_1\in P_1,\ldots,y_n\in P_n, z_1\in Q_1,\ldots,z_m\in Q_m} \right\}_{x\in V}$$

and $\operatorname{dom}\theta\circ\eta = \operatorname{dom}\eta = V$, $\operatorname{cod}\theta\circ\eta = \operatorname{cod}\theta$, *and the product of $\theta \circ \eta$ is*

$$\langle P_1 \times \cdots P_n \times Q_1 \times \cdots \times Q_m, (a_1, \ldots, a_n, b_1, \ldots, b_m)\rangle \quad .$$

As the reader may check, the grammar characters along with all the possible constructions and with the composition just defined, forms a category.

Although this category is of interest, as we will briefly discuss in the following, the kernel of the formal model concentrates upon another category.

Definition B.1.6 (Grammar category) *Consider a small category $\mathbb{C}$, a set of grammar characters on it $G = \{G_i\}_{i\in GC}$ with $W \in G$, and a set C of grammar constructions over some product on $\mathbb{C}$, G, W from a grammar character in G. Then, the structure $\langle \mathbb{C}, G, W, C\rangle$ is said to be a* grammar category *when it satisfies the following properties:*

1. *There is a collection M of objects in $\mathbb{C}$, called the* atoms of the category, *such that, for each $x \in \operatorname{Obj}\mathbb{C}$, there is an arrow f in $\mathbb{C}$ with $\operatorname{dom} f \in M$ and $\operatorname{cod} f = x$; moreover, if $x \in M$, then $f = \mathrm{id}_x$;*

2. *For each arrow $f \colon a \to b$ in $\mathbb{C}$, there is $\eta_{P,V} \in C$ such that $a \in V$ and $f \in \eta_{P,a}$, where $\eta_{P,a}$ is the a-component of $\eta_{P,V}$;*

3. *Every construction $\eta \in C$ is the composition of a sequence $\theta_1, \ldots, \theta_n$ of constructions in C each one having degree* 1*;*

4. *Each construction in C has exactly one conjugate construction for every element in its product.*

Informally, a grammar category is a category whose arrows can be grouped into a collection of constructions satisfying a number of regularity properties: the collection comprises all the arrows in the category; every object can be reached from a set of atoms; each construction has all its conjugates; every construction can be reduced to a sequence of one-step constructions.

In the whole book we dealt with a subclass of grammar categories.

Definition B.1.7 (Adpositional category) *A grammar category* $\mathbb{A}$ *is* adpositional *when* $\mathbb{A} = \langle \mathbb{C}, G, U, C \rangle$ *and G is indexed by*

$$GC = \{A, E, O, U\} \cup \left\{ I_p \colon p \text{ is a grammar product on } \mathbb{C}, G, U \right\} \ .$$

We require that the collection of I_p*s is wide enough to allow the definition of all the related constructions, mainly valency.*

B.2 Adtrees

Let $\langle \mathbb{C}, G = \{G_i\}_{i \in GC}, W, C \rangle$ be a grammar category and let η be a grammar construction over $\langle Y_1 \times \cdots \times Y_n, (a_1, \ldots, a_n) \rangle$ from the object x:

$$\eta = \left\{ f_{(y_1,\ldots,y_n)} \colon x \to c_{(x,y_1,\ldots,y_n)} \right\}_{y_1 \in Y_1, \ldots, y_n \in Y_n} \ .$$

We can equivalently write each arrow in η as[a]

$$x \xrightarrow{f_{(y_1,\ldots,y_n)}} a_n(y_n, a_{n-1}(\ldots, a_1(y_1, x)\ldots)) = c(x, y_1, \ldots, y_n) \in \operatorname{cod} \eta \ .$$

The above representation is evidently unique, as far as x is the governor and $(y_1, \ldots, y_n)$ are the dependents taken in the given order, so we have shown that every object in a grammar category can be represented as an adtree whose leaves are atoms. Of course, if we take a conjugate construction [1] η^*, the governor would be y_k and the dependents would be $(y_1, \ldots, y_{k-1}, x, y_{k+1}, \ldots, y_n)$, corresponding to the representation

$$a_n(y_n, a_{n-1}(\ldots, a_{k_1}(y_{k-1}, a_1(x, a_{k+1}(\ldots(a_k(y_1, y_k))\ldots))\ldots))$$

[a] We use linearised trees for compactness.

of the same object $c(x, y_1, \ldots, y_n)$.

To some extent, the converse also holds: if we take as objects the adtrees, as arrows the constructions described in the book along with their conjugates, we obtain an adpositional category. In this case, we have that a linguistic construction η generates a set of grammar constructions, itself plus the constructions which are conjugates to η.

It is interesting to notice that grammar categories, when thought of as categories of adtrees, give raise to a category of grammar characters.

Theorem B.2.1 *Fixed a grammar category* $\langle \mathbb{C}, G = \{G_i\}_{i\in GC}, W, C\rangle$*, if*

$$\eta = \left\{\left\{f_{(x,y_1,\ldots,y_n)} : x \to c_{(x,y_1,\ldots,y_n)}\right\}_{y_1\in Y_1,\ldots,y_n\in Y_n}\right\}_{x\in V}$$

is a construction over $\langle Y_1 \times \cdots \times Y_n, (a_1, \ldots, a_n)\rangle$ *from the grammar character* V*, then, for every* $(y_1, \ldots, y_n) \in Y_1 \times \cdots \times Y_n$*, consider*

$$\theta_{(y_1,\ldots,y_n)} = \left\{f_{(x,y_1,\ldots,y_n)}\right\}_{x\in V} .$$

The structure having GC as objects and the θ*'s above as arrows, is a category*

Proof: Immediate from the definitions, noticing that each θ is a set-theoretic function. ∎

The meaning of the theorem is that we are allowed to construct new grammar categories literally over the old ones, by putting new grammar characters and new constructions over the category of the theorem. This operation is exactly what we have done when moving from the morphemic constructions to the adtree constructions in Chapter 4.

A property of grammar category which is important for computational purposes is that every object can be generated from atoms and constructions.

If a grammar category is such that every object can be represented by an adtree which is unique modulo conjugates, we will say that the category is *pure*. In this case, having a dictionary, which describes the set of atoms and the possible constructions, allows to generate all the possible adtrees.

In the adpositional case, it is evident that some adtrees will not correspond to well-formed linguistic productions, because, e.g., they will not agree on the gender. In general, all the adtrees generated via constructions from atoms will have a correct structure, but their leaves may not conform to the redundancies rules that every natural language has. In principle, it should be possible to cope with redundancies augmenting the number of grammar characters, as it is usual in the formal Chomsky grammars. But, in practice, this approach would multiply the number of constructions as well, obscuring the structural unity of a language. In this respect, we prefer to accept the generation of adtrees which fail to meet the redundancies of the language; we claim that it is possible to check redundancies in an adtree and to correct them after the generation of the adtree.

It is impossible to prove the previous claim except by an exhaustive analysis of the redundancies in the existing natural languages. And, we are not going to do that. Instead, we support the claim by observing that redundancy rules are always "local", that is, they apply on specific (and small, in practice) subtrees of a given adtree. Also, such rules can be expressed as a tree-pattern which induces a substitution on the match.

For example, a rule for articles *vs* nouns in Italian is *if the noun is feminine, the article must be feminine as well; if not, change it to a feminine article!*. We can formalise this rule as *if* $\epsilon(L, N)$ *is your adtree, and it is constructed via the "determiners" construction (see Chapter 3), and N has the "feminine" attribute, which is an attribute in the grammar category of morphemes, and L has not the "feminine" attribute, then substitute "la" for L.* Of course, in a complex adtree, we should apply recursively this rule to every subtree.

In this respect, the computational analysis of algorithms on adtrees can be somewhat simplified by the following result. We say that an adtree is *linear* when its governor and all its dependents are atoms. Moreover, if the adtree $a(x, T_0)$ is obtained by means of the η construction, then $a(T_0, x)$ is obtained by means of the η^* conjugate construction. We say that $T_1 \sim T_2$ when $T_1 = T[a(x, T_0)]$ and $T_2 = T[a(T_0, x)]$, i.e., when T_1 and T_2 differ by a single subtree which appears in the regular form in T_1 and in the conjugate form in T_2, and, evidently it holds that the object denoted by T_1 equals the object denoted by T_2. We denote by $\approx$ the reflexive, symmetric and transitive closure of the $\sim$ relation. Evidently, $\approx$ is an equivalence relation over adtrees. We call it equivalence modulo conjugates, or c-equivalence for short.

Theorem B.2.2 *Let $\mathbb{A}$ be a grammar category. Then, every object of $\mathbb{A}$ is denoted by an adtree x, and there is an adtree y, denoting the same object, such that $x \approx y$ and y is linear.*

Proof: We prove that every adtree T is c-equivalent to some linear adtree L denoting the same object as T. We proceed by induction on the structure of T:

- (Basis) If T is an atom, $L = T$ is linear and c-equivalent to T by reflexivity, and L and T denote the same object;
- (Induction step) If $T = a(T_2, T_1)$ then, by induction hypothesis, there are

$$L_1 = b_k(t_k, b_{k-1}(\ldots, b_1(t_1, t_0)\ldots))$$

and L_2, both linear, such that $T_1 \approx L_1$, $T_2 \approx L_2$, T_1 and L_1 denote the same object, and T_2 and L_2 denote the same object. Thus, $T \approx a(L_2, L_1)$ and they denote the same object, deriving from an application of the same construction to identical objects. So, considering the conjugate of the composed construction,

$$\begin{aligned} T &\approx a(L_2, b_k(t_k, b_{k-1}(\ldots, b_1(t_1, t_0)\ldots)) \approx \\ &\approx b_1(t_0, b_k(t_k, b_{k-1}(\ldots, a(t_1, L_2)\ldots)) = L \ , \end{aligned}$$

and L is linear and denotes the same object as T. ■

Corollary B.2.3 *In a grammar category, every adtree $T[T_1]$, i.e., an adtree having T_1 as a subtree, is c-equivalent to an adtree having T_1 as governor.*

Proof: The same induction argument proves the statement. ■

Corollary B.2.4 *Given a grammar category, the constructions can be partitioned into the basic ones and their conjugates. Then, every adtree constructed via basic constructions, is c-equivalent to a linear adtree and vice versa.*

Proof: A tedious analysis of the proof of the theorem, where we keep track of conjugates, allows to prove the result. ■

Corollary B.2.5 *In a pure grammar category, every object is uniquely identified with the quotient of the set of its adtrees via the $\approx$ relation.*

As a side effect of corollary B.2.4 we get an abstract parsing algorithm for any grammar category. In fact, a linear adtree is the same thing as a sequence of morphemes where adpositions, which can be mechanically extracted, are put in the “right” places. So, the meaning of corollary B.2.4 is that it is possible to reconstruct from such a linear adtree another adtree which is built by basic constructions only, i.e., an adtree in the format we used along all the book. Of course, such an algorithm is "universal", since it applies to any grammar category, but desperately inefficient, since it is highly non-deterministic.

Such a parsing algorithm is unable to distinguish different representations of the same objects unless they are c-equivalent. So, it should be used only on pure grammar categories. By the way, all the grammar categories we dealt with in this monograph are pure.

B.3 Transformations

Several times it has been said that indicators are essential to define and to understand transformations. The reader may have noticed that we have no space for indicators, or other attributes of adpositions and morphemes in the formal model. In fact, there is no need for a distinct treatment of linguistic features like indicators, as well as for the attributes (like gender or number) regulating the redundancies of the language.

Let us consider the case of indicators: if U is the set of adpositional morphemes in the language, it suffices to take $\overline{U} = U \times \{\leftarrow, \rightarrow, \leftrightarrow\}$ as the set of adpositions in the grammar category to ensure that each adposition is decorated by an indicator. Other attributes are treated similarly.

Moreover, we usually require that there is a special morpheme, ϵ, in the set of adpositions, representing the empty adposition. We also require that $\square$ is an element of each grammar character, representing the empty element of that class; to be precise, we should speak of $\square_A$, $\square_E$, $\square_O$, ...which are distinct objects in the grammar category, but the ambiguous $\square$ symbol does not cause any harm in the usual practice as it is always in a context clarifying its correct meaning.

Furthermore, we frequently used cancelled atoms to denote objects which are present in a linguistic structure but they are absent from its textual representation—we said they are "hidden". Again, these elements require no special treatment in the definition of grammar categories: if M is the set of atoms in the language, it suffices to take $\overline{M} = M \sqcup M$, the disjoint union of M with itself, as the set of atoms in the grammar category. The first copy of M stands for the atoms, while the second one is for the cancelled atoms; constructions and grammar characters are defined accordingly, making no distinction whether an atom is cancelled.

These manipulations on atoms and adpositions are silently assumed in the following, as they are frequently used to encode transformations. Also, they are transparent to the formal machinery that constitute the basis of the formal model.

Informally, a transformation is a map from adtrees to adtrees in a given grammar category.

As we have seen in section B.2, fixed a grammar category $\mathbb{C}$, an adtree is a representation of an instance of a construction which makes explicit its product, its dependents and its governor, organising them in a (binary) tree structure. Also, every object in $\mathbb{C}$, except for adpositions, can be represented by an adtree, and usually by more than one.

We will call $\mathrm{Ad}(\mathbb{C})$ the collection of all adtrees in $\mathbb{C}$, plus the empty adtree, denoted by $\perp$; in fact, $\mathrm{Ad}(\mathbb{C})$ is a category whose arrows are the arrows of $\mathbb{C}$, i.e., the instances of the constructions of $\mathbb{C}$, thought of as applied to adtrees instead of being applied to objects. Moreover, we assume that there is a unique arrow from $\perp$ to any other adtree on $\mathbb{C}$, so $\perp$ is an initial object in $\mathrm{Ad}(\mathbb{C})$.

Definition B.3.1 (Transformation) *Given a grammar category* $\mathbb{C}$*, a* transformation on $\mathrm{Ad}(\mathbb{C})$ *is an endofunctor in* $\mathrm{Ad}(\mathbb{C})$.

The previously described redundancy transformations satisfy the above definition. In general, all the "transformations" we considered in this book are transformations in the defined formal sense, except for "ϵ-transformations", which, as it has been remarked, are just a convenient way to condense adtrees, that is, a matter of human readability.

It is worth describing in some detail how redundancy transformations operate, both to clarify how they should be conceived and understood, and to introduce in a formal way the notion of correct adtrees with respect to redundancies.

As an example, consider the redundancy rule which says "if the noun is feminine, modify the article to feminine". In the grammar category for the Italian language, this rule becomes a functor F on adtrees as follows: an adtree T becomes the $F(T)$ adtree where every occurrence of a subtree $\epsilon(a, n)$ with a article and n feminine noun is substituted with $\epsilon(a', n)$ with a' the feminine article; every arrow, i.e., every instance of some construction, is mapped as obvious.

Since every redundancy transformation is analogous to the shown case, the example is completely general. In fact, every such transformation can be reduced to the substitution of an adtree pattern with a variant of its instance. Functoriality ensures the correct spreading of the reduced definition in complex adtrees.

Given an endofunctor F, we say that an adtree T is a *fixed point of F* if $T = F\,T$. Also, we say that the f adtree is *final for F on the adtree t* if f is a fixed point of F and $f = F \circ \cdots \circ F\,t$, i.e., if f is the result of a finite composition of F with itself applied to t. The meaning of this definition is that a transformation F is applied to an adtree t to "correct" its "wrong" redundancies; the transformation is iteratively applied until all the "mistakes" are removed and the resulting adtree does not change anymore.

The set of final adtrees for a redundancy transformation R deprived of $\perp$, the empty adtree, if present, is the set of adtrees which are *correct with respect to* R. Usually, we associate to every grammar category a set T_R of transformations which are responsible for discriminating the *correct adtrees for the grammar category*, which is the set of adtrees correct with respect to each functor in T_R.

In the book appears another type of transformations, the ones used to model complex constructions like auxiliary verbs or the relation between active and passive tenses. These transformations are endofunctors on the category of adtrees of a peculiar class of grammar categories. In some cases, it suffices to consider an adpositional category describing the language, like the English grammar category and the active-passive transformation.

In other cases, the linguistic phenomena under examination can be explained only considering the interplay between the linguistic level, modelled by the linguistic adpositional category, and the corresponding morphemic level, modelled by the morphemic adpositional category.

It has been claimed and explained that phenomena like the participle is a sort of morphemic summary for a more complex linguistic construction which is semantically equivalent. In the following, we will define the exact mathematical framework that enables a formal and precise definition of these phenomena as formal transformations.

As explained in Chapters 2, 3 and 4, the linguistic and morphemic levels of a natural language are modelled by two distinct adpositional categories. The atoms of the linguistic category are the objects corresponding to the correct adtrees for the morphemic category, in the formal sense previously explained. In fact, exploiting this link into an operation that combines the two categories into one, is the mathematical environment allowing to represent the phenomena of interest as transformations, i.e., endofunctors on the combined category.

Assume to have a grammar category $\mathcal{M} = \langle \mathbb{M}, G_M, W_M, C_M \rangle$, called the *category of morphemes*, and another grammar category $\mathcal{L} = \langle \mathbb{L}, G_L, W_L, C_L \rangle$, called the *linguistic category*, and a map $\phi\colon \operatorname{Obj}\mathbb{M} \to \operatorname{Obj}\mathbb{L}$ whose image lies in the atoms of $\mathcal{L}$. Define $C = \langle \mathbb{C}, G, W, C \rangle$ where

- the category $\mathbb{C}$ has as objects $\square_M$ plus $\operatorname{Obj}\mathbb{L} \sqcup \operatorname{Obj}\mathbb{M}$, where $M = \{\square_M\}$

and $\square_M$ is not an object of $\mathbb{L}$ and $\mathbb{M}$; the arrows of $\mathbb{C}$ are the arrows of $\mathbb{L}$ and those of $\mathbb{M}$ plus the instances of ϕ;

- the atoms of C are, clearly, the atoms of $\mathcal{M}$ plus $\square_M$;
- the grammar characters of $\mathcal{M}$ are $G = G_L \sqcup G_M \sqcup M$;
- the adpositions of C are $W = W_L \sqcup W_M \sqcup \{\epsilon_M\}$;
- the constructions of C are

$$C = C_L \sqcup C_M \sqcup \left\{ \left\{ \left\{ \theta_j \colon x \to \phi(x) \right\}_{j \in M} \right\}_{x \in \alpha} : \alpha \in G_M \right\} .$$

Evidently, C is a grammar category.

Informally, C corresponds to the disjoint union of $\mathcal{M}$ and $\mathcal{L}$ where the atoms of $\mathcal{L}$ are substituted, modulo an ϵ-transformation which hides the θ construction, by the morphemes, i.e., the objects of $\mathcal{M}$, via the ϕ correspondence.

In the C grammar category, every endofunctor of $\mathcal{L}$ and $\mathcal{M}$ can be trivially extended to act on C, so to become a transformation on C. Extending the redundancies transformations of $\mathcal{M}$ and $\mathcal{L}$, and considering their union, one naturally defines the concept of correct adtrees on C.

But, the additional endofunctors, the ones not being extension of endofunctors on $\mathcal{L}$ or $\mathcal{M}$, are precisely those that are used to provide the formal counterpart of the transformations we want to model, the ones described in Chapter 4.

B.4 An abstract view

In this section, we want to introduce the preliminaries of a possible future development of the formal model. The idea is to abstract over the concrete details in the definition of a grammar category, and to consider the overall structure from an external point of view.

The core of the abstract view is to consider the language as a formal space where constructions take place. Thus, studying a grammar means to understand the inner relations among constructions in a structural way. To achieve this goal, we follow the rather standard mathematical technique of looking at a category as a space via the so-called Grothendieck topologies. Then, we are allowed to study the inner structure of the obtained site via the sheaves on that site.

We will not develop here the whole mathematical machinery, which is rather complex, but we want to give the reader an intuition of what to expect and why we believe this approach could be of interest.

We recall that a *sieve on an object* c in a category $\mathbb{C}$ is a set S of arrows of $\mathbb{C}$ with codomain c such that, if $f \in S$ and $f \circ h$ is defined, then $f \circ h \in S$. Specialising this notion to pure grammar categories, we get that a sieve S on c is a set of instances of constructions in bijective correspondence with a set S' of adtrees defined as follows: if $f : b \to c \in S$, then $b \in S'$ and, for every d such that there is $g : d \to b$, $d \in S'$. Intuitively, a sieve S on c fixes a set B of instances of constructions of c, whose governors are in S', and the sieve contains all the instances of constructions of c whose last step is in B.

Dualising, a *cosieve* S *on* c in a pure grammar category is a set of instances of constructions of c in bijective correspondence with a set S' of adtrees defined as follows: if $f : c \to b \in S$, then $b \in S'$ and, for every adtree d whose governor is b, the adtree $d' \in S'$, where d' is obtained by d expanding the construction of b so to make c its governor. In other words, cosieves are built fixing a set B of constructions having c as governor, and taking all the constructions which are extensions of those in B.

Examples of sieves are as follows: suppose c is the result of a valency construction over the product $\langle O, (\epsilon)\rangle$ from v in some adpositional category; then, the sieve S generated by this arrow contains all the constructions of c having v as governor, eventually substituted by its constructions. Similarly, the cosieve on the object v generated by the arrow as above, is nothing else that the collection of instance of constructions having the arrow as the first step, i.e., it is isomorphic to the set of phrases whose principal sentence is the given verb construction.

Also, considering all the constructions obtained by adding a fixed circumstantial to an element of the I grammar characters, we obtain a sieve whose base arrows are the conjugates of the I-modifier construction.

In general, in a pure grammar category, fixing a collection of instances of constructions of the same object, a sieve contains all the (instance of constructions of the) objects represented by adtrees obtained by expanding the governors of those instances. On the other side, if we fix a collection of adtrees all having the same governor, the elements of the corresponding cosieve are all the (instance of constructions of the) adtrees whose governors are the fixed ones. In a general grammar categories, sieve and cosieve are understood in a similar way, except that the adtree representation is not unique modulo conjugates.

As sieves and cosieves represent interesting collections of (instances of) constructions, it is worth considering the meaning of Grothendieck topologies in the context of pure grammar categories.[2] We recall that a *Grothendieck topology on a category* $\mathbb{C}$ is a function J mapping every object c of $\mathbb{C}$ into a collection $J(c)$ of sieves on c such that

1. $t_c = \{f : \operatorname{cod} f = c\} \in J(c)$, i.e., the maximal sieve on c is always in $J(c)$;
2. if $S \in J(c)$ then, for every $h : d \to c$ in $\mathbb{C}$,
$$h^*(S) = \{g : \operatorname{cod} g = d \text{ and } h \circ g \in S\} \in J(d) \ ;$$
3. if $S \in J(c)$ and R is any sieve on c such that $h^*(R) \in J(d)$ for all $h : d \to c$ in S, then $R \in J(c)$.

A *site* is a pair $\langle \mathbb{C}, J \rangle$ where J is a Grothendieck topology on $\mathbb{C}$. Intuitively, a site is a topological space built on a category instead of a set. The reader should think to such a space as a potentially pointless space, where the basic elements are neighbours not necessarily containing points, which may be absent. Although these mathematical structures are not immediate to the geometrical intuition, their properties are essentially the same as the ones of usual spaces, only in a more abstract setting.

To clarify the meaning of a Grothendieck topology on a pure grammar category, one may think of a sieve $S \in J(c)$ as a set of admissible instances of constructions with respect to J. So, $h^*(S)$ becomes the set of instances of constructions that, composed with h, give an admissible construction of the object c. Hence, the stability axiom, i.e., the second requirement in the definition of a topology, is nothing more than requiring that $h^*(S)$ is admissible, which is a very natural condition. The third requirement amounts to a generalised transitivity. Thus, composing admissible constructions with themselves produces admissible constructions, which is the meaning of transitivity.

As a consequence, a Grothendieck topology J on a pure grammar category $\mathbb{C}$ is a way to coherently identify admissible instances of constructions. But the *sheaves on a site* $\langle \mathbb{C}, J \rangle$ form a topos $\operatorname{Sh}(\mathbb{C}, J)$ that contains the whole amount of information on the site. We remind that a *presheaf on* $\mathbb{C}$ is simply a functor $\mathbb{C}^{\mathrm{op}} \to \mathbf{Set}$ and a *sheaf on* $\langle \mathbb{C}, J \rangle$ is a presheaf P on $\mathbb{C}$ such that, for every object c in $\mathbb{C}$ and every sieve $S \in J(c)$, the inclusion $S \rightarrowtail \operatorname{Hom}(-, c)$ induces an isomorphism $\operatorname{Hom}(S, P) \cong \operatorname{Hom}(\operatorname{Hom}(-, c), P)$.

We do not want to explain here the details of what a sheaf is in a pure grammar category and the precise consequences of $\operatorname{Sh}(\mathbb{C}, J)$ being a topos, as these explanations are far beyond the scope of the present text. It suffices to say that, if we think of admissible constructions as a sort of abstract syntax for the language modelled by the category $\mathbb{C}$, then the topos $\mathrm{Sh}(\mathbb{C}, J)$ is able to provide a logical description for that syntax, which becomes an abstract grammar for the language, as well as a geometrical interpretation of the language as a mathematical space, which is a novel and different way to look at linguistic phenomena.

We believe that the analysis of $\mathrm{Sh}(\mathbb{C}, J)$ when $\mathbb{C}$ is a pure grammar category will be a difficult, but deeply rewarding future development for the formal model that may undercover and explain many subtleties of natural languages, and that may lead to a deeper understanding and unification of linguistic phenomena on a completely abstract basis, founded on the pure topos-theoretic framework.

B.5 Links with other formal models

In this section we want to clarify the relation between our formal model and context-free languages. Although the relations we draw can be generalised to context-sensitive languages and, beyond, to all decidable languages (and most of their sub-classes), we limit ourselves to the context-free case for clarity.

In the following, we assume acquaintance with the basic theory of formal languages. As a reference text, we will follow Martin (1997). Firstly, we recall the fundamental definition of context-free grammar and language. Then, we will show how to characterise a context-free language as a subset of objects in a suitable grammar category. The proposed representation is general and it can be easily adapted to wider and richer classes of languages and grammars. After, we will present a more specific construction which embodies the fundamental character of context-free languages, and provides a rather different characterisation of those languages in our formal model.

Definition B.5.1 *A* context-free grammar *is a structure* $\mathbb{G} = \langle V, \Sigma, S, P\rangle$ *where* V *and* Σ *are disjoint finite sets,* $S \in V$ *and* P *is a finite set of expressions of the form* $A \to \alpha$*, where* $A \in V$ *and* $\alpha \in (V \cup \Sigma)^*$. *The elements of* V *are called the* non-terminal symbols*, those of* Σ *are called the* terminals*.* S *is called the* start symbol *and the elements of* P *are called the* productions *or* derivation rules*.*

Definition B.5.2 *Let* $\mathbb{G} = \langle V, \Sigma, S, P\rangle$ *be a context-free grammar. The* language generated by $\mathbb{G}$ *is*

$$L(\mathbb{G}) = \left\{ x \in \Sigma^* : S \xrightarrow{\mathbb{G}} x \right\} \quad ,$$

where $\xrightarrow{\mathbb{G}}$ *is the reflexive and transitive closure of the* $\blacktriangleright$ *relation which, given* $\alpha, \beta \in (V \cup \Sigma)^*$*, is defined as* $\alpha \blacktriangleright \beta$ *if there is* $v \in V$ *such that* $\alpha = \alpha' \cdot v \cdot \alpha''$ *and* $\beta = \alpha' \cdot \beta' \cdot \alpha''$ *where* $(v \to \beta') \in P$.

A language L is said to be context-free *if there is a context-free grammar* $\mathbb{G}$ *such that* $L = L(\mathbb{G})$.

Let $\mathbb{G} = \langle V, \Sigma, S, P\rangle$ be a context-free grammar. Consider the relation $\xrightarrow{\mathbb{G}} \subseteq V \times (V \cup \Sigma)^*$; by definition, it is reflexive and transitive, so it naturally induces a composition operation. Define $\mathbb{C}'$ as the category whose objects are the strings σ on the $V \cup \Sigma$ alphabet such that $S \xrightarrow{\mathbb{G}} \sigma$, and whose arrows are exactly the instances of the $\xrightarrow{\mathbb{G}}$ relation.

Now, define $\mathbb{C}$ as the category $\mathbb{C}' \sqcup \mathbf{1}$ where $\sqcup$ is the coproduct of categories and $\mathbf{1}$ has only one object, ϵ, with its identity as the unique arrow.

Le G be the discrete partition of $\operatorname{Obj}\mathbb{C}$, whose classes are $\{c\}$ for every object c. Obviously, G is indexed by $\operatorname{Obj}\mathbb{C}$. Fixing as C all the instances of the $\xrightarrow{\mathbb{G}}$ relation, one obtains in a trivial way that $\overline{\mathbb{G}} = \langle \mathbb{C}, G, \{\epsilon\}, C\rangle$ is a grammar category [3] whose atoms are $\{S, \epsilon\}$.

The language $L(\mathbb{G})$ coincides with the set of objects in $\overline{\mathbb{G}}$ which are not the domain of any arrow except identities, as it is immediate to prove. [4] The corresponding adtrees represent the possible constructions as sequences of applications of rules in P.

In a different, more abstract way, one could equivalently say that $L(\mathbb{G})$ is the collection of maximal arrows in the maximal cosieve on S. This characterisation in not limited to context-free languages, but applies to all families of languages in the Chomsky's hierarchy. [5]

Moreover, the reader is invited to notice how this characterisation relates to the discussion on "correct" adtrees and redundancy transformations: in fact, we can equivalently say that $L(\mathbb{G})$ is isomorphic to the set of non-empty adtrees in $\overline{\mathbb{G}}$ which are fixed points with respect the transformation mapping each adtree denoting $c \in \Sigma^*$ whose governor is S into itself, and each other adtree into the empty adtree.

Unambiguous context-free languages can be characterised in a different and more specific way as special elements in a grammar category. Firstly, suppose Λ, the empty string of terminal symbols, is not in $L(\mathbb{G})$, where $\mathbb{G} = \langle V, \Sigma, S, P\rangle$ is some unambiguous context-free grammar.

A standard result states that, if $\mathbb{G} = \langle V, \Sigma, S, P \rangle$ is a context-free grammar, then there is another context-free grammar $\mathbb{G}' = \langle V', \Sigma, S, P' \rangle$ such that $L(\mathbb{G}) = L(\mathbb{G}')$ and $\mathbb{G}'$ is in Chomsky normal form, i.e., every production in p' is either of the form $A \to B \cdot C$ with $B, C \in V'$, or of the form $A \to a$ with $a \in \Sigma$.

Define $\mathbb{C}^*$ as the category having as objects the strings on Σ^* and as arrows the derivation rules defined as follows, along with constructions and grammar characters:

- if $(A \to a) \in P'$ then $a \in G_A$;
- if $(A \to B \cdot D) \in P'$ and $b \in G_B$, $d \in G_D$, then $b \cdot d \in G_A$; moreover, the arrows $b \to b \cdot d$ and $d \to b \cdot d$ are in $\mathbb{C}^*$. Finally, these arrows are instances of the obvious constructions over the products $\langle G_D, (\epsilon) \rangle$ from G_B, and over $\langle G_B, (\epsilon) \rangle$ from G_D, respectively. These construction are in the set C of constructions. We observe that the latter construction is the conjugate of the former one.

Now, call $\mathbb{C}$ the full subcategory of $\mathbb{C}^*$ whose objects are in $(\bigcup_{v \in V'} G_v) \cup \{\epsilon\}$. Evidently,

$$\left\langle \mathbb{C}, \{\epsilon\} \cup \bigcup_{v \in V'} G_v, \{\epsilon\}, C \right\rangle$$

is a grammar category, and $L(\mathbb{G}) = L(\mathbb{G}') = G_S$. In a trivial way, the correct adtrees are defined as the non-empty fixed points of the transformation which maps every adtree denoting an object in G_S into itself, and any other adtree into the empty one.

It is interesting to notice that the adtrees in this grammar category, when only conjugate constructions are allowed, are exactly the usual derivation trees in the Chomsky tradition.

Index

Entries are ordered alphabetically; Greek letters are ordered according to their pronunciation, e.g., λ is listed under L.
Some entries are marked with ► : this sign indicates to see another entry, listed below. Some entries refers to other, related entries: these references are introduced by *also* and the entry they refer to.
Finally, entries are referred to as ► *A* ▹ *B* ▹ *C*, meaning that the referred entry is *C* which is listed under the entry *A* and the sub-entry *B*.

A

B

D

G

H

I

N

O

P

Q

R

S

T