

Chapter 1

Introduction

1.1 The Logical Space of Meaning

If the space of physics *is* mathematics, the space of meaning *is* logic. In the first case, mathematics produces objects and structures which are precisely those which make the external world understandable. It is only a small step to assume that this external reality *consists in* those mathematical objects. In support of this view, we must note that there is, for instance, no coherent formulation of quantum mechanics outside the equations of mathematics, so that this immaterial reality (the reality of mathematics) *exceeds* the reality of common sense. Mental entities are in many ways different from physical ones. It is not to say that mental entities cannot be thought of in physical terms in the last resort, but simply that so-called mental entities have distinct properties with regard to directly physical ones. Those properties *make* (in the strong sense) our mental life. According to Schrödinger (1958), the condition which gave birth to physics lies in that particular move of the mind extracting itself from reality, a process he named *elision*, and which preconditions the remote observation of physical objects, the price to pay being the rejection of subjectivity from science. In the case of mental objects, the subjective element cannot be constructively abstracted away because of *intrinsic reflexivity*: mental structures may always be applied to themselves. Nevertheless, a knowledge of them may come from a small gap introduced between the mental reality and itself, a gap which may be compared to *parallax*. Generally this reflection of thought on itself, where a discrepancy may occur, showing something of its structure, is provided by communication between people by means of language. This appears especially in dialogues, and it is no coincidence that the great works of our cultures are often based on them (if we think for instance of such

different traditions as the Greek on one side, illustrated by Plato, and the Indian on the other side, as witnessed for instance by Nāgārjuna). Or as the famous cognitivist Steven Pinker (2007) said, “language emerges from interaction between human minds”. Let us assume for instance a speaker A who claims *Every linguist knows an African language* to speaker B. B may respond, *I know* (acknowledgement), or *I’m not sure Paul, who is a linguist, knows any African language*, or she may answer, *Paul and Mark, who are linguists, don’t know the same languages*. In the latter case, B shows that she had a slightly different comprehension from the two former cases. The feedback thus provided by B commits A to perceive that her sentence may have different meanings and realize that such a difference lies in the logical structure. Dialogue thus creates the opportunity for analyzing thought and meaning even if they cannot be remote from our mind. This aspect of meaning will be addressed mainly in the last part of this book, when more traditional ways of treating it will have been surveyed. In the small gap thus revealed between a thought and its reflection in another one, lies the possibility of logic, as a phenomenological approach to meaning.

On the other hand, as pointed out by F. Récanati (2008) among others, meanings may be assembled like syntactic constituents. We could make the assumption that human language is oriented towards dynamicity of meanings, which could not be connected without the help of syntactic structures. These syntactic structures form the core of syntax. It follows that it is not possible to study meaning without studying the syntax-semantics interface.

To study the space of meaning therefore implies that we have to address two domains simultaneously: logic and syntax. There is no guarantee that they can be harmoniously articulated. Like we shall see in the second part of this book, the study of logic is more and more oriented towards the discovery of deep symmetries, like those between the rules of linear logic in its sequent presentation, or between the two sides of a *sequent* $\Gamma \vdash \Delta$ either in classical logic or in linear logic. On the other hand, syntax seems to be a domain of massive anti-symmetry, at least if we follow the assumptions made by some of the most eminent linguists, like R. Kayne (1994) who stated the so-called *anti-symmetry* principle in generative syntax. There seems to be therefore a challenge in the idea of dealing simultaneously with both domains. That did not prevent many scholars from attempting to make a connection between them, above all since the seminal work by Jim Lambek (1958, 1961, 1988, 1993) who, in 1958, provided the foundations

for the logical study of syntax in an authentic “syntactic” calculus. This work has been continued since the early 1980s, particularly by Glyn Morrill (1994) and Michael Moortgat (1988, 1997) and their followers. This led to a theory of language presented as *grammatical reasoning*. Such an enterprise is oriented to the search of *constants* which would play the same role with regard to language that constants, like *connectives* and *quantifiers*, play with regard to logic. On the path to this research, we meet recent evolutions of logic (Girard, 1987, 1995, 2001; Schroeder-Heister and Dosen, 1993) which show that by playing with the so-called structural rules in the sequent calculus (weakening, contraction and permutation), it is possible to enrich the spectrum of these connectives. It also appears that previously assumed primitive constants of logic (like the implicative connective) may actually be decomposed into more primitive symbols, thus allowing a more flexible management of formulae in the search for proofs.

Nevertheless, language is not logic. If we try to represent the more syntactic-semantic phenomena that we can in a logical system, probably we shall have to import into it devices which have little independent motivation within logic. This is the case for instance for *structural modalities*, which have been introduced since the 1980s, in the Lambek calculus in order to surpass its limitations (Kurtonina and Moortgat, 1997). Either we restrict the Lambek calculus \mathbf{L} to a so-called “non-associative” Lambek calculus (\mathbf{NL}), wishing to keep the notion of constituency which seems to be crucial in syntax, and so are obliged to introduce these devices to relax the constraints in some places, or we extend it, for instance, to accept non-peripheral extraction, but in this case we get an overgenerating system that we have to constrain if we don’t wish to accept too many sentences. From our viewpoint, this reflects the conflict we mentioned between the fundamental symmetries of logic and the dissymmetries of syntax. In fact, strong arguments exist (see Girard, 2006) against non-associative logic. Non-associative logic would be contradictory with one of the main foundations of logic as they are provided by category theory.¹

Our position in this book is not to hide the discrepancies between logic and syntax but to try to use the symmetries when possible, as much as possible. For instance, the use of symmetries in the computational applications

¹Let us recall that a *category* is defined as a pair of sets, *objects* and *morphisms*, so that morphisms are supplied with an operation of composition which has a neutral element and is associative.

of classical logic will prove useful in representing elegantly some meaning phenomena, like scope ambiguities.

Another point which will be addressed in this book is a criticism of the widespread conception of meanings as sets of *truth conditions*, a view which dates from Frege and has been widely accepted in the studies of formal semantics by Montague (1974a,b) and Heim and Kratzer (1998), among others. The alternative view will be hinted at here, but developed more fully elsewhere.

1.2 The Aim of This Book

This book aims at presenting some of the main advances in natural language semantics and the interface between syntax and semantics in the last two decades, based on some of the most recent logical theories, mainly linear logic, and on precise analyses of computing due to advanced theories in theoretical computer sciences, like the extensions of the Curry–Howard isomorphism to classical logic which has led to several variants of calculi, like Parigot’s $\lambda\mu$ -calculus, Curien and Herbelin’s $\lambda\mu\tilde{\mu}$ -calculus, Wadler’s dual calculus, the Lambek–Grishin calculus revisited by Moortgat and Bernardi and so on (Parigot, 1992; Curien and Herbelin, 2000; Herbelin, 2005; Wadler, 2003; Bernardi and Moortgat, 2010). The book surveys many solutions which have been proposed for the syntax-semantics interface, taking into account the specificities of the linguistic signs and fundamental mechanisms brought to light by the linguists and particularly the generativists. It ends with a presentation of ludics, a framework which allows us to characterize meaning as an invariant with regard to interaction between processes (Girard, 2001, 2003, 2006).

1.3 Starting from Traditional Formal Semantics

These advances described are not only technical: they are also philosophical. In this book, we start from a well-known stage of formal semantics, the one provided by Montague grammar, and we point out one of the problems it had to face: the question of scope construal. It has been well known for a long time that sentences with several quantifiers may be ambiguous. A question is whether such ambiguity is due to various potential syntactic analyses (as Montague and after him the most eminent linguists like

Chomsky seem to believe) or simply to various strategies for evaluating the logical meaning, as would seem to be the case if we took seriously the idea that this kind of ambiguity is properly semantic. Some linguists (like Culicover and Jackendoff (2005) when they called for a “simpler syntax”) seem to call for a theory where the respective tasks of syntax and semantics would be quite separate and where, for instance, syntax would only deal with pure questions of assembling lexical signs, leaving to semantics the task of combining meanings. Of course, this trend still keeps close to the compositionality principle (Janssen, 1997), according to which the meaning of a sentence is a function of the meanings of its parts and the way they are combined, but there is perhaps a (later) stage in the interpretation (perhaps a “silent” one) where meanings are still combined without any syntactic counterpart. Some works based on continuation semantics (de Groote, 2001b) show that such a machinery is easily conceivable, by means of the particular reduction rules of the $\lambda\mu$ -calculus. If it is so, we are confronted with an interesting question which regards the history of logic.

1.4 Semantics and the History of Logic (1): Intuitionism

1.4.1 *Curry–Howard*

One of the biggest discoveries in logic during the twentieth century was its *algorithmic content* (Howard, 1980). The Curry–Howard correspondence between formulae and types, and between proofs and programs revealed that proofs in intuitionistic logic may be coded by λ -terms, and that therefore if we represent, like in the Montagovian tradition, meanings by λ -terms, we have a way of computing meanings if we are able to display the analysis of a sentence as an intuitionistic proof. This had been exactly realized by Jim Lambek, several years before the “official” formulation of the Curry–Howard correspondence, in his seminal article (Lambek, 1958).

1.4.2 *Lambek and the substructural hypothesis*

The Lambek calculus is an intuitionistic framework since to recognize a sequence of words in that calculus precisely amounts to constructing a proof of the validity of a sequent with only one formula (or type) on its right-hand side (and, moreover, with a non-empty left-hand side, something which has actually nothing to do with logic, but with grammar, if that were not the case, we could insert “empty” categories with appropriate types anywhere,

thus leading to strange sentences!). Of course, it is not a complete logic, since, for instance, it has no negation and no disjunction. Moreover it is a variant of so-called *substructural logics*, that is, logic where structural rules are missing. We recall here that a *structural rule* is a rule which allows the easy management of a sequence of formulae (or types). For instance the weakening rule is also known elsewhere as a monotonicity rule: it ensures that if something has been proven without a given formula A , it can be *a fortiori* proven if we add A to the set of premises. The contraction rule allows consideration of two occurrences of the same formula as the duplication of this formula and the permutation rule of course allows permutation of premises if needed in the course of a demonstration. Suppress one or more (eventually all) of these rules and you get what we call a *resource-sensitive logic*: exactly the kind of thing we seem to need in order to deal with “material” entities like *signs*. Of course, if you demonstrate that the sequence of words *Peter feeds the cat* is a correct sentence, by using the convenient types associated with the words, you would not be allowed to accept that the sequence *Peter Mary feeds the cat* is also a sentence. In the same way, the sentences *I call a cat* and *I call a spade a spade* are not similar in meaning, as would be the case if we had a contraction rule at our disposal!

Nevertheless, many linguistic phenomena are such that they need some violation of this strict *linearity* principle (we call *linearity* the property of a system based on a discipline of *consumption*: any object must be used once and only once), as is the case for anaphora (*he thinks that he is clever*) or parasitic gapping (*the book that I shelved without reading*).

1.5 Semantics and the History of Logic (2): Classicism

There have been a lot of works exploring the potential algorithmic content of classical logic (Girard, 1991; Vauzeilles, 1993; Danos *et al.*, 2003). In fact, it was shown, in particular by the categoricist Joyal (Lambek and Scott, 1986), that, strictly speaking, classical logic is not *constructive*! This is shown with category theory tools: if proofs are interpreted as morphisms in a category where formulae are objects, the constructive character of intuitionistic logic is revealed in its semantics in terms of a Cartesian closed category: each proof is an individual (modulo normalization), whereas in classical logic, the category obtained collapses into a simple Boolean algebra; that means that there is only one morphism corresponding to a provable formula, and

this morphism only says that the formula is “true”. Another aspect of the lack of constructivity of classical logic is provided by the fact that the normalization process in classical logic does not possess the Church–Rosser property: it is not confluent. It may happen that two different paths in the reduction lead to two different results. These negative facts did not prevent some researchers from dealing with classical logic, exploring the attempt to recover some determinism in the normalization process by *controlling* the strategy of reduction.

Computer scientists familiar with functional programming know that a symbolic form may be evaluated according to two different strategies: *call-by-value* and *call-by-name*. In call-by-value (CBV), each argument of a function must be evaluated (transformed into a *value*) before being passed to the function. In call-by-name (CBN), this is not required: arguments may be passed to the function before being evaluated. This gives an advantage when the same complex object is passed several times as an argument to a given function. We may summarize this opposition by saying that in CBV, *values* must be computed first, and that in CBN, *contexts* are computed first. In most programming systems, the two strategies lead to the same results (*confluence*), but it would be possible to extend those programming systems and then determinism would require following only one strategy. This is what is realized in the $\lambda\mu$ -calculus. Nevertheless, for purposes other than strictly computational ones, we can imagine a calculus where two different strategies are equally usable, thus leading to different results. We claim that this is just what happens in language when we try to evaluate a sentence which contains several quantifiers.

If things are this way, it is reasonable to conclude that *semantic ambiguity* comes from the *classical* (and not *intuitionistic*) nature of language. However, there are still other tracks to follow. After all, these so-called *symmetric calculi* are fascinating but strange to manage: they are based on a permanent need for dissymmetrization. At each step, a formula must be distinguished (as an *active* one) and the story of activations and deactivations is stored in the stack of μ - (or $\tilde{\mu}$ -) operators. Linear logic provides an entirely different way of making a symmetric calculus constructive.

1.6 Semantics and the History of Logic (3): Linear Logic

Removing the structural rules of weakening and contraction opens the field to another constructive logic: linear logic. In this case, the *functional*

interpretation of logic is given up. Intuitionistic logic was based on a fundamental dissymmetry: several *inputs* for only one *output*. Now, in linear logic, we have a symmetry between inputs and outputs. Therefore the preferred interpretation is no longer in terms of functionality but in terms of interaction: if inputs and outputs are symmetric, it is possible to exchange them and to read a result either as a flow from inputs towards outputs or as a flow in the reverse direction. This suggests the interpretation of proofs as *processes*. Removing weakening and contraction moreover gives space for more connectives. The sequent calculus has two variants which coincide in the classical frame: the multiplicative and the additive ones. In general, we distinguish between the active formula (the one to which the rule is applied) and the inactive ones, which are its *contexts*. In an introduction rule for a given connective (\wedge or \vee , on the left or on the right) we may formulate the rule either by mentioning disjoint contexts in the premises, or by mentioning identical ones: because of weakening and contraction, that does not matter. But in the absence of these structural rules, it matters, and the multiplicative and additive variants are no longer equivalent. Therefore we get *multiplicative* conjunction (\otimes) and disjunction (\wp), and *additive* conjunction ($\&$) and disjunction (\oplus). *Negation* is simply a duality operator: every formula may change its side in one sequent by being negated. *Linear implication* is a multiplicative connective. $A \multimap B$ is defined as $A^\perp \wp B$; it is this connective which expresses resource consumption: against A , you obtain B . As seen above, a natural interpretation of linear logic is in terms of processes. The linear implication gives an illustration of that: when you give A in order to obtain B , there are two entangled processes, one of giving A and one of receiving B in exchange. Therefore the multiplicative disjunction \wp is naturally interpreted as a *parallelization* of processes, whereas \otimes is interpreted as a *sequentialization* of processes ($A \otimes B$ is either A and then B or the other way round²). Game semantics interpretations have been provided very early on for linear logic, at least since the work of A. Blass (1992). In computational terms, a type could be regarded as a server from which a client can get, in one access, an element of that type; the client need not do anything more than show up. This kind of type is said to be simple because the client has nothing to do, but it is also possible to build *complex* types. In fact we can understand $A \& B$ as

²C. Retoré (1993) explored this in the context of non-commutative linear logic. For this purpose, he invented a connective $<$ such that $A < B$ is interpreted as A before B .

a *choice*: the resources you have can provide you with A as well as with B , therefore they provide you with a choice between two kinds of data. The complex type may be therefore interpreted as a *server* from which the client can get either A or B . Because \oplus has a symmetric law with regard to $\&$, it is natural to understand it as a dual choice: the client has nothing to do because the server makes the choice itself. Now, because of De Morgan's laws, *negation* may be interpreted as an exchange between the two roles, of server and of client. Blass still proposes a more elaborate interpretation by means of games. He says, "a play of $A \otimes B$ consists in interleaved runs of the two constituents A and B ". Whenever it is the proponent's turn (= server's turn) to move, he or she must move in the same component in which the opponent (= client) last moved, while the opponent is free to switch components. Finally, $A \otimes B$ is a complex game with the impossibility for the proponent to make use of the information obtained from one component when playing in the other, while $A \wp B$ means such a game where he or she can.

This game interpretation is something that we get in the more recent *ludics* (Girard, 2001, 2003, 2006; Fleury and Quatrini, 2004; Curien, 2003; Lecomte and Quatrini, 2010). Ludics starts from the observation that proofs can always be represented as alternations of positive and negative steps (Andréoli, 1992), where a negative step is a reversible one (such that we may always make the reverse step because we never lose contextual information when making that step) and a positive one is non-reversible. Seeing attempts to find a proof as such successions of negative and positive steps allows us to see those attempts as games, where positive steps are moves by the proponent and negative ones are records of the expected answers by the opponent. Because the roles of proponent and opponent may be exchanged, we see that a similar attempt to build a proof exists from the opponent's viewpoint, but his or her attempt to build a proof has as its goal the negative sequent with regard to the sequent defended by the first player. Proofs are thus opposed to counter-proofs, in a space where both are coexisting (the space of *paraproofs*). An *orthogonality relation* may then be defined and in analogy to vector spaces, a null vector is introduced, which belongs to the proofs as well as to the counter-proofs: this paradoxical object is a *paralogism* that Girard calls the *daimon*.

The "morality" of this story is that we are led to consider "true" proofs and "false" ones (that is, unsuccessful attempts) on the *same* level. What interests us above all is their interaction (called normalization) and not

particularly their “truth” or their “falsity”. In fact, this is exactly what is needed when we try to isolate various meanings that a sentence can have. A meaning is determined by the counter-meanings with which it interacts: there is no need to use truth values as primitive objects for representing that, as it suffices to have a formalization of the notion of interaction, something provided by ludics.

1.7 Presentation of the Book

This book may be read by beginners: it does not require much preliminary knowledge, except for some notions of predicate logic and λ -calculus that everyone can find in many textbooks (see for instance Gamut, 1991). It is particularly intended for students in computer science who wish to learn about applications of theoretical computer science to the understanding of natural language, for students in linguistics who wish to learn some logical and computer-scientist aspects of the study of language, and for students in philosophy who wish to have an overview of topics which are discussed in the area of language studies and which necessitate some technical background.

It is divided into 13 chapters (including this introduction and a conclusion). We start from a well-known state of the art: most formal semantics studies nowadays are based on two famous approaches, which share much in common, Montague’s semantics and Heim and Kratzer’s conception of semantics in generative grammar. That is not to claim our attachment to these frameworks, but to state things as they actually are. Nevertheless, this chapter states the connection between generative grammar and compositional semantics. We recall the main characteristics of the Montagovian approach to formal semantics, putting emphasis on the connection between syntactic and semantic rules and suggesting a move towards a more uniform treatment of compositional meaning. Heim and Kratzer’s approach is also presented. It has the peculiarity with regard to Montague’s of being directly based on a model-theoretic perspective. The question which arises from this chapter will be that of *uniformity*: how to deal in the most uniform way with all the problems which are posed from an analysis of scope construal?

Chapter 3 recalls Lambek’s syntactic calculus, a calculus particularly suitable for getting a semantic translation from a syntactic derivation, seen

as a proof in a fragment of an intuitionistic and resource-sensitive logic, but which is limited when considering for instance non-peripheral extractions or non-linear phenomena. Moreover, the pure Lambek calculus is not sensitive to the phrase structure of a sentence. In this chapter, a first comparison is attempted with *minimalist grammars*, which represent the current state of generative grammar (Stabler, 1997, 2003). Because a deeper study of such systems involves more knowledge in proof theory, Chapter 4 offers such a digression, presenting all the main tools that are nowadays used in this discipline, with some linguistic illustrations (like for *proof nets*).

The Curry–Howard correspondence is presented in Chapter 5.

Chapter 6 presents variants of the Lambek calculus, which are able to deal with the phrase structure (non-associative Lambek calculus). Various applications of this framework, as well as close variants of it are presented, like Hendricks’ flexible types (Hendricks, 1993) and Jacobson’s variable free semantics (Jacobson, 2000). Extensions are also shown, like the Lambek–Grishin calculus and Abrusci’s non-commutative linear logic (Grishin, 1983; Abrusci, 1991).

Chapter 7 explores what Moortgat has called *grammatical reasoning*, that is, the view according to which complex linguistic analyses may be performed by logical means. These “logical” means include structural modalities (distinct from the usual modalities in modal logic, and therefore more or less tied to the linguistic purpose).

Chapter 8 goes back to the so-called Minimalist Program, proposing type-theoretical formulations for minimalist grammars, which would make the interface between syntax and semantics more tractable. In Chapter 9 other grammars are studied which propose similar but distinct views, like Pollard’s convergent grammars (Pollard, 2007) and Anoun and Lecomte’s linear grammars (Anoun and Lecomte, 2006).

Chapter 10 introduces the notion of *continuation*, a notion issued from computer science, which supports the management of *contexts* as well as of *terms* (or programs). It is shown that earlier solutions to some problems of the syntax-semantics interface, by Montague (like the systematic use of type raising), were already compatible with the idea of continuation: when an object of type e is seen as an object of lifted type $(e \rightarrow t) \rightarrow t$, it is transformed into a function which takes a continuation (something of type $e \rightarrow t$) as its argument. Barker (2002) and Shan (2004) were among the first to propose *continuized grammars* which precisely allow the kind of uniformity we are seeking. On the logical side, to systematically use continuations

calls for a symmetric calculus where contexts and terms would be treated on a par. This chapter presents some variants of such a symmetric calculus, like the $\lambda\mu$ - and $\lambda\mu\tilde{\mu}$ -calculus. It shows how to directly apply these calculi to the framework of the Lambek grammars in order to overcome some of their limitations, in a direction which is currently explored by Bernardi and Moortgat (Moortgat, 2007; Bernardi and Moortgat, 2010). Chapter 11 is devoted to one of the most original contributions that proof theory can bring to semantics, that is, the constructive type theory approach, first advocated by Martin-Löf (1984) and then deepened by Ranta (1994). We try then to go further on to the dynamic aspect of language by taking (dialogical) *interaction* into account.

It is in Chapter 12 that we pose the main philosophical problems linked to the study of language: Is a non-denotational semantics possible? How to reconcile the *meaning as proof* paradigm (like it is formulated in the Brouwer–Heyting–Kolmogorov hypothesis, and used in natural language semantics by Ranta) with the game semantics paradigm developed by Hintikka and Sandu (1997) and Hintikka and Kulas (1983)? Ludics, we claim, provides an answer to these questions, and in this chapter, co-authored with Myriam Quatrini, the main concepts of ludics are presented, and linguistic applications are shown, concerning the notion of logical form, the semantics of questions and the study of discursive relations.

Finally, Chapter 13 tries to plan future works on the basis of the last part of this book, showing links with other perspectives like Kempson and Colleagues' dynamical syntax, Hamblin's perspective on fallacies, Groenendijk and Roelofsen's inquisitive semantics and works on semantics inspired by coherent spaces (Kempson *et al.*, 2003; Hamblin, 2004; Groenendijk and Roelofsen, 2010).