Chapter 1
Synopsis

BEFORE I start with the technical discussion it is perhaps worthwhile to discuss the relevance of the concepts. I shall begin with some notes on the historical context and the current developments before I turn to the questions that I have tried to answer in this book.

Modern linguistics begins with de Saussure, yet he wrote surprisingly little on the subject matter. The famous *Cours de linguistique générale* exists in several editions none of which were published by de Saussure himself. Some years ago, however, a bundle of autographs was found in his home in Geneva, which are, I think, of supreme importance (see the English edition (Saussure, 2006)). We see de Saussure agonize over some quite basic and seemingly innocent problems: one is the distinction between what he calls "parole", a continuous object of changing and elusive nature and "langue", a system of oppositions, in other words a structured object. De Saussure constantly reminds us that all the objects we like to talk about in linguistics are abstractions: meanings, letters, phonemes and so on. The second problem that he deals with, and one that will be central to this book, is that language is a *relation* between form and meaning and not just a system of well-formed expressions.

One might think that one hundred years later we have settled these issues and found satisfactory answers to them. I think otherwise. Both of the problems are to this day unsolved. To understand why this is so it is perhaps useful to look at Chomskyan linguistics. The basic ingredients of Generative Grammar are a firm commitment to discrete objects and the primacy of form over meaning. There is no room for gradience (though occasional attempts have been made even by Chomsky himself to change this). Grammars are rule systems. Moreover, linguistics is for the most part the study of form, be it phonology, morphology or syntax. The rise of Montague Grammar has changed that to some degree but not entirely. One reason for this is that Montague Grammar itself, like Generative Grammar, is rooted in metamathematics, which puts the calculus, the mindless symbolic game, into the centre of investigation.

The present book took its beginning in the realization that what linguists (and logicians alike) call meaning is but a corrupted version thereof. A second, related insight is that linguists rarely if ever think of language as a *relation*. The ambition of the present monograph is to change that. What I shall outline here is a theory of formal languages that are not merely collections of syntactic objects but relations between syntactic objects and their meanings.

M. Kracht, *Interpreted Languages and Compositionality*, Studies in Linguistics and Philosophy 89, DOI 10.1007/978-94-007-2108-1_1,

> Throughout this book, *language* means a set of signs. Signs are pairs consisting in syntactic objects and meanings. Languages are sets of signs and hence relations between syntactic objects and meanings.

This calls for a complete revision of the terminology and the formal framework. Consider by way of example the syntactic rule

$$\texttt{S} \rightarrow \texttt{NP VP} \tag{1.1}$$

This rule can be used to replace the string /S/ by the string /NP VP/. (I use slashes to enclose strings so as to show explicitly where they begin and end in running text.) Yet, if language consists in syntactic objects together with their meanings we must ask what the meaning of /S/ is, or, for that matter, /NP VP/. If anything, the meaning of /S/ is the disjunction of all possible meanings of sentences of the language, or some such object. However, notice that /S/ is not an object of the language. The whole point of auxiliary symbols in the grammar is that they are not *meant* to be part of the language for which they are used. And if they are not in the language then they have no meaning, for a language by definition endows only its own constituents with meaning.

Notice that the problem existed already at the inception of grammar as production rules. Grammars never generated only the language they were designed to generate but a host of strings that do not belong to the language. Again this was precisely because they contained auxiliary symbols. While it was unproblematic if only string generation was concerned, the problem becomes more urgent if meanings are considered as well. For now we need to replace the rule by something that replaces not only strings but signs, like this:

$$\langle \texttt{S}, x \rangle \rightarrow \langle \texttt{NP}, y \rangle \ \langle \texttt{VP}, z \rangle \tag{1.2}$$

This means something like this: an /S/ that means x can be decomposed into an /NP/ that means y and a /VP/ that means z. This formulation however is unsatisfactory. First, we have lost the idea that /S/ is replaced by the *sequence* of /NP/ followed by /VP/, for we needed to annotate, as it were, the parts by meaning. Second, there is no unique way to derive y and z from x; rather, x is unique once y and z are given. In Montague Semantics, following Frege, z is a function and $x = z(y)$, the result of applying z to y. Thus, it is actually more natural to read the rule from right to left. In this formulation it now reads as follows: given an object α of category NP and meaning y and an object β of category VP and meaning z, the concatenation $\alpha^\frown\beta$ is an object of category S and meaning $z(y)$. The objects can be anything; however, I prefer to use strings. Notice now that we have variables for strings and that we have (de facto) eliminated the syntactic categories. The rule looks more like this now:

$$\langle \alpha, y \rangle, \langle \beta, z \rangle \rightarrow \langle \alpha^\frown\beta, z(y) \rangle \tag{1.3}$$

There is a proviso: α must be of category NP, β of category VP. To implement this we say that there is a function f that takes two signs and returns a sign as follows.

$$f(\langle \alpha, y\rangle, \langle \beta, z\rangle) := \begin{cases} \langle \alpha^\frown \beta, z(y)\rangle & \text{if } \alpha \text{ is of category NP} \\ & \quad \text{and } \beta \text{ of category VP,} \\ \text{undefined} & \text{otherwise.} \end{cases} \tag{1.4}$$

This is the formulation that we find in Categorial Grammar and variants thereof. It is, as I see it, the only plausible way to read the rules of grammar. In this formulation the category is not explicit, as we are generating objects of the language intrinsically. The fact that the generated string $\alpha^\frown \beta$ is an S is therefore something that we must be able to recover from the sign itself. Notice that this problem exists also with the input: how do we know whether α is a string of category NP? Where does this knowledge reside if not in the grammar? I shall answer some of these questions below. They show surprising complexity and contrary to popular opinion it is not necessary to openly classify strings into categories.

From this moment on we are faced with plenty of choices. The binary function f takes as its input two signs, each of which consists in two parts. Thus it has in total four inputs. The question is whether the function is decomposable into simpler functions. Some people would argue that this is not so and some theories encode that dictum in one or another form. Yet, from a theoretical point of view it is not good to drop a stronger hypothesis unless one really has to. The plausible hypothesis is this.

> **Independence.** The functions of the grammar that create signs create the components of the signs independently of each other.

This thesis has two parts. One is the so called *Autonomy of Syntax Thesis* and the other the *Compositionality Thesis*. For convenience I spell them out for our example. The autonomy thesis says that whatever $f(\langle \alpha, x\rangle, \langle \beta, y\rangle)$ may be in a given language, the form (or morphology) of the sign is a function of α and β alone, disregarding x and y. The compositionality thesis says that whatever $f(\langle \alpha, x\rangle, \langle \beta, y\rangle)$ may be in a language, its semantics depends only on x and y and nothing else. Thus we have functions f^γ and f^μ such that

$$f(\langle \alpha, x\rangle, \langle \beta, y\rangle) = \langle f^\gamma(\alpha, \beta), f^\mu(x, y)\rangle \tag{1.5}$$

Informally, this says that whatever form the expression takes, it does not depend on the meaning of the component expressions; and whatever meaning the expression has, it does not depend on the form of the component expressions.

What does this Principle of Independence actually say? It is at this point where many linguists start to be very creative. Anything goes in order to prove languages to be compositional. But the problem is that there is little room for interpretation. A language is a relation R between expressions and meanings. What we postulate in the case of f is that there is a pair of binary functions $f^\gamma : E \times E \to E$ and $f^\mu : M \times M \to M$ such that (1.5) holds. What is important is that the input signs are taken from the language R and the output sign must be in R too. Thus, independence means that we have a set of functions that generate R from the lexicon.

All functions are allowed to be partial. Partiality is essential in the generation of the signs. For example, suppose we want to account for the fact that it is grammatical to say "Jack drove the car." but not "Jack drove the bicycle.". Clearly, we must say that "drive" requires a certain kind of vehicle. The nature of the restriction may now be either morphological or semantic. If it is morphological then it may be formulated as a restriction on the function f^{γ} on the expressions. If however it is semantic, what should we do? There are various options. The best is probably to say that the type of vehicle is already implied by the expression and so we cannot use a different one on pain of contradiction. If one dislikes this solution, here is another one. Create two modes, f_1 and f_2 and declare that $f_1^{\mu}(x, y)$ is defined only if y is a motorized (earth bound) vehicle, while $f_2^{\mu}(x, y)$ is defined in cases where y is a different kind of vehicle. What we cannot do, however, is add some material in the syntactic structure that replicates the semantic properties, such as $\texttt{car}_{\mathsf{motorized}}$ and $\texttt{bicycle}_{\neg\mathsf{motorized}}$. This is effectively what has been proposed with θ-roles. More often than not they have been used to encode semantic properties in syntax. The converse has also often been done: encode a syntactic restriction in semantics.

There is a lot of terminological ground to be covered here. If the formation of signs is a partial operation the question is whether we can at all distinguish syntactic from semantic deviance. Chomsky argued that we can, and I wish to basically agree with his observation even though it does seem to me that it often requires some education to disentangle ungrammaticality and semantic deviance. If it is therefore possible to distinguish ungrammaticality from semantic deviance, what could be the source of that distinction? It could be this: a sentence is syntactically well-formed if it could be generated looking only at the syntactic composition functions and semantically well-formed if its meaning could be generated looking only at the semantic composition functions. Thus, the fact that we can distinguish between these two notions of (un)acceptability requires that we have independent knowledge of both the syntactic functions and the semantic functions. However, notice that the definition I gave is somewhat strange: how can we know the meaning of an ungrammatical sentence? What is the meaning that it has despite the fact that it is ungrammatical? Unfortunately, I do not have an answer to this question, but it is these kinds of questions that come to the fore once we make a distinction between different kinds of well-formedness. Another problem is why it is that an ungrammatical sentence at all has a meaning. Why is it that sometimes the semantic functions are more general than the syntactic functions and sometimes the syntactic functions more general than the semantic functions? This is not only a theoretical problem. It is important also in language learning: if a child hears only correct input, it will hear sentences that are both grammatical and meaningful, so it can never (at least in principle) learn to distinguish these concepts. Again I have not much to say, I simply notice the problems. In part it is because I am not concerned with learning. Partly, however, it is that—surprisingly—setting up something as simple as a formal theory of interpreted languages as opposed to a formal theory of string languages requires much more care in the definitions, and this task has to come first. Despite the fact that the language is given in a relational form it is not clear how we can or should define from that a grammar that manipulates syntax and semantics independently.

Parts of Chapter 3 are consumed by disentangling various notions of autonomy and compositionality.

Now, as much as one would like to agree with my insistence that the language R is given a priori and cannot be adapted later, there is still a problem. Namely, no one knows for sure exactly how R looks like. This is not only due to the somewhat insufficient knowledge of what is a grammatical constituent. It has to do more with the problem of knowing exactly what the meaning of a given expression actually is. For example, what is the meaning of "drive"? Is it a function, an event, an algorithm? Is it extensional, intensional, time dependent? My own stance here is that basically expressions have propositional content and the meaning of a proposition is its truth conditions. This implies that it is not a function in the sense of Frege (from individuals to truth values) and that the dependencies it displays result from the conditions that it places on the model. Yet, what exactly the formal nature of truth conditions is, is far from clear. Logicians have unfortunately also been quite complacent in thinking that the calculi they have formulated are compositional. They mostly are not. For this reason I have to take a fresh start and develop something of a calculus of truth conditions. The problem is that certain vital constructs in logic must be discarded when dealing with natural language semantics. One of them are variables, another is type theory. To see why this is so we must simply ask ourselves what the semantics of a variable, say, "x" is and how it differs from the semantics of a different variable, say "y". Moreover, these meanings should be given *independently of the form of the expression*. The result is that there is nothing that can distinguish the meaning of "x" from that of "y" because all there is to the difference is the difference in name. Consequently, if names are irrelevant, the meaning of the expression "R(x,y)" is the same as "R(y,x)", that is, we cannot even distinguish a relation and its converse!

This observation has far reaching consequences. For if we accept that we cannot explicate same or different references in terms of variables then the composition of meanings is severely restricted. Indeed, I shall show that it amounts to the restriction of predicate logic to some finite variable fragment. On the other hand, I will argue that nevertheless this is precisely what we want. Consider an ergative language like Dyirbal. Dixon (1994) translates the verbs of Dyirbal by their passives in English. So, the verb meaning "hit" is translated by "is hit by". This makes a lot of sense for Dyirbal, as it also turns out that the transitive object in Dyirbal is the syntactic pivot in coordination. Yet, we may wonder how come that "hit" can at all *mean the same thing as* "is hit by", for "John hits Rover." does not mean the same as "John is hit by Rover.". The answer lies here in a distinction between meaning and meaning composition. The way the verb "hit" composes with a subject expression is certainly different from that of "is hit by". And yet, both have as their truth conditions that someone hit someone.

Similarly, the issue of types is a difficult one. Take once again the meaning of the transitive verb "hit". Montague gave it the type $e \to (e \to t)$ (it is enough to look at the extensional type). This means that it is a function that, when given an object, returns an intransitive verb, which in turn is a function that returns a truth value when given an object. So the first object supplied is the direct object. We could think however that it is just the other way around (compare Dyirbal for that matter):

the first to be supplied is the subject and the direct object comes next. Alternatively we may give it the type $e \bullet e \rightarrow t$, which returns a truth value when given a subject paired with an object. Which of the three is correct? The problem is that they are all equivalent: choose one, get the others for free. From a technical viewpoint this is optimal, yet from our viewpoint this says that there is no a priori way to choose the types. Moreover, from a philosophical point of view this gives rise to what has been termed Benaceraff's Dilemma after Benaceraff (1973): if we cannot choose between these formalizations how can we know that any of them is correct? That is, if there are such objects as meanings but they are abstract then how can we obtain knowledge of them? If we are serious about meanings then either we must assume that they are real (not abstract) or else that they do not exist. In particular, the idea that types are abstract properties of objects is just an illusion, a myth. Types are introduced to smoothen the relationship between syntax and semantics. They are useful but not motivated from semantics. In this connection it is important to realize that by semantics I do *not* mean model theoretic semantics. If I did, then any type assignment could be motivated from a needed fit with a particular formal model. Instead, I think of semantics primarily as truth conditions in the world.

In order to understand how this affects thinking semantically, take the sentence "John is hitting Rover.". How can we judge whether this sentence is true? Obviously, it is of no help to say that we have to look whether or not the pair consisting in John and Rover is in the hit-relation. For it is the latter that we have to construct. That we somehow possess a list of pairs where we can look the facts up is no serious suggestion. Obviously, such a list if it ever existed has to be compiled from the facts out there. But how? Imagine we are witnessing some incident between John and Rover or watching a film—where is that relation and how are we to find it? Clearly, there must be other criteria to tell us who is the subject (or first argument) and who is the object (or second argument). So, for a given situation we can effectively decide which object can fill the first slot and which one the second slot so that they come out as a pair in the hit-relation. Once we have established these criteria, however, there is no need to appeal to pairs anymore. For whatever it is that allows us to judge who will be subject, it is this procedure that we make use of when inserting the subject into the construction but not earlier. The pair has become redundant.

A type theorist will object and say: so you are in effect changing the nature of meanings. Now they are functions from scenes (or films) to objects or whatever but still you uphold type distinctions and so you are not eliminating types. I actually agree with this criticism. It is not types as such that I wish to eliminate. There are occasions when types are necessary or essentially equivalent to whatever else we might put in their place. What I contest is the view that the types tell us anything of essence about the syntax of the expressions. We can of course imagine languages where the fit is perfect (some computer languages are of that sort) but the truth is that natural languages are definitely *not* of that kind.

I have said above that language is a relation, that is, a set of pairs. This relation is many-to-many. A given meaning can be expressed in many ways, a given expression may have many meanings. However, one may attempt to reduce the complexity by a suitable reformulation. For example, we may think that an expression denotes

not several meanings but rather a single one, say, the set of all its meanings. Call this kind of meaning *set-meaning* and the other the *ground meaning*. Thus, /crane/ denotes a set of ground meanings, one covering the bird meaning and the other the machine meaning. This technical move eliminates polysemy and makes language a function from expressions to (set-)meanings. There are however many problems with this approach. The first is that the combination of two set-meanings is much more complex than the combination of ground meanings, for it must now proceed through a number of cases. Consider namely how complex signs are being made. Given a two place function f, a complex sign is made from two simple signs, each being an expression paired with a ground meaning. It is thus particular expressions with particular ground meanings that are composed via f and not expressions with all their meanings or meanings with the totality of their expressions. If an expression is polysemous the claim is therefore that it must enter with any one of its meanings in place of the collection of all its meanings. The expression /big crane/ can therefore be formed with two particular meanings for /crane/, each of them however taken on its own. The expression is thus again polysemous insofar as the combination of "big" with any of the two ground meanings makes sense. Similarly, /all cranes/ can never be a quantification over objects of the expression /cranes/ in both senses simultaneously. It can only be either of them: a quantification over some birds, or a quantification over some machines. Lumping the two meanings into a set therefore creates options that languages do not seem to have. Or, more precisely, the fact that a given expression has two ground meanings (= is polysemous) is technically different from it having a set-meaning.

As the reader will no doubt notice the present monograph is quite technical. This is because I felt it necessary to explore certain technical options that the setup allows for. Since the details are essentially technical there is no point in pretending that they can be dealt with in an informal way. Moreover, if we want to know what the options are we better know as exactly as possible what they consist in. It so turns out that we can obtain certain results on the limitations of compositionality. Moreover, I show that certain technical manoeuvers (such as introducing categories or eliminating polysemy) each have nontrivial side effects that need to be addressed. By doing this I hope to provide the theoretical linguist with a tool for choosing among a bewildering array of options.