Case at the Syntax/Semantics Interface

Marcus Kracht
II. Mathematisches Institut
Arnimallee 3
D – 14195 Berlin
kracht@math.fu-berlin.de

Abstract

For most theories of semantics cases are irrelevant at best. For these theories the ideal language has fully regimented word order but no morphology. This means that cases have to disappear at the interface from syntax to semantics. The role of syntax is among other to provide a structural substitute for cases. This is also the philosophy of the Minimalist Program, where cases are treated as uninterpretable features. We argue here in line with [9] that case is not only independent from syntactic structure, but that it can be a substitute for it. Moreover, we shall provide a semantics in which cases are instrumental in providing the string–to–meaning translation. This shows that there is any reason to believe that cases survive at the interface.

Why is there Case? Case is certainly a syntactic and a morphological category. Theories of syntax and morphology give ample evidence of this. But it is not at all clear whether case is also a semantic category. In the Minimalist Program, for example, case is the prototypical example of a feature that serves no interpretive purpose and hence must be deleted before the structure is shipped to the syntax–to–meaning interface. Support from this view comes also from semantical theory. Montague Grammar, for example, has no home for case. In MG, the string–to–meaning translation is effected in two steps:

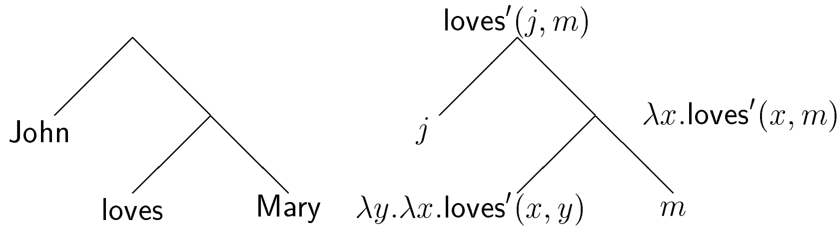
1. Assign a binary analysis to the string.

2. Apply the semantic translation rules bottom up.

Take for example the string (1) possesses the analysis (1a) and (1b).

(1) John loves Mary.
(1a) [John [loves Mary]]
(1b) [[John loves] Mary]

However, since the type assignment is such that John cannot be combined with loves, the analysis (1b) is ruled out. So, John loves Mary will be translated as $\mathsf{loves}'(j, m)$ in virtue of there being only one analysis compatible with the syntactic types. [1]



From the standpoint of MG, there is little in addition to the semantical types that is needed in order to get both the syntax and the semantics right. A string is a grammatical sentence if and only if it has a translation if and only if it has the syntactic type t. In its original formulation, the extra ingredient was the *directionality* of syntactic types.

Montague dealt in his papers exclusively with English. Since English has only very marginal morphology, Montague did not spend much energy in integrating it into his system. Cases and agreement appear in the system as morphological side conditions on the rules. For example, there is a rule saying that if instead of Mary we use a pronoun, it must be put into the accusative, and if we use a pronoun instead of John it must appear in the nominative. If one extends MG to languages with richer morphology, one is driven to the proposal by [1] (adopted by [3]) to annotate for each syntactic argument whether there is a morphological condition on this argument. The argument structure for the German verb lieben (*to love*) is in this system the

[1] To makes matters as simply as possible, we do not typeraise NPs. Here, NPs will be semantic arguments of the verb. The reader is assured that the arguments put forward here do not depend on this choice.

following.

$$\begin{array}{ccc} \lambda x. & \lambda y. & \mathsf{love}'(y,x) \\ \vdots & \vdots & \\ \text{ACC} & \text{NOM} & \end{array}$$

The intention is that whenever this structure takes an argument, then the morphological annotation associated with the relevant abstracted variable must be respected. This makes sure that the first argument with which lieben is combined will be in the accusative case. Notice that the conditions are external stipulations, there is nothing in the λ–calculus that corresponds to them and they cannot be integrated other than by typing the arguments. It is however not very helpful to think of a semantical type *nominative*, or of *accusative* even though such speculations have occasionally be made (for example — though not explicitly — in HPSG). Morphology simply appears to be a needless complication of the sentence. The most ideal language from the standpoint of MG would be Polish or Reverse Polish Notation: all heads universally precede or follow their arguments, and the arguments come in a fixed order. This includes the word orders VSO, VOS, SOV and OSV. SVO and OSV are not uniquely readable, but the claim of MG is that the different analyses of strings do correspond to the ambiguities in meaning. So, languages with regimented word order appear to fit rather nicely with Montague Grammar in absence of morphological complications.

Now, languages without an equivalent of case morphology have several disadvantages. The main disadvantages are the following.

- There is no possibility to drop arguments without introducing massive ambiguity. The sentence Love John could either mean Love John someone or Love someone John.

- No word order variation is allowed. Love John Mary and Love Mary John do not mean the same.

- Polyvalence introduces ambiguity.

Positively speaking, anything other than structure that lets us recover the argument structure of the sentence gives us some freedom either to drop arguments, or to vary the word order variation without increasing the ambiguity. This remark is of purely information theoretic character. There is no point in denying that. It is of course another question whether languages reject ambiguity to the extent that it is rejected in formal semantics, say, but the burden this places on communication is certainly clear.

Therefore, cases introduce redundancy into language if we would simply add them in addition to full word order regimentation. However, as a matter of fact, cases are often *not redundant*, since languages with rich enough case morphology relax the word order regime to some extent. Such is the case with German.

(2) Der Vater sieht den Sohn.
the-NOM father-NOM/ACC sees the-ACC son-NOM/ACC
The father sees the son.

(3) Den Vater sieht der Sohn.
the-ACC father-NOM/ACC sees the-NOM son-NOM/ACC
The son sees the father.

(4) Die Mutter sieht die Tochter.
the-NOM/ACC nother-NOM/ACC sees the-NOM/ACC daughter-NOM/ACC
The mother sees the daughter.
**The daughter sees the mother.*

As (2) and (3) show, with the cases interchanged the subject and object change places as well. Notice that the word Vater is ambiguous between nominative, accusative and even dative (which we did not annotate here). But when it goes together with the definite determiner der it must be nominative since the latter is, and when it goes together with den, it must be accusative since the latter is. German has, as opposed to English, freer word order, which in turn makes the cases nonredundant. Notice that in absence of any information (4) is read as SOV, and not OSV, a fact which shows that it is overt cases, not the abstract cases that are at issue.

In fact, in many languages there is free word order (either free on the level of constituents eg German or at the level of individual words eg Jiwarli). In such languages, cases are absolutely essential in recovering the meaning of the sentence. For example, any permutation of the following Latin sentence is again a (grammatical) Latin sentence and with roughly the same meaning.

(5) Senātus librum cōnsulī dat.
senate-NOM book-ACC consul-DAT gives.
The senate gives the book to the consul.

Sentence (5) raises a problem for syntactic theory. How can we ensure that any permutation of it will get the same meaning? Of course, we have not proved that all these permutations do have the same meaning, and we cannot be sure that no meaning difference will be (or has already been) discovered. [2] So, let us be a little bit more careful:

> How can the association between functor and argument be ensured in presence of free word order?

The answers to this are manyfold. However, the following is — we think — an uncontroversial thesis:

> Cases serve to disambiguate sentences in presence of free word order. Lack or loss of case distinctions leads in absence of any other factors to lack or loss of free word order.

It has been pointed out to me at certain occasions that lack of distinction in, say, subject and object, does not mean a language has fixed order of subject and object. Counter evidence is given in German, where (4) can be read as either SOV (the mother seeing the daughter) or as OSV (the daughter seeing the mother). Without going too much into this claim, notice that ambiguities of this sort are not negligeable in any language, since people dealing with people is what we are most concerned with. Second, the tendency to favour SOV *in absence of any other information* is so overwhelming that I have difficulties understanding that there is no tendency towards fixed word at all. For exactly such arguments and others drawing from the historical development of English been defended the reader is referred to [12].

Now, it is compatible with this thesis that cases are there *in addition to structure* and that cases are an overt indicators of the structure of a sentence which is often enough not so easily recovered from the string. So, it is conceivable that even if there are cases, there is enough structure so that the case relations can be recovered from the structure alone. This is the claim in Transformational Grammar and subsequent theories. It is our next goal to discredit this viewpoint.

The sentence (5) and its permutations will be our main concern in this paper. Up to now there exists to our knowledge no coherent semantic theory

[2] I have assured by Alan Dench (p.c.) that in some Australian languages, in particular Warlpiri and Jiwarli, no difference in meaning exists between different word orders analogous to those of (5) (on condition that they are permissible), not even a pragmatic difference. This implies that the problem exists if not for Latin then at least for *some* languages.

that can treat all admissible word order in a unified or principled way.

Case is Independent of Syntactic Structure. In much of the transformational literature cases are viewed as derived from the syntactic configuration. In particular, the accusative case is defined by means of a structural relation between the verbal head and its complement, namely sisterhood. Against this argues [3]. The innermost (ie most immediate) complement of the verb aussetzen is the dative complement, while for zumuten it is the accusative complement. Haider gives one main argument for his classification: focus spreading. According to [4] the focus domain is extensible if and only if the carrier of focus is contained in the most deeply embedded phrase. Hence, focus is most neutral if the carrier is a nonscrambled complement of the verb. It is therefore predicted that the complements are in base position in (6) but not in (7) since (6) is a good answer to a question *What happened?* while (7) is not. Similarly with (8) and (9).

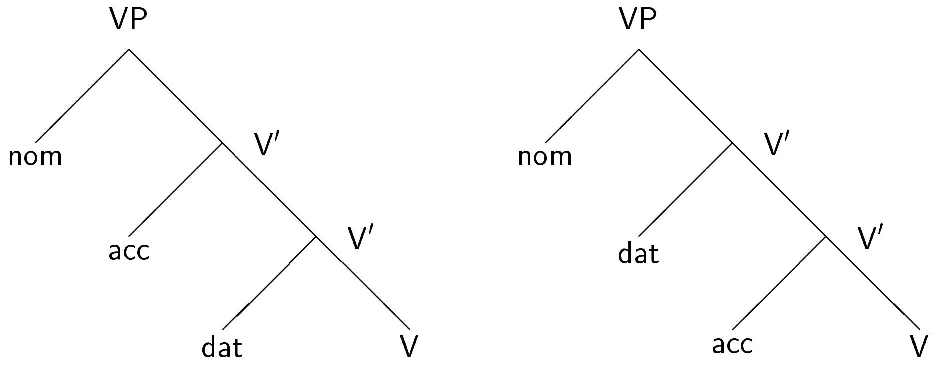
(6) Er hat die Kinder einer solchen GeFAHR ausgesetzt.
He has exposed the children to such a DANGer.

(7) Er hat einer solchen Gefahr die KINDer ausgesetzt.
He has exposed to such a danger the CHILdren.

(8) Er hat seinen Studenten eine solche KlauSUR zugemutet.
He has given his students such a TEST.

(9) Er hat eine solche Klausur einen StuDENten zugemutet.
He has given such a test to his STUdents.

It follows that some verbs select the dative complement first, others the accusative. We therefore have the following schematic structures.



This demonstrates that cases are not predictable from the structure alone. [3] An even stronger viewpoint is expressed by [9]. She claims the following:

> *Generalisation A.* Case morphology can construct grammatical relations on a par with, and independent from, phrase structure.
>
> *Generalisation B.* Case morphology can construct the larger syntactic context, including and providing complex information about the clause.

For us, Generalisation A is of immediate relevance. It claims not only that case is independent of structure but also that case is — in some circumstances — *a substitute of syntactic structure*. In other words, there may even be languages in which there is no syntactic structure and just words. We have studied such languages from a formal point of view in [8].

Case and Word Order in Categorial Grammar. There are alternative ways in categorial grammar to account for case and word order. Cases are treated by annotations on the categorial labels. The verbs above receive the following type assignment:

zumuten	$(\mathrm{NP}[acc]\backslash(\mathrm{NP}[dat]\backslash(\mathrm{NP}[nom]\backslash S)))$
aussetzen	$(\mathrm{NP}[dat]\backslash(\mathrm{NP}[acc]\backslash(\mathrm{NP}[nom]\backslash S)))$

In Unification Categorial Grammar (UCG), category labels were anyway complex feature structures, so that case can be specified in addition. In CCG (Combinatory Categorial Grammar, see [11]) cases are treated as type lifters. Nominative case lifts the NP to a function over intransitive verbs yielding sentence. However, if cases are independent of transitivity (as for example in German), this approach is bound to fail for accusative and dative. (There are verbs selecting nom–dat (gefallen, *to please*), and others selecting nom–acc (mögen, *to like*). Also the facts mentioned above militate against this view.) The problem with the first proposal is that it allows also for adjectives to change the case of the noun, an option that is not found in languages. For example, the following is the type of a premodifier changing the case of the argument from dative to accusative:

$$\mathrm{NP}[acc]/\mathrm{NP}[dat]$$

[3] In the current theory, the Minimalist Program, this argument is no longer valid, since arguments are invariably never found in their base position. This makes matters worse for the MP, since it looses the explanation for the contrast between (6) and (7) and between (8) and (9).

Categorial Grammar provides no principled way of excluding such things.

Word order also is a problematic issue. Fixed word order is no problem. [4] However, as soon as there is freedom, problems start to arise. The easiest road is to introduce lexical rules. However, lexical rules are too unconstrained to give us any valuable insights. It has therefore been proposed to use a nondirectional slash (here written $\multimap$), which is order insensitive. This allows for certain types of free word order, for example that of Sanskrit as proposed by [10]. Staal notes that Sanskrit allows all four orders SOV, SVO, VOS and OVS. This is easily captured in order insensitive slash. However, languages with free word order of the Latin type still remain problematic. The CCG analysis of [11] needs no stipulations at all. It allows V and S to combine directly. The idea is that a subject is analysed as $V_1 \multimap S$ (here V_1, V_2, V_3 are the categories of intransitive, transitive, ditransitive verbs, respectively, an object as $V_2 \multimap V_1$. In addition to application, function composition is a rule of combination:

$$\begin{array}{ccc} V & S & O \end{array}$$
$$\dfrac{V_2 \quad \dfrac{V_1 \multimap S \quad V_2 \multimap V_1}{V_2 \multimap S}}{S}$$

Still, even under these generous assumptions on typing, Latin remains a problem. The following word orders are ruled out by Steedman's system contrary to fact: IVSO and OSVI. This means that even CCG rules out some permutations of (5).

Discourse Representation Theory. As is well known, existentially quantified variables are accessible even outside of their scope. This has led Kamp to propose a new kind of semantics, called Discourse Representation Theory (DRT) (see [5] for an introduction), in which in contrast to Montague Grammar variables appear as discourse objects and the structures are expanded simply by adding new material to the DRS. A DRS is a pair $\delta = [V : \Delta]$, where V is the *head section* consisting of a set of discourse objects and Δ is a set of *conditions*, consisting of either formulae or DRSs. 'And' is interpreted by $\bullet$, called the *Zeevat–Merge*. Formally, it is defined by

$$[V_1 : \Delta_1] \bullet [V_2 : \Delta_2] := [V_1 \cup V_2 : \Delta_1 \cup \Delta_2]$$

[4]Certainly, for the orders SOV, SVO, OVS and VOS there is no problem. The object is the first argument and te subject the second. Only the order is specified in one or the other way. If the verb is allowed to consume its subject first — and form a constituent with it — then also VSO and OSV can be accounted for.

Here is an example.

(10) John sees a woman. She is walking.

This is translated by translating each word into a DRS and then applying the Zeevat–Merge. The last step is this:

$$\begin{array}{c} \mathsf{/John\ sees\ a\ woman/} \\ \begin{array}{|c|} \hline x \quad y \\ \hline x \doteq j; \mathsf{woman}'(y); \\ \mathsf{see}'(x, y) \\ \hline \end{array} \end{array} \bullet \begin{array}{c} \mathsf{/She\ is\ walking./} \\ \begin{array}{|c|} \hline z \\ \hline z \doteq y; \\ \mathsf{walk}'(z) \\ \hline \end{array} \end{array}$$

$$= \begin{array}{c} \mathsf{/(9)/} \\ \begin{array}{|c|} \hline x \quad y \\ \hline x \doteq j; \mathsf{woman}'(y); \\ \mathsf{see}'(x, y); z \doteq y; \mathsf{walk}'(z) \\ \hline \end{array} \end{array}$$

Technically, the pronoun introduces the discourse referent z and adds the condition $z \doteq y$. The anaphoric binding potential of a pronoun is captured as follows:

> A DRS is *closed* if every discourse object occurring in some formula must occur in the head section of an accessible DRS.

Accessibility is a technical notion, which need not concern us here. For the moment we shall just say that it means that V contains all variables occurring in Δ. The reason why pronouns want to be bound is that all DRSs corresponding to a full text must be closed.

It has subsequently been shown that this effect can also be achieved by introducing a new semantics for predicate logic, the so–called dynamic semantics. Here, a variable can in effect be bound by an existential quantifier even outside its scope. The formal translation of (10) is then simply (11), which has the truth conditions of (12) of ordinary predicate logic:

(11) $(\exists y.\mathsf{see}'(j, y)) \wedge \mathsf{walk}'(y)$

(12) $(\exists y.\mathsf{see}'(j, y) \wedge \mathsf{walk}'(y))$

Marrying Dynamic Semantics and MG, Groenendijk and Stokhof have developed the so–called Dynamic Montague Grammar (DMG) ([2]), which allows for dynamic binding. Dynamic Semantics has become a very dominant interpretation model in present day semantical theory.

Referent Systems. In a series of papers, Albert Visser and Kees Vermeulen have scrutinised the methods of dynamic semantics (see among others [13] and [14]). Basically, what is lost when we step into dynamic semantics is the substitutivity of bound variables. Already, this problem has appeared in MG with pronouns. The problem is this: there is no unique logical formula or semantic unit that is substitutable for a given word in each context. For example, suppose the DRS for dog was

$$\begin{array}{|c|}\hline \varnothing \\ \hline \mathsf{dog}'(x) \\ \hline \end{array}$$

Then the translation for (12) would be the following DRS:

$$\begin{array}{|c|}\hline x \\ \hline \mathsf{dog}'(x); \mathsf{see}'(x,x) \\ \hline \end{array}$$

(12) A dog sees a dog.

However, this is contrary to fact. Instead, when the phrase a dog occurs for the second time, we must choose a different variable, eg y, so that we end up with the following correct translation:

$$\begin{array}{|c|}\hline x \quad y \\ \hline \mathsf{dog}'(x); \mathsf{dog}'(y); \\ \mathsf{see}'(x,y) \\ \hline \end{array}$$

So, words do not have a unique translation, rather the variables must be carefully chosen before insertion into the translation. In fact, without a proper choice we will not get even the simplest facts right, eg the identification of subject and object. The following question therefore arises:

> Who chooses the variables for us?

DRT uses the syntactic parse as input with full indexation. So, rather than (10), what the semantic interface is given is (14):

(14) John_1 $\mathsf{sees}_{1,2}$ a_2 woman_2. $\mathsf{She}_{2,3}$ is_3 $\mathsf{walking}_3$.

Having gone this far, we are back at square one: unlike MG we do not even get the translation from the syntactic structure, we must get the indexing of the variables as well. Now, it is clear that syntax does only part of the indexing (if at all), so the whole algorithm must be supplemented by an extra component that takes care of the variable handling. In MG this component is the λ–calculus. However, MG suffers from the same problem: it does not do *all* the variable handling, only parts of it. Roughly, the set of cases handled by MG coincides with that of syntax. Now if that is so, we can either renounce the problem or start again by considering an explicit mechanism that does the handling. This has been the concern of [13] and the subsequent [14].

Visser and Vermeulen argue that the problem lies with the Zeevat–Merge. By using the name x, two different DRSs are taken to be talking about the same discourse object. They argue that instead DRSs should by default be taken to talk about *different* objects. The default merge is therefore

$$[V_1 : \Delta_1] \circ [V_2 : \Delta_2] := [V_1 \times \{1\} \cup V_2 \times \{2\} : \Delta_1^1 \cup \Delta_2^2]$$

where Δ_1^1 results from Δ_1 by replacing each variable x by $\langle x, 1\rangle$ (which we write x^1) and Δ_2^2 results from Δ_2 by replacing each variable x by $\langle x, 2\rangle$ (which we denote by x^2). This seems to make matters worse. For then (15) means something like (16):

(15) John sees Mary.
(16) *There is John. Someone sees someone. There is Mary.*

This is where the referent systems come in. Referent system is a device for the explicit handling of variables. By choosing in addition to the usual letter x, y etc a so–called *name* for their variables, referent systems communicate which variables they are using should be taken as talking about the same thing. [5] Variables with the same name will end up as the same variable in the merge, but variables with different names will end up as different variables. This is the key idea. To put this into a slogan:

> Referent Systems provide means of making the variable handling explicit rather than leaving it implicit.

This means specifically that we can use genuine linguistic information to do the variable handling and drive the translation. A *semantic structure* looks

[5] In what is to follow, we will ignore the distinction between a variable and a referent. A variable is a particular member of the set of variables. However, we never write down variables, we write metavariables. This hardly creates a confusion and the distinction is hardly maintained even in logic textbooks. Likewise, we propose to be less strict in our terminology in the hope that no confusion will arise as well.

like this:

$$\begin{array}{|c|}\hline R \\ \hline \hline V \\ \hline \Delta \\ \hline \end{array}$$

where $[V : \Delta]$ is a DRS and R a special kind of referent system, which we call the *argument structure*. This is what we shall turn to now.

Argument Structure in Referent Systems. The very basic idea is as follows. Each referent comes with a name. For simplicity, this name shall be a case. Some referents have no name, they are called *anonymous*. If two DRSs are merged, then those referents that have the same name click together. All others are treated as distinct entities. The basic rule is this (where $B \neq A$):

$$\begin{array}{lcccl} \langle x, A\rangle & \circ & \langle y, A\rangle & = & \langle x^1, A\rangle \\ \langle x, A\rangle & \circ & \langle y, B\rangle & = & \text{undefined} \end{array}$$

Argument structure is a set of such pairs. Merge is by merging the sets pairwise (this is not always well defined). The following example gives an idea of the mechanics of merge:

$$\begin{array}{|c|}\hline \langle x, nom\rangle, \langle y, dat\rangle \\ \hline \hline x \\ \hline \varphi(x, y, z) \\ \hline \end{array} \circ \begin{array}{|c|}\hline \langle x, nom\rangle, \langle z, dat\rangle, \langle y, acc\rangle \\ \hline \hline y \\ \hline \psi(x, y, z) \\ \hline \end{array}$$

$$= \begin{array}{|c|}\hline \langle x^1, nom\rangle, \langle y^1, dat\rangle, \langle y^2, acc\rangle \\ \hline \hline x^1 \quad y^2 \\ \hline \varphi(x^1, y^1, z^1); \\ \psi(x^1, y^2, y^1) \\ \hline \end{array}$$

We allow referents to have sets of names, for example $\{nom, acc\}$. This model is however overly naive. All referents in the nominative would be taken as talking about the same thing. Hence, we must assume that variables can loose their names under certain circumstances. We shall therefore annotate the name as follows: we write $\vartriangle$ if the argument structure supplies the name, $\triangledown$ if it consumes the name, and $\Diamond$ if it does both. $\Diamond$ is an abbreviation of $\{\vartriangle, \triangledown\}$. Finally, we write $\varnothing$ if it does neither. In that case the referent is as if anonymous. We call these marks *vertical diacritics*.

$$\begin{array}{lclcl}
\langle x, \triangle, A\rangle & \circ & \langle y, \triangledown, A\rangle & = & \langle x, -, A\rangle \\
\langle x, \triangle, A\rangle & \circ & \langle y, \Diamond, A\rangle & = & \langle x, \triangle, A\rangle \\
\langle x, \Diamond, A\rangle & \circ & \langle y, \triangledown, A\rangle & = & \langle x, \triangle, A\rangle \\
\langle x, \Diamond, A\rangle & \circ & \langle y, \Diamond, A\rangle & = & \langle x, \Diamond, A\rangle
\end{array}$$

So, in each case a pair consisting of a $\triangle$ and a $\triangledown$ is cancelled. We also say that if the argument structure contains x with diacritic $\triangle$ ($\triangledown$, $\Diamond$) that it is an *x–argument* (*x–functor*, *x–adjunct*).

The next complication is that we introduce order sensitivity. If the vertical diacritic is $\triangledown$ or $\Diamond$, an additional *horizontal diacritic* is added. It is associated with the $\triangledown$ of the vertical diacritic. It is a subset of $\{\curvearrowleft, \curvearrowright\}$. We write $\circlearrowright$ instead of $\{\curvearrowleft, \curvearrowright\}$. When the $\triangledown$ and $\triangle$ are cancelled, the referent carrying the $\triangledown$ wants the referent carrying $\triangle$ to its right ($\curvearrowright$), left ($\curvearrowleft$) or on either side ($\circlearrowright$). If the horizontal diacritic is $\varnothing$, no merge succeeds.

An AHS is a triple consisting of a referent, a pair of diacritics, and a set of names. It looks like this:

$$\begin{array}{lcccc}
\alpha = & \langle x, & \triangledown & \curvearrowright, & acc\rangle \\
 & \text{referent} & \text{handling} & \text{directionality} & \text{name}
\end{array}$$

α says in informal talk: when two structures are merged, the referent x is identified with whatever referent in the other structure has the name *acc*, if that structure is to the right ($\curvearrowright$). $\triangledown$ specifies that the name is thrown away after merge. We first do the merge of two single AHSs. Legitimate examples:

$$\begin{array}{lclcl}
\langle x, \triangledown \curvearrowright, nom\rangle & \circ & \langle y, \triangle\, \varnothing, nom\rangle & = & \langle x^1, \varnothing\varnothing, -\rangle \\
\langle x, \Diamond \curvearrowright, nom\rangle & \circ & \langle y, \triangle\, \varnothing, nom\rangle & = & \langle x^1, \triangle\, \varnothing, nom\rangle \\
\langle y, \triangle\, \varnothing, nom\rangle & \circ & \langle x, \triangledown \curvearrowleft, nom\rangle & = & \langle y^1, \varnothing\varnothing, -\rangle \\
\langle y, \Diamond \curvearrowleft, \{nom, acc\}\rangle & \circ & \langle x, \Diamond \curvearrowleft, nom\rangle & = & \langle y^1, \varnothing \curvearrowleft, nom\rangle
\end{array}$$

Definition 1 *An* ***argument handling statement over N*** *is a triple $\langle x, \delta, S\rangle$, where x is a variable, Δ a pair consisting of a vertical diacritic and a horizontal diacritic, and $S \subseteq N$ a set of names. An* ***argument structure (over N)*** *is a sequence of AHSs over N such that (1) no variable appears twice, (2) exactly one variable has the diacritic $\Diamond$ or $\triangle$. This variable is called the* ***head*** *of the argument structure.*

We write *nom* rather than $\{nom\}$ for singleton sets. The merge of argument structure is described below.

Table 1: The Basic Syntactic Types

	$\triangle$	$\lozenge$
object	noun	adjective
event	verb	adverb

Basic Syntax. An argument structure contains only one head. The head decides the syntactic category of the item. We have two basic semantic types (there may be more): objects and events. Now let x be the head of R. Then we assign a category to R in the following way. We have two basic dichotomies. The vertical diacritic of x can be $\triangle$ or $\lozenge$, and the type of x can be that of an object or that of an event. This gives four possibilities, which are summarized in Table 1. So, the basic categories fall out from the argument structure. [6] Further, there is no need to distinguish syntactically between heads and phrases. We shall assume that an argument structure is *phrasal* if it contains only a single AHS.

For English nouns we get for example:

/dog/
$\langle x, \triangle\ \varnothing, nom\rangle$
$\varnothing$
$\mathsf{dog}'(x)$

/teacher/
$\langle x, \triangle\ \varnothing, nom\rangle, \langle y, \triangledown \curvearrowleft, gen\rangle$
$\varnothing$
$\mathsf{teach}'(x, y)$

Here is the structure of an English transitive verb (here $\star$ is some unspecific value):

/sees/
$\langle e, \triangle\ \varnothing, \star\rangle$, $\langle x, \triangledown \curvearrowright, nom\rangle, \langle y, \triangledown \curvearrowleft, acc\rangle$.
$\varnothing$
$\mathsf{see}'(e); \mathsf{exp}'(e) \doteq x;$ $\mathsf{thm}'(e) \doteq y$

Now how are argument structures combined? We shall assume throughout that we are dealing with configurational languages. Now, let A and B be

[6] Certainly, the type of x needs to annotated somewhere. However, we shall ignore this question of detail here. See [7] for an extensive discussion.

Table 2: Access Restrictions and Language Type

Access	Directionality		Word Order	Language
E	S: $\curvearrowleft$	O: $\curvearrowright$	SVO	English
E	O: $\curvearrowright$	S: $\curvearrowright$	VSO	Berber, Toba–Batak
E	S: $\circlearrowright$	O: $\circlearrowright$	$\{\mathrm{SVO}, \mathrm{SOV}, \mathrm{VOS}, \mathrm{OVS}\}$	Sanskrit
G	S: $\curvearrowleft$	O: $\curvearrowright$	$\{\mathrm{S(VO)}, \mathrm{(SV)O}\}$	?
G	S: $\curvearrowleft$	O: $\curvearrowleft$	SOV+scrambling	German
G	S: $\circlearrowright$	O: $\circlearrowright$	Free	Latin, Ancient Greek

two argument structures. We say that $\langle \alpha, \beta \rangle$ is a *matching pair* of A and B if $\alpha \circ \beta$ is defined. Now, $A \circ B$ is defined iff there exists a matching pair of A and B such that α accesses β and β accesses α. There are two possibilities for access. Languages will differ in whether they allow G–access or E–access.

1. α *E–accesses* β if β is the last member of B.

2. α *G–accesses* β if β is the last member of B having the same name as α.

The difference between the two types of access is this: if the access is E–type, then the arguments in the argument structure must be discharged in the order listed. Word order variation is possible, but the configurationality is fixed. If the access is G–type, then the configurationality is flexible, allowing for what looks like scrambling (if arguments are uniformly on one side). Table 2 presents a nonexhaustive list of the possibilities for word order variation: See [10] for word order in Sanskrit, [6] for Berber and Toba–Batak. Word order in English and German are folklore (see [3]). In standard grammars of Latin and Ancient Greek one will find support for the claim that they have free word order. Otherwise, free word order has been claimed for Wambaya in [9]. See also references therein. There are a number of possibilities that have not been tried. We may — for example — argue that English allows both S(VO) and (SV)O, whereby quantifier raising from direct object over the subject is made possible. This can be implemented in the following way. We may allow to define access in such a way that it is either directionality insensitive or directionality sensitive. The previous definitions of access can be classified as directionally insensitive. The directionally sensitive versions are as follows:

1. α *ED–accesses* β if β is the last member of B with the same horizontal diacritic as α.

2. α *GD–accesses* β if β is the last member of B having the same name as α and the same horizontal diacritic.

It is easy to see that the in a verb peipheral language (word orders SOV, OSV, VSO, VOS and combinations thereof) there is no difference between E–access and ED–access and likewise no difference between G–access and GD–access for the verbal arguments. Only in verb medial constructions it is manifest, as in English. It is therefore predicted that German will not allow for object quantifier raising, if the latter is a corollary of the different syntactic structure. Indeed, German does not seem to allow for object quantifier raising. We are now left with a more fine grained notion of access. It is not clear yet whether this distinction has any significance beyond the facts just noted. This needs close investigation.

So, not only basic word order is accounted for but also word order freedom. Take the permutation of (5) which is problematic for CCG:

(17) Cōnsulī dat senātus librum.

$$
\begin{array}{c}
/\text{cōnsulī}/ \\
\begin{array}{|c|}\hline \langle x, \triangle\ \varnothing, dat\rangle \\ \hline x \\ \hline \mathsf{consul}'(x) \\ \hline \end{array}
\end{array}
\circ
\begin{array}{c}
/\text{dat}/ \\
\begin{array}{|c|}\hline \langle e, \triangle\ \varnothing, \star\rangle, \langle x, \triangledown\ \circlearrowright, nom\rangle, \\ \langle y, \triangledown\ \circlearrowright, dat\rangle, \langle z, \triangledown\ \circlearrowright, acc\rangle. \\ \hline \varnothing \\ \hline \mathsf{give}'(e); \mathsf{thm}'(e) \doteq y; \\ \mathsf{agt}'(e) \doteq x; \mathsf{goal}'(e) \doteq z. \\ \hline \end{array}
\end{array}
\circ
\begin{array}{c}
/\text{senātus}/ \\
\begin{array}{|c|}\hline \langle x, \triangle\ \varnothing, nom\rangle \\ \hline x \\ \hline \mathsf{senate}'(x) \\ \hline \end{array}
\end{array}
$$

$$
\circ
\begin{array}{c}
/\text{librum}/ \\
\begin{array}{|c|}\hline \langle x, \triangle\ \varnothing, acc\rangle \\ \hline x \\ \hline \mathsf{book}'(x) \\ \hline \end{array}
\end{array}
$$

$$
=
\begin{array}{c}
/(17)/ \\
\begin{array}{|c|}\hline \langle e^{211}, \triangle\ \varnothing, \star\rangle. \\ \hline x^{111} \quad y^{211} \quad x^{211} \\ \hline \mathsf{give}'(e^{211}); \mathsf{thm}'(e^{211}) \doteq y^{211}; \\ \mathsf{agt}'(e^{211}) \doteq x^{211}; \mathsf{goal}'(e^{211}) \doteq z^{211}; \\ \mathsf{consul}'(x^{111}); \mathsf{book}'(y^{211}); \mathsf{senate}'(x^{211}). \\ \hline \end{array}
\end{array}
$$

We have chosen here the bracketing $((A_1 \circ A_2) \circ A_3) \circ A_4$. A different bracketing would have resulted in an alphabetic variant of the structure for (17). The second line of Table 2 is also interesting. If a language has VSO order, then the only way to assign a constituent structure is (VS)O. So the verb must have the subject lower in the argument structure. This is line with [6].

One consequence is worth noting. The availability of free word order depends in this model on the case names. From a speakers point of view, the labels are distinct, so (4) should be parseable either as SOV or as OSV. However, from a hearers point of view, only the reading SOV will be generated. For if a complement could be the object, then by the rules of G–access it is. There is no choice. Dutch has less distinctive morphology, and scrambling in Dutch is more restricted than in German.

Case Morphology and Case Agreement. While the verb selects case, nouns exist in various cases. To account for that, we shall assume that there is a symbol $\star$. Hence the set of formal case names for German is

$$\{nom, gen, dat, acc, \star\}$$

Nouns have the following structure in the lexicon:

$$\begin{array}{|c|}
\multicolumn{1}{c}{/\mathsf{Hund}/} \\ \hline
\langle x, \triangle\ \varnothing, \star\rangle \\ \hline
\varnothing \\ \hline
\mathsf{dog}'(x) \\ \hline
\end{array}$$

For each morphological case there is a morpheme that changes $\star$ into the corresponding case name. Here we assume that functional elements have the power to change the name of a referent (which lexical elements cannot do). This is indicated by the symbol $A \mapsto B$:

$$\begin{array}{|c|}
\multicolumn{1}{c}{/\textsc{nom}/} \\ \hline
\langle x, \Diamond \curvearrowleft, \star \mapsto nom\rangle \\ \hline
\varnothing \\ \hline
\varnothing \\ \hline
\end{array}$$

Case is a suffix (it has the diacritic $\curvearrowleft$), and we get for example:

$$\begin{array}{|c|}
\multicolumn{1}{c}{/\mathsf{Hund}^\frown\textsc{nom}/} \\ \hline
\langle x, \triangle\ \varnothing, nom\rangle \\ \hline
\varnothing \\ \hline
\mathsf{dog}'(x) \\ \hline
\end{array}$$

These case morphemes can also be added to adjectives:

$$\begin{array}{c} \text{/blau/} \\ \hline \multicolumn{1}{|c|}{\langle x, \Diamond \curvearrowright, \star \rangle} \\ \hline \multicolumn{1}{|c|}{\varnothing} \\ \hline \multicolumn{1}{|c|}{\mathsf{blue}'(x)} \\ \hline \end{array} \circ \begin{array}{c} /\textsc{nom}/ \\ \hline \multicolumn{1}{|c|}{\langle x, \Diamond \curvearrowleft, \star \mapsto \mathit{nom} \rangle} \\ \hline \multicolumn{1}{|c|}{\varnothing} \\ \hline \multicolumn{1}{|c|}{\varnothing} \\ \hline \end{array}$$

$$= \begin{array}{c} \text{/blau}^\frown\textsc{nom}/ \\ \hline \multicolumn{1}{|c|}{\langle x, \Diamond \curvearrowright, \mathit{nom} \rangle} \\ \hline \multicolumn{1}{|c|}{\varnothing} \\ \hline \multicolumn{1}{|c|}{\mathsf{blue}'(x)} \\ \hline \end{array}$$

So, /blau⌢NOM/ and /Hund⌢NOM/ can be merged to form an expression which is spelled out as blauer Hund. However, we could also merge blau and Hund without adding the case suffix. Although this is illegitimate in German, the corresponding example is grammatical in Hungarian.

(18) a kék kutyá-t
DET blue dog-ACC

(19) a kék kutyá-nak
DET blue dog-DAT

To account for the difference between German and Hungarian, we assume that languages differ in whether case is attached to the word or is a phrasal affix. Alternatively, one may argue that morphology specifies for the inflectional categories how their argument structure must look like to be an independent unit. This yields the following classification:

Agreement		Type	Languages
+ noun	+ adjective	case agreement in NP	Latin, German
+ noun	– adjective	phrasal case	Hungarian, Japanese
– noun	+ adjective	modifier case	??
– noun	– adjective	no overt case	Chinese, English

The absence of the third type follows from the tendency of languages to mark the head if they mark the adjunct. As I have shown elsewhere (see [7]), this conception results in a model where functional elements fill the argument structure in much the same way as in Anderson's model of Morphosyntactic Structures.

Conclusion. Using referent systems, we see that cases (in particular core cases) provide information about the identification of variables in the semantics. This is a decisive step ahead from previous models, which could use as their source of linguistic input only the structure, without agreement morphology. In these theories, cases had to be thrown away before syntax hands the representation over to semantics. In just the same way, the Minimalist Program takes cases as features of grammar that must disappear before the structure is shipped to the interpretive component.

In our semantics however cases are an integral part even of the representation. If this line is taken, it is now possible to use any kind of information that the structure provides for the syntax/semantics interface. For example, using φ–features a model of agreement as well as anaphoric binding can be developed.

References

[1] Manfred Bierwisch. On the grammar of local prepositions. In M. Bierwisch, W. Motsch, and I. Zimmermann, editors, *Syntax, Semantics, and Lexicon*, Studia Grammatica XXIX, pages 1 – 65. Akademie Verlag, Berlin, 1988.

[2] Jeroen Groenendijk and Martin Stokhof. Dynamic predicate logic. *Linguistics and Philosophy*, 14:39 – 100, 1991.

[3] Hubert Haider. *Deutsche Syntax — generativ. Vorstudien zur Theorie einer projektiven Grammatik.* Gunter Narr Verlag, Tübingen, 1993.

[4] Tilman Höhle. Explikation für 'normale betonung' und 'normale wortstellung'. In W. Abraham, editor, *Satzglieder des Deutschen*, pages 75 – 133. Gunter Narr Verlag, Tübingen, 1982.

[5] Hans Kamp and Uwe Reyle. *From Discourse to Logic. Introduction to Modeltheoretic Semantics of Natural Language, Formal Language and Discourse Representation.* Kluwer, Dordrecht, 1993.

[6] Edward L. Keenan. On semantics and binding theory. In John A. Hawkins, editor, *Explaining Language Universals*. 1988.

[7] Marcus Kracht. Agreement Morphology, Argument Structure and Syntax. Manuscript, 1999.

[8] Marcus Kracht. The Combinatorics of Cases. Manuscript, 1999.

[9] Rachel Nordlinger. *Constructive Case. Evidence from Australian Languages.* CSLI Dissertations in Linguistics. CSLI, Stanford, 1998.

[10] J. F. Staal. *Word Order in Sanskrit and Universal Grammar.* Foundations of Language, Supplementary Series No. 5. Reidel, Dordrecht, 1967.

[11] Mark Steedman. Gapping as constituent coordination. *Linguistics and Philosophy*, 13:207 – 263, 1990.

[12] Theo Vennemann. An explanation of drift. In C. N. Li, editor, *Word Order and Word Order Change*, pages 269 – 305. University of Texas Press, Austing and London, 1975.

[13] Kees F. M. Vermeulen. Merging without mystery or: Variables in dynamic semantics. *Journal of Philosophical Logic*, 24:405 – 450, 1995.

[14] Albert Visser and Kees F. M. Vermeulen. Dynamic bracketing and discourse representation. *Notre Dame Journal of Formal Logic*, 37:321 – 365, 1996.