Linguist Philos (2007) 30:47–95
DOI 10.1007/s10988-006-9011-5

RESEARCH ARTICLE

The emergence of syntactic structure

Marcus Kracht

Received: 9 June 2005 / Accepted: 29 October 2006 / Published online: 16 March 2007

Abstract The present paper is the result of a long struggle to understand how the notion of compositionality can be used to motivate the structure of a sentence. While everyone seems to have intuitions about which proposals are compositional and which ones are not, these intuitions generally have no formal basis. What is needed to make such arguments work is a proper understanding of what meanings are and how they can be manipulated. In particular, we need a definition of meaning that bans all mentioning of syntactic structure; it is not the task of semantics to state in which way things are put together in syntax. The present paper presents such a theory of meaning. This, in tandem with some minimal assumptions on the syntactic process (that there can be no deletion) yield surprisingly deep insights into natural language. First, it rehabilitates a lot of linguistic work as necessary on semantic

This paper has been presented first at the 9th South Californian Philosophy Meeting in November 2004. The ideas go a long way back. My deepest intellectual credits are to Albert Visser and Kees Vermeulen. They have opened my eyes to the fact that indices are not what we really want. Unfortunately, Kees cannot see the fruit of our long conversations in 1991/1992 when I was a visitor at the philosophy department in Utrecht. His untimely death leaves a gap no one can fill. I will try my best to give credit to his contributions to semantics and continue where he had left things. I have toyed with the idea of performing the elimination of indices for a long time without major success. I was unable to see what to put in their place. The present paper tells me why this was so. What unfolded in front of my eyes was a maze of technical apparatus that is needed in order to push through. I have no regrets; I think the work had to be done. In the process of getting my ideas out, I had the benefit of help from Hans-Martin Gärtner, Ben Keil, Greg Kobele, Ed Keenan, Philippe Schlenker, Marcus Smith, Dominique Sportiche, and Ed Stabler. Needless to say they might not share my views on the matter and thus should not be held accountable for what I say in the sequel. The responsibility is entirely my own.

M. Kracht (✉)
Department of Linguistics,
UCLA, 3125 Campbell Hall, PO Box 951543, Los Angeles,
CA 90095-1543, USA
e-mail: kracht@humnet.ucla.edu

M. Kracht
MTA Nyelvtudományi Intézet, Benczur u. 33, H-1068, Budapest, Hungary

grounds and defends it against potential claims of redundancy. For example, θ-roles and linking are an integral part of semantics, and not syntax. To assume the latter is to put the cart before the horse. Second, as a particular example we shall show that Dutch is not strongly context free even if weakly context free. To our knowledge, this is the first formal proof of this fact.

Keywords Compositionality · Syntax · Semantics

In memory of Kees Vermeulen (1966–2004)

1 Introduction

By and large, language presents itself to us in the form of strings of sounds (or letters). Yet, linguists of all persuasion have argued that there is evidence for more structure than meets the eye. There are what we call constituents. There are two types of definitions that are being given for constituent: one is syntactic and the other is semantic. In structuralism, a constituent is defined syntactically, via the notion of substitutability in a context. On the other hand, constituents do seem to have meaning, while nonconstituents very often do not. This opens the way to define constituents as semantically meaningful substrings.[1] The idea of compositionality is based on these ideas in the following way. Let's assume the right model of syntax is a context free grammar. This means that it is described by means of context free rules like (1).

(1) $\quad S \to NP \quad VP$

As a claim concerning the generated string language it says that if a given string $\vec{x}$ is an NP and $\vec{y}$ is a VP, then the string $\vec{x}^\frown\Box^\frown\vec{y}$ is (among other things) an S.[2] If we assume compositionality, then we additionally claim the following: there is a function F such that if M is the meaning of the NP $\vec{x}$ and N is the meaning of $\vec{y}$ then $F(M,N)$ is the meaning of $\vec{x}^\frown\Box^\frown\vec{y}$. Montague was the first to work this idea out for linguistics. Following Frege he assumed that for F one basically has only one choice: function application. To explain this further: in his grammar, if we can form a constituent from two parts, the meaning of one of them is going to be a function that can take the meaning of the other as its argument. Since the context free rules encode order as well, and since the VP may be either to the right (as in English) or the left, it will turn out that we either have to apply the meaning of the left hand constituent to that of the right hand constituent, or that we have to do the converse.

With the introduction of Montague Grammar, the notion of compositionality was put into the spotlight. But what does it actually mean? It is usually defined like this: the meaning of a constituent is a function of the meaning of its immediate parts, the function being determined by the mode of composition. Notice that it says ''meaning of the parts'', so it presupposes that the parts do have meaning, and that this meaning is there to begin with. We are not free to introduce new meanings to hitherto unknown parts. The parts that are used in the course of this ''composition''

[1] I am tacitly assuming that constituents are continuous; I am doing this only for expository purposes.

[2] We use $^\frown$ to denote string concatenation and $\Box$ for the blank. We distinguish plain concatenation $\vec{x}^\frown\vec{y}$ from word concatenation $\vec{x}^\frown\Box^\frown\vec{y}$. This is not an accurate assessment of the orthographic facts; it only serves to remind us that on the side of strings even in context free grammars there is not just concatenation.

are the constituents mentioned above.[3] So, compositionality says that constituents are those parts that are used in the derivation of the meaning. In Kracht (2003) (see Sects. 3.1 and 5.7) I have spent considerable energy in pinning down in more detail what the principle actually says and what it does not say. Part of this is repeated below. I assume that language is a set of signs, consisting of a phonological representation (here: a string or a sequence thereof), a category, and a meaning. There is a restriction on the syntactic side: you are not allowed to destroy anything, and any string that you have formed on the way must be a substring of the entire string you are analysing. I do allow for empty strings; there are invisible signs, so to speak.

One problem that I have not addressed previously was: what are meanings? This question seems to be a hopeless one, and I am not pretending to have a definitive answer. Instead, I shall propose a few things that I claim do not get represented at all in semantics. These are the following:

(a) indices,
(b) order, and
(c) multiplicity.

As for indices, I propose that semantics has no notion of index in the sense of logic. Thus there are no variables x_8, x_{1043} or the like; the only existing things are what is called anonymous variables in computer science. In order to access a location where information on an individual has been stored you have to describe where to find it, or say what it contains. This idea has been proposed by Albert Visser and Kees Vermeulen in several papers (see Vermeulen, 1994, 1995, 1996), and by Fine (2003). Their solutions differ (and mine is still different), but we all agree that for the purposes of communicating meanings there is no such thing as x_{76}. Syntax may tag indices onto syntactic items but semantics will pay no attention to them. This thwarts any hopes of executing the program (Kamp & Reyle, 1993) without an additional layer that translates indices into something more useful for semantics. Kees Vermeulen has introduced referent systems in Vermeulen (1995) to supply such a layer. Referent systems have anonymous variables but you can access the location by using specially designed names. While the referent system may (or may not) use variables in the usual sense, it does not show them to you. All you can do is use names to call them. These names are agreed on beforehand (for example, 'subject', 'object', or 'nominative', 'accusative' etc.). This system also does away with the second complaint, the problem of order. In referent systems, the fact that underlyingly predicates are sets of n-tuples does little to help you find the linking of the arguments to the positions in the sequence.

My only worry about referent systems is the fact that they are not compositional. The names are something that has no meaning (or need not have any). They constitute an interface between syntax/morphology and semantics, an interface in the true sense of the word.[4] They were designed to link argument positions to slots in the

[3] Just as an etymological remark: `compositionality` has the Latin words `con` 'with' and `ponere` 'to put' in it. It actually also says that the meaning of the object is obtained by actually forming constituents and accompanying this with a formation of the associated meaning. This means that the accounts of semantics favoured in the Minimalist Program, which assume that meanings are computed at LF by induction on the structure, are not compositional in this sense. Yet they are subject to the structural constraints of compositionality. More in Sect. 20.

[4] Also, the names could in principle be anything. There is no limit on how many names you use. This allows to replicate indices of predicate logic.

predicate. To achieve that they use morphological properties (cases, grammatical relations) and link them with variable names. It is this point where the present investigation took its beginning. In order to have a truly compositional semantics, the lexical representations have no place for an interface. Syntax should be autonomous, and so should be semantics, by compositionality. Semantics should only contain meaning. Or rather: it should only contain truth-conditions.[5] What followed were attempts to remove any excess information that standard predicate logic provides over truth conditions. The concerns that our notational systems are less than adequate are not new. Quine, speaking about linear notation of sets once used the phrase ''excess of notation over subject matter''. The string $\{\varnothing, \{\varnothing\}\}$ involves two token inscriptions for the empty set where there is only one empty set to begin with.[6] Kit Fine (p.c.) in connection with variables said that the standard semantics is not ''alphabetically innocent''.[7] For the purposes of meaning it should not matter what you called the variable; the only thing that matters is that you have distinct names for distinct variables. This is implemented below in a particular way. What it effectively says is that semantics ignores particular names for variables as ''excess information''.

All these measures take care of (a) and (b). A last loophole to be closed is that of multiplicity. DRT makes use of condition such as $x_6 \doteq x_{75}$. In Kamp and Reyle (1993) and similar proposals they are used in an essential way, namely to link meanings to each other. Names of variables are global, they are used by the assignment function. In this interpretation, $x_6 \doteq x_{75}$ is not vacuous; it says that your assignment gives the same value to x_6 and x_{75}. However, what I fail to see is how they enter the picture in the first place. What is it in the meaning of a verb, say `see`, that allows to actually link a variable, say x_6, to the subject? In DRT, all you can do is use a special variable for the subject, say x_1, which is linked to x_6 in interpretation (either by renaming or by adding the equation $x_1 \doteq x_6$). Once the sentence is complete, it expresses a fair amount of conditions on specific variables. It is these conditions whose reality I fail to see. What is the precise contribution of $x_6 \doteq x_{75}$ in the DRS of a given sentence? If you can rename variables (as I propose) it would mean the same as $x_{98} \doteq x_{101}$ if you are allowed to map, say, x_6 to x_{98} and x_{75} to x_{101}. So far there is nothing to prevent you from inserting equations. Thus, just like $\varphi \wedge \varphi$, the iteration of a variable under a different name is simply excluded (or rendered meaningless).

Thus we arrive at a particular version of meaning where you cannot give particular names to variables, and you cannot give different names to the same object. Once we have come this far, we turn around and ask: how must language be like if it is compositional and uses these kinds of meanings? This question, I think, is a very natural one, so natural that it makes me wonder why virtually no one is concerned about it. One interpretation is that up until now the question could not be meaningfully asked. There was no satisfactory notion of semantics that would allow to

[5] To prevent confusion I should stress that I am not advocating a specific semantics. Many cognitive grammars contain visual representations (see in particular Langacker (1987)). But the contrast between a visual representation and one in terms of formulae (or graphs) is purely one of surface appearance. Even a cognitive linguist will admit that the representations have content that can be checked against reality. It is this content that I wish to tease out here. In that sense the 'concepts' that will be defined below despite looking like formulae from predicate logic are in substance much more like the structures found in Langacker (1987).

[6] I owe this quote to Hans-Martin Gärtner, who has made that point for syntax in Gärtner (2002).

[7] This phrase he used in a lecture at UCLA explaining his Fine (2003). The paper does not seem to contain that rather apt expression, though.

draw conclusions of that sort. I am inclined to think that there is an additional deeper reason: linguistic training mostly involves learning to think in terms of syntax—as a consequence few have reliable intuitions about meanings and they have also been told that it does not matter. (Well, metamathematics with its talk of inscriptions has had its hand in there, too.) Folklore says syntax has a mind of its own, and does not need semantics. I wish to add here: the same goes for semantics! I agree that semantics is not what Montague Grammar makes us believe it is—and I shall give arguments in that direction below. But I do not agree that sentence structures cannot be motivated from semantics. Even if we grant that syntax has its own way, the fact that at the end of the day we want to communicate certain meanings will put pressure on language to put up with the requirements of semantics. In other words: both syntax and semantics will have certain intrinsic properties, and they will conspire to produce the systems that we call natural languages.

Be this as it may, the present paper will show that many features of natural language are rooted in certain constraints that originate not in syntax, but in semantics. It will show that simply because semantics is weaker than we originally thought, it needs syntax to schedule the composition of constituent in such a way that semantics with its limited capacity is actually able to come up with the right result.

2 Syntactic structure

Let us return to the idea of strings and constituent structure. An overwhelming majority of syntactic theories start with the following assumption: a *constituent* is a substring of the given string, and constituents are in general formed from two strings by concatenation. I call this *syntactic locality*. I stress outright that this is an assumption we make, and it is—I think—a reasonable one. An underlying rationale for positing this restriction might be the following. Constituents have meaning; on the assumption of compositionality, the meaning of the next higher constituent is formed from the meanings of its parts. Thus, the meanings of the parts are engaging in a process of some sort. From the standpoint of the hearer, s/he needs to figure out which of the parts of a sentence are constituents and which ones are not. The idea is now that the closer a string is to the other, the closer the connection. Closeness indicates relevance. The closest you can be is be adjacent. Thus, we expect a constituent to be adjacent to the one with which it forms the constituent next up in the tree.

The trouble starts at two points, actually. First, notice that for any given constituent, if it is a mere string, it only has two points where a constituent can be joined. A ditransitive verb thus poses a dilemma for its three arguments; at least one of them cannot be adjacent. (Imagine having three children: with only two hands you have trouble holding each of them by the hand.) Many grammatical theories welcome this on the grounds that the ditransitive does not seek its arguments at once; it will accept them only one by one. Each time it does, the next argument has two places to choose. They will deny that in

(2) `John calls Harry an idiot.`

the phrase `an idiot` enters into a constituent with the verb; rather, it enters into a constituent with `calls Harry`.[8] This argument rests on the idea that the notion of

[8] Or they will claim that at another level it forms a constituent `calls an idiot`. But that does not make a difference, the problem is the same at that level.

calling [someone] an idiot has no linguistic home. You may think about such a concept, but syntax will not allow you to pronounce it without first filling the argument slot of the direct object. I have met this kind of reasoning a lot; some would think that there is no sense to the thought in the first place. If you can't say it does not exist. I find that simplistic. There certainly is a notion of insulting. It means, well, to throw bad words at someone—for example, by calling him an idiot. You are certainly able to think of the notion of spitting out bad words without asking who is addressed. There is also a way to put this into words: `to swear`, or `to issue an insult`. As far as I can see this is not syntactically deficient, so it ought to exist. Any semanticist believing in Montague Grammar should actually agree: it just takes applying some combinator shuffling the argument places and the concept is there.

Now, as a matter of fact, word order in natural languages is a tricky matter. Things do not work out in the way sketched above. In a topicalised sentence, for example, the object is at the left edge of the sentence, so if adjacency is to hold, it is the subject that forms a constituent with the verb. Different scenarios have been developed. We may allow the verb to change its meaning so as to accommodate the subject before the object. The drawback of this solution is that it does not tell us why we do not do that when the object is right adjacent to the verb. We may however admit that latter option too. At that point we have given up on the motivating story above. The second scenario is to give up adjacency and make discontinuous constituents a first class citizen. It is this option that I favour. The third option is by far the most popular. It is to assume locality at some other level, and to assume that there are structural operations that may change the arrangement of constituents. This is the story of transformational grammar.[9]

3 Transformations

In particular, within transformational grammar, arguments have been presented to show that the syntactic structure of sentences is quite complex: they are generated in a two-layered process, the first of which establishes a basic structure, where the function-argument relation is established and the second rearranges the elements into their surface position. This theory has a lot to recommend itself. I mention only the fact that it explains the syntax of questions in English rather well. All we have to assume is that first we generate the question as if it were an assertion and then front the auxiliary and finally the question word.

(3) `you have seen the movie with whom?`

(4) `with whom have you seen the movie?`

At the beginning of the inception of transformations by Zellig Harris, they were claimed to leave meaning invariant. Later this position was abandoned. Moreover, transformations were included into the generative procedure for sentences. Chomsky argued that transformations existed because of a need to render a logically correct

[9] Recent developments within the Minimalist Program suggest a rapprochement between these views. The idea of sideward movement is effectively a way to reconcile the need for locality with the desire not to insert an element early into a structure. The connection between sideward movement and the tuple based syntax that I favour has been worked out in detail by Stabler (2006a, b).

structure into one which conforms to the surface laws of language. For example, in the above case there is a condition of English that requires at exactly one question word to appear at the beginning of the sentence. Moreover, there is another condition that requires the auxiliary to be in second place. Therefore, even though (3) is formed according to the typical grammar (subject–verb–object), the particular requirements of English require a repair which is offered by moving certain parts to the left. Eventually, one reaches (4), which conforms to the laws of English surface syntax.

At times it has also been argued that a particular syntactic analysis is needed because without this assumption the sentence would not be interpretable. In the present version of the transformational theory, the Minimalist Program, no assumption of that sort is made. Syntax projects deep structures and goes through a cycle of transformations, which eventually yield a surface string and a logical or semantic representation. This latter architecture if correct leads to a scenario where interpretation can be compositional only at LF. Compositionality however requires to build meanings in tandem with structure; for the purposes of MP we have to pretend the structures of LF are assembled in the obvious way. I will return to the viability of this approach at a later stage in Sect. 20.

Here I will discuss some foundational issues of the program itself. At its earliest stages, transformational grammar was heavily based on string rewriting, though it always assumed a structure to go with it. This has meant that a crosslinguistic rule of passivisation could not be formulated, because the surface realisation of passives were too diverse to be captured by string rules. This led to the birth of relational grammar (see Perlmutter, 1983, 1984). For a while, transformation grammar, then known under the name GB, eliminated order from syntax. Later, with antisymmetry hypothesis by Kayne (1994), order was paired with structure so that effectively it was put back into syntax. Now however it imposed a rather quirky structure on syntax, because linear order had to be recoverable from c-command. The price to be paid was the postulation of an armada of new constituents which eventually rendered the relation between deep structure and surface structure completely opaque. The theory culminates in the claim that all languages have the same deep word order, head medial, a claim for which no independent evidence exists. Not everyone agrees, though, that this is a correct assessment, see for example Haider (2000a, b) for the claim that SOV must be included in the list of fundamental orders. Of course, it may be said that the refutation of Kayne's thesis is a theory internal matter, so it does not make sense to ask for motivating evidence in the first place. I will not even bother to dismantle such claims.

Another consequence of Kayne's theory was this. The newly postulated constituents had to have labels to make their assumption reasonable for a linguist. If one looks at the labels one is struck by the fact that they are *exclusively semantic in nature*: negation, aspect, tense, distributivity and so on figure in the names of these categories. What is more, in an extensive study of the order of adverbials (Cinque, 1999), Cinque not only proposes that the order is more or less fixed by their semantic type, but also proposes that this is basically due to syntactic constraints. I do not dispute his claims; I only point out that what the theory does not explain is why the adverbials come in the order they do.[10] It is precisely the Prague school, in the

[10] Another point that needs to be raised is why it is that adverbials are not moved while DPs are. (Cinque (1999) uses this so that he can use surface order as a diagnostic instrument for deep order.) I do not know of a syntactic argument why this is so.

notion of communicative dynamism (see for example Firbas, 1992), which has long ago addressed the issue, claiming that the order is motivated by communicative/pragmatic constraints. If that is so, we ask: how come syntax mirrors the extrinsically motivated order if the labels for the constituents are purely abstract? How come a negative word such as `not` will actually ever originate in NEG^0 if syntax does not know what all these things mean?

The predicament is I guess a very real one. On the one hand syntax is claimed to be abstract, on the other hand we use semantic terms to identify syntactic labels, so that we have *celerity aspect phrases* (for the adverbial constituents denoting speed), *durative aspect phrases* (for the constituents denoting or qualifying an ongoing action) and so on. The crosslisting of semantic categories in syntax should raise doubts as to whether the explanations are syntactic in nature. What is at stake, then, is not the validity of the observations concerning the surface order but the arguments that relate them to syntactic structure.

4 Montague grammar

In semantics, an analogous confusion can be noted. Many semantic theories (not the least Montague's own) import syntactic notions in order to get the mechanics of meaning right. For example, in Montague Grammar a transitive verb such as `see` is translated into a typed expression that consumes first the object and then the subject. Thus, given that the only rule of combination is function application and that this in turn is paired with concatenation, the constituent structure that emerges is [S[V O]], order irrelevant. For a language like Gaelic, which has (in part) VSO order, the needed constituent cannot be formed. At the heart of this failure is the notion of *Currying* a function.[11] Even if a predicate actually needs two arguments at once, we shall turn it into a function that will take them one at a time. This is the same idea that we talked about earlier in connection with ditransitives: we claim that some arguments actually do not enter into a constituent with the head alone, but with the head plus some argument slots already filled. The problem with this solution is again a foundational one: if the predicate is first, and the Currying comes later, who decides in which way to Curry the function? This is the question that Dowty raised in Dowty (1979). However, what he did not talk about was the other question: how do we get from the predicate to its Curried equivalent? Suppose you call your meaning $\mathsf{see}'(x_6, x_{75})$. Then you abstract the object by abstracting over x_{75}; you abstract the subject by abstracting over x_6. Thus we get the following two Curried versions:

$$(5) \qquad \lambda x_{75}.\lambda x_6.\ \mathsf{see}'(x_6, x_{75}), \qquad \lambda x_6.\lambda x_{75}.\ \mathsf{see}'(x_6, x_{75})$$

Now assume, as I do, that there is no way to point at variable names directly. Then you have to give me a way to identify the subject and object independently of the way they have been called. (This is quite reasonable as the variables you use to store the concept may be different from mine. Yet, we should get the same function in the end.) If that can be solved, however, we may ask: what is the need of function abstraction in the first place? The need arose in the first place because one sought to follow Frege in assuming that syntactic juncture is accompanied by function application. But was he

[11] Another escape hatch is to relax constituency; I have proposed this in Kracht (2003). Effectively, this move will be made below. It does not solve the principled problem that we discuss here, though.

right? He said the predicate is unsaturated; you needed to fill in a subject to obtain a complete thought. I ask: why is see not a complete thought?[12] My answer is: because syntax makes us think that way. I may think: there is music in the hallway, because I hear that someone is singing ''Sweet Georgia Brown''. What is my thought? Should it be (6), or (7)? Or rather (8)?

(6) Someone is singing "Sweet Georgia Brown".

(7) There is singing of "Sweet Georgia Brown".

(8) I hear a song. It's "Sweet Georgia Brown".

Anyone wishing to persuade me that the thought of (7) without the accompanying thought of someone actually singing ''Sweet Georgia Brown'' is incomplete will have to claim that there are sentences which express incomplete thoughts. But they may actually be true or false in the same way (6) may be true or false. So who is to decide?

Actually, I doubt very much that many semanticists really care about Frege's worries. For them, the typed λ-calculus provides a convenient way to write down meanings. What it does for them is what I call variable administration. Even though I am convinced that it is not the best way to use it for, there are more substantial arguments against the adoption of λ-calculus. Suppose namely that meanings are Curried functions. Then the notion of an object, say, can be defined purely on the basis of the type. The Curried function projects a syntactic structure that makes the object of the verb its sister. (This is reminiscent of Chomsky's definition of the term subject and object.) This means that there is nothing left to explain. The fact that, for example, the more agent like argument ends up in subject position is a nice thing to have, but it is simply a regularity of the lexicon that may or may not be there. Given that there seem to be cognitive roots for this phenomenon we seem to miss a generalisation here. The meanings do not put pressure on the system to arrive at a consensus about the assignment of argument roles. As far as the λ-calculus is concerned, it's all the same. Also, notice that languages will not have both a relation and its converse in the lexicon (apart from spatial relations, that is). For example, there is no basic verb that means ''to be taught by'' in English and any language I know of. From the standpoint of λ-calculus this is an unexplained coincidence, for on other occasions we do seem to find words whose meanings could be derived from other word meanings (like causatives; actually ''teach'' may be analysed as ''to make learn'', just like the textbook causative ''to kill'', analysed as ''to make die'', and so on).

Another thing that has no home in Montague and Categorial grammar is agreement morphology. In a language like Latin, where cases signal argument status, cases are vital in assuring the correct linking in presence of free word order. Both Montague Grammar and Transformational Grammar pass this problem with silence: the derivation does not need this information. The problem is not that morphology cannot be added; the problem is that it appears to be unmotivated to begin with. This in turn is due to the fact that both theories assume that everything starts with

[12] Probably, if we were Japanese speakers we would not think that way. My hunch is that the theory too easily follows the lead of one's own language. Might that also hold for the debate between SVO and SOV?

syntactic structure. Once we have stripped off the additional layer of Currying, however, things start to look somewhat different. Then we must seriously ask the question that we have so far covered up: how is the linking of argument places in the head with syntactic arguments actually carried out? Again, if we do afford the luxury of variable names, this can be made a trivial matter of just adding some predefined identity statement. Once we remove that we begin to see why languages are the way they are: because they are blindfolded! They cannot see all these things theory makes us believe are there. If we assume predicate logic to start, then we need a theory of which abstraction is applied first. There is none that I know of.

5 Dutch

To make the presentation focused, I shall present a particular example. The order of words in Dutch in infinitives is markedly different from that of English.

(9) `Ik zei dat Jan Marie Hans het kind zag laten leren zwemmen.`
I said that Jan Marie Hans the child saw let teach swim

(10) `Ich sagte, dass Jan Marie Hans das Kind schwimmen lehren lassen sah.`
I said that Jan Marie Hans the child swim teach let saw

(11) `I said that Jan saw Mary let Hans teach the child to swim.`

As can be gleaned from the word to word gloss, the order of the verbs in Dutch is the same as that of English. In German, on the other hand, it is inverted. Rather than saw-let-teach-swim we have swim-teach-let-saw. The fact that we use embedded that-clauses only has to do with the fact that main clauses have different word order in both Dutch and German, a fact that we shall ignore by turning to the analysis of just the subordinate clauses.

It is not hard to see that the English sentences can be produced by the following grammar.

(12)
$$\begin{aligned}
\mathrm{S}[-\mathrm{i}] &\to \mathrm{NP}\ \mathrm{VP}[-\mathrm{i}]\\
\mathrm{S}[+\mathrm{i}] &\to \mathrm{NP}\ \mathrm{VP}[+\mathrm{i}]\\
\mathrm{VP}[-\mathrm{i}] &\to \mathrm{VR}[-\mathrm{i}]\mathrm{S}[+\mathrm{i}] \mid \mathrm{VB}[-\mathrm{i}]\\
\mathrm{VP}[+\mathrm{i}] &\to \mathrm{VR}[+\mathrm{i}]\mathrm{S}[+\mathrm{i}] \mid \mathrm{VB}[+\mathrm{i}]\\
\mathrm{NP} &\to \texttt{Jan} \mid \texttt{Marie} \mid \texttt{Hans} \mid \texttt{the child}\\
\mathrm{VR}[+\mathrm{i}] &\to \texttt{see} \mid \texttt{let} \mid \texttt{teach}\\
\mathrm{VB}[+\mathrm{i}] &\to \texttt{swim}\\
\mathrm{VR}[-\mathrm{i}] &\to \texttt{saw} \mid \texttt{let} \mid \texttt{taught}\\
\mathrm{VB}[-\mathrm{i}] &\to \texttt{swam}
\end{aligned}$$

Likewise, the German sentences can be generated by the following grammar:

$$
\begin{aligned}
\mathrm{S}[-\mathrm{i}] &\rightarrow \mathrm{NP\ VP}[-\mathrm{i}] \\
\mathrm{S}[+\mathrm{i}] &\rightarrow \mathrm{NP\ VP}[+\mathrm{i}] \\
\mathrm{VP}[-\mathrm{i}] &\rightarrow \mathrm{S}[+\mathrm{i}]\mathrm{VR}[-\mathrm{i}] \mid \mathrm{VB}[-\mathrm{i}] \\
\mathrm{VP}[+\mathrm{i}] &\rightarrow \mathrm{S}[+\mathrm{i}]\mathrm{VR}[+\mathrm{i}] \mid \mathrm{VB}[+\mathrm{i}] \\
\mathrm{NP} &\rightarrow \texttt{Jan} \mid \texttt{Marie} \mid \texttt{Hans} \mid \texttt{das Kind} \\
\mathrm{VR}[+\mathrm{i}] &\rightarrow \texttt{sehen} \mid \texttt{lassen} \mid \texttt{lehren} \\
\mathrm{VB}[+\mathrm{i}] &\rightarrow \texttt{schwimmen} \\
\mathrm{VR}[-\mathrm{i}] &\rightarrow \texttt{sah} \mid \texttt{ließ} \mid \texttt{lehrte} \\
\mathrm{VB}[-\mathrm{i}] &\rightarrow \texttt{schwamm}
\end{aligned}
\tag{13}
$$

Apart from the actual words that get inserted, the difference is in the third and fourth row: the verb *follows* its sentential complement rather than preceding it.

Both grammars are context free: they generate the sentences by starting with the string S[−i] through successive replacement of one token by the right hand side of a rule (if there is a | there is a choice between the item to the left of the slash and the one to the right), until no more replacements are possible. The grammar assigns structure to the strings in an obvious way. If the process replaces the symbol X by the string $\vec{x}$ then the occurrence of $\vec{x}$ is taken to be a constituent of label X. The labels are otherwise arbitrary.

We now ask: is there a grammar that generates the sentences of Dutch? At first glance the answer seems to be yes. Just choose

$$
\begin{aligned}
\mathrm{S}[-\mathrm{i}] &\rightarrow \mathrm{NP\ VB}[-\mathrm{i}] \mid \mathrm{NP\ VP}[-\mathrm{i}] \\
\mathrm{VP}[-\mathrm{i}] &\rightarrow \mathrm{S1}[-\mathrm{i}]\mathrm{VB}[+\mathrm{i}] \\
\mathrm{S1}[-\mathrm{i}] &\rightarrow \mathrm{NP\ VP1}[-\mathrm{i}] \\
\mathrm{VP1}[-\mathrm{i}] &\rightarrow \mathrm{S1}[-\mathrm{i}]\mathrm{VR}[+\mathrm{i}] \mid \mathrm{NP\ VR}[-\mathrm{i}] \\
\mathrm{NP} &\rightarrow \texttt{Jan} \mid \texttt{Marie} \mid \texttt{Hans} \mid \texttt{het kind} \\
\mathrm{VR}[+\mathrm{i}] &\rightarrow \texttt{zien} \mid \texttt{laten} \mid \texttt{leren} \\
\mathrm{VB}[+\mathrm{i}] &\rightarrow \texttt{zwemmen} \\
\mathrm{VR}[-\mathrm{i}] &\rightarrow \texttt{zag} \mid \texttt{liet} \mid \texttt{lerte} \\
\mathrm{VB}[-\mathrm{i}] &\rightarrow \texttt{zwam}
\end{aligned}
\tag{14}
$$

This grammar generates the Dutch as if it was German, only that the finite verb is placed at the beginning. There is wide consensus that this grammar assigns incorrect structures to the Dutch sentences. To wit (Huybregts, 1984) has argued that the structures are incorrect, if we assume that selectional restrictions (subject is animate or not) are actually syntactic in nature. For then it can be shown that the association between subjects and their verbs is grammatically relevant. If this can be made manifest in terms of syntactic marking, then that constitutes a proof. This has been the source of the proof by Shieber (1985) that Swiss German is not context free. However, Huybregts himself was not at ease with this type of argument because he felt that the restriction was rather of semantic nature. Thus, it remained open whether or not Dutch must be seen as syntactically context free. What I aim to show here is that such worries are orthogonal to the linguist's question of what the appropriate

structure for Dutch should be. Here, namely, we feel compelled to recognise the cross serial dependencies as syntactically real. To make such an argument, however, requires a number of assumptions, which I shall clarify below in Sect. 18.

Standardly, linguists are likely to use their instinctive syntactic analysis as a basis of judgement. The problem however is that this line of thinking does not take us very far. First, there are languages spoken in Australia which are said to have very free word order. Although not totally free (typically clauses present a constituent boundary), the order may be virtually free within a single clause of several constituents. This has prompted the claim that these languages are actually of a totally different syntactic type (*nonconfigurational*). What is of interest here is the fact that these languages allow to disrupt the continuity of constituents to a degree deemed impossible by a speaker of a Western European language. Yet these languages are perfectly intelligible. In Ebert and Kracht (2002), together with Christian Ebert I have given a semantic account of them in terms of this nonconfigurational structure, which is computationally even simpler than Montague Semantics. Therefore, the mere instinct that free or alternative word order is impossible is not enough to exclude a given syntactic analysis. What we need is a proof. We shall provide such a proof below. Certainly, it is a proof only inasmuch as you believe my story and inasmuch as the purported facts of Dutch hold, but they are uncontroversial. One can easily see from the proof that it uses practically nothing else but the fact that Dutch has cross-serial dependencies.

If that is so, this constitutes the first proof of its kind. Elsewhere, in Kracht (2003, pp. 444–445), I have shown that Chinese A-not-A questions are not strongly context free, given that the structures are as in Radzinski (1990). However, it is not clear that the syntactic facts concerning Chinese are as portrayed in that paper. Moreover, syntactic copying is marginal, while crossing dependencies are more widespread. Therefore, with Dutch, matters are quite different. If the pattern is repeatable without bound (as we shall assume here), the argument goes through. Thus, we shall establish the impossibility of certain constituent structures over a given sentence based purely on the fact that a compositional interpretation cannot otherwise be given. To provide this argument, we must make precise what we mean by compositionality, and what we mean by meaning.

6 Compositionality

We shall assume the background framework of Kracht (2003, Sect. 3.1). A *sign* is a triple $\sigma = \langle e, c, m\rangle$, where e is the *exponent*, c the *category* and m the *meaning* of σ. We write $\varepsilon(\sigma)$ for e, $\kappa(\sigma)$ for c and $\mu(\sigma)$ for m. *Languages* are sets of signs. A *signature* is a pair $\langle F, \Omega\rangle$, where F is a finite set of so-called *modes* and Ω: $F \to \mathbb{N}$ a function assigning each symbol from F a so-called *arity*. A *grammar* for a language L is a map G, which assigns to each $f \in F$ a partial, computable function $G(f)$: $L^{\Omega(f)} \to L$ such that L is the least set closed under the $G(f)$; or, equivalently, if L is generated with the help of the $G(f)$. This means the following. Each f such that $\Omega(f) = 0$ is assigned an element $G(f)$ of L; the set $\{G(f)\text{: } \Omega(f) = 0\}$ is the *lexicon*. All other signs are created from the lexicon using the functions $G(f)$ with arity > 0. Apart from the lexicon, there are typically very few such functions. For example, Montague Grammar uses besides the lexicon only the modes $\mathtt{A}_<$ and $\mathtt{A}_>, \Omega(\mathtt{A}_<) = \Omega(\mathtt{A}_>) = 2$. The exponents are strings over the alphabet, including the

blank ('□'); the only available operation is concatenation. The categories are terms over the set of basic categories formed with the infix symbols / and \. Meanings are typed λ-terms over a signature of predicate logic. The modes are interpreted as follows.

(15) $\mathtt{A}_{>}(\langle \vec{x}, \gamma/\delta, M\rangle, \langle \vec{y}, \delta, N\rangle) := \langle \vec{x}^\frown\Box^\frown\vec{y}, \gamma, M(N)\rangle$

(16) $\mathtt{A}_{<}(\langle \vec{x}, \delta, M\rangle, \langle \vec{y}, \gamma\backslash\delta, N\rangle) := \langle \vec{x}^\frown\Box^\frown\vec{y}, \gamma, N(M)\rangle$

The functions are partial; $\mathtt{A}_{>}(\sigma, \sigma')$ is defined only when the category of σ' is of the form δ and the category of σ is of the form γ/δ for some γ. Likewise for $\mathtt{A}_{<}$. Montague Grammar also is compositional.[13]

Definition 1 A grammar for L is *compositional* if for every mode $f \in F$ there are partial functions f^ε, f^κ and f^μ on the exponents, categories and meanings, respectively, such that

❶ $G(f)(\sigma_1, \ldots, \sigma_{\Omega(f)})$ is defined iff
(a) $f^\varepsilon(\varepsilon(\sigma_1), \ldots, \varepsilon(\sigma_{\Omega(f)}))$ is defined and
(b) $f^\kappa(\kappa(\sigma_1), \ldots, \kappa(\sigma_{\Omega(f)}))$ is defined and
(c) $f^\mu(\mu(\sigma_1), \ldots, \mu(\sigma_{\Omega(f)}))$ is defined.

❷ If $G(f)(\sigma_1, \ldots, \sigma_{\Omega(f)})$ is defined then

$$G(f)(\sigma_1, \ldots, \sigma_{\Omega(f)}) = \begin{array}{l} \langle f^\varepsilon(\varepsilon(\sigma_1), \ldots, \varepsilon(\sigma_{\Omega(f)}), \\ f^\kappa(\kappa(\sigma_1), \ldots, \kappa(\sigma_{\Omega(f)}), \\ f^\mu(\mu(\sigma_1), \ldots, \mu(\sigma_{\Omega(f)}))\rangle \end{array}$$

This notion of compositionality requires each of the three components to be autonomous of each other; the only channel of communication is the choice of the mode. So, if a complex sign is formed using a mode, the meaning of that sign depends only on the meaning of the parts and the mode applied. And likewise for the category and the exponent. We shall introduce restrictions on possible functions as we go along.

7 How trivial is compositionality?

There exist proofs that every language is compositional (see Zadrozny, 1994) or at least every recursively enumerable language (Janssen, 1997). However, what Janssen and Zadrozny call *compositionality* does not do justice to our intuitive understanding. These proofs consist in both cases in postulating *additional signs*.[14] But we

[13] One should not confuse the assertion that Montague Grammar is compositional with the claim that languages are compositional. What I claim is that although it is compositional (except for the rules for pronouns) but that it uses the wrong type of semantics. Thus the question of compositionality of languages is an entirely different matter.

[14] Zadrozny massages the semantics into some suitable form, while Janssen assumes that only sentences have meaning and that everything else can be fiddled to fit the needs. Both come down to the same.

have dismissed these options above. Still, in Kracht (2003, Theorem 3.14ff) I gave arguments that English (and presumably every other natural language) can be generated by a 'compositional' grammar. The proof is based on the existence of infinitely many signs of the form

$$\begin{array}{ll} & \langle \texttt{one}, v, 1\rangle \\ & \langle \texttt{one plus one}, v, 2\rangle \\ (17) & \langle \texttt{one plus one plus one}, v, 3\rangle \\ & \langle \texttt{one plus one plus one plus one}, v, 4\rangle \\ & \ldots \end{array}$$

where v is an arbitrary category, say that of numerals. Notice that there is a one-to-one correspondence between form and meaning; the exponent is predictable from the meaning and the meaning from the exponent. The function $\mathrm{num}(-)$ is bijective:

$$\mathrm{num}(\vec{x}) := \text{number of occurrences of } \texttt{one} \text{ in } \vec{x} \tag{18}$$

Moreover, the set of expressions of the form above is recursive in the set of all exponents, which is to say that given a string $\vec{x}$ we can tell whether it is of the form of any of the exponents of (17). The only remaining assumption is that the expressions of English of any given category are recursively enumerable and that there are finitely many categories. Thus, assume that there is a finite set Γ such that

$$L = \bigcup_{\gamma\in\Gamma} L_\gamma \tag{19}$$

where

$$L_\gamma := \{\sigma \in L : \kappa(\sigma) = \gamma\} \tag{20}$$

Moreover, assume that for each $\gamma \in \Gamma$ there is a recursive function $\rho^\gamma\colon \mathbb{N}\twoheadrightarrow L_\gamma$. The grammar is this. We write a grammar to generate the signs from (17). We need only two lexical elements (one for `one` and one for `plus`) in addition to $\mathtt{A}_>$. In addition to this grammar we assume for every $\gamma \in \Gamma$ a unary mode g^γ, which is defined as follows.

$$g^\gamma(\langle \vec{x}, v, n\rangle) := \langle \varepsilon(\rho^\gamma(\mathrm{num}(\vec{x}))), \gamma, \mu(\rho^\gamma(n))\rangle \tag{21}$$

So, given a sign from (17), the exponent is obtained by calculating the number n that the exponent represents, then calculating the nth member of the enumeration of the signs and taking the exponent of that sign.

We shall rule out such an example as follows (see Kracht, 2003, Sect. 5.7). The functions on the exponents are required to not destroy any material. In general, exponents are tuples of strings.[15] If exponents are strings, the requirement is furthermore that nothing but polynomials based on string concatenation are available. This successfully rules out many artificial examples, but it cannot cope with the problem of Dutch. The problem with Dutch is not that we failed to provide a reasonable grammar, but that we need to show that we can come up with functions on the meanings that make the grammar compositional. There is so far no reason why we cannot use an analogous trick.

[15] We shall not deal here with phonological representations, nor are we concerned with the abstractness of the orthographic system. The idea that we are dealing with tuples of strings is a suitable simplification and we ignore the low level complications that phonology and morphology pose. For the purpose of this paper I am basing my arguments of something like deep phonological representations, in the way of Mel'čuk (2000).

This convoluted definition (21) makes sure that the algorithm proceeds strictly inside the language. It uses the fact that inside the language there is a definable set of signs where the number can be calculated from the exponent alone. In general, if there is a computable function ξ from meanings to exponents (in every given category) with computable inverse, then the autonomy of the meanings is trivially obtained. Let me illustrate this with $\mathtt{A}_{>}$. Given the meanings of the two arguments, m and m', we first look up the exponents $e := \xi(m)$ and $e' := \xi(m')$ that denote them, using our computable function ξ from meanings to exponents. We combine e and e' (here by concatenation), and then look up the meaning that this complex expression has, using the computable ξ^{-1}.

$$\mathtt{A}_{>}(\langle e, \gamma/\delta, m\rangle, \langle e', \delta, m'\rangle) := \langle e^\frown\square^\frown e', \gamma, \xi^{-1}(\xi(m)^\frown\square^\frown\xi(m'))\rangle \tag{22}$$

Thus, we effectively 'spy' on syntax to determine what to do with our meanings. This has a lot to do with our problem of Dutch: unless it can be argued that the meanings of some of the expressions are actually identical there is no hope to argue that there is no compositional context free grammar for Dutch. We can take the one we have found above in (14) and calculate with the method shown here. In this way, semantics becomes a slave of syntax, deriving the meanings from the expressions as a whole, not necessarily depending on the meanings of the parts previously established in any direct way. One may find this totally absurd. Indeed, many linguists would argue that the kinds of functions needed to make this work are unnatural and not available in semantics. But I should stress that so far no one has successfully delineated what kinds of functions are admissible or available in semantics and which ones are not. Until that is done, the dismissal of this proposal as unnatural is without theoretical basis. We need to look harder.

8 Meanings are truth conditions

The idea that I will pursue here is all problems arise from an improper understanding of what meanings actually are. The crux is that as long as anything can pass for 'meaning' no theoretical claim will ever be proved. Just let meanings contain a record of the syntactic structure and compositionality comes for free. Therefore, I wish to exclude any semantic representation that keeps a record of the syntactic structure (unless I can be convinced that it must be there). The challenge therefore is to come up with a proper notion of meaning.

My stance at the matter is this. I assume that meanings are no more and no less than *truth conditions*. This said, the judicious application of this assumption requires a lot of thought. Namely, because truth conditions are normally communicated using languages that merely describe them (such as predicate logic), we have to remind ourselves of the fact that different descriptions might actually be descriptions of the same thing, that is, of the same truth condition. We have to be aware that there may be variation in expression that is actually irrelevant.

Thus, I shall assume that meanings are *expressed* by formulae of many sorted first order predicate logic but I reject the view that this is what they *are*. Before I can say what I think meanings are (or better: how they function), let us briefly fix our lingua franca. The language has variables x_i, $i \in \mathbb{N}$, and certain function and relation symbols. We shall later also introduce variables of different type, but for now there

shall only be one, the type of objects. The *model structures* are pairs $\mathscr{M} = \langle M, \mathfrak{I}\rangle$, where M is a set, the *universe*, and $\mathfrak{I}$ a function assigning to an n-ary function symbol f a function $\mathfrak{I}(f)\colon M^n \to M$, and to an n-ary relation symbol R a relation $\mathfrak{I}(R) \subseteq M^n$. (Actually, the language is many sorted and so the definition of model structures and so on would have to be more complex. As the sorts complicate the notation I shall omit them whenever possible.) A *valuation* is a function from the variables into M.[16] By induction on the structure of the formula one defines whether a valuation makes a formula true in a structure. In the standard semantics this means that we may view meanings simply as functions from valuations into a model into a truth value. Fixing a particular first-order structure $\mathfrak{M}$, the meaning of φ is simply a set of valuations.

$$[\varphi]_{\mathscr{M}} := \{\beta : \langle \mathfrak{M}, \beta\rangle \models \varphi\} \tag{23}$$

For example, let our model be the numbers modulo 3. Then

$$[(\exists x_2)(x_2^2 = x_1)]_{\mathscr{M}} = \{\beta : \begin{array}{l}\beta(x_1) = 0 \text{ and } \beta(x_2) = 0\\ \text{or } \beta(x_1) = 1 \text{ and } \beta(x_2) = \pm 1\}\end{array} \tag{24}$$

As is well known, the meaning of any given formula of the language of predicate logic can be computed from the meaning of its parts. Thus if we can translate expressions into formulae of this language, it can be cascaded with this translation to yield an effective form-to-meaning translation. This is the basis of Montague Grammar.

Meanings are thus construed as sets of assignments. However, I shall change this slightly and say that an assignment is an infinite sequence $\vec{a} = \langle a_i\colon\ i \in \mathbb{N}\rangle$. The *sequence* $\vec{a}$ represents the assignment $\beta_{\vec{a}}\colon x_i \mapsto a_i$. The meaning of a formula is thus a set of infinite series, defined bottom up in the following way.

$$\begin{aligned} [R(x_{i_1}, \ldots, x_{i_n})]_{\mathscr{M}} &:= \{\vec{a}\colon \langle a_{i_1}, \ldots, a_{i_n}\rangle \in \mathfrak{I}(R)\}\\ [\neg\varphi]_{\mathscr{M}} &:= M^\omega - [\varphi]_{\mathscr{M}}\\ [\varphi \wedge \chi]_{\mathscr{M}} &:= [\varphi]_{\mathscr{M}} \cap [\chi]_{\mathscr{M}}\\ [\exists x_i.\varphi]_{\mathscr{M}} &:= C_i.[\varphi]_{\mathscr{M}} \end{aligned} \tag{25}$$

Let $[b\colon i]\vec{a}$ denote the sequence obtained by replacing the member a_i by b. The operation C_i of *cylindrification* is defined by

$$C_i.A := \{\vec{a}\colon \text{there is } b \text{ such that } [b : i]\vec{a} \in A\} \tag{26}$$

If L_n is the language obtained by using only the variables x_1 through x_n, then meanings will just be n-tuples of elements. The clauses above remain the same (modulo replacing M^ω by M^n). The presentation of meanings as certain sets has been given by Tarski and has led to the introduction of *cylindric algebras*. They are noteworthy in the present context since they can be said to provide actual meanings for formulae (and thus are a semantics in the proper sense of the word) rather than algorithms for assessing their truth. They will be found inadequate, though. The notation will nevertheless be useful. We add some more notation. There is a family of diagonals:

[16] This is a suitable moment to clarify a few things. The present approach uses a *single* model as the semantics. This model may be abstract (meanings modulo equivalence), so this actually not much of a commitment. There are no meaning postulates; identity in meaning is identity in denotation. (See Zimmermann (1999) on this question.) Our approach seems to be extensional; however, using many sorted predicate logic allows to add as many parameters as one wishes, for example worlds and times.

(27) $d_{i,j} := \{\vec{a}\colon a_i = a_j\}$

Notice that $d_{i,j} = [x_i = x_j]_{\mathscr{M}}$. Also, we write

(28) $\Pi_i.A := \{\langle a_1, \ldots, a_{i-1}, a_{i+1}, a_{i+2}, \ldots\rangle : \vec{a} \in A\}$

$\Pi_i.A$ is the projection of A obtained by removing the ith column. If $A \subseteq M^n$ then $\Pi_i.A \subseteq M^{n-1}$ provided that $i \leq n$.

There is a branch in semantics called *variable free semantics* which articulates meanings (and operations on meanings) in the form of algebraic operations. As I just said, algebraic semantics is the proper choice if we allow free variables. For, as has been noted quite early on (see Ferreirós, 2001) the standard semantics in terms of truth under a valuation is not compositional. This is because the value of a quantified expression $(Qx)\chi(x)$, for example $(\exists x)\chi(x)$, cannot be computed unless the values of $\chi(x)$ are known for different choices of x rather than just one. So, the meanings must be in some sense sets of assignments. These sets form an algebra, namely a cylindric algebra. Algebraic semantics and variable free semantics are in fact quite similar.[17] This is due to the insight is that substitution is actually definable in terms of cylindrification (see Dresner, 2001 for a discussion), so that it is possible to interpret a relation symbol by a certain set of assignments (corresponding to, say, the set $[R(x_1, \ldots, x_n)]_{\mathscr{M}}$) and then perform operations in terms of substitution and the operations above.

9 Alphabetical innocence

Unfortunately, the meanings just presented suffer from a defect. They depend on the names of the free variables chosen. This means that changing the names of the variables affects the meaning, while it is clear that the choice of the particular variables to fill in the empty slots in $\mathsf{see}'(\_,\_)$ is immaterial. This problem needs to be addressed. Fine (2003) has noted that the ordinary semantics for predicate logic is not compositional for precisely the reason just noted. We are interested in a similar problem: it is the fact that predicate logic for this reason is not suitable to represent meanings. Thus, while '$(\exists x_2)x_2^2 = x_4$' can be thought of as having different meaning than '$(\exists x_2)x_2^2 = x_1$' as far as predicate logic goes, we deny that they represent different *meanings* of natural language expressions. The distinction is between the name of the variable and the role it plays in the representation. The point that the actual name is irrelevant as long as different variables are given different names has been made over and over in the work of Vermeulen (1995) and Visser (1996). They have developed an alternative theory of so-called *referents*, a theory that focuses on the process of baptism of the individual variables. It is not necessary to explain this particular proposal, all we need to note is that their views conspire to the same requirement on meanings that Kite Fine called *alphabetical innocence*.

> Alphabetic Innocence. If φ' is the result of replacing the free variables of φ by free variables so that different variables are replaced by different variables, then φ' expresses the same meaning as φ.

We shall henceforth employ the following notation. For a formula of predicate logic φ let $\langle\!\langle\varphi\rangle\!\rangle_{\mathscr{M}}$ represent the meaning expressed by φ in the model structure $\mathscr{M}$.

[17] See the collection (Böttner & Thümmel, 2002) on variable free semantics.

(We shall define this notion below in exact detail.) Then alphabetical innocence implies among other the following.

(29) If s: Var $\to$ Var is injective then $«\varphi^s»_{\mathscr{M}} = «\varphi»_{\mathscr{M}}$

The consequences of this requirement are far reaching. The meaning cannot be computed on the basis of the set of variables that occur in it. Now, even if one cannot tag variables with numbers, still one should be able to distinguish variables that would ordinarily receive different numbers. Fine (2003) solves the problem by requiring to keep score of (physical) positions. This, however, puts the numbers back on the table if only as labels of positions. Vermeulen (1995) has proposed to assign variables names of any sort (they could be numbers) but requires that the algorithms be independent of the actual number chosen. We sympathise with the second solution because it presupposes no linear order.[18] Let us see what the consequences are. The first consequence is that every formula may after some substitution be brought into the form $\varphi(x_1, \dots, x_n)$, where the variables x_1 to x_n are exactly the free variables of the formula. However, notice that we are factually unable to say which of the original variables is now called x_1, which one is x_2, and so on. Thus the present theory differs also from variable free semantics in an important way. In variable free semantics it is effectively always possible to trace the name of a variable in a formula. For in variable free semantics substitutions must be made explicit. The effect of substituting x_7 by x_4, for example, corresponds to application of an operator, say, $O_{4:7}$ on the meaning of the formula. Further, in variable free semantics the converse of a relation is distinguishable from the relation itself; given alphabetical innocence it is not: $R(x_1, x_2)$ is an alphabetical variant of $R(x_2, x_1)$! Let us make this formally more explicit. Let π be a permutation of the set $\{1, 2, \dots, n\}$. (A *permutation* is just a one-to-one and onto map.) For an n-tuple $\vec{a}$ put

(30) $\pi(\vec{a}) := \langle a_{\pi(1)}, a_{\pi(2)}, \dots, a_{\pi(n)} \rangle$

Furthermore, for a set P set

(31) $\pi[P] := \{\pi(\vec{a})\colon \vec{a} \in P\}$

Then from alphabetical innocence it follows that $\pi[P]$ must be regarded the same as P. This is a welcome consequence. Williamson (1985) and Fine (2000) have argued that a relation and its converse are actually the same. While both wish to reform the definition of a relation itself, I remain conservative and use relations as before, but insist that the actual meanings (called concepts below) must be free of positional bias, in the way both Williamson and Fine suggest for relations.

There is however more that we want to require. If a variable does not occur in a formula it should be possible to ignore it completely. Because meanings are represented by first order formulae, only finitely many variables occur freely. This allows us to finitise the meanings as follows. In a given structure $\mathscr{M} = \langle M, \mathfrak{I} \rangle$, meanings are represented as subsets of M^n for a suitable n. However, not all such sets actually denote different meanings. Notice for example that the formula φ is equivalent to the formula $\varphi \wedge x_n \doteq x_n$, regardless of whether x_n actually occurs in φ or not.

[18] The linear order in predicate logic (and in virtually all formal languages) is not appropriate for human languages. We rather think that meanings are abstract schemes (say, in the way outlined in cognitive grammar), but it is not necessary here to commit to one of the many views. Instead, we have seen to it to obtain maximal abstractness for the names while still using the linear notation.

(32) $\quad «\varphi»_{\mathscr{M}} \;=\; «\varphi \wedge (x_n \doteq x_n)»_{\mathscr{M}}$

Thus, the set $P \times M$, where P is the meaning of φ, represents the same meaning as P.

Finally, suppose that you decided to have two variables, x_i and x_j that receive the same value by any valuation that make the formula true. Then the difference in name should actually be immaterial, and you are safe to replace one by the other. Thus, if in P, for every $\vec{a} \in P$, $a_i = a_j$, then we may 'cut' the jth argument place.[19] Thus P is regarded the same meaning as $\Pi_i.P$.

Definition 2 Let $\mathscr{M} = \langle M, \mathfrak{I} \rangle$, be a first-order structure. A *relation* over $\mathscr{M}$ is a subset of M^n for some n. We say that two relations P and Q *express the same concept*, in symbols, $P \approx Q$, if Q can be obtained from P by any number of the following operations.

(S1) Adding a trivial column: $P \mapsto P \times M = \{\langle \vec{a}, m \rangle : \vec{a} \in P, m \in M\}$.
(S2) Removing a trivial column: $P \times M \mapsto P$.
(S3) Permuting the columns: $P \mapsto \pi[P]$, π a permutation of $\{1, 2, \ldots, n\}$,
(S4) Shrinking identical columns: $P \mapsto c_i[P]$, provided that there is a $j \neq i$ such that for every $\vec{a} \in P$ $a_i = a_j$.
(S5) Expansion: $P \mapsto \{\langle \vec{a}, a_i \rangle : \vec{a} \in P\}$.

Write $[\![P]\!]_{\mathscr{M}} := \{Q : Q \approx P\}$. A set of relations over M is called a *concept* if it has the form $[\![P]\!]_{\mathscr{M}}$ for some relation P.

Thus if $P \approx Q$ you do not know any more by what name you called the variables, you do not know how often you called something by a different name, and you do not know how many variables you actually used. These things (although appearing in the notation) are not considered part of the meaning. In linear notation we are forced to linearise the members of a concept by writing for example $\langle a, c, d, z \rangle$. The places for the elements are called *slots*; a is for example in slot 1, c in slot 2. The tuples however are aligned simultaneously.

(33) $\quad H = \{\langle a, d \rangle, \langle b, c \rangle\}$

Then a and b must always appear in the same slot: thus we say they are in the same *column*. Similarly for d and c. Thus the following is not a member of the concept of H: $\{\langle d, a \rangle, \langle b, c \rangle\}$. In writing a relation, columns are mapped in a standard way to slots: columns number i appears in slot number i. However, this obtains only if we consider minimal members. As soon as we allow expansion there is a certain ambiguity in the association of slots with columns. This can be avoided completely by alway using relations of minimal length in the algorithms.

Let me stress that meaning identity is neither entirely a matter of form, neither one of equivalence. First, if two formulae are equivalent, then they have the same meaning. But there are nonequivalent formulae which do have the same meaning, for example $x_1 \doteq x_3$ and $x_1 \doteq x_2$. The best way to think about $\approx$ is equivalence (in the ordinary sense of predicate logic) plus alphabetic innocence.

Given a concept $\mathfrak{c}$, there is a P such that $\mathfrak{c} = [\![P]\!]_{\mathscr{M}}$. P is *minimal* if it has minimal length among the members of $\mathfrak{c}$. A concept $\mathfrak{c}$ has *arity* n iff its minimal members have arity n. Consider now the set $H = \{\langle a, b \rangle, \langle a, c \rangle, \langle b, c \rangle\}$. Its converse is $H^{\smile} = \{\langle b, a \rangle, \langle c, a \rangle, \langle c, b \rangle\}$, and $H \approx H^{\smile}$. Thus, the concept $[\![H]\!]_{\mathscr{M}}$ contains both H

[19] By permutation invariance, we may also cut the ith place but not both.

and $H^{\smile}$. This seems pretty odd: it means that the meaning of `left of` is the same as the meaning of `right of`, the meaning of `south of` the same as the meaning of `north of`. It will be my task to show that the situation is not as impossible as it may appear. For as it turns out, even though the concepts may be same, we need not *use* them in the same way.

As a point of notation: for a function $f\colon M \to N$ and a subset $X \subseteq M$ write

$$f[X] := \{f(x) : x \in M\} \tag{34}$$

The following is immediate from the definitions.

Lemma 3 *If P and Q are n-ary relations on which* (S2) *and* (S4) *cannot be applied, then $P \approx Q$ iff there is a permutation π such that $\pi[P] = Q$.*

We briefly mention the following interesting fact.

Theorem 4 *Let $L(\Omega)$ be the language of predicate logic over a fixed relational signature Ω; and let $L_n(\Omega)$ be the restriction of $L(\Omega)$ to the set of formulae where the only occurring variables are $x_1, \ldots, x_n$. There is a compositional grammar for $\mathrm{FL}_n := \{\langle \varphi, \text{«}\varphi\text{»}_{\mathscr{M}}\rangle\colon \varphi \in L_n\}$. There is however no compositional grammar for $L(\Omega)$ (given some nontriviality assumptions on Ω and $\mathscr{M}$).*

For a proof see Kracht (2006).

10 A simple grammar

We shall present here a grammar for some fragment of English. It is based on the nonlogical words `Alex`, `Bert`, `Cindy`, `Danielle`, `man`, `woman`, `sees`, `likes`, `walks`, `talks`, and some logical words such as `and`, `or`, `who`, and `some`. The phrase structure skeleton is as follows.

$$\begin{aligned}
\mathrm{S} &\to \mathrm{NP\ VP}\\
\mathrm{N} &\to \texttt{man}|\texttt{woman}\\
\mathrm{VP} &\to \texttt{walks}|\texttt{talks}|\mathrm{VT\ NP}|\mathrm{VP}\ \texttt{and}\ \mathrm{VP}|\mathrm{VP}\ \texttt{or}\ \mathrm{VP}\\
\mathrm{VT} &\to \texttt{sees}|\texttt{likes}\\
\mathrm{NP} &\to \texttt{some}\mathrm{N1}|\texttt{Alex}|\texttt{Bert}|\texttt{Cindy}|\texttt{Danielle}\\
\mathrm{N1} &\to \mathrm{N}
\end{aligned} \tag{35}$$

Like Montague grammar, the logical words are syncategorematic. This can be avoided, but that introduces notational complications which I'd like to avoid. This grammar generates among other the following sentences:

(36) `Alex sees Cindy.`

(37) `Some man talks.`

(38) `Some man sees some woman.`

We fix a language $\mathcal{L}$ of predicate logic that contains the constants $\underline{A}, \underline{B}, \underline{C}, \underline{D}$, man′, woman′, talks′, walks′, likes′, and sees′, and we fix a model structure $\mathscr{M} = \langle M, \mathfrak{I}\rangle$

that interprets them. This is the structure $M = \{A, B, C, D\}$, where $A = \mathfrak{I}(\underline{A})$, $B = \mathfrak{I}(\underline{B}), C = \mathfrak{I}(\underline{C})$ and $D = \mathfrak{I}(\underline{D})$. Furthermore,

$$
\begin{aligned}
\mathfrak{I}(\mathsf{man}') &= \{A, B\} \\
\mathfrak{I}(\mathsf{woman}') &= \{C, D\} \\
\mathfrak{I}(\mathsf{walk}') &= \{A, C\} \\
\mathfrak{I}(\mathsf{talk}') &= \{D, C\} \\
\mathfrak{I}(\mathsf{see}') &= \{\langle B, C\rangle, \langle B, D\rangle, \langle C, A\rangle, \langle C, C\rangle\} \\
\mathfrak{I}(\mathsf{like}') &= \{\langle A, A\rangle, \langle A, B\rangle, \langle A, C\rangle, \langle B, A\rangle, \langle B, D\rangle\}
\end{aligned} \tag{39}
$$

The lexical modes are the following:

$$
\begin{aligned}
G(\ell_0) &:= \langle \mathtt{man}, \mathrm{N}, \text{«}\mathsf{man}'(x_1)\text{»}_{\mathscr{M}}\rangle \\
G(\ell_1) &:= \langle \mathtt{woman}, \mathrm{N}, \text{«}\mathsf{woman}'(x_1)\text{»}_{\mathscr{M}}\rangle \\
G(\ell_2) &:= \langle \mathtt{talks}, \mathrm{VP}, \text{«}\mathsf{talks}'(x_1)\text{»}_{\mathscr{M}}\rangle \\
G(\ell_3) &:= \langle \mathtt{walks}, \mathrm{VP}, \text{«}\mathsf{walks}'(x_1)\text{»}_{\mathscr{M}}\rangle \\
G(\ell_4) &:= \langle \mathtt{sees}, \mathrm{V}, \text{«}\mathsf{sees}'(x_1, x_2)\text{»}_{\mathscr{M}}\rangle \\
G(\ell_5) &:= \langle \mathtt{likes}, \mathrm{V}, \text{«}\mathsf{likes}'(x_1, x_2)\text{»}_{\mathscr{M}}\rangle \\
G(\ell_6) &:= \langle \mathtt{Alex}, \mathrm{NP}, \text{«}x_1 = A\text{»}_{\mathscr{M}}\rangle \\
G(\ell_7) &:= \langle \mathtt{Bert}, \mathrm{NP}, \text{«}x_1 = B\text{»}_{\mathscr{M}}\rangle \\
G(\ell_8) &:= \langle \mathtt{Cindy}, \mathrm{NP}, \text{«}x_1 = C\text{»}_{\mathscr{M}}\rangle \\
G(\ell_9) &:= \langle \mathtt{Danielle}, \mathrm{NP}, \text{«}x_1 = D\text{»}_{\mathscr{M}}\rangle
\end{aligned} \tag{40}
$$

Here are two unary modes:

$$
\begin{aligned}
G(\nu_0)(\langle e, c, m\rangle) &:= \begin{cases} \langle \text{some}^\frown\Box^\frown\text{e}, \mathrm{NP}, m\rangle & \text{if } c = \mathrm{N1} \\ \text{undefined} & \text{else} \end{cases} \\
G(\nu_1)(\langle e, c, m\rangle) &:= \begin{cases} \langle e, \mathrm{N1}, m\rangle & \text{if } c = \mathrm{N} \\ \text{undefined} & \text{else} \end{cases}
\end{aligned} \tag{41}
$$

The rest of the grammar consists in binary modes. To define them we need a special device, called a *linking aspect*.

Definition 5 A *linking aspect* is a partial function on concepts which for each $\mathfrak{c}$ if defined yields a minimal member of $\mathfrak{c}$.

For the purpose of defining the grammar we define the following linking aspect.

$$
Y(\mathfrak{c}) := \begin{cases} P & \text{if } c \text{ and } P \text{ are unary and } c = \text{«}P\text{»}_{\mathscr{M}} \\ \mathfrak{I}(\mathsf{like}') & \text{if } c = \text{«}\mathsf{like}'(x_1, x_2)\text{»}_{\mathscr{M}} \\ \mathfrak{I}(\mathsf{see}') & \text{if } c = \text{«}\mathsf{see}'(x_1, x_2)\text{»}_{\mathscr{M}} \\ \text{undefined} & \text{else} \end{cases} \tag{42}
$$

We assume that in both cases x_1 is the subject and x_2 the object in the relation. This is not necessary, but makes reading the formulae easier.

$$
\begin{aligned}
&G(\theta_0)(\langle e, c, m\rangle, \langle e', c', m'\rangle)\\
&\quad := \begin{cases} \langle e^\frown\Box^\frown\texttt{and}^\frown\Box^\frown e', \mathrm{VP}, [\![Y(m) \cap Y(m')]\!]_{\mathscr{M}}\rangle \\ \quad \text{if } c = c' = \mathrm{VP} \\ \text{undefined else} \end{cases}\\
&G(\theta_1)(\langle e, c, m\rangle, \langle e', c', m'\rangle)\\
&\quad := \begin{cases} \langle e^\frown\Box^\frown\texttt{or}^\frown\Box^\frown e', \mathrm{VP}, [\![Y(m) \cup Y(m')]\!]_{\mathscr{M}}\rangle \\ \quad \text{if } c = c' = \mathrm{VP} \\ \text{undefined else} \end{cases}\\
&G(\theta_2)(\langle e, c, m\rangle, \langle e', c', m'\rangle)\\
&\quad := \begin{cases} \langle e'^\frown\Box^\frown e, \mathrm{S}, [\![\Pi_1.(Y(m) \cap Y(m'))]\!]_{\mathscr{M}}\rangle \\ \quad \text{if } c = \mathrm{VP} \text{ and } c' = \mathrm{NP} \\ \langle e^\frown\Box^\frown e', \mathrm{VP}, [\![\Pi_2.(Y(m) \cap (M \times Y(m')))]\!]_{\mathscr{M}}\rangle \\ \quad \text{if } c = \mathrm{VT} \text{ and } c' = \mathrm{NP} \\ \text{undefined else} \end{cases}
\end{aligned}
\tag{43}
$$

Let us stop here and see how we derive (37) and (38). First (37). As is customary in Montague Grammar and elsewhere, we define *analysis terms*. These are terms in the signature Ω, which the grammar evaluates into signs. For (37) the analysis term is $t := \theta_2(\ell_2, v_0(v_1(\ell_0)))$. The value is denoted by $\iota_G(t)$. We evaluate it step by step:

$$
\begin{aligned}
&\iota_G(\theta_2(\ell_2, v_0(v_1(\ell_0))))\\
=&G(\theta_2)(G(\ell_2), G(v_0)(G(v_1)(\langle \texttt{man}, \mathrm{N}, \text{«}\mathsf{man}'(x_1)\text{»}_{\mathscr{M}}\rangle)))\\
=&G(\theta_2)(G(\ell_2), G(v_0)(\langle \texttt{man}, \mathrm{N1}, \text{«}\mathsf{man}'(x_1)\text{»}_{\mathscr{M}}\rangle))\\
=&G(\theta_2)(G(\ell_2), \langle \texttt{some man}, \mathrm{NP}, \text{«}\mathsf{man}'(x_1)\text{»}_{\mathscr{M}}\rangle)\\
=&G(\theta_2)(\langle \texttt{talks}, \mathrm{V}, \text{«}\mathsf{talk}'(x_1)\text{»}_{\mathscr{M}}\rangle, \langle \texttt{some man}, \mathrm{NP}, \text{«}\mathsf{man}'(x_1)\text{»}_{\mathscr{M}}\rangle)\\
=&\langle \texttt{some man talks}, \mathrm{S}, \text{«}\exists x_1.\mathsf{talk}'(x_1) \wedge \mathsf{man}'(x_1)\text{»}_{\mathscr{M}}\rangle\\
=&\langle \texttt{some man talks}, \mathrm{S}, \varnothing\rangle
\end{aligned}
\tag{44}
$$

Only the last two steps involves some nontrivial manipulations. The actual definition of the semantics is as follows:

$$
[\![\Pi_1.(Y(\text{«}\mathsf{talk}'(x_1)\text{»}_{\mathscr{M}}) \cap Y(\text{«}\mathsf{man}'(x_1)\text{»}_{\mathscr{M}}))]\!]_{\mathscr{M}} \tag{45}
$$

Now, since man' and talk' denote unary relations, we find that

$$
\begin{aligned}
Y(\text{«}\mathsf{man}'(x_1)\text{»}_{\mathscr{M}}) &= [\mathsf{man}'(x_1)]_{\mathscr{M}} (= \{A, B\})\\
Y(\text{«}\mathsf{talk}'(x_1)\text{»}_{\mathscr{M}}) &= [\mathsf{talk}'(x_1)]_{\mathscr{M}} (= \{C, D\})
\end{aligned}
\tag{46}
$$

Next we use the identity $[P(x_1) \wedge Q(x_1)]_{\mathscr{M}} = [P(x_1)]_{\mathscr{M}} \cap [Q(x_1)]_{\mathscr{M}}$, (45) becomes

$$
\begin{aligned}
&[\![\Pi_1.[\mathsf{talk}'(x_1)]_{\mathscr{M}}) \cap Y(\text{«}\mathsf{man}'(x_1)\text{»}_{\mathscr{M}}))]\!]_{\mathscr{M}}\\
=&[\![\Pi_1.[\mathsf{talk}'(x_1) \wedge \mathsf{man}'(x_1)]_{\mathscr{M}}]\!]_{\mathscr{M}}
\end{aligned}
\tag{47}
$$

This can be rewritten using $\Pi_i.[\chi]_{\mathscr{M}} = [\exists x_i.\chi]_{\mathscr{M}}$ into

(48) $[\![[\exists x_1.\mathsf{man}'(x_1) \wedge \mathsf{talk}'(x_1)]_{\mathscr{M}}]\!]_{\mathscr{M}}$

Finally, observe that $\text{«}\chi\text{»}_{\mathscr{M}} := [\![[\chi]_{\mathscr{M}}]\!]_{\mathscr{M}}$. So we finally get the desired result. The concept is actually $\varnothing$, or simply ''false'', as can be computed from the model.

Now we step over to (38). Here, the linking aspect is actually doing real work. The analysis term is $u = \theta_2(\theta_2(\ell_2, v_0(v_1(\ell_1))), v_0(v_1(\ell_0)))$. We have:

(49)
$$\begin{aligned} \iota_G(v_0(v_1(\ell_0))) &= \langle \texttt{some man}, \mathrm{NP}, \text{«}\mathsf{man}'(x_1)\text{»}_{\mathscr{M}}\rangle \\ \iota_G(v_0(v_1(\ell_1))) &= \langle \texttt{some woman}, \mathrm{NP}, \text{«}\mathsf{woman}'(x_1)\text{»}_{\mathscr{M}}\rangle \end{aligned}$$

The next step is to compute $\iota_G(\ell_2, v_0(v_1(\ell_1)))$, which comes down to this:

(50)
$$\begin{aligned} & G(\theta_2)(\langle \texttt{sees}, \mathrm{V}, \text{«}\mathsf{see}'(x_1,x_2)\text{»}_{\mathscr{M}}\rangle, \\ & \qquad \langle \texttt{some woman}, \mathrm{NP}, \text{«}\mathsf{woman}'(x_1)\text{»}_{\mathscr{M}}\rangle \\ =& \langle \texttt{sees some woman}, \mathrm{VP}, \\ & \qquad [\![\Pi_2.Y(\text{«}\mathsf{see}'(x_1,x_2)\text{»}_{\mathscr{M}}) \cap Y(\text{«}\mathsf{woman}'(x_1)\text{»}_{\mathscr{M}})]]_{\mathscr{M}}\rangle \\ =& \langle \texttt{sees some woman}, \mathrm{VP}, \\ & \qquad [\![\Pi_2.Y(\text{«}\mathsf{see}'(x_1,x_2)\text{»}_{\mathscr{M}}) \cap (M \times Y(\text{«}\mathsf{woman}'(x_1)\text{»}_{\mathscr{M}}))]\!]_{\mathscr{M}}\rangle \\ =& \langle \texttt{sees some woman}, \mathrm{VP}, \\ & \qquad [\![\Pi_2.[\mathsf{see}'(x_1,x_2)]_{\mathscr{M}}) \cap (M \times [\mathsf{woman}'(x_1)]_{\mathscr{M}}))]\!]_{\mathscr{M}}\rangle \\ =& \langle \texttt{sees some woman}, \mathrm{VP}, \\ & \qquad [\![\Pi_2.([\mathsf{see}'(x_1,x_2)]_{\mathscr{M}} \cap [\mathsf{woman}'(x_2)]_{\mathscr{M}})]\!]_{\mathscr{M}}\rangle \\ =& \langle \texttt{sees some woman}, \mathrm{VP}, \\ & \qquad [\![\Pi_2.[\mathsf{see}'(x_1,x_2)) \wedge \mathsf{woman}'(x_2)]_{\mathscr{M}})]\!]_{\mathscr{M}}\rangle \\ =& \langle \texttt{sees some woman}, \mathrm{VP}, \\ & \qquad [\![[\exists x_2.\mathsf{see}'(x_1,x_2)) \wedge \mathsf{woman}'(x_2)]_{\mathscr{M}})]\!]_{\mathscr{M}}\rangle \\ =& \langle \texttt{sees some woman}, \mathrm{VP}, \\ & \qquad \text{«}\exists x_2.\mathsf{see}'(x_1,x_2)) \wedge \mathsf{woman}'(x_2))\text{»}_{\mathscr{M}}\rangle \\ =& \langle \texttt{sees some woman}, \mathrm{VP}, [\![\{B\}]\!]_{\mathscr{M}}\rangle \end{aligned}$$

The important steps are the transition from $Y(\text{«}\mathsf{see}'(x_1,x_2)\text{»}_{\mathscr{M}})$ to $[\mathsf{see}'(x_1,x_2)]_{\mathscr{M}}$, which holds by definition of Y; and the transition from $M \times [P(x_1)]_{\mathscr{M}}$ to $[P(x_2)]_{\mathscr{M}}$, which holds by definition of $[-]_{\mathscr{M}}$. From here on things proceed as in (37).

11 Adding relative clauses

We shall now add more modes in order to generate relative clauses. Here the semantics adds no complication; instead, it is the syntax that needs attention. Notice that relative clauses work by putting the relative pronoun at the beginning regardless of whether it is subject or object. We shall analyse them in the fashion of GPSG: there are special categories of sentences-missing-a-subject (S[s]) and sentences-missing-an-object (S[o]), as well as VP-missing-an-object (VP[o]). The rule set that is added to the original grammar is as follows:

$$\begin{aligned}
\text{N1} &\to \text{N RelC}\\
\text{RelC} &\to \texttt{who}\ \text{S[s]} \mid \texttt{who}\ \text{S[o]}\\
\text{S[o]} &\to \text{NP VP[o]}\\
\text{S[s]} &\to \text{VP}\\
\text{VP[o]} &\to \text{VT}
\end{aligned} \tag{51}$$

It will thus be possible to generate the following sentences:

(52) `Some woman who sees some man walks.`

(53) `Some man walks or likes some woman who talks.`

The grammar needs the following unary modes:

$$G(\nu_2)(\langle e, c, m\rangle) := \begin{cases} \langle \texttt{who}^\frown\Box^\frown e, \text{RelC}, m\rangle & \text{if c} \in\{\text{S[s], S[o]}\}\\ \text{undefined} & \text{else}\end{cases}$$

$$G(\nu_3)(\langle e, c, m\rangle) := \begin{cases} \langle e, \text{S[s]}, m\rangle & \text{if } c = \text{VP}\\ \text{undefined} & \text{else}\end{cases} \tag{54}$$

$$G(\nu_4)(\langle e, c, m\rangle) := \begin{cases} \langle e, \text{VP[o]}, m\rangle & \text{if } c = \text{VT}\\ \text{undefined} & \text{else}\end{cases}$$

Finally, two binary modes are needed as well:

$$\begin{aligned}
&G(\theta_3)(\langle e, c, m\rangle, \langle e', c', m'\rangle)\\
&\quad := \begin{cases} \langle e^\frown\Box^\frown e', \text{N1}, [\![Y(m) \cap Y(m')]\!]_{\mathscr{M}}\rangle\\ \quad \text{if } c = \text{N and } c' = \text{RelC}\\ \text{undefined else}\end{cases}\\
&G(\theta_4)(\langle e, c, m\rangle, \langle e', c', m'\rangle)\\
&\quad := \begin{cases} \langle e^\frown\Box^\frown e', \text{S[o]}, [\![\Pi_1.(Y(m') \cap Y(m))]\!]_{\mathscr{M}}\rangle\\ \quad \text{if } c = \text{NP and } c' = \text{VP[o]}\\ \text{undefined else}\end{cases}
\end{aligned} \tag{55}$$

Let us see how the new grammar deals with (52) and (53). For (52) what is new is the relative clause and the way it modifies the noun. The analysis term for the relative clause is $r = \nu_3(\nu_2(\theta_2(\ell_4, \nu_0(\nu_1(\ell_0)))))$. We calculate the outcome (this time using sets rather than formulae):

$$\begin{aligned}
&\iota_G(\theta_2(\ell_4, \nu_0(\nu_1(\ell_0))))\\
=&G(\nu_2)(G(\nu_3)(G(\theta_2)(G(\ell_4), \langle \texttt{some man}, \text{NP}, [\![\{A, B\}]\!]_{\mathscr{M}}\rangle)))\\
=&G(\nu_2)(G(\nu_3)(\langle \texttt{sees some man}, \text{VP}, [\![\{C\}]\!]_{\mathscr{M}}\rangle))\\
=&G(\nu_2)(\langle \texttt{sees some man}, \text{S[s]}, [\![\{C\}]\!]_{\mathscr{M}}\rangle)\\
=&\langle \texttt{who sees some man}, \text{RelC}, [\![\{C\}]\!]_{\mathscr{M}}\rangle
\end{aligned} \tag{56}$$

Finally, it is computed that the modified noun has the analysis term $\theta_3(\ell_1, r)$. It is computed that

$$
\begin{aligned}
&\iota_G(\theta_3(\ell_1, r))\\
=&G(\theta_3)(\langle \texttt{woman}, \mathrm{N}, [\![\{C, D\}]\!]_{\mathscr{M}}\rangle,\\
&\quad \langle \texttt{who sees some man}, \mathrm{RelC}, [\![\{C\}]\!]_{\mathscr{M}}\rangle)\\
=&\langle \texttt{woman who sees some some man}, \mathrm{N1}, [\![\{C\}]\!]_{\mathscr{M}}\rangle
\end{aligned}
\tag{57}
$$

Finally, let us turn to (53). Here we meet the logical word `or`. We compute the analysis terms for `likes some woman who talks`, which is $w = \theta_2(\ell_5, v_0(\theta_3(\ell_1, v_2(v_3(\ell_2)))))$. Its value is

$$\langle \texttt{likes some woman who talks}, \mathrm{NP}, [\![\{A, B\}]\!]_{\mathscr{M}}\rangle \tag{58}$$

Finally, we calculate the value of $\theta_1(\ell_5, w)$:

$$
\begin{aligned}
&G(\theta_1)(\langle \texttt{walks}, \mathrm{VP}, [\![\{A, C\}]\!]_{\mathscr{M}}\rangle,\\
&\quad \langle \texttt{likes some woman who talks}, \mathrm{NP}, [\![\{A, B\}]\!]_{\mathscr{M}}\rangle)\\
=&\langle \texttt{walks or likes some woman who talks},\\
&\quad \mathrm{VP}, [\![\{A, B, C\}]\!]_{\mathscr{M}}\rangle
\end{aligned}
\tag{59}
$$

The rest of the derivation is as above.

12 Linking

We see that the main difference between the present semantics and Montague Grammar is that instead of a calculus of application and abstraction that makes the identification of variables across constituents automatic we need an additional mechanism of linking. Linking depends on the linking aspect. This is apparent in the case of a transitive verb. Suppose we defined the following aspect Z:

$$
Z(\mathfrak{c}) := \begin{cases} P & \text{if } c \text{ is unary and } c = \text{«P»}_{\mathscr{M}} \\ \mathfrak{I}(\mathsf{like}')^{\smile} & \text{if } c = \text{«like}'(x_1, x_2)\text{»}_{\mathscr{M}} \\ \mathfrak{I}(\mathsf{see}') & \text{if } c = \text{«see}'(x_1, x_2)\text{»}_{\mathscr{M}} \\ \text{undefined} & \text{else} \end{cases}
\tag{60}
$$

If the grammar would use Z in place of Y, the subject of 'likes' would be the theme and the object the experiencer. In Montague Grammar the same result would have been achieved by Currying $\mathsf{like}'(x_1, x_2)$ in a different way. The difference lies not so much in what is expressed but rather in the way the syntactic knowledge is encoded into the language. In Montague Grammar the linking is part of the meaning; a different linking is effected by a different meaning. Here, linking is part of the grammar:[20] it is encoded in the way rules combine two concepts.

It is worth looking again at variable free semantics. In variable free semantics we work not with concepts but rather with relations, that is, objects of the form $[\chi]_{\mathscr{M}}$ for some formula χ. The positions reflect variable names in a direct way: position number 1 shows the value of variable x_1, position number 2 shows the value of variable x_2, and so on. Therefore, in variable free semantics the relations $[\mathsf{like}'(x_1, x_2)]_{\mathscr{M}}$ and $[\mathsf{like}'(x_2, x_1)]_{\mathscr{M}}$ are different. Therefore, they may be linked

[20] Notice the ambiguity in the word 'grammar'. Here it means grammar to generate signs, while elsewhere it means rather syntax. In stratificational terminology we have moved the linking information from the sememes to the semotactics.

independently from each other. There is alphabetical innocence only in a superficial way: the meanings do not contain variable names; on the other hand, the relations encode such names via the columns.

The linking aspect is part of the grammar. It must therefore be a finite object. In the case above, there is no problem. The linking aspect is needed only when we combine a transitive verb with its object. However, the way it is specified leaves something to desire. For we have said that $Y(\text{«}\mathsf{like}'(x_1,x_2)\text{»}_{\mathscr{M}})$ is the *relation* $[\![\mathsf{like}'(x_1,x_2)]\!]_{\mathscr{M}}$. This is possible in virtue of the fact that the interpretation of the primitive symbol like' is given as a relation. So, we seem to maintain that we associate relations at least to *some* elements of the language. But this is not necessary. I describe below two ways of defining the linking aspect without assuming such knowledge.

The first approach is via a *critical set*.

Definition 6 Suppose that P is minimal in its concept. A set A is *critical* for P, if for all minimal $Q \approx P$: if $A \subseteq Q$ then $Q = P$.

Proposition 7 *Every relation minimal in its concept has a finite critical set.*

Proof Fix P of length k, and let $[\![P]\!]_{\mathscr{M}} = \{P, Q_1, Q_2, \ldots, Q_n\}$. (Since there are at most $k!$ permutations of the sequence $1, 2, \ldots\ k$, we know that $n \leq k!$, so the set is finite.) For every $i > 0$, let $\vec{x}_i$ be a k-tuple such that $\vec{x}_i \in P - Q_i$. Put $A := \{\vec{x}_i : 1 \leq i \leq n\}$. This set is critical for P. For if $A \subseteq Q_i$ we get $\vec{x}_i \in Q_i$, contradicting our choice of $\vec{x}_i$. □

Thus, given the concept $\text{«}\mathsf{like}'(x_1,x_2)\text{»}_{\mathscr{M}}$ we only need a single pair $\langle a, b\rangle$ such that $\langle a, b\rangle \in [\![\mathsf{like}'(x_1,x_2)]\!]_{\mathscr{M}}$ but $\langle a, b\rangle \notin [\![\mathsf{like}'(x_2,x_1)]\!]_{\mathscr{M}}$. Then we may define $Y(\text{«}\mathsf{like}'(x_1,x_2)\text{»}_{\mathscr{M}})$ to be that minimal member of the concept that contains $\langle a, b\rangle$. Consequently, the linking aspect of the grammar defined above is defined by giving two pairs of objects: one for the concept of seeing, one for the concept of liking. Recall that

$$\mathfrak{I}(\mathsf{like}') = \{\langle A, A\rangle, \langle A, B\rangle, \langle A, C\rangle, \langle B, A\rangle, \langle B, D\rangle\} \tag{61}$$

Then $\{\langle A, C\rangle\}$ is a critical set for the concept of liking.

There are two cases for a binary relation: it is either symmetric, or it is not. If it is symmetric, the concept contains only one pair, and the critical set may in fact be empty. If it is not symmetric, one pair is enough. For higher order relations and concepts more tuples might be needed.

A second procedure is this. In place of knowing about a particular tuple that it is contained in the relation, we may know that the various arguments of the concept can be told apart by some inherent properties. There is, say, a unary predicate $\mathsf{exp}'(x_1)$ which holds of the experiencers. If we pick from the concept $\text{«}\mathsf{like}'(x_1,x_2)\text{»}_{\mathscr{M}}$ the set $\mathfrak{I}(\mathsf{like}')$ we expect that the first projection (the set $\{A, B\}$) is the set of experiencers of that concept. However, since seeing someone is also being an experiencer, the set of experiencers may not actually be identical to the previous. Assume, for example, that it is $\mathfrak{I}(\mathsf{exp}') = \{A, B, C\}$ (which turns out to be the set of individuals that either see or like someone). Then we can also tell apart the columns; suppose, namely, that we pick the relation $\mathfrak{I}(\mathsf{like}')^{\smile}$. Its first projection is

$\{A, B, C, D\}$, which contains D, a nonexperiencer. In logical terms, we have the following situation:

$$(62) \qquad \begin{array}{l} \mathscr{M} \vDash \mathsf{like}'(x_1, x_2) \rightarrow \ \mathsf{exp}'(x_1) \\ \mathscr{M} \nvDash \mathsf{like}'(x_1, x_2) \rightarrow \ \mathsf{exp}'(x_2) \end{array}$$

This translates into the following identities between relations:

$$(63) \qquad \begin{array}{l} [\mathsf{like}'(x_1, x_2)]_{\mathscr{M}} \subseteq \ [\mathsf{exp}'(x_1)]_{\mathscr{M}} \times M \\ [\mathsf{like}'(x_1, x_2)]_{\mathscr{M}} \nsubseteq \ M \times [\mathsf{exp}'(x_2)]_{\mathscr{M}} \end{array}$$

The following, however, does *not* hold: $\mathscr{M} \nvDash \neg(\mathsf{like}'(x_1, x_2) \rightarrow \mathsf{exp}'(x_2))$. This is because if someone is liked, s/he may still be an experiencer. (For example: A likes B and B likes A. Both are therefore experiencers.) Also, what we do *not* have is $\mathscr{M} \vDash \mathsf{exp}'(x_1) \rightarrow \mathsf{like}'(x_1, x_2)$. For someone may be an experiencer without liking someone. For example, if C sees A, C is an experiencer in virtue of seeing C, but C does not like anyone.

The previous method is reminiscent of θ-roles: in order to align the columns we make use of certain semantic relationships that hold between the concept and other concepts. However—and this may well hold for natural language as well—the method does not need to proceed using inherent properties. Moreover, it does not need to provide inherent properties for all its arguments. Let's deal with the second point. If we have a binary relation, we need to identify only one of the columns, the other one is fixed as a consequence. We need to know only who is experiencer, the theme is then clear as a result. This is interesting insofar as it turns out that especially the *theme* is notoriously ill-defined. There seems to be no definition that would reliably pick out the theme from a concept, in distinction to others, such as *experiencer*. But it is also not necessary to have such a definition as long as the other arguments can be picked out. Furthermore, to return to the first point, it is not necessary to be able to pick out every argument independently from the others. For example, in a ternary concept, it is enough to be able to pick out one of the arguments in terms of the two others, and subsequently one more argument in terms of the remaining one. This results in the following definition.

Definition 8 Let $\varphi(x_1, \ldots, x_n)$ be a formula. A θ-*cascade* for $\varphi(\vec{x})$ is a series of formulae $\chi_i, 0 < i < n$, such that

(1) $\chi_i = \chi_i(x_1, \ldots, x_i)$,
(2) $\mathscr{M} \vDash \varphi(x_1, \cdots, x_n) \rightarrow \chi_i(x_1, \cdots, x_i)$
(3) For every injective function $\pi : \{1, 2, \cdots, i\} \rightarrow \{1, 2, \cdots, i+1\}$, if π is not the identity, $\mathscr{M} \nvDash \varphi(x_1, \cdots, x_n) \rightarrow \chi_i(x_{\pi(1)}, \cdots, x_{\pi(i)})$.

For example, $\langle \mathsf{exp}'(x_1) \rangle$ is a θ-cascade for $\mathsf{like}'(x_1, x_2)$. That cascades are necessary is exemplified by such notions as beneficiary. In an event that has a beneficiary, the beneficiary is often an intended recipient by the actor, as in 'John paints a picture for Mary.' Thus, we cannot define what it is to be a beneficiary without taking recourse to the subject.

In comparing these two ways of defining a linking aspect, note that both of them have disadvantages. θ-roles do not always work. While verbal heads seem to allow

for differentiation of arguments (however see Levin & Hovav, 2005 for a discussion of the complexity of this issue) there are clear cases where such a differentiation is not possible. Consider the case of `greater` in the domain of integers. This word is interpreted by the relation $>= \{\langle m, n\rangle : m > n\}$; both projections of this relation are $\mathbb{Z}$. For it is the case that for every number there is a larger number; and it is likewise the case that for every number there is a smaller number. It follows that there is no intrinsic characterisation for either position. Thus there is no cascade for this concept. We can use critical sets, though. The set $\{\langle 1, 0\rangle\}$ is critical for $>$. The method of critical pairs, however, has the disadvantage of providing only a case by case analysis. What we really wish to have is some general algorithm to establish the linking aspect, and this is why cascades are preferred.

13 How general is this?

Now that we have defined a grammar and shown that it can correctly handle the cases, we need to ask just how generic the grammar is. In other words: will this toy grammar really scale up to natural language in the way Montague Grammar does? Or does it have inherent limitations, and if so, which ones?

I first discuss aspects where I foresee no problems in generalising the grammar and then turn to problems that I have so far identified. On the positive side let us note that we can practically introduce primitive relations of any arity. We are not bound to binary relations. If we want to use a ternary predicate, we introduce, say, a rule

(64) $\mathrm{VP} \to \mathrm{VD\ NP\ NP}$

where VD is the class of ditransitives, for example

(65) $\mathrm{VD} \to$ `call`

For the purpose of linking, after the VP is formed, we get a unary concept, and so the linking aspect is needed only once.

Another point to mention is the fact that many properties actually contain many more variables than we have displayed. For example, any realistic semantics of the word `president` will need to include a time point, a world (or situation) and a variable defining the institution of which the person is president. Such added parameters become vital in giving a successful semantics for sentences such as the following:

(66) `The president met the ex-director of the bank when they were attending high school.`

This sentence is full with reference to time; the subject is president now, the object is director at some earlier time point, and subject and object attended school (presumably) even earlier than that. I call such added variables *parameters*. Such ubiquitous parameters make life difficult in semantics. However, in our case the situation is actually simpler than for most others. Variables of different sort can never be identified with each other. Thus, if a concept involves variables of different sort, critical sets need to be established only up to confusion of variables of identical sort. To give an example, if we decide to render the semantics of `president` as

president$'(t, w, x, y)$ (x is president of institution y at time t in world w), then if x and y, say, are of identical sort, then it is only x and y that can be confused: t, the time point, is sortally distinct from w, x and y; w, the world, is sortally distinct from t, x and y.

Now we come to the downsides of this. First, notice that the only rules of coordination we have is VP coordination. This is no accident. If we were to admit the coordination of, say, transitive verbs, we must also define the linking aspect on the resulting concept, since the VP-formation rule $G(\theta_2)$ uses Y to do the linking. In the sentence

(67) `Some man likes and sees some woman.`

we shall form the concept of liking-and-seeing someone, and then link it to the object. At this point the aspect Y is invoked. So we require it to be defined. Another problem arises with the ditransitives. For the syntactic evidence suggests that rather than feeding two objects at once, syntax feeds them one by one. This means that we rather that (64) we want the following rule:

(68) $\mathrm{VT} \rightarrow \mathrm{VD\ NP}$

This allows a ditransitive to combine with one of its objects first, and then with the second one. This again requires that the linking aspect be defined for complex concepts, not just those that the lexicon supplies. A last problem concerns scope. One of the success stories of formal semantics was its ability to explain different readings in terms of scope differences. Montague Semantics also showed a way to generate them in a systematic way. The present theory however returns in some aspects to a pre-Montagovian analysis: there is no obvious way in which alternative scopes can be derived. For the time being, each argument is quantified off when it is supplied. That feature can be eliminated, but that creates problems of its own (see below in Sect. 17).

It is to be noted, though, that what appears here under the heading 'downside' is not really to be regarded as an argument against the proposal. Rather, it is my conviction that precisely these limitations provide some insight into the quirky nature of language. We noted, for example, that coordination of transitive verbs, indeed heads in general, is problematic. If we read (Keenan & Faltz, 1985) we are thus compelled to think that we must discard this kind of semantics. However, I suggest it rather means that we have to rethink our syntax of coordination. On the other hand, looking closer we can also see that not all coordinated structures are equally good. Consider

(69) `John hates or somewhat dislikes his donkey.`

(70) `?John likes or beats his donkey.`

(71) `John walks or talks.`

(72) `?John is seen or walks.`

Though intelligible, these sentences seem odd. But there is no prohibition against them. What I suggest happens (more with disjunction that with conjunction, by

the way) is that the formation of a new concept to be combined with others is not successful if no uniform linking aspect can be found. One way of guaranteeing a uniform linking aspect is if the two are similar in terms of θ-roles and linking aspect. I admit that the intuitions are vague; moreover, a theoretical explanation would have to go into the detail of the computation of the linking aspect. I suggest leaving that topic for further research. Instead I turn now to diathesis.

14 Grammatical roles and diathesis

As we have seen above, there is—least in some cases—a possibility to organise linking in terms of θ-roles or θ-cascades. What needs to be discussed, though, is the fact that the same predicate can be linked differently, due to diathesis.

The grammatical roles are arbitrary. This is to be expected. Consider the sentence

(73) `Bert is seen by Alex`.

Here, Bert is the subject even though it is object under Y. This brings us to the notion of *diathesis*. For simplicity, let us assume that there is a transitive verb `is seen by`. The concept associated with it is the same as that of `sees`. It follows that we must link the same concept differently in the passive. We can either use a different linking aspect (say, by using Z defined by $Z(\mathfrak{c}) := Y(\mathfrak{c})^{\smile}$) or we can use the same linking aspect and just link the arguments in a different way. The first option then makes us define the following rule $G(\theta_2)$:

$$
\begin{aligned}
&G(\theta_2)(\langle e, c, m\rangle, \langle e', c', m'\rangle)\\
&:= \begin{cases}
\langle e'^{\frown}\Box^{\frown}e, \mathrm{S}, [\![\Pi_1.(Z(m) \cap Z(m'))]\!]_{\mathscr{M}}\rangle\\
\qquad \text{if } c = \mathrm{VP} \text{ and } c' = \mathrm{NP}\\
\langle e^{\frown}\Box^{\frown}e', \mathrm{VP}, [\![\Pi_2.(Z(m') \cap (M \times Z(m)))]\!]_{\mathscr{M}}\rangle\\
\qquad \text{if } c = \mathrm{VT} \text{ and } c' = \mathrm{NP}\\
\text{undefined else}
\end{cases}
\end{aligned}
\tag{74}
$$

The second option rather asks us to define the rule as follows:

$$
\begin{aligned}
&G(\theta_2)(\langle e, c, m\rangle, \langle e', c', m'\rangle)\\
&:= \begin{cases}
\langle e'^{\frown}\Box^{\frown}e, \mathrm{S}, [\![\Pi_1.(Y(m) \cap Y(m'))]\!]_{\mathscr{M}}\rangle\\
\qquad \text{if } c = \mathrm{VP} \text{ and } c' = \mathrm{NP}\\
\langle e^{\frown}\Box^{\frown}e', \mathrm{VP}, [\![\Pi_1.(Y(m') \cap (Y(m) \times M))]\!]_{\mathscr{M}}\rangle\\
\qquad \text{if } c = \mathrm{VT} \text{ and } c' = \mathrm{NP}\\
\text{undefined else}
\end{cases}
\end{aligned}
\tag{75}
$$

One is as good as the other. Notice however that these definitions of $G(\theta_2)$ are different from the original one. So, what we need to do is to differentiate the active

from the passive. This is done by adding the feature $[\pm p]$ to the verb, so that a full definition runs as follows:

$$G(\theta_2)(\langle e, c, m\rangle, \langle e', c', m'\rangle) := \begin{cases} \langle e'^\frown\Box^\frown e, \mathrm{S}, [\![\Pi_1.(Y(m) \cap Y(m'))]\!]_{\mathscr{M}}\rangle \\ \quad \text{if } c = \mathrm{VP} \text{ and } c' = \mathrm{NP} \\ \langle e^\frown\Box^\frown e', \mathrm{VP}, [\![\Pi_2.(Y(m') \cap (M \times Y(m)))]\!]_{\mathscr{M}}\rangle \\ \quad \text{if } c = \mathrm{VT[-p]} \text{ and } c' = \mathrm{NP} \\ \langle e^\frown\Box^\frown e', \mathrm{VP}, [\![\Pi_1.(Y(m') \cap (Y(m) \times M))]\!]_{\mathscr{M}}\rangle \\ \quad \text{if } c = \mathrm{VT[+p]} \text{ and } c' = \mathrm{NP} \\ \text{undefined else} \end{cases} \tag{76}$$

This solves the problem of diathesis. However, the addition of gratuitous syntactic features potentially proliferates inattestable distinctions. The problem is that we may in fact use the syntactic categories to transmit information from syntax to semantics (for example: although we have eliminated all reference to order in a concept, we may keep track of an intended ordering by annotating the linking aspect in the syntactic category). To prevent this abuse I propose to implement the following principle. Say that for a context free grammar G, $\vec{x}$ *belongs to category* A *in* G iff $A \Rightarrow_G^* \vec{x}$. G defines constituent occurrences of $\vec{x}$ in the obvious way.

> Identity of Indiscernibles. Suppose that each constituent occurrence of $\vec{x}$ can be substituted for every constituent occurrence of $\vec{y}$ and vice versa. Then $\vec{x}$ and $\vec{y}$ belong to the same categories of G.

Let us see what options languages have. First, as we have said above, by the principle of identity of indiscernibles, there must a difference in category between actives and passives that reflects different distribution. This is indeed the case. Thus, verbs come in two forms, *active* and *passive*. These can be distinct in two ways: they can be distinct in form (exponent); or they can be distinct in category. If distinct in form the linking is triggered via the difference exponent. If distinct in category, the linking is conditioned by the syntactic context. Both possibilities exist. English dative shift, for example, leaves no morphological trace. But it changes the syntactic environment. The verb does not expect a `to`-PP any more, but instead two DPs.

(77) `Alex gave a book to Cindy.`

(78) `Alex gave Cindy a book.`

The principle of identity of indiscernibles does not rule that out. The form of the verb (77) is the same as in (78) but it enters different syntactic contexts. The two therefore have different category and are not indiscernible.

Our theory predicts that what will not happen is that there is a rule of passive that exchanges subject and object. For by the Identity of Indiscernibles, active and passive will then be identical in category. If that is to, they will enter the same constructions. There is nothing that can trigger the different choice of linking aspect.

15 Identity and reflexives

I shall briefly discuss the impact of our definitions on two topics: identity and reflexives. First, notice that the concept associated with $x_1 = x_2$ is actually $\{\varnothing\}$, the 'true concept'. This is so because it is generated by the set $\{\langle x, x\rangle : x \in M\}$, which by (S4) is reducible to $\{\langle x\rangle : x \in M\}$, which is the same as M. Using (S5) and the fact that $M \cong 1 \times M$ this is further reduced to 1 (which is the set $\{\varnothing\}$). Thus the fact that some variable is identical to another makes no contribution. But surely we can issue statements to the effect that one thing is the same as another. How is this therefore possible?

The answer lies in the following. From a metaphysical point of view identity is indeed trivial: every object is identical to itself, nothing else needs to be said. There can also be no two identical things. Thus identity statements really reveal the identity not of objects but of descriptions thereof. Cognitively speaking I wish to think of the mental representations as not containing duplicates of the same thing either. If we have different images of Brutus the son of Caesar and of Brutus the murderer of Caesar, then we think they are different. When we learn that they are the same, however, our mental representation will change, too. It will no longer keep separate images of the two. (Well, it might, but then I'd say we have not fully implemented the identity.) I'd like to speak of identity therefore as *process meaning*. Its ''content'' (qua concept) is trivial, but its effect in constructions can be substantial.

This may explain why the syntax of ''to be'' is special. If we were to treat it like a transitive verb, say ''like'', it would enter the construction with a meaning that is empty—a needless thing indeed. Therefore, some languages decide to leave the copula empty. In Hungarian, for example, the third person copula (van) is left out in predicational sentences:

Ez az ember jó
(79) This DEF man good
This man is good.
(80) *Ez az ember jó van

Other languages, like English and German, keep the copula. However, the construction is never symmetric. It seems that the copula rather functions to promote the postcopular constituent to a predicate. In Finnish, where there is a special case (the essive) the construction therefore looks like an overkill:

Jussi on sairana.
(81) Jussi is sick-ESS
Jussi is sick.

Here the copula seems to be required for the sole purpose of having an inflection carrier, for example to spell out tense and mood. Indeed, in Hungarian the copula will appear as soon as we use past tense.

This behaviour of the semantics has another consequence. Consider a reflexive verb like ''to wash oneself''. It is theoretically possible to see this as a transitive verb with the added semantic condition that the subject is identical to the object. So, the binary constant $\mathsf{wash\text{-}r}'$ is interpreted as follows:

(82) $\mathfrak{I}(\mathsf{wash\text{-}r}') = \mathfrak{I}(\mathsf{wash}') \cap d_{1,2} = \{\langle a, a\rangle : \langle a, a\rangle \in \mathfrak{I}(\mathsf{wash}')\}$

Call a predicate *diagonal* if $P(x,y)$ implies $x = y$. In Ancient Greek, the mediopassive formed a diagonal predicate from a binary relation. What is observed is however that mediopassive verbs are never transitive again. In Montague Grammar this could be explained by saying that the mediopassive has the following semantics:

(83) $\lambda\mathcal{P}.\lambda x.\mathcal{P}(x)(x)$

Apply it to something of type $e \to (e \to t)$ and you get something of type $e \to t$. However, in principle it is possible to assign to the mediopassive the following semantics:

(84) $\lambda\mathcal{P}.\lambda x.\lambda y.(\mathcal{P}(y)(x) \wedge x = y)$

This returns something of type $e \to (e \to t)$ again. Thus, Montague Grammar has no semantic explanation for this fact. It can only appeal to common sense.

In the present semantics, however, the result follows. The semantics insists that the mediopassive is a unary concept, not a binary one. This follows directly from the rule (S4). The rule obligatorily applies to all diagonal predicates. (This does not exclude that *syntax* treats a diagonal predicate as a binary one by using a transitive verb; to exclude that we need to restrict the syntax–semantics coupling. This is however not our concern here.) This should raise at least some suspicion: while the mediopassive overtly forms a diagonal predicate, it might not be always obvious that a predicate is diagonal. Say you form the following predicate: x **inc** y iff x is incident with all the lines that go through x. Then this is in a diagonal predicate in standard Euclidean geometry and so our semantics should view this as a unary concept, not a binary one. But it takes us a while to see this (if at all). I fully agree. On the one hand, however, this is a performance problem: we might not be aware of the consequences. On the other hand we should realise that the model structure $\mathscr{M}$ is a private object (otherwise the computations cannot be performed in the head at all). It follows that there is no a priori reason to assume that a given predicate is seen as diagonal by someone else. Only ostensible diagonal predicates like mediopassives are exempt from this problem. The example predicate **inc** and many others may not be unary in someone else's model structure.[21] Hence we arrive at the conclusion that for ostensibly diagonal predicates (mediopassives) there is pressure for syntax to treat them differently from other binary predicates.

This is indeed the case; I mention here only one fact. Languages with double agreement have interesting gaps in the paradigm. I give an example. In Mordvin the verb has agreement markers for both subject and object agreement (see Keresztes, 1990). Intransitive verbs conjugate only for subject agreement, but transitives conjugate in addition for object agreement. However, while there are agreement markers for 3rd subject and 3rd object agreement (eg 'he sees him'), there is no 1st subject and 1st object marker (eg 'I see myself'). Keresztes (p.c.) confirms that in such cases the verbs must be reflexivised and then conjugated intransitively. This would be the same if we want to express 'he sees himself' (as opposed to 'he sees him'). This shows that Mordvin really treats diagonal concepts differently. Similarly the missing agreement markers in Potawatomi (cf. Halle & Marantz, 1993) can be explained.

[21] And be it only for the reason that the person is not fully aware of the standard principles of geometry. That people agree on all facts of the world, or at least on the denotation of words, is wishful thinking.

16 Inverse marking

The previous sections have exposed the standard techniques of systematic argument linking: grammatical and thematic roles. We shall look briefly at a method that defines linking independently of the actual *syntactic* structure. It allows to perform linking of any number of arguments based on semantics of the NPs alone. We remain in our original model. An NP denotes a subset of this set. Let $<$ be an ordering on this set; just any ordering, for example this one. (This ordering is to be kept constant.)

$$\begin{aligned}(85)\qquad & \varnothing < \{A\} < \{B\} < \{C\} < \{D\} < \{A,B\} < \{A,C\} < \{A,D\} < \{B,C\} \\ & < \{B,D\} < \{C,D\} < \{A,B,C\} < \{A,B,D\} < \{A,C,D\} < \{B,C,D\} \\ & < \{A,B,C,D\}\end{aligned}$$

Then define the following function. It takes three inputs, a binary concept $\mathfrak{m}$ and two unary concepts $\mathfrak{p}$ and $\mathfrak{q}$. We take a linking aspect Y.

Case 1. $Y(\mathfrak{p}) < Y(\mathfrak{q})$. Then put

$$(86)\qquad T(\mathfrak{m},\mathfrak{p},\mathfrak{q}) := [\![Y(\mathfrak{m}) \cap (Y(\mathfrak{p}) \times Y(\mathfrak{q}))]\!]_{\mathscr{M}}$$

Case 2. $Y(\mathfrak{p}) \geq Y(\mathfrak{q})$. Then put

$$(87)\qquad T(\mathfrak{m},\mathfrak{p},\mathfrak{q}) := [\![Y(\mathfrak{m}) \cap (Y(\mathfrak{q}) \times Y(\mathfrak{p}))]\!]_{\mathscr{M}}$$

This algorithm does the following. The first step is as usual the choice of a minimal set. The linking is now done independently of the surface order of the arguments; rather, it is done on the basis of the linear order. It may be checked that

$$(88)\qquad T(\mathfrak{m},\mathfrak{p},\mathfrak{q}) = T(\mathfrak{m},\mathfrak{q},\mathfrak{p})$$

For suppose that $Y(\mathfrak{p}) < Y(\mathfrak{q})$. Then on the left hand side we are in Case 1, while on the right hand side we are in Case 2. But the two cases link the arguments inversely. Similarly if $Y(\mathfrak{p}) > Y(\mathfrak{q})$. The case $Y(\mathfrak{p}) = Y(\mathfrak{q})$ means $\mathfrak{p} = \mathfrak{q}$, and so again the result follows. So the function does not care even in which order the NPs are arranged.

Such systems do exist. Inverse marking is an implementation of this idea. It starts with a notion of rank, typically *animacy*. Animacy is a semantic notion: the rank is not decided on the basis of what a thing is called but on the basis of what it is. In Plains Cree the following hierarchy is used:

$$(89)\qquad 2 > 1 > 1\text{dual inclusive} > 3\text{prox} > 3\text{obv}$$

Finally, the linking aspect is defined as follows. Let $\mathfrak{m} = [\![M]\!]_{\mathscr{M}}$ for some binary relation M. We find (at least 'normally') that in M either for all $\langle x,y\rangle \in M$ x is higher in agentivity than y; or for all $\langle x,y\rangle \in M$ x is lower in agentivity than y. In the first case let $Y(\mathfrak{m}) := M$; in the second let $Y(\mathfrak{m}) := M^{\smile}$. This fixes the argument places. Now, the verb has two arguments, and we assume that they are both immediate constituents of the sentence. Then the verb combines with both arguments at the same time; there is no subject and object. The meanings are combined using T. The two arguments may or may not be positionally distinguished.

If English were like Plains Cree, English could afford freer word order. For the following sentences would equally mean 'you see me':

(90) I see you.

(91) You see me.

The system has an immediate problem: How do we express the meaning 'I see you'? For that we need different mode of composition. Define in Case 1 above:

(92) $U(\mathfrak{m},\mathfrak{p},\mathfrak{q}) := [\![Y(\mathfrak{m}) \cap (Y(\mathfrak{q}) \times Y(\mathfrak{q}))]\!]_{\mathscr{M}}$

In Case 2 put

(93) $U(\mathfrak{m},\mathfrak{p},\mathfrak{q}) := [\![Y(\mathfrak{m}) \cap (Y(\mathfrak{p}) \times Y(\mathfrak{q}))]\!]_{\mathscr{M}}$

Finally, here is the mode for sentence meanings:

(94) $c_1(\langle e, \mathrm{V}, \mathfrak{p}\rangle, \langle e', \mathrm{NP}, \mathfrak{m}\rangle, \langle e'', \mathrm{NP}, \mathfrak{n}\rangle) := \langle e^\frown\Box^\frown e'^\frown\Box^\frown e'', \mathrm{S}, T(\mathfrak{p},\mathfrak{m},\mathfrak{n})\rangle$

(95) $c_2(\langle e, \mathrm{VI}, \mathfrak{p}\rangle, \langle e', \mathrm{NP}, \mathfrak{m}\rangle, \langle e'', \mathrm{NP}, \mathfrak{n}\rangle) := \langle e^\frown\Box^\frown e'^\frown\Box^\frown e'', \mathrm{S}, U(\mathfrak{p},\mathfrak{m},\mathfrak{n})\rangle$

Here, VI is the category of inverse marked verbs. There is an affix in Plains Cree that tells us whether or not a verb is interpreted directly (that is, using T) or inversely (U).

Additional complications quickly arise. The ordering (89) is actually far from linear. Basically, it fails to distinguish any 3rd participants from each other. To make up for that one can mark them to be 'proximate' and 'distal' or 'obviative'. The proximate takes the slot of the higher ranked argument. There are ways to implement that strategy too (it is basically a form of case marking).

Notice that as we have repeatedly argued, there can be no rule that simply exchanges subject and object. Inverse marking looks deceptively like that. However, it turns out that direct and inverse are not syntactically identical. The pronouns are of a different grammatical category. In Plains Cree, for example, the sentence 'I hit the man' cannot be used with proximate marking on man, if the verb is marked 'direct'; this is because 1st person is higher and direct marking makes it the subject. Proximate marking is licit with the inverse form, though. In this way, direct and inverse marked verbs are syntactically distinct.

17 Keeping the scope open

The idea of quantifying an object away as soon as the function has been applied, has been the basic principle of Montague Grammar. Montague Grammar interprets every expression by a closed λ-term and the only admissible interpretation is function application. If one is using relations rather than functions, then one has to translate $\lambda x.f(x)$ into $x_2 \doteq f(x_1)$, and use identification of variables with additional quantification to achieve the same result. If we systematically eliminate λ-abstraction, we end up with functions of the form $f(\vec{x})$, which take as input a sequence of elements of certain basic type and return a value of given type. (If you like, they are elements of a many sorted algebra). Function application is a binary schematic operation that takes two such functions and identifies the result of the second with a given argument of the first and then quantifies away the auxiliary variables.

Thus, function application becomes the following map:

(96) $\langle x_2 \doteq f(x_1, \vec{y}), h(x_3, \vec{z})\rangle \mapsto \exists x_2.\exists x_3.x_2 \doteq f(x_1, \vec{y}) \wedge h(x_3, \vec{z}) \wedge x_1 \doteq x_3$

One of the intermediate variables can be made to disappear by applying a substitution:

(97) $\langle x_2 \doteq f(x_1, \vec{y}), h(x_3, \vec{z})\rangle \mapsto \exists x_2.x_2 \doteq f(x_1, \vec{y}) \wedge h(x_1, \vec{z})$

In this way we can keep the number of free variables rather low. However, in order to solve the problem of inverted quantifier scopes, Montague departed from the previous scenario as follows.

The function $\lambda x_1.\lambda x_2.\mathsf{see}'(x_2, x_1)$ (the meaning of `sees`) was applied first to some variables, say x_8 and x_{67} to give $\mathsf{see}'(x_{67}, x_8)$. These variables are quantified away only later. With the names of variables now on display the approach is vulnerable to the objections raised above. This is because the name of the variables is immaterial and nothing in the surface string tells us which one to choose. What is more, there is a popular doctrine (first implemented in DRT) that the meaning of pronouns is something like $x_i \doteq x_j$, where x_i is a fresh variable and x_j is a variable previously introduced. If this is so, and given that pronouns refer to elements outside of the sentence, we cannot even assume that all variables are quantified away at the end of the sentence. This has led to several changes in semantic theory, all trying to capture the fact that variables are visible as far to the right as possible by semantic principles. (For example, variables are not visible if inside a negation or a universal quantifier, but they are visible if inside an existential.) This means that the strategy we have employed above of quantifying away variables when no longer needed does not work in natural language. We should refrain from quantifying them away.

Again there are many ways in which this may upset the compositional process. First, in keeping the variables alive we may create concepts of ever increasing length. However, the linking mechanisms are generally defined only for predicates of low arity; at a certain point the linking mechanism becomes indeterminate. (This is factually behind the Theorem 4.) We shall give only one among many examples. Consider

(98) `Alex thinks that Bert thinks that he is a fool.`

The pronoun `he` has three potential antecedents: `Alex`, `Bert` or some other individual. There are several ways in which we can remove the ambiguity of the sentence. We may point at the person in question. This is tantamount to adding more information to the pronoun; in order to deal with that we need a more comprehensive treatment of meanings, one that includes gestures. Similarly, in sign languages, arguments are put into virtual space, they are assigned spatial positions and are retrieved from there. If I point at a location, I mean the individual that has been assigned that location. This is a way to establish indices without numbers; this is a viable procedure but it is not the one that spoken languages use. Mathematical discourse is again different: we assign names to things for the purpose of unique identification. (`Let PQR be a triangle...` is a way to introduce a triangle defined by three points, P, Q and R.) Again, this is perfectly acceptable but not the way languages work. Instead, languages operate by what (Fiengo & May, 1994) call *vehicle change*. The expression used to identify an object may be different depending on syntactic criteria. In the case of Fiengo and May they are mostly interested in the question of pronouns versus reflexives

versus empty and the way that reflects on syntactic identity. But the metaphor may be used here too: rather than use a pronoun one may use a description that is enough to single out the correct antecedent. This is reminiscent of the distinction between referential and attributive description. The referential part of the description actually serves in establishing the linking while both the referential and attributive parts enter the semantic representation. We shall not elaborate that further; suffice it to say that the present theory predicts the necessity of descriptive content in referential expressions if a potentially unbounded number of columns is kept.

18 Dutch again

Now we are ready to face the question: is there a compositional context free grammar for Dutch? Even though there can be no formal proof that no such grammar ever exists (which in turn *can* be proved) there are good reasons why such grammar is not available for Dutch. The present proof is based on the following assumptions.

(1) Dutch has cross serial dependencies which are unbounded in length.
(2) Meanings are concepts (in the technical sense of this paper).
(3) The admissible operational meanings are: identification of arguments and existential quantification.

The last point needs emphasis: I am restricting the behaviour of the meanings that the construction alone can add to the following: an operation that takes, say, two concepts $\mathfrak{c}$ and $\mathfrak{d}$ as input, may only align them using a linking aspect (possibly different aspect for different arguments), then identify certain columns (intersect with some of the diagonals $d_{i,j}$), and apply some of the projections Π_i. Finally, it must return the concept. In this way we ensure that semantics is not destructive: every concept created is used in an essential way in the structure.

Suppose there is a compositional context free grammar; then by the Pumping Lemma large enough strings can be decomposed into

(99) $\vec{u}\vec{v}\vec{x}\vec{y}\vec{z}$

such that all of the following strings are also in the language:

(100) $\vec{u}\vec{v}^n\vec{w}\vec{x}^n\vec{y}$

It is not hard to see that $\vec{v}$ must consist of a sequence of noun phrases, and $\vec{y}$ of a sequence of verbs. In what is to follow we need one more piece: the existence of some $\vec{v}_1$ with different meaning that can be put in for $\vec{v}$. In other words, the following should be a subset of Dutch:

(101) $\{\vec{u}\vec{c}_i^n\vec{w}\vec{x}^n\vec{y} : n \in \mathbb{N}, \text{for all } i \leq n : \vec{c}_i \in \{\vec{v}, \vec{v}_1\}\}$

To see that we can indeed have $\vec{v}_1$, notice that there are at least two different NPs (in fact, many more). Even if we assume that there is only one raising verb, the number of sentences of length n grows exponentially in n, since any noun phrase slot can be filled with at least two members (irrespective of the structure that we assign to it; this

just concerns the number of Dutch sentences). If only the pumping pair $\langle \vec{v}, \vec{x} \rangle$ existed, not enough strings could be generated. (Their number would in that case be linear.)[22] The same argument can be used to show that also some $\vec{x}_1$ exists with meaning different from $\vec{x}$, but we do not need that.

Now we turn to the semantics to see whether this grammar allows for compositional interpretation. We assume that the meaning of $\vec{w}$ has been established and is $\langle\!\langle \varphi \rangle\!\rangle_{\mathcal{M}}$. In the next step we take in the meanings of $\vec{v}$ and $\vec{x}$. However, notice that since we can have any number of $\vec{v}$ and $\vec{x}$ in a row, we cannot simply identify the variables. Let us look at this in more detail. To make matters simple, we assume that the meaning of $\vec{v}$ is $\langle\!\langle \beta(x_1) \rangle\!\rangle_{\mathcal{M}}$, with just one free variable, the meaning of $\vec{v}_1$ is $\langle\!\langle \beta_1(x_1) \rangle\!\rangle_{\mathcal{M}}$, and the meaning of $\vec{x}$ is $\langle\!\langle \gamma(x_1, x_2) \rangle\!\rangle_{\mathcal{M}}$ (these verbs have a subject and an object). Also, we think of the $\vec{v}$ and $\vec{v}_1$ as noun phrases (rather than sequences thereof) and of $\vec{x}$ as a verb, with a subject and an object (in addition to a complement infinitive). Once the argument is established it is straightforward to extend it to the case where the formulae contain more free variables and the constituents are less simple. For simplicity we also ignore variables present in other formulae. Let's align strings and meanings.

$$
\begin{array}{lllll}
\vec{u} & \vec{v}_1 & \vec{v}_1 & \vec{v} & \\
\langle\!\langle \alpha \rangle\!\rangle_{\mathcal{M}} & \langle\!\langle \beta_1(x_1) \rangle\!\rangle_{\mathcal{M}} & \langle\!\langle \beta_1(x_1) \rangle\!\rangle_{\mathcal{M}} & \langle\!\langle \beta(x_1) \rangle\!\rangle_{\mathcal{M}} & \\
\vec{w} & \vec{x} & \vec{x} & \vec{x} & \vec{y} \\
\langle\!\langle \varphi \rangle\!\rangle_{\mathcal{M}} & \langle\!\langle \gamma(x_1, x_2) \rangle\!\rangle_{\mathcal{M}} & \langle\!\langle \gamma(x_1, x_2) \rangle\!\rangle_{\mathcal{M}} & \langle\!\langle \gamma(x_1, x_2) \rangle\!\rangle_{\mathcal{M}} & \langle\!\langle \delta \rangle\!\rangle_{\mathcal{M}}
\end{array}
\tag{102}
$$

Now, the minute we enter these constituents into the derivation we have to replace the variables in the formulae to avoid a clash. First of all, since the verbs link subject and object, we can use the γ's to establish an order on the variables. So, we fix the variable names by looking at the γ. Namely, rename the variables such that you only have formulae of the form $\gamma(x_i, x_{i+1})$. Now, if the first variable on the left is eventually called x_1, too, it becomes the highest subject, and if the second variable on the left receives the same name x_2 it becomes the second highest subject, and so on. This means that the lower row is as follows:

$$
\begin{array}{lllll}
\vec{w} & \vec{x} & \vec{x} & \vec{x} & \vec{y} \\
\langle\!\langle \varphi \rangle\!\rangle_{\mathcal{M}} & \langle\!\langle \gamma(x_1, x_2) \rangle\!\rangle_{\mathcal{M}} & \langle\!\langle \gamma(x_2, x_3) \rangle\!\rangle_{\mathcal{M}} & \langle\!\langle \gamma(x_3, x_4) \rangle\!\rangle_{\mathcal{M}} & \langle\!\langle \delta \rangle\!\rangle_{\mathcal{M}}
\end{array}
\tag{103}
$$

(To make this clear: the actual name of the variable is arbitrary, so we are allowed to fix their names arbitrarily at this point. It would be more exact to use the brackets $[-]_{\mathcal{M}}$, since we want to use the names later. Nothing should hinge on the choice of names, though.) Thus, trusting that we can in this way identify the variables used the γs, let's turn to the β formulae. In a nested structure, their order would be $x_3x_2x_1$ in

[22] There is a loophole in the argument: it could be that we have a pumping pair $\langle \vec{v}_1, \vec{x}_1 \rangle$ which simply adjoins at different places than $\langle \vec{v}, \vec{x} \rangle$. We have sidestepped that possibility, the proof will go through regardless.

the first row. Notice however that the assignment $x_2x_1x_3$ yields a nondistinct meaning from the assignment $x_1x_2x_3$:

$$\begin{aligned}&\text{«}\alpha \wedge \beta_1(x_2) \wedge \beta_1(x_1) \wedge \beta(x_3) \wedge \varphi \wedge \gamma(x_1, x_2)\\ &\quad \wedge\gamma(x_2, x_3) \wedge \gamma(x_3, x_4) \wedge \delta)\text{»}_{\mathscr{M}}\\ =&\text{«}\alpha \wedge \beta_1(x_1) \wedge \beta_1(x_2) \wedge \beta(x_3) \wedge \varphi \wedge \gamma(x_1, x_2)\\ &\quad \wedge\gamma(x_2, x_3) \wedge \gamma(x_3, x_4) \wedge \delta)\text{»}_{\mathscr{M}}\end{aligned} \tag{104}$$

Let us now trace the steps in the derivation. We start with $\vec{w}$ and meaning $\text{«}\varphi\text{»}_{\mathscr{M}}$. The first step is to form the constituent $\vec{v}\vec{w}\vec{x}$ with meaning

$$\text{«}\beta(x_3) \wedge \varphi \wedge \gamma(x_1, x_2)\text{»}_{\mathscr{M}} \tag{105}$$

Thus, the variables of the noun phrases are not identified with any of the variables of the verbs. Assume that in the second step we get $\vec{v}^2\vec{w}\vec{x}^2$ with meaning

$$\text{«}\beta_1(x_2) \wedge \beta(x_3) \wedge \varphi \wedge \gamma(x_1, x_2) \wedge \gamma(x_2, x_3)\text{»}_{\mathscr{M}} \tag{106}$$

with the variables correctly identified. Then we have committed ourselves to the identification of the leftmost noun phrase as the first level object (and second level subject), and the rightmost noun phrase with the second level object (and hence the third level subject). This is not in itself dangerous; however, we do also need to care about the case in which we want to iterate more than three times. In this case, we do not identify and instead get

$$\text{«}\beta_1(x_4) \wedge \beta(x_5) \wedge \varphi \wedge \gamma(x_1, x_2) \wedge \gamma(x_2, x_3)\text{»}_{\mathscr{M}}. \tag{107}$$

Suppose we continue on this line, keeping all the variables distinct, up to level n, where we want to start identifying. In comes a constituent $\vec{x}$ with meaning $\gamma(x_1, x_2)$ and either a constituent $\vec{v}$ with meaning $\beta(x_1)$ or $\vec{v}_1$ with meaning $\beta_1(x_1)$. We quickly rename the variables in $\gamma(x_1, x_2)$ to $\gamma(x_n, x_{n+1})$. We now however also want to identify the variable of the rightmost $\beta(x_1)$ with the lowest subject, which is now called x_n. The situation is now this: the variables occurring in a β-formula form a set U, the variables occurring in a β_1-formula form another set U_1 disjoint from U. One of $U \cup U_1$ must be identified with x_n. But which one? It must be in U if the last member of the noun phrase series is $\vec{v}$, and U_1 otherwise. But precisely this information is lost. This completes the argument.

Recall also that the meaning you have in your hands is quite a weak one: it is something like *there is a Mary, there is a John, there is a child, someone lets someone* (*do something*), *someone helps someone* (*to do something*), *someone is swimming* and so on. But any indication as to who is doing what to whom is missing. It was encoded in the order but now it has been lost. The only way to make the order information available to semantics is by projecting it into the syntactic derivation. That is, if you need to communicate to semantics the fact that the string for x_2 precedes the string for x_3 in syntax, you must see to it that x_2 is processed before x_3. This is why Dutch cannot be context free.

Notice that I have not assumed that merge is obligatorily accompanied by any identification of variables, as I did in Kracht (1999). Although I think the latter view is the correct one, the argument goes through anyway.

19 Viable structures for Dutch

The previous discussion was still a little informal. Moreover, it is not clear which structures if any would allow for a compositional interpretation. We shall therefore take a second look at the matter. We shall study again the cross-over dependencies. The constituents in a sentence each contribute a meaning $«\varphi(\vec{x})»_{\mathscr{M}}$ containing at least one variable. The NPs are different from infinitives in the following way. The different NPs each contribute at least one new variable, but the variables they contribute are independent of each other. The infinitives however contribute two variables each (at least), and they introduce a dependency relation between them.

We make the following assumption. The meanings of raising predicates have the form $«\gamma(e,x_1,x_2)»_{\mathscr{M}}$, for some event e and objects x_1, x_2. If e and e' are different events, or if $x_1 \neq x_1'$ or if $x_2 \neq x_2'$ then we assume $«\gamma(e,x_1,x_2)»_{\mathscr{M}} \neq «\gamma(e',x_1',x_2')»_{\mathscr{M}}$. (If this does not hold, matters can only get worse.) A meaning that does not involve a raising predicate is said to be of level 0; an event is of level $n+1$ if it has the form $«\gamma(e,x_1,x_2)»_{\mathscr{M}}$, where γ is a raising predicate, and e is of level n. This allows for every meaning $\mathfrak{m}$ to say which level it has. In what is to follow we shall suppress the event variable.

Suppose we want to compositionally attribute meanings to a sentence. We assume that constituents are any sets of occurrences of the sentence, not necessarily a continuous one. The constituent has an associated meaning $\mathfrak{m}$, which is computed from the immediate subconstituents, using the above operations. We assume that the NPs contribute formulae of the form $\alpha(x_1)$ (just one variable free). Possible NPs are `Jan`, `een kind` ('a child'), `een vrouw` ('a woman'), and so on. A verb contributes a formula of the form $\gamma(x_1,x_2)$. Let the entire sentence be as follows.

(108)

NP_1	NP_2	NP_3	$\dots$	V_1	V_2
$\alpha^1(x_1)$	$\alpha^2(x_1)$	$\alpha^3(x_1)$	$\dots$	$\gamma^1(x_1,x_2)$	$\gamma^2(x_1,x_2)$
	V_3	$\dots$			
	$\gamma^3(x_1,x_2)$	$\dots$			

A constituent consists in a subset of the NPs and a subset of the Vs. Let NP_i be in a constituent H with meaning $\mathfrak{m} = «\varphi»_{\mathscr{M}}$. We say that NP_i is *unattached* if there is j such that $\varphi \vDash \alpha^i(x_j)$ and moreover: if $\varphi \vDash \delta(x_j)$ then $\alpha^i(x_j) \vDash \delta(x_j)$. This means that the NP-meaning is concatenated, no identification of variables has been applied. This means that the NP does not know to which of the verbs it belongs. The notion of unattachedness is translated into structure as follows.

Lemma 9 *NP_i is unattached in H iff H does not contain V_i or V_{i+1} (the latter only if V_{i+1} exists).*

Proof Let O be the set of i such that NP_i is in H; and P the set of i such that V_i is in H. The meaning of H is (n large enough):

$$(109) \qquad «\bigwedge_{i\in O} \alpha^i(x_{n+i}) \wedge \bigwedge_{i\in P} \gamma^i(x_i,x_{i+1}) \wedge \bigwedge_{i\in O\cap P} x_{n+i} \doteq x_i»_{\mathscr{M}}$$

(This applies when H does not contain the lowest, nonraising verb, but the other case is quite similar.) When NP_i is entered, the variable of NP_i is identified to some other variable iff there is a verb that takes the NP_i as its argument. □

Lemma 10 *For every compositional grammar of Dutch there is a bound on the number of unattached NPs.*

Proof Suppose that we only have one function for the composition of meanings. We shall show that the bound is 1. Suppose the contrary. Then there is a constituent H' containing H at which either of the NPs, say NP_i, is attached. Without loss of generality we may assume that H' is the immediate constituent above H. To form the correct meaning of H' we must add the clause $x_i = x_{n+i}$. However, nothing prevents us from adding $x_j = x_{n+i}$ and thus end up with the wrong meaning.

The argument goes as follows. Case 1. H' does not contain NP_j. Form the sentence S^* by exchanging NP_i and NP_j, and assume that it has the constituents that S has, with NP_i and NP_j exchanged. In particular, it has a constituent H^* and a constituent H'^*. It turns out, though, that since H^* also contains NP_i and NP_j, and since both are unattached, H^* has the same meaning as H. When H' is formed, NP_i becomes attached. This can be so only because V_i or V_{i+1} is added. Therefore, NP_j becomes attached in H'^* (because it takes the place number i in S^*). This time, however, we want to add the equation $x_j = x_{i+n}$. Case 2. H' does contain NP_j. Similarly. Contradiction.

It thus turns out that in order to do the next step we need to know something about the order of NP_i with respect to NP_j. In principle, there could be a convention that orders any two sets of objects, so that if we have some number of unattached NPs, they are implicitly ordered. The meaning function that assembles them uses that ordering to determine which equation to add. However, if that function is to do its job properly, we need at least n different functions if the number of unattached NPs is at least n. Since the number of functions is finite, this concludes the proof. □

Let us see why we could not prove the stronger claim that the bound is 1. We may introduce an ordering on sets of individuals. At each stage, when we have two yet unattached NPs, say `een kind` and `een vrouw`, we use the convention to implicitly order them. Notice namely that our meaning so far is something like

$$(110) \qquad «\mathsf{woman}'(x_1)»_{\mathscr{M}} \bullet «\mathsf{child}'(x_1)»_{\mathscr{M}} \bullet «\varphi»_{\mathscr{M}} = [W \times C \times M]$$

where P is a relation in $«\varphi»_{\mathscr{M}}$. This allows us to use names on the variables again to say which one should be identified with the subject or object of some verb. We have a function $f_\bullet$ that always chooses the variable of the lowest ranked NP, and a function $f^\bullet$ that always chooses the higher ranked variable. Suppose that $W < C$. Then $f_\bullet$ will attach the variable x_1 of the first argument, $f^\bullet$ the variable of the second. This is precisely the strategy of inverse marking that we have talked about earlier, now taken to its limit.

Notice however that all this means that the order to the NPs has been coded, though in some hidden fashion. How can that be? It can be in the following way: the category of a constituent H is different from that of H' if H arises from permuting two unattached NPs. Thus, by devising a suitable category system it is possible to encode into the syntactic category what permutation has been applied to deviate from the 'standard' order. So far, nobody has proposed such a system at full scale and it does not seem to be realised in languages except for monoclausal verbs. In this linking system, the arguments may come in any order, and when they are merged into the verb it is decided on their relative rank whether they will be subject or object. Notice however that in order to keep a bound on the number of syntactic categories, the system can apply effectively to a bounded number of arguments only.

This is virtually the same with all the systems that we have looked at (grammatical roles, thematic roles). Dutch at any rate does not have such a system.[23]

The interested reader is asked to verify that CFGs require the introduction of constituents with an unbounded number of unattached NPs, thus proving again our claim. Now however we shall propose two structural accounts that work in the correct way.

① The verb clustering account (due to Evers (1975)): form a constituent from the verbs, taking them in either from left to right or from right to left. After having formed the verb cluster, take in the NPs in descending order. These are the structures that CCGs generate (see Steedman, 2000).
② Form the constituents by taking in NP_i together with V_i. This is the approach proposed by Mike Calcagno in Calcagno, 1995 for Swiss German. This leads to discontinuous constituents.

Recently Haider (2004) has argued in favour of an account that cyclically reorders the sequence of verbs. This is also compatible, provided that we accept compositionality. One can mix these strategies. Notice that German can be both generated using a CFG and using verb cluster formation, but there seems to be evidence (from coordination) that both structures exist concurrently. Structures that definitely do not work include the LFG analysis of Bresnan, Kaplan, Peters, and Zaenen (1987) which uses an NP-cluster.

20 Compositionality and transformational grammar

Let us return to the issue of compositionality in transformational grammar. There are two notion of compositionality here: one is based on the derivational cycle, and the other is based on the final output (LF). The first conception starts with meaning assigned by the lexical items and applies semantic rules in tandem with syntactic rules. This is historically the older proposal. Meanings were established at deep structure. Furthermore, it was assumed that transformations do not change the meaning of the sentence. The latter is as far as I can see an additional stipulation, not warranted by the requirement of compositionality. It is this latter stipulation that led to the new conception after it was shown that the two assumptions were not co-tenable. It is nowadays agreed that meanings are assigned to structures at LF and that at LF the assignment is compositional in the sense that it can be computed bottom up. To the extent that LF is itself derived in the transformational cycle it may be possible to make the latter algorithm truly compositional: it may be possible to transform it into a procedure that is coroutined with the *actual* structure building operation. This would require the following setup: we start with meanings assigned in the lexical signs and apply move and merge to them. Transformations can be effect by operations on meanings of the component signs, just like merge. I am not

[23] This requires proof even though the matter is only of peripheral interest. Effectively, if any two NPs with different meaning give rise to the same linking so that linking is completely dependent on the position in the linear string, this eliminates any hope of finding an inverse system in Dutch. This is seen as follows. In the inverse linking scheme there is no difference in NP_1NP_2 and NP_2NP_1. Thus, if order has an effect on linking this must be made known to semantics. The only way to this is to establish enough syntactic categories to discriminate the linking patterns. Again, using the Identity of Indiscernibles we see that Dutch does not have them.

aware of such a proposal within the current framework but it seems to me the only viable one.

Heim and Kratzer (1998) present the now current approach to semantics in transformational grammar: meanings are assigned to LFs and are computed bottom up. Inasfar as that computation of meaning is a process (here: a mental process) that involves meanings, the only difference between proposals inside MP and others are that the first assumes a different constituent structure as a basis for computation. In order to show that this proposal is successful, two things need to be established. The first: that LF can be generated by itself, without recourse to semantics. The second: that the structures generated at LF are compositionally interpretable. I think that it is possible to define syntactic structures that meet all these criteria. This follows for example from a combination of three results: that minimalism can be given the form of Stabler (1997), that this version of minimalism is weakly equivalent with Linear Context Free Rewrite Systems (LCFRSs) Michaelis (2001) with enough overlap in constituent structure and that the sentential structures of Dutch can be described using a 2-LCFRS.

Notice that weak equivalence is not enough, for we need to be able to have the constituents that allow for computation of the meanings. For example, the classical GB analysis of Dutch (and German for that matter) has been the following. Clauses are derived in centre embedding fashion, like this (constituents shown using brackets):

$$(111) \qquad [\mathrm{NP}^1 \quad [\mathrm{NP}^2 \quad [\mathrm{NP}^3 \cdots \mathrm{V}^3] \quad \mathrm{V}^2] \quad \mathrm{V}^1]$$

Next the verbs move, starting with the lowest and adjoin to the next higher head either to the left (German) or to the right (Dutch). This gives

$$(112) \qquad [\mathrm{NP}^1 \quad [\mathrm{NP}^2 \quad [\mathrm{NP}^3 \cdots \mathrm{t}_{n-2}] \quad \mathrm{t}_{n-1}] \quad \mathrm{t}_n][[[\cdots \mathrm{V}^3_{n-2}] \quad \mathrm{V}^2_{n-1}] \quad \mathrm{V}^1_n]$$

$$(113) \qquad [\mathrm{NP}^1 \quad [\mathrm{NP}^2 \quad [\mathrm{NP}^3 \cdots \mathrm{t}_{n-2}] \quad \mathrm{t}_{n-1}] \quad \mathrm{t}_n][\mathrm{V}^1_n \quad [\mathrm{V}^2_{n-1} \quad [\mathrm{V}^3_{n-2} \cdots]]]$$

These structures unfortunately will not qualify. Both effectively create an NP-cluster and a V-cluster. As we have shown above, there is no way to create the meanings compositionally. This incidentally also holds for the constituent structure argued for in Bresnan et al. (1987). However, the double-structure approach of LFG calls for the application of the principle of compositionality to the derivation and not the c-structure. It is beyond the scope of this paper to do that.

Let us look in a more detail at the approach that is advocated in Heim and Kratzer (1998). The algorithm is bottom up. Syntactic structures are trees where the labels contain categories and indices. The terminal strings contain the lexical entries, complete with a typed λ-term for the denotation of the respective entries. Some empty categories are interpreted as variables whose index is the one that syntax assigns them. (The text employs a different notation; that is a matter of superficial detail.) There are a handful of operations (see Page 95). However, like Montague's T14 and T16 (Dowty, Wall, & Peters, 1981), there are formation rules that are parametric. They use a number as a parameter (see Kracht, 2003, p. 440ff., for an analysis in relation to compositionality). The basic schema is this. The sentence

(114) `Paul loves every woman.`

is given the structure

(115) Every woman_5[Paul[loves t_5]]

In the first step, t_5 is translated as x_5 and fed to the function love'. Then Paul is translated into the constant p and fed as well:

(116) $[\![\text{every woman}]\!]^5\mathsf{love}'(p, x_5)$

At this point the choice of indices becomes relevant. The phrase 'every woman_5' is interpreted as quantifying over some variable, namely x_5, and it is the same variable that sits inside the VP, producing accidental capture of x_5.[24]

$$\begin{aligned}&[\![\text{every woman}]\!]^5(\mathsf{love}'(p, x_5))\\ =&(\lambda\mathcal{P}.\forall\, x_5\ \mathsf{woman}'(x_5) \to \mathcal{P})\mathsf{love}'(p, x_5)\\ =&\forall x_5.\ \mathsf{woman}'(x_5) \to \mathsf{love}'(p, x_5)\end{aligned} \tag{117}$$

Under the current approach, it is not possible to share names of variables in this way. It actually does not matter whether one thinks of the previous as being derived as given or whether one likes to insert an abstraction, as Heim and Kratzer (1998) suggest. It only shifts the problem to the definition of abstraction. Namely, Heim and Kratzer suggest that the meaning of 'every woman' is

(118) $[\![\text{every}]\!] = \lambda\mathcal{Q}.\lambda\mathcal{P}.\forall x.\mathcal{Q}(x) \to \mathcal{P}(x)$

The structure is now the following in place of (115).

(119) Every woman [5[Paul[loves t_5]]]

The interpretation of the constituent $[i\ \ \alpha]$ is $\lambda x_i.[\![\alpha]\!]$. It is at the abstraction step where we need information about the index.

The mechanism assumes that syntax supplies the indices to semantics which then uses them in whatever way it pleases. The same holds for Kamp and Reyle (1993). Effectively, syntax assigns indices to constituents which get translated into variables; the index i at man_i says that the variable x_i must be fed to the function man'. (In DRT, it is taken to mean that x_i must replace some canonical variable, say x_1, in the open formula, here $\mathsf{man}'(x_1)$. But these are questions of detail.) DPL and subsequent proposals have not managed to change that. Alphabetic innocence has not been regained, except in the form that Kees Vermeulen gave it in Vermeulen (1995).

Apart from the use of indices in predicting the distribution of reflexives and pronouns, indices are actually a purely semantic device. Their only reason of existence is to get the interpretation right. Surface strings do not contain indices. There is no expression $\texttt{man}_1$ in English. (That's why the meanings are assigned to LF, where such things are claimed to exist...) But suppose we grant the use of variables. Suppose we grant that syntax happily assigns indices: it still won't help. For as I have been arguing at length, semantics does actually not use them. Even if syntax passes down certain numbers, semantics will kindly ignore them. The variables are there

[24] Although it the capture of the variable is intended, you should think of it rather as an effect of the way syntax supervises semantics. For semantics the capture looks accidental.

only for the eye. Not only does it not make a difference whether you call a variable x_1 rather than x_5 in a formula—I claim that there is no variable with that name to begin with. I will not repeat the arguments given so far. Here I will point out a few facts about the present view that one must address before rejecting it. The first is that the present proposal shows why languages employ devices such as θ-roles, for example. Transformational grammar as far as I see has only been concerned with the question of how θ-roles function rather than why they are there to begin with. Similarly for agreement, diathesis and many other basic traits of languages. If it seriously envisages their elimination from syntax this only narrows the scope of the theory, although technically it is the correct move to make. It would not, of course, make it any easier to say why languages have them. Under the official view, it is the job of semantics to do that. However, I see no theory in formal semantics that would explain the existence of θ-roles rather than telling us how they function if they were to exist.

The next objection I raise is that the use of indices results in a misunderstanding of the role of the languages of logic. It is a nice thing to use logical formulae to explicate meanings, as Montague did; it is another to couple that with a mentalist conception of language. The use of indices for our mental language in the way they function in logical languages encapsulates assumptions about the human mind that may well turn out be false. Indeed, I claim that they are.

But now I hear my critics say: listen, surely we need some kind of register, a peg that unites all the information concerning a certain individual, and it is precisely the index that provides this peg. Can we not simply point at an abstract location and keep it fixed for a while like names in a mathematical proof? Fair enough, I say, but that does not mean that we use numbers to do that. What if you have to call a variable by a property in the sense of a θ-role? What if there is a limited array of names that you may use to call a variable? Moreover, there are different ways that variables may be used. Maybe it so happens that what we are really using are anonymous variables, of the sort occurring in Prolog, whose name is given by Prolog itself, so you cannot make prior reference to it since you cannot call it by its name. Or perhaps we are dealing with the variable _ or the backreferences $\backslash 1, \backslash 2$ etc. of Perl, whose value changes depending on the context? To use variables does not mean you use them the way that logicians use them.

These questions are not minor issues. To request an unlimited resource of memory locations together with a device to manage them is no small wish. Recall that you can code any amount of data into a number, so there is unlimited communication between syntax and semantics possible, at least in principle. Thus, anyone wishing to give the principle of compositionality some bite will have to exclude this device.

21 Conclusion

The present work argues for a radical rethinking of what the semantic representations are. I claim that these representations do not contain any indication of variable names, nor of their multiplicity. This move complicates the algorithm of meaning composition substantially; one must so to speak establish a temporary naming scheme for the variables, and then manipulate the meanings

before the variables sink back into anonymity. In view of the fact that the variables are anonymous, the temporary naming scheme must be established using some kind of meaning based criteria, be it in form of a critical set that aligns the variables by using concrete tuples, or by means of thematic roles that manage to single out the variable we look for in a sufficiently precise way. This plot motivates a number of devices that one finds on a regular basis in languages: thematic roles, grammatical roles, diathesis, and so on. Furthermore, principles like UTAH, predicting a uniform deep structure for verbs with identical θ-grids suggest that our view is correct. For they say that at the interface between syntax and semantics all that matters are thematic roles. It also shows us that the composition algorithm trades on an understanding of how to extend the relevant concepts to yet unseen cases and expand the linking aspect. Though the theory may become technically difficult, I claim that the complexity is not an artefact of the theory. It is a result of the way things are. It puts our noses right into the middle of the problems that natural languages actually face on a daily basis.

As a particular benefit I have looked at the sentential structure of Dutch. This is an interesting case insofar as the theory exposed here actually predicts that Dutch is not strongly context free even if weakly CF. Previously, such claims were mere speculations.

Appendix: How many meanings?

To get used to these new concepts, let us calculate a few of the meanings in the above sense. First we need a few definitions on permutations. A permutation can be written in two ways. The first, more explicit version is this.

(120) $$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 1 & 2 & 4 & 7 & 6 & 5 \end{pmatrix}$$

This says that the permutation maps 1 to 3, 2 to 1, 5 to 7, and so on. A more compact way of saying this is as follows. We write the numbers down in the order the permutation maps them to each other. We start with 1. It is mapped 3, 3 to 2, 2 to 1; once we have returned to an already existing number, we enclose the sequence in brackets like this: (132). In this sequence, each element is mapped to the next in the sequence, and the last is mapped to the first. Next we pick an element that we have to yet mentioned, say 4. It is mapped to itself, so we add (4). Next we pick 5. It is mapped to 7 and 7 to 5, so we add (57). This is now the complete representation:

(121) (132)(4)(57)(6)

Each bracketed sequence is called a *cycle*. The number of elements are the *length* of the cycle. Cycles of length 1 are generally dropped, so we arrive at the following notation.

(122) (132)(57)

The identity function will have an empty representation (all cycles have length 1 and may be dropped). We write it as (). Notice that the order inside the cycles is important, while the order of the cycles is not.

If π and π' are permutations of the same set, then $\pi \circ \pi'$ also is a permutation, where $(\pi \circ \pi')(i) = \pi(\pi'(i))$. Also, the inverse π^{-1} is a permutation. The set of permutations of a set is thus closed under composition (also called multiplication) and inverse; such sets are called **groups**.[25]

Now let $U = \{a, b\}$ be a universe with just two objects. There are exactly two 0-ary relations, $\varnothing$ and $\{\varnothing\}$. They are the denotation of «$\bot$»$_{\mathscr{M}}$ and «$\top$»$_{\mathscr{M}}$. As for unary relations, notice that $\{a, b\}$ is the universe, and by (S2) the same as $\{\varnothing\}$. Trivially, the empty unary relation is the same as the empty 0-ary relation, so have added only two meanings: «$x_1 \doteq a$»$_{\mathscr{M}}$ and «$x_1 \doteq b$»$_{\mathscr{M}}$.

Next we look at binary relations. There are four pairs of objects:

$$A := \langle a, a\rangle, \quad B := \langle a, b\rangle, \quad C := \langle b, a\rangle, \quad D := \langle b, b\rangle \tag{123}$$

If we permute the first and the second column, we map A to itself, B to C, C to B and D to itself. There are 16 sets that can be formed from the four entries. However, the empty set and the set $\{A, B, C, D\}$ are already 0-ary meanings; similarly, $\{A, B\}, \{A, C\}, \{B, D\}$ and $\{C, D\}$ are already unary meanings, by (S2). This leaves us with the following:

$$\begin{aligned} &M_1 := \{A\}, M_2 := \{B\}, M_3 := \{C\}, M_4 := \{D\}, M_5 := \{A, D\},\\ &M_6 := \{B, C\}, M_7 := \{A, B, C\}, M_8 := \{A, B, D\}, M_9 := \{A, C, D\},\\ &M_{10} := \{B, C, D\} \end{aligned} \tag{124}$$

M_1, M_4, and M_5 are subject to contraction and therefore eliminated. The permutation maps M_2 to M_3, and M_8 to M_9. This gives us the following 2-sets with corresponding formulae (for the given universe $\{a, b\}$).

$$\{M_2, M_3\} = \text{«}(x_1 \doteq a) \wedge (x_2 \doteq b)\text{»}_{\mathscr{M}} \tag{125}$$

$$\{M_6\} = \text{«}\neg(x_1 \doteq x_2)\text{»}_{\mathscr{M}} \tag{126}$$

$$\{M_7\} = \text{«}\neg(x_1 \doteq b) \vee \neg(x_2 \doteq b)\text{»}_{\mathscr{M}} \tag{127}$$

$$\{M_8, M_9\} = \text{«}\neg(x_1 \doteq a) \vee \neg(x \doteq b)\text{»}_{\mathscr{M}} \tag{128}$$

$$\{M_{10}\} = \text{«}\neg(x_1 \doteq a) \vee \neg(x_2 \doteq a)\text{»}_{\mathscr{M}} \tag{129}$$

Thus, there are only 5 2-sets, or 5 genuinely binary relations up to equivalence.

Let us now turn to ternary relations. Put

$$\begin{array}{ll} A := \langle a, a, a\rangle & E := \langle a, b, b\rangle\\ B := \langle a, a, b\rangle & F := \langle b, a, b\rangle\\ C := \langle a, b, a\rangle & G := \langle b, b, a\rangle\\ D := \langle b, a, a\rangle & H := \langle b, b, b\rangle \end{array} \tag{130}$$

[25] More precisely, the quadruple $\langle G, \cdot, ^{-1}, 1\rangle$ is a group, where $\cdot$ is a binary associative operation, x^{-1} the inverse with respect to $\cdot$, and 1 the unit.

There are $2^8 = 256$ subsets. However, these quickly reduce. First, notice that there are six permutations of positions: (), (12), (13), (23), (123), (132). Here is how they permute the triples:

(131)		()	(12)	(13)	(23)	(123)	(132)
	A	*A*	*A*	*A*	*A*	*A*	*A*
	B	*B*	*B*	*D*	*C*	*C*	*B*
	C	*C*	*D*	*C*	*B*	*D*	*D*
	D	*D*	*C*	*B*	*D*	*B*	*C*
	E	*E*	*F*	*G*	*E*	*F*	*G*
	F	*F*	*E*	*F*	*G*	*G*	*E*
	G	*G*	*G*	*E*	*F*	*E*	*F*
	H	*H*	*H*	*H*	*H*	*H*	*H*

We can exclude right away all sets that contain just one triple; this is because in a triple two elements must be equal, to the set is subject to contraction. This leaves us with sets of cardinality 2–7. As it turns out, there are 65 3-sets. To give just one more number: there are 3983 4-sets (out of a total of $2^{16} = 65536$ sets of 4-tuples). The numbers are large, but far smaller than the sets of all relations.

References

Böttner, M., & Thümmel, W. (Eds.). (2000). *Variable-free semantics*, Vol. 3 of *Artikulation und Sprache*. Osnabrück: secolo Verlag.

Bresnan, J., Kaplan, R. M., Peters, S., & Zaenen. A. (1987). Cross–serial dependencies in Dutch. In W. Savitch, E. Bach, W. Marsch, & G. Safran-Naveh (Eds.), *The formal complexity of natural language* (pp. 286–319). Dordrecht: Reidel.

Calcagno, M. (1995). A sign–based extension to the Lambek calculus for discontinuous constituents. *Bulletin of the IGPL, 3*, 555–578.

Cinque, G. (1999). *Adverbs and functional heads. A cross linguistic perspective*. Oxford: Oxford University Press.

Dowty, D. (1979). *Word meaning and montague grammar*. Dordrecht: Reidel.

Dowty, D. R., Wall, R. E., & Peters, S. (1981). *Introduction to Montague semantics*. Number 11 *in Synthese library*. Dordrecht: Reidel.

Dresner, E. (2001). Tarski's restricted form and Neale's quantificational treatment of proper names. *Linguistics and Philosophy, 24*: 405–415.

Ebert, C., & Kracht, M. (2002). Formal syntax and semantics of case stacking languages. In *Proceedings coling 2000*, pp. 250–256.

Evers, A. (1975). *The transformational cycle in Dutch and German*. PhD thesis, University of Utrecht.

Ferreirós, J. (2001). The road to modern logic—an interpretation. *The Bulletin of Symbolic Logic, 7*: 441–484.

Fiengo, R., & May, R. (1994). *Indices and identity*. Number 24 *in Linguistic inquiry monographs*. Cambridge, Massachussetts: MIT Press.

Fine, K. (2000). Neutral relations. *The Philosophical Review, 109*, 1–33.

Fine, K. (2003). The role of variables. *Journal of Philosphy, 50*, 605–631.

Firbas, J. (1992). *Functional sentence perspective in written and spoken communication*. Studies in English language. Cambridge: Cambridge University Press.

Gärtner, H.-M. (2002). *Generalized transformations and beyond*. Akademie Verlag.

Haider, H. (2000a). Branching and discharge. In P. Coopmans, M. Everaert, & J. Grimshaw (Eds.), *Lexical specification and insertion, number* 197 *in Current issues in linguistic theory* (pp. 135–164). Amsterdam: John Benjamins.

Haider, H. (2000b). OV is more basic than VO. In P. Svenonius, (Ed.), *The derivation of VO and OV* (pp. 45–67). Amsterdam: John Benjamins.
Haider, H. (2004). V-Clustering and clause union. *Causes and effects*. In P. Seuren, & G. Kempen, (Eds.), *Verb constructions in German and Dutch* (pp. 91–126). Amsterdam: John Benjamins.
Halle, M., & Marantz, A. (1993). Distributed morphology and the pieces of inflection. In K. Hale & S. J. Keyser (Eds.), *The view from building 20: Essays in Honour of Sylvain Bromberger*, (pp. 111–176). MIT Press.
Heim, I., & Kratzer, A. (1998). *Semantics in generative grammar*. Oxford: Blackwell Publishers.
Huybregts, R. (1984). Overlapping dependencies in Dutch. *Utrecht Working Papers in Linguistics, 1*, 3–40.
Janssen, T. (1997). Compositionality. In J. van Benthem, & A. ter Meulen, (Eds.), *Handbook of logic and language* (pp. 417–473). Amsterdam: Elsevier.
Kamp, H., & Reyle, U. (1993). *From discourse to logic. Introduction to modeltheoretic semantics of natural language, Formal language and discourse representation*. Dordrecht: Kluwer.
Kayne, R. S. (1994). *The antisymmetry of syntax*. Number 25 *in Linguistic inquiry* monographs. MIT Press.
Keenan, E. L., & Faltz, L. L. (1985). *Boolean semantics for natural language*. Dordrecht: Reidel.
Keresztes, L. (1990). *Chrestomathia Mordvinica*. Budapest: Tankönyvkiadó.
Kracht, M. (1999). Agreement morphology, Argument structure and syntax. Manuscript.
Kracht, M. (2003). *The mathematics of language*. Berlin: Mouton de Gruyter.
Kracht, M. (2006). Lectures on interpreted languages and compositionality. Manuscript.
Langacker, R. W. (1987). *Foundations of cognitive grammar* (Vol. 1). Stanford: Stanford University Press.
Levin, B., & Hovav, M. R. (2005). *Argument realization. Research surveys in linguistics*. Cambridge University Press.
Mel'čuk, I. (2000). *Cours de Morphologie Générale. (General Morphology. A Course-book)* (Vol. 1–5). Les Presses de l'Université de Montréal.
Michaelis, J. (2001). *On formal properties of minimalist grammars*. PhD thesis, Universität Potsdam.
Perlmutter, D. M. (1983). *Studies in relational grammar* (Vol. 1). Chicago and London: The Chicago University Press.
Perlmutter, D. M. (1984). *Studies in relational grammar* (Vol. 2). Chicago and London: The Chicago University Press.
Radzinski, D. (1990). Unbounded syntactic copying in Mandarin Chinese. *Linguistics and Philosophy, 13*, 113–127.
Shieber, S. (1985). Evidence against the context–freeness of natural languages. *Linguistics and Philosophy, 8*, 333–343.
Stabler, E. P. (1997). Derivational minimalism. In C. Retoré (Ed.), *Logical aspects of computational linguistics (LACL '96)*, Number 1328 *in Lecture notes in artificial intelligence* (pp. 68–95). Heidelberg: Springer.
Stabler, E. P. (2006a). Tuple pregroup grammars(?). Manuscript, UCLA.
Stabler, E. P. (2006b). Sidewards without copying. In P. Monachesi, G. Penn, G. Satta, & S. Wintner, (Eds.), *Proceedings of the 11th conference on formal grammars*. (pp. 133–146). Stanford: CSLI.
Steedman, M. (2000). *The syntactic process*. Cambridge (Mass.): MIT Press.
Vermeulen, K. F. M. (1994). *Explorations in the dynamic environment*. PhD thesis, Department of Philosophy, University of Utrecht.
Vermeulen, K. F. M. (1995). Merging without mystery or: Variables in dynamic semantics. *Journal of Philosophical Logic, 24*: 405–450.
Vermeulen, K. F. M., & Visser, A. (1996). Dynamic bracketing and discourse representation. *Notre Dame Journal of Formal Logic, 37*, 321–365.
Williamson, T. (1985). Converse relations. *The Philosophical Review, 94*, 249–262.
Zadrozny, W. (1994). From compositional semantics to systematic semantics. *Linguistics and Philosophy, 17*, 329–342.
Zimmermann, T. E. (1999). Meaning postulates and the model–theoretic approach to natural language semantics. *Linguistics and Philosophy, 22*, 529–561.