978-1-107-00539-6 - Lexical Meaning in Context: A Web of Words
Nicholas Asher
Excerpt
More information

PART ONE

FOUNDATIONS

1

Lexical Meaning and Predication

To build a formal model of predication and to express lexical meaning, I will use the lambda calculus. The lambda calculus is the oldest, most expressive, and best understood framework for meaning representation; and its links to various syntactic formalisms have been thoroughly examined from the earliest days of Montague Grammar to recent work like that of de Groote (2001), Frank and van Genabith (2001). Its expressive power will more than suffice for our needs.[1]

The pure lambda calculus, or λ calculus, has a particularly simple syntax. Its language consists of variables together with an abstraction operator λ. The set of terms is closed under the following rules: (1) if v is a variable, then v is a term; (2) if t is a term and v a variable, then λvt is also a term; (3) if t and t' are terms, then *the application* of t to t', $t[t']$, is also a term. We can use this language to analyze the predication involved when we apply a predicate like an intransitive verb to its arguments. The meaning of an intransitive verb like *sleeps* is represented by a lambda term, $\lambda x\, \text{sleep}'(x)$; it is a function of one argument, another term like the constant j for *John* that will replace the λ bound variable x and yield a logical form for a larger unit of meaning under the operation of β reduction. β reduction, also known as β conversion, is a rule for inferring one term from another. β reduction is the formal counterpart in the λ calculus of the informal operation of predication. One can also think of reduction as the rule governing application, and so I shall call it the rule of *Application*.[2] I'll write such a rule in the usual natural deductive format.

[1] There are other formalisms that can be used—for instance, the formalism of attribute value matrices or typed feature structures with unification. This formalism, however, lacks the operation of abstraction, which is crucial for my proposals here.

[2] Besides Application, there are other rules standardly assumed for the λ calculus—for example, α conversion, which ensures the equivalence of bound variables, rules for equality, and the following rules which validate a rule of Substitution that I shall introduce subsequently:

- Application:

$$\frac{\lambda x \phi[\alpha]}{\phi(\frac{\alpha}{x})}$$

The λ calculus as our representational language tells us in principle what our lexical entries should look like. For example, if we decide that a word like *cat* is a one place predicate, then our lexical entry for this word should have the form $\lambda x\, \text{cat}'(x)$, where *cat′* is an expression in our language for logical forms that will, when interpreted, assign the right sort of denotation to the word and contribute to the right sort of truth conditions for sentences containing the word. Of course, there are lots of decisions to be made as to what *cat′* should be exactly, but we will come back to this after we have taken a closer look at predication.

1.1 Types and presuppositions

Sometimes predications go wrong. This is something that lexical semantics has to explain.

(1.1) a. ?That person contains an interesting idea about Freud.
b. That person has an interesting idea about Freud.
c. That book contains an interesting idea about Freud.
d. That person is eating breakfast.
e. That book is red.
f. #That rumor is red.
g. # The number two is red.
h. # The number two is soft.
i. # The number two hit Bill.
j. The number two is prime.
k. John knows which number to call.
l. *John believes which number to call.

The predications in (1.1f,g,h) or (1.1i) are malformed—each contains what Gilbert Ryle would have called a category mistake. Numbers as abstract objects can't have colors or textures or hit people; it's nonsensical in a normal

- $t = t' \rightarrow t[t''] = t'[t'']$
- $t = t' \rightarrow f[t] = f[t']$
- $t = t' \rightarrow \lambda x\, t = \lambda x\, t'$

Church (1936) shows how to encode Boolean functions within the λ calculus, once we have decided on a way of coding up truth functions.

conversation to say something like the number two is red, soft, or that it hit Bill.[3] The mismatch between predicate and argument is even more blatant in (1.1l).

One has to exercise some care in understanding why a predication like (1.1a) sounds so much odder than (1.1b–d). In some sense people can contain information: spies have information that they give to their governments and that counter-spies want to elicit; teachers have information that they impart to their students. But one can't use the form of words in (1.1a) to straightforwardly convey these ideas. The predication is odd; it involves a misuse of the word *contain*. If it succeeds at all in making sense to the listener, it must be subject to reinterpretation.

It's important to distinguish between necessary falsity and the sort of semantic anomaly present in (1.1a) and (1.1f–i). In the history of mathematics, many people, including famous mathematicians, have believed necessarily false things. But competent speakers of a language do not believe propositions expressed by a sentence with a semantically anomalous predication. (1.1a) or (1.2c,d) are semantically anomalous in a way that (1.1b–d) or (1.2a,b) below are not.[4]

(1.2) a. Tigers are animals.
b. Tigers are robots.
c. #Tigers are financial institutions.
d. #Tigers are Zermelo-Frankel sets.

Many philosophers take (1.2a) to be necessarily true and (1.2b) to be necessarily false.[5] Nevertheless, according to most people's intuitions, a competent speaker could entertain or even believe that tigers are robots; he or she could go about trying to figure this out (e.g., by dissecting a tiger). It is much harder to accept the possibility, or even to make sense of, a competent speaker's believing or even entertaining that tigers are literally financial institutions, let alone ZF style sets. Thinking about whether a competent speaker could entertain or believe the proposition expressed by a sentence gives us another means to distinguish between those sentences containing semantically anomalous expressions and those that do not.

[3] As the attentive reader may have already guessed, besides "normal" conversations, there are also "abnormal" discourse contexts—contexts that would enables us to understand these odd sentences in some metaphorical or indirect way, or that even enable us to reset the types of words. More on this later.

[4] Thanks to Dan Korman for the first two examples.

[5] The reason for this has to do with a widely accepted semantics of natural kinds due to Hilary Putnam and Saul Kripke, according to which *tigers* picks out a non-artifactual species in every possible world.

The reason why (1.1a), (1.1f–i) or (1.2c,d) are semantically anomalous, while the other examples above are not, is that there is a conflict between the demands of the predicate for a certain type of argument and the type of its actual argument. People aren't the right type of things to be containers of information, whereas tapes, books, CDs, and so on are. Rumors aren't the right type of things to have colors, and tigers aren't the right type of things to be sets or financial institutions.

We can encode these humdrum observations by moving from the pure lambda calculus to a typed lambda calculus. The reason why some predications involve misuses of words, don't work, or require reinterpretation, is that the types of the arguments don't match the types required by the predicates for their argument places. (1.1a) involves a misuse of the language. *Contain*, given the type of its direct object, requires for its subject argument a certain type of object—a container of information; and persons are of not of this type—they don't contain information the way books, journal articles, pamphlets, CDs, and the like do. On the other hand, there is no such problem with (1.1b); books are the sort of object that are containers of information. (1.1c) is also fine, but that is because the verb *have* doesn't make the same type requirements on its arguments that *contain* does.

The typed lambda calculus, developed by Church (1940), assumes that every term in the language has a particular type. This places an important constraint on the operation of Application. Assume that every term and variable in the lambda calculus is assigned a type by a function TYPE.

- Type Restrictied Application:

$$\frac{\lambda x \phi[\alpha]}{\phi(\frac{\alpha}{x})}$$

provided $\textsc{type}(x) = \textsc{type}(\alpha)$.
$\lambda x \phi[\alpha]$ is undefined, otherwise.

In what follows, I'll encode TYPE with the usual colon notation; $\alpha\colon a$ means that term α has type a.

The *typed lambda calculus* has many pleasant semantic and computational properties. This has made it a favorite tool of compositional semanticists since Montague first applied it in developing his model theoretic notion of meaning in the sixties. I will model predication as type restricted application and lexical entries as typed lambda terms. This will require that each term gets a type in a given predicational context. Moreover, each term will place restrictions on the type of its eventual arguments. The data just discussed indicates that the set of relevant types for a theory of predication and lexical meaning are quite fine-grained; in the next chapter we will see how this data and other data lead to the hypothesis of a great many more types than envisaged in Montague Grammar or standard compositional semantics.

Before addressing questions about types, I want to investigate some implications of Type Restricted Application for a theory of predication. There is a compelling analogy between the way types and type requirements work in the typed lambda calculus and the linguistic phenomenon known as presupposition. Linguists take certain words, phrases, and even constructions to generate presupposed contents as well as "proffered" contents; the latter enter into the proposition a sentence containing such items expresses, whereas the presupposed contents are conceived of as constraints on the context of interpretation of the sentence. For instance, in

(1.3) The dog is hungry

the definite determiner phrase (DP) *the dog* is said to generate a presupposition that there is a salient dog in the discourse context. If such a presupposition is satisfied in a discourse context, the presupposition is said to be *bound*; if it cannot be bound, the presupposition is *accommodated* by making the supposition that the discourse context contains such a salient dog. However, there is a certain cost to such suppositions; if there really is no salient dog in the context, (1.3) is difficult to interpret. Frege and Strawson proposed that in cases where no salient dog can be found, a sentence like (1.3) cannot be literally interpreted and fails to result in a well-formed proposition capable of having a truth value.[6] This view of presupposition, though it has its detractors, is well established in linguistics and has received a good deal of empirical support and formal analysis (Heim (1983), van der Sandt (1992), Beaver (2001)). Type Restricted Application says something very similar to the doctrine of presupposition: a type concordance between predicate and argument is required for coherent interpretation. If an argument in a predication cannot satisfy the type requirements of the predicate, then the predication cannot be interpreted and fails to result in a well formed logical form capable of having a truth value.

There are other similarities between presupposition and type requirements. A common test for presupposition is the so-called projection test: presuppositions "project" out of various operators denoting negation, modality, or mood. So if the type requirements of a predicate are a matter of presupposition, then semantically anomalous sentences like (1.1g,h) should remain anomalous

[6] This is known as the "Frege-Strawson" doctrine.

when embedded under negation, interrogative mood or modal operators. This is indeed the case:

(1.4) a. # The number two could have been red.
b. # Is the number two soft?
c. # The number two didn't hit Bill.[7]

The sentences in (1.4) all convey presuppositions that are absurd and that cannot be met—namely, that the number two is a physical object. Other tests for presuppositions concern the non-redundancy of presupposed content and the inability to make certain discourse continuations on presupposed content.[8] These tests apply to type requirements of predicates as well. It is not redundant to say *the abstract object two is prime* instead of *two is prime*, and it seems impossible to make discourse continuations on the type requirements, since the latter are not even propositional contents. Thus, it seems that the type requirements of predicates provide a kind of presupposed content. I shall call these *type presuppositions*.

Two features of presuppositions will be very important for the study of predication in this book. The first is the variability among terms that generate presuppositions to license accommodation. It is standardly assumed that the adverb *too* generates a presupposition that must be satisfied in the given discourse context by some linguistically expressed or otherwise saliently marked content. Thus, in an "out of the blue" context, it makes no sense to say

(1.6) Kate lives in New York too.

even though as a matter of world knowledge it is clear that the presupposition of *too* in this sentence is satisfied—namely, that there are other people besides Kate who live in New York. Even if the proposition that there are other people besides Kate who live in New York is manifestly true to the audience of (1.6), (1.6) is still awkward, unless the presupposed content has been made salient somehow in the context. The presupposed, typing requirements of the predicates in (1.1) and (1.4) resemble the behavior of the presupposition of *too*; they

[7] A presuppositional view should allow that this sentence has a perfectly fine reading where the negation holds over the type requirements as well. But typically such readings are induced by marked intonation. If this sentence is read with standard assertion prosody, then it is as anomalous as the rest.

[8] The continuation test says that one cannot elaborate or explain or continue a narrative sequence on presupposed content. Thus, one cannot understand the example below as conveying that John regretted that he yelled at his girlfriend and that then after fighting with her he went to have a drink.

(1.5) John regrets that he fought with his girlfriend. Then he went to have a drink.

have to be satisfied in their "predicative" context in order for the sentences containing them to receive a truth value. Accommodation of these type presuppositions is impossible. The sentences that fail to express a coherent proposition capable of having a truth value do so, because the relevant type presuppositions cannot be satisfied, given that the arguments and predicates therein mean what they standardly mean and have the types that they standardly do.

On the other hand, some presupposition introducing phrases like possessive DPs readily submit to accommodation. For instance, *Sylvain's son* presupposes that Sylvain has a son, but this information is readily accommodated into the discourse context when the context does not satisfy the presupposition.

(1.7) Sylvain's son is almost three years old.

Other definite descriptions can be satisfied via complex inferences. The example below, which features a phenomenon known as "bridging," features such an inference; the definite *the engine* is "satisfied" by the presence of a car in the context—the engine is taken to be the engine of the car:

(1.8) I went to start my car. The engine made a funny noise.

In the following chapters we will see cases of type presuppositions that can either be satisfied in complex ways like the bridging cases or can be accommodated via a "rearrangement" or modification of the predicative context, if the latter fails to satisfy the type presuppositions in a straightforward way. Figuring out when presupposed typing requirements can be accommodated and when they cannot will be a central task of this book.

Another important property of presuppositions is their sensitivity to discourse context. For instance, if we embed (1.7) in the consequent of a conditional, the presupposition that projects out from the consequent can be bound in the antecedent and fails to project out further as a presupposition of the whole sentence (1.9):

(1.9) If Sylvain has a son, then Sylvain's son is almost three years old.

A similar phenomenon holds for type presuppositions. Consider (1.4a) embedded as a consequent of the following (admittedly rather strange) counterfactual.

(1.10) If numbers were physical objects, then the number two could have been red.

The presupposition projected out from (1.4a) is here satisfied by the antecedent of the counterfactual and rendered harmless. Thus, category mistakes for the most part must be understood relative to a background, contextually supplied set of types, a background that may itself shift in discourse.

1.2 Different sorts of predication

Having introduced types as part of the apparatus of predication, let me come back to predication itself. I have spoken so far of predication as a single operation of applying a predicate to its arguments. But in fact predication takes many forms in natural languages, some particular to particular languages, others more general. Even among ordinary predications, linguists distinguish between:

- predication of a verb phrase to a subject or a transitive verb to an object
- adjectival modification with different types of adjectives—e.g., evaluative adjectives like *good rock*, *bad violinist*, material adjectives like *bronze statue*, *paper airplane*, and manner adjectives like *fast car*, *slow cigar*
- adverbial modification and modification of a verb phrase with different prepositional phrases or PPs—e.g., the distinction between *load the wagon with hay* and *load the hay on the wagon*.

Beyond these are more exotic forms of predication:

- metaphorical usage (extended predication)

  (1.11) John is a rock.

- restricted predication

  (1.12) John as a banker makes $50K a year but as a plumber he makes only $20K a year.

- copredication

  (1.13) The lunch was delicious but took forever.
  (1.14) The book has a purple cover and is the most intelligible introduction to category theory.
  (1.15) #The bank is rising and specializes in IPOs.

- loose predication

  (1.16) That's a square (pointing to an unpracticed drawing in the sand).

- resultative constructions

  (1.17) a. Kim hammered the metal flat.
  b. * Kim hammered the metal gleaming.

  (1.18) depictives

  a. Pat swims naked.
  b. *Pat cooks hot.

- the genitive construction

(1.19) a. Kim's mother
b. Kim's fish

- noun noun compounds

(1.20) a. lunch counter
b. party favor

Each one of these forms of predication presents its own challenges for lexical and compositional semantics; the lexical theory must assign to the words in these constructions the right sort of meaning and postulate the right sort of composition rules for predication so as to get the right result. In addition, a lexical theory must specify what morphological processes and elements affect meaning and how; it must give those processes and elements a meaning. A lexical theory using the typed lambda calculus can provide the right sort of picture to tackle these issues.

Let's consider these forms of predication in a bit more detail. Loose predication is a difficult and well-known problem in philosophy.[9] But other forms of predication mentioned above, which linguists think also provide challenges for lexical theory, have not received so much philosophical scrutiny or formal analysis. Copredication, for instance, which is a grammatical construction in which two predicates jointly apply to the same argument, has proved a major challenge. Languages, as we shall see in the next chapter, distinguish between events and objects; the predicates that apply the one type do not apply in general to the other type literally. It turns out that some objects, however, are considered both events *and* physical objects in some sense. Consider, for instance, lunches. Lunches can be events but they are also meals and as such physical objects. As a result, *lunch* supports felicitous copredications in which one predicate selects for the event sense of *lunch* while the other selects for the physical object or meal sense.

(1.21) Lunch was delicious but took forever.

It turns out that many words behave like *lunch* in (1.21) and denote objects with multiple senses or aspects. I will call predications like those in (1.21) *aspect selections*, and I will analyze these predications as predications that apply to selected aspects of the object denoted by the surface argument.

In trying to account for instances of copredication that involve aspect selection like (1.21), standard, typed theories of predication and lexical semantics confront some difficult if not unanswerable questions. How can a term have two incompatible types, as is apparently the case here? How can one term

[9] Loose predication is related to vagueness, and vague predication might be considered another form to be studied. But I shall not do that here.