

## 2 Semantic interpretation

---

### 2.1 Introduction

In this chapter, I look at previous approaches to semantics and semantic interpretation in linguistics and NLU. The work I will examine addresses one or more of the following questions:

- What kind of formalism, representation, or model can adequately capture the semantics of a natural language utterance? That is, what is a “semantic object”?
- What kind of process can map a natural language utterance to these semantic objects?
- Can these semantic objects be used in artificial intelligence applications? Once a sentence has been interpreted, how can its meaning be applied by the system to which it was input?

All three questions are of interest to AI researchers, but linguists care only about the first two and philosophers only about the first. We should therefore not be surprised or too disappointed if, in looking at work on semantics outside AI, we find that it stops short of where we wanted to go. Conversely, we may also find that a semantic theory is adequate for AI without satisfying the linguists or philosophers.

It is worth emphasizing the point that our concerns in semantics are not identical to everyone else’s. We can identify two distinct (but obviously related) concerns (*cf.* Winograd 1984):

- The study of abstract meaning and its relation to language and the world.
- The study of how agents understand the meaning of a linguistic utterance in the world.

Too often the label *doing semantics* is used vaguely to mean one or the other of these, or some random, shifting mixture of both, and the word *semantics* is unwittingly used differently by different participants in discussions. In this research, our concern is the second of those above. We will, of course, look at work on the other, and try hard not to get ourselves too confused.

What exactly is it that we are looking for in a semantic theory? We can identify the following properties as desirable:<sup>1</sup>

- The semantic objects, whatever they are, must be amenable to computational manipulation, supporting inference and problem solving.
- The theory must account for the relationship between the meaning of a sentence and the meanings of its components; that is, the theory must be compositional.
- The theory must show what role syntactic structure plays in the meaning of a sentence (Tarnawsky 1982: 113).
- The theory should be able to deal with intensions, opaque contexts, generics, and the like.
- The theory must relate semantics to syntax in such a way as to be able to provide feedback to a parser. (Preferably, the theory should allow the earlier parts of a sentence to be interpreted even before the later parts are seen or heard, as human sentence comprehension clearly has this property.)

Most of these properties are self-explanatory once I have defined the terms; skip to the next section if you know the terms already. An INTENSION may best be thought of as a description, an individual instantiation of which is its EXTENSION. For example, in (2-1), the noun phrase *the president* refers to an intension:

(2-1) The president is elected every four years.

That is, it doesn't follow from (2-1) that if George Washington is the president, then George Washington is elected every four years; (2-1) has a DE DICTO reading. In a different context, the same noun phrase can also refer to an extension, that is, have a DE RE reading:

(2-2) The president was discussing foreign policy with his horse when a White House aide arrived with the bad news.

A sentence can have both an intensional and extensional reading; (2-3) could refer to whoever is president or to George Washington in particular:

(2-3) The president has too much power.

An OPAQUE or OBLIQUE CONTEXT is one in which referential substitution cannot occur. For example, even if Nadia's pet unicorn is a Montague semanticist, we cannot conclude from (2-4):

(2-4) Ross believes that Nadia's pet unicorn is persecuting him.

that (2-5) holds:

(2-5) Ross believes that a Montague semanticist is persecuting him.

<sup>1</sup> Conspicuously missing from the desiderata is the requirement (Tarnawsky 1982) that the theory account for the processes of meaning acquisition and meaning extension. I will have nothing to say on these matters.

as Ross may be unaware of the unicorn's particular field of research.

A **GENERIC** noun phrase is one intended to refer to all members of the class of which it is an instance. Thus (2-6) is intended to be true of all golden tamarins and all cats:

(2-6) The golden tamarin is smaller than a cat, but twice as cute.

By **COMPOSITIONALITY** we mean that the meaning of the whole is a systematic function of the meaning of the parts.<sup>2</sup> At first glance this sounds trivial; if the noun phrase *my pet penguin* denotes by itself some particular entity, namely the one sitting on my lap as I write this paragraph, then we do not expect it to refer to a different entity when it is embedded in the sentence *I love my pet penguin*, and a semantic system that did not reflect this would be a loser indeed. Yet there are alternatives to compositional semantics.

The first alternative is that the meaning of the whole is a function of not just the parts but also the situation in which the sentence is uttered. For example, the possessive in English is highly dependent upon pragmatics; the phrase *my penguin* could refer, in different circumstances, to the penguin that I own, to the one that I am now carrying but don't actually own, or to the one that I just bet on at the penguin races. My definition of semantic interpretation in section 1.1.1 excluded this sort of consideration, but this should not be regarded as uncontroversial.

The second alternative is that the meaning of the whole is not a *systematic* function of the parts in any reasonable sense of the word, but rather that the meaning of an individual word varies idiosyncratically with the other words in the same sentence. We will see an example of a semantic system with this property in section 2.3.1. Generally, non-compositional semantics can get very messy. Maintaining compositionality permits us to apply our semantic techniques, whatever they might be, recursively to a sentence and each of its components, in a uniform (and, we hope, elegant) manner. In section 3.8 we will examine some of the limitations of compositional semantics.

The requirement that the theory be able to interpret the first part of the sentence even before the rest is seen is not strictly necessary. It does, however, accord with our intuition on how people understand sentences (Swinney 1982; Marslen-Wilson and Tyler 1980) (*see section 2.4*). The result of hearing (2-7):

(2-7) Ross gave Nadia the ...

is some kind of mental structure representing Ross giving something to Nadia, and if the sentence broke off completely after the *the*, the hearer could still answer the question *Whom did Ross give something to?* To be able to provide an interpretation

<sup>2</sup>By *parts*, we mean of course syntactically well-formed parts. We do not expect there to be a meaning to the substrings *again Nadia* or *mother's industrial* in sentence (i):

- (i) If Ross brings that sheepdog to our meditation session again, Nadia is going to have to borrow her mother's industrial vacuum cleaner.

of a constituent as soon as it is complete seems also a desirable property for a computer NLU system, especially one that may have to use the context of the first part of the sentence to resolve ambiguity in a later part. A suitably compositional semantic theory should be able to meet this need.

## 2.2 Semantic interpretation and linguistic theory

In this section I look at two semantic theories from linguistics. The first is decompositional semantics, the second model-theoretic semantics.

### 2.2.1 *Decompositional semantics*

Theories of decompositional semantics<sup>3</sup> attempt to represent the meaning of each word by decomposing it into a set of semantic primitives. For example, the word *chair* could be represented as shown in (2-8) (Katz 1972, JD Fodor 1977: 148):

- (2-8) (Object), (Physical), (Non-living), (Artifact),  
 (Furniture), (Portable), (Something with legs),  
 (Something with a back), (Something with a seat),  
 (Seat for one)

In practice, such a lexical decomposition program is problematic. It is extremely difficult, if not perhaps impossible in principle (see Kempson 1977: 96–101 for discussion), to find a suitable, linguistically universal collection of semantically primitive elements in which all words (of all languages) could be decomposed into their necessary properties. For example, not all the properties shown in (2-8) are defining features of a chair, or are even present in all modern chairs, and yet they seem necessary in distinguishing chairs from other seating equipment (JD Fodor 1977: 148). Even simple words whose meanings seem straightforward are extremely difficult to characterize (Winograd 1976). Decompositional semantics is also problematic in its notion of how a sentence is represented (JD Fodor 1977: 160–165). One cannot simply hang the representations of the individual words from the parse tree of a sentence and call the result the meaning, for that would predict that the slightest variation in syntactic structure changes the meaning of the sentence. But a “less structured” approach does not work well either; Katz and JA Fodor (1963) simply strung the word representations together in order thus:

---

<sup>3</sup>I do not attempt to list or catalogue here the many theories that come under the heading of decompositional semantics; perhaps the best-known versions are those of Katz and JA Fodor (1963) and Jackendoff (1972). JD Fodor 1977, Kempson 1977, and Tarnawsky 1982 may be consulted for detailed reviews of various theories. See also Nida 1975 for a detailed example of such a theory.

- (2-9) **The man hits the colorful ball.**  
 [Some contextually definite] → (Physical Object) →  
 (Human) → (Adult) → (Male) → (Action) → (Instancy)  
 → (Intensity) → [Collides with an impact] →  
 [Some contextually definite] → (Physical Object) →  
 (Color) → [[Abounding in contrast and variety  
 of bright colors] [Having globular shape]]

It is by no means clear what to do with a thing like that, but obviously it fails to satisfy the requirement that a semantic theory show the contribution to meaning of a sentence's syntactic structure.

In particular, these problems can be seen to be fatal in any consideration of the theory for use in NLU. Representing a word as a set of primitives is by itself useless when the theory can provide neither a suitable structure for putting them together to represent a sentence, nor methods for deep inference in context upon the resultant structures.<sup>4</sup> There have been attempts in AI to make decompositional semantics usable by adding these missing elements to the theory. Notable among these are the Preference Semantics system of Wilks (1973, 1975a, 1975b, 1975c, 1975d) and the early conceptual dependency representations of Schank (*e.g.*, 1973, 1975); in particular, Schank had a principled system for building his primitives into a structure that represented a sentence and performing inference upon it.<sup>5</sup> I will return to this point in section 2.3.3.

The result of decomposition is usually compost.  
 —Nadia Talent<sup>6</sup>

## 2.2.2 *Model-theoretic and truth-conditional semantics*

In his well-known “PTQ” paper (Montague 1973), Richard Montague presented the complete syntax and semantics for a small fragment of English. Although it was limited in vocabulary and syntactic complexity,<sup>7</sup> Montague's fragment dealt

<sup>4</sup>Nor does decompositional semantics even constitute an adequate semantic theory from a linguistic standpoint, since all that has happened in a decompositional analysis is that one set of symbols has been translated into another set, without any relationship being specified between the world and that new set. For critiques of decompositional semantics see Putnam 1970, Winograd 1976, and Tamawsky 1982.

<sup>5</sup>As Waltz (1982) points out, Schank's set of primitives is incomplete and is unable to capture nuances of meaning; the embedding of conceptual dependency representations in Schank's newer, larger memory structures (*e.g.*, Schank and Abelson 1977, Schank 1982a, 1982b) is of no help. Waltz proposes a decompositional representation intended to capture subtle differences in verb meaning (see also DeJong and Waltz 1983).

<sup>6</sup>Personal communication, 22 February 1983.

<sup>7</sup>Hausser (1984) has significantly extended the fragment, much modifying the system in the process.

with such semantic problems as intensions and opaque contexts, different types of predication with the word *be*, and the “THE TEMPERATURE IS NINETY” PROBLEM.<sup>8</sup> Montague’s formalism is exceedingly complex, and I make no attempt to present it here, discussing rather the formalism’s important theoretical properties. The reader interested in the details will find Dowty, Wall, and Peters 1981 (hereafter *DWP*) a useful introduction.

Montague’s theory is TRUTH-CONDITIONAL and MODEL-THEORETIC. By *truth-conditional* we mean that the meaning of a sentence is the set of necessary and sufficient conditions for the sentence to be TRUE, that is, to correspond to a state of affairs in the world (DWP 1981: 4–6). By *model-theoretic* we mean that the theory uses a formal mathematical model of the world in order to set up relationships between linguistic elements and their meanings. Thus semantic objects will be entities in this model, namely individuals and set-theoretic constructs defined on entities. Since sentences are not limited to statements about the present world as it actually is, Montague employs a set of POSSIBLE WORLDS; the truth of a sentence is then relative to a chosen possible world and point in time (DWP 1981: 10–13). A possible world–time pair is called an INDEX.

Montague takes the word to be the basic unit of meaning, assuming that for each index there is an entity in the model for each word of the language. The same entity could be represented by more than one word, of course: thus at some index, the words *unicorn* and *pigeon* could denote the same set of individuals—in particular, they could both denote the empty set. The converse, an ambiguous word representing different entities in different linguistic contexts but at the same index, was not allowed in Montague’s formalism; this matter is dealt with at length in Part II of this book.

For Montague, then, semantic objects, the results of the semantic translation, are such things as INDIVIDUALS in (the model of) the world, INDIVIDUAL CONCEPTS (which are functions to individuals from the set of indexes), properties of individual concepts, and functions of functions of functions. At the top level, the meaning of a sentence is a truth condition relative to an index. These semantic objects are represented by expressions of an INTENSIONAL LOGIC; that is, instead of translating English directly into these objects, a sentence is first translated to an expression of intensional logic for which, in turn, there exists an interpretation in the model in terms of these semantic objects.

Montague has a strong THEORY OF TYPES for his semantic objects: a set of types that corresponds to types of syntactic constituents. Thus, given a particular syntactic category such as proper noun or adverb, Montague was able to say that the

<sup>8</sup>That is, to ensure that (i):

(i) The temperature is 90 and the temperature is rising.

cannot lead to the inference (ii):

(ii) 90 is rising.

meaning of a constituent of that category is a semantic object of such and such a type.<sup>9</sup> Montague's system of types is recursively defined, with individuals, truth values, and intensions as primitives, and other types defined as functions from one type to another in such a manner that if syntactic category  $X$  is formed by adding category  $Y$  to category  $Z$ , then the type corresponding to  $Z$  is functions from senses of the type of  $Y$  to the type of  $X$ .<sup>10</sup>

Montague's system contains a set of syntactic rules and a set of semantic rules, and the two are in one-to-one correspondence. Each time a particular syntactic rule applies, so does the corresponding semantic rule; while the one operates on some syntactic elements to create a new element, the other operates on the corresponding semantic objects to create a new object that will correspond to the new syntactic element. Thus the two sets of rules operate in TANDEM.

The syntactic rules are a simple categorial (Ajdukiewicz) grammar (see DWP 1981: 182). A typical rule is (2-10), the rule for combining a noun and a determiner to make a noun phrase:

- (2-10) If  $\zeta$  is a noun, then *every*  $\zeta$  and *the*  $\zeta$  are noun phrases, and so is *a*  $\zeta$  or *an*  $\zeta$  according as  $\zeta$  takes *a* or *an*.

The words *every*, *the*, *a*, and *an* are said to be introduced by the rule SYNCATEGOREMATICALLY. Many rules just concatenate constituents:

- (2-11) If  $\delta$  is a sentence-taking verb phrase and  $\beta$  is a sentence, then  $\delta\beta$  is an intransitive verb phrase. [Example: *believe that* + *John loves a unicorn*]

There are three types of semantic rule. The first type is basic rules that just provide translations of most individual words. The second type translates syntactic constituents with syncategorematic words in them. Here is the semantic rule that corresponds to the noun-phrase rule above:

- (2-12) If  $\zeta$  translates into  $\zeta'$ , then *every*  $\zeta$  translates into:  
 $\lambda P [\forall x [\zeta'(x) \Rightarrow P\{x\}]]$ ;  
*the*  $\zeta$  translates into:  
 $\lambda P [\exists y [\forall x [\zeta'(x) \Leftrightarrow x = y] \wedge P\{y\}]]$ ;  
and *a*  $\zeta$  and *an*  $\zeta$  translate into:  
 $\lambda P [\exists x [\zeta'(x) \wedge P\{x\}]]$ ,  
where  $P\{x\}$  denotes the application of the EXTENSION of  $P$  to  $x$ .

(You needn't be worried if this doesn't mean a great deal to you; all you need to notice is that the translation of the noun phrase is a function—in particular, a function in intensional logic—that includes the translation of the noun.) The third type of semantic rule is rules of functional application: the translation of the

<sup>9</sup>To be precise: the semantic type of a proper noun is *set of properties of individual concepts*; that of an adverb is *function between sets of individual concepts* (DWP 1981: 183, 187).

<sup>10</sup>For example, the semantic type of prepositions is *functions mapping senses of the semantic type of noun phrases to the semantic type of prepositional phrases*.

new constituent is formed by functionally applying the translation of one of its components to the other. The rule that corresponds to (2-11) is this:

(2-13) If the translation of  $\delta$  is the function  $\delta'$  and that of  $\beta$  is  $\beta'$ , then the translation of  $\delta\beta$  is  $\delta'(\beta')$ , where  $\beta'$  denotes the intension of  $\beta$ .

Because of the strong typing, the tandem operation of the two non-basic types of rule, and the fact that the output of a semantic rule is always a systematic function of its input, Montague semantics is compositional. (Because verb phrases are generally analyzed right to left, however, many constituents are uninterpreted until the sentence is complete.)

Although Montague semantics has much to recommend it, it is not possible to implement it directly in a practical NLU system, for two reasons. The first is that Montague semantics as currently formulated is computationally impractical. It throws around huge sets, infinite objects, functions of functions, and piles of possible worlds with great abandon. Friedman, Moran, and DS Warren (1978a) point out that in the smallest possible Montague system, one with two entities and two points of reference, there are, for example,  $2^{2^{522}}$  elements in the class of possible denotations of prepositions, each element being a set containing  $2^{512}$  ordered pairs.<sup>11</sup>

The second reason we can't use Montague semantics directly is that truth-conditional semantics is not useful in AI. We are interested not so much in whether a state of affairs is or could be true in some possible world, but rather in the state of affairs itself; thus AI uses KNOWLEDGE-BASE SEMANTICS in which semantic objects tend to be symbols or expressions in a declarative or procedural knowledge-representation system (see section 2.3.2). Moreover, truth-conditional semantics really only deals with declarative sentences (DWP 1981: 13) (though there has been work attempting to extend Montague's work to other types of sentence; e.g., Hamblin 1973); a practical NLU system needs to be able to deal with commands and questions as well as declarative sentences.

There have, however, been attempts to take the intensional logic that Montague uses as an intermediate step in his translations and give it a new interpretation in terms of AI-type semantic objects, thus preserving all other aspects of Montague's approach; see, for example, the paper of Hobbs and Rosenschein (1977) and Smith's (1979) objections to their approach. There has also been interest in using the intensional logic itself (or something similar) as an AI representation<sup>12</sup> (e.g., Moore 1981). But while it may be possible to make limited use of inten-

<sup>11</sup>Despite this problem, Friedman, Moran, and DS Warren (1978b, 1978c) have implemented Montague semantics computationally, using techniques for maintaining partially specified models. However, their system is intended as a tool for understanding Montague semantics better rather than as a usable NLU system (1978b: 26).

<sup>12</sup>Ironically, Montague regarded intensional logic merely as a convenience in specifying his translation, and one that was completely irrelevant to the substance of his semantic theories.

sional logic expressions,<sup>13</sup> there are many problems that need to be solved before intensional logic or other flavors of higher-order logical forms could support the type of inference and problem solving that AI requires of its semantic representations; see Moore 1981 for a useful discussion. Moreover, Gallin (1975) has shown Montague's intensional logic to be incomplete. (See also the discussion of work using logical forms, in section 8.3.1.)

Nevertheless, it is possible to use many aspects of Montague's approach to semantics in AI (*cf.* DS Warren 1985). The semantic interpreter that I will describe in Chapter 3 has several of the properties of Montague semantics that we described above, and I therefore refer to it as "Montague-inspired".

If your thesis is utterly vacuous,  
Use first-order predicate calculus.  
With sufficient formality,  
The sheerest banality  
Will be hailed by all as miraculous.

But for theses you fear indefensible,  
Reach for semantics intensional.  
Your committee will stammer  
Over Montague grammar,  
Not admitting it's incomprehensible.

—Henry Kautz<sup>14</sup>

### 2.3 Semantic interpretation and artificial intelligence

The development of artificial intelligence has necessarily included much research on knowledge-representation formalisms and systems. Two major classes of representation have been used:

- Logical representations: predicate logic, higher-order logic, various forms of intensional logic (*e.g.*, Moore 1981, DS Warren 1983). A knowledge base consists of assertions in the formalism that are known to be true.
- Knowledge structures: semantic nets, frames, scripts, etc. (*e.g.*, Charniak 1976, Schank and Abelson 1977). A knowledge base consists of a set of data objects in a structure that denotes relationships between the objects.

The two classes are not antithetical; indeed, predicate calculus is isomorphic to a simple semantic network or frame system. Representations such as Frail, described in section 1.3.1, attempt a synthesis that provides the advantages of both classes. (Note, however, my remarks in the previous section about higher-order logic as an AI representation.)

<sup>13</sup>Godden (1981) in fact uses them for simple translation between Thai and English.

<sup>14</sup>Personal communication.

Regardless of the representation used, a knowledge base can be thought of as a model of a world in exactly the sense used in model-theoretic semantics (*see section 2.2.2*).<sup>15</sup> That is, it provides a way of deciding the truth of a statement about the world that it represents: a true statement is one that is represented in the knowledge base or is provable from it, and a false statement is one whose negation is true. In practice, of course, incompleteness of the knowledge base or Gödelian omeriness will make some statements undecidable.

In this section I look at semantic theories from artificial intelligence. I will use three theories as representatives of AI: procedural semantics, knowledge semantics, and object-oriented semantics. The theories differ in the degree to which the entities that they use are INTERPRETED. Thus, the theories represent three points on a spectrum; many AI systems may be considered hybrids that fall elsewhere on the line.<sup>16</sup> I will be arguing that things are best at the object-oriented end (the interpreted end) of the line.

### 2.3.1 Procedural semantics and Woods's semantic interpreter

Research on semantic interpretation in artificial intelligence goes back to William Woods's dissertation (1967, 1968),<sup>17</sup> which introduced PROCEDURAL SEMANTICS to NLU in a natural-language front-end for an airline reservation system. Input sentences were translated into procedure calls that retrieved information from a database, and the meaning of a sentence was identified with the corresponding procedure call. For example:

- (2-14) AA-57 is non-stop from Boston to Chicago.  
`equal (numstops (aa-57, boston, chicago), 0)`
- (2-15) They serve breakfast on flight AA-57.  
`mealserv (aa-57, breakfast)`
- (2-16) Every flight that leaves Boston is a jet.  
`(for every X1/flight: depart (X1, boston); jet (X1))`
- (2-17) What is the departure time of AA-57 from Boston?  
`list (dtime (aa-57, boston))`
- (2-18) Does AA-57 belong to American Airlines?  
`test (equal (owner (aa-57), american-airlines))`

<sup>15</sup>This is not to say that the knowledge base will necessarily meet the formal requirements of a model for any particular model-theoretic system, such as Montague's.

<sup>16</sup>This spectrum is, in principle, independent of the two classes of representation mentioned above. In practice, however, frame systems tend to be at the interpreted end of the line, and logic systems closer to the other end.

<sup>17</sup>See section 1.2 of Woods 1967 for the relevant prehistory. See Simmons and Burger 1968 and Wilks 1968 for approaches contemporary to Woods's.

Thus the representation is essentially first-order. Note that the system accepted only questions for retrieval from the database, not declarative sentences for updating it, a point that will be important later. Nevertheless, for simplicity I follow Woods's style of showing most example sentences in a "raw" declarative form, rather than an interrogative form. The representation of the interrogative form was the declarative form given as an argument to the procedures *test* (for *yes/no* questions) or *list* (for *wh*- questions), which formatted the output accordingly. For example, in (2-18), the procedure *equal* will check for the equality of its arguments, and return *true* or *false*; the procedure *test* will then present the result to the user as *yes* or *no*.

Woods's system had rules with patterns that, when they matched part of the parsed input sentence, contributed a string to the semantic representation of the sentence. This string was usually constructed from the terminals of the matched parse tree fragment. The strings were combined to form the procedure call that, when evaluated, retrieved the appropriate database information. The rules were mostly rather ad hoc; they looked for certain key items in the parse tree, and their output was quite unconstrained. For example, the rule that applied to (2-14) was this:

```
(2-19)  1-(G1: flight((1)) and (2) = be) and
        2-(G4: (1) = non-stop) and
        3-(G3: (1) = from and place((2))) and
        4-(G3: (1) = to and place((2)))
        => equal (numstops (1-1, 3-2, 4-2), 0)
```

G1, G3, and G4 are the names of partial parse-tree patterns; for example, G1 is the partial S tree with an NP labeled (1) and a VP with a verb labeled (2). The first line of the rule thus matches a sentence that starts with a noun phrase that is a **flight** and whose verb is *be*. The subsequent lines require an adjective phrase *non-stop* and two prepositional phrases, one with *from* and a **place**, the other with *to* and a **place**. If these conditions are satisfied, then the string specified in the last line of the rule results. In the string, items of the form *x-y* are replaced by the *y*th node of the pattern matched on line *x* of the rule; thus 3-2 will be replaced by the place name in the *from* PP. Note that the word *non-stop* does not appear in the output at all; rather, it has been syncategorematically changed into *equal (numstops (...), 0)* by the rule. The verb of the sentence does not appear in the output either, not even in disguise.

We can identify two major shortcomings of this system: as the above example suggests, it is ad hoc and non-compositional. Table 2.1 shows how the interpretation of the verb *depart* varies as different prepositional phrases are attached to it (Woods 1967: A-43–A-46).<sup>18</sup> This variation is possible because the rules are both

<sup>18</sup>I have simplified a little here in order to make my point. In fact, sentences like those with prepositional phrases in table 2.1 would actually cause the execution of two semantic rules: one for the

very specific and very powerful. For example, rule (2-19) could not handle sentence (2-14) if its verb were changed to a synonym, even though the verb itself is not used in the output:

(2-20) AA-57 flies non-stop from Boston to Chicago.

(2-21) AA-57 goes non-stop from Boston to Chicago.

The rule could, of course, be extended to look for these other verbs as well as *be*, but the point is that the system is inherently unable to handle such synonymy except by exhaustively listing synonyms in each rule in which they might occur. And (2-19) is also tied to a very particular sentence structure; separate rules would be needed for paraphrases:

(2-22) AA-57 doesn't stop between Boston and Chicago.

Moreover, the output is not tied in any way to the input; a rule can ignore any or all of its input, or make syncategorematic changes that are quite inconsistent with those of other rules.<sup>19</sup>

Non-compositionality was necessitated by the particular set of primitives that Woods used, selected for being "atomic" concepts in the domain of discourse (1967:7-4-7-11) rather than for promoting compositionality.<sup>20</sup> Woods's semantics could probably be made reasonably compositional by appropriate adjustment of the procedure calls into which sentences are translated. However, the system would still not be compositional BY DESIGN, and it would be easy to inadvertently lose compositionality again when extending the system. The problem is that the rules are too powerful.

Adding an ability to update the database would also be antithetical to compositionality in the system, for then either the meaning of a procedure would have to vary with context, or the translation of the whole sentence would have to vary with sentence form. To see the problem, consider the sentence (2-23):

(2-23) AA-57 is non-stop from Boston to Chicago.

---

complete sentence, and one for the subsentence it happens to contain, *AA-57 departs from Boston*. The resulting interpretation would be the conjunction of the output from each rule (Woods 1967:9-5):

(i) AA-57 departs from Boston to Chicago.

(ii) depart (aa-57, boston) and connect (aa-57, boston, chicago)

Woods left it open (1967:9-7) as to how the semantic redundancy in such expressions should be handled, but one of his suggestions is a filter that would remove conjuncts implied by others, giving, in this case, the interpretations shown in table 2.1.

<sup>19</sup>Nor is there anything preventing the construction of rules that would result in conjunctions with conflicting, rather than merely redundant, terms.

<sup>20</sup>DS Warren (1983) points out that a first-order representation, such as Woods's, is inadequate in principle if both compositionality and a suitable typing are to be maintained.

Table 2.1. *Noncompositionality in Woods's system*


---



---

AA-57 departs from Boston. depart (aa-57, boston)
AA-57 departs from Boston to Chicago. connect (aa-57, boston, chicago)
AA-57 departs from Boston on Monday. dday (aa-57, boston, monday)
AA-57 departs from Boston at 8:00 a.m. equal (dtime (aa-57, boston), 8:00am)
AA-57 departs from Boston after 8:00 a.m. greater (dtime (aa-57, boston), 8:00am)
AA-57 departs from Boston before 8:00 a.m. greater (8:00am, dtime (aa-57, boston))

---



---

Previously, I said that the “raw” meaning of this sentence was (2-24):

(2-24) equal (numstops (aa-57, boston, chicago), 0)

and that therefore the meaning of the interrogative form, (2-25):

(2-25) Is AA-57 non-stop from Boston to Chicago?

is (2-26):

(2-26) test (equal (numstops (aa-57, boston, chicago), 0))

But if we are to allow sentence (2-23) as input to modify the database, we have to think more carefully about its translation. Possibilities include its “raw” form, (2-24), and a form more analogous to (2-26), such as (2-27):

(2-27) assert (equal (numstops (aa-57, boston, chicago), 0))

But in either case, the meaning of equal has suddenly changed; instead of being a predicate, it is now an assignment statement. The alternative is to translate (2-26) into an entirely different procedure call:

(2-28) make-equal (numstops (aa-57, boston, chicago), 0)

We must thus choose from two unpalatable situations: one is to say that equal shall somehow be sensitive to the context in which it is called and adjust its behavior accordingly (a dubious idea both semantically and from a programming viewpoint); the other is to double the number of rules and predicates needed in

the system, having one set for interrogative forms, another for declaratives, again defeating compositionality.<sup>21</sup>

Another problem with Woods's approach is that semantic interpretation necessarily occurs after the parsing of the sentence is complete, and so the interpretation of the first part of the sentence is not yet available to aid the parser if a structural ambiguity arises later in the sentence.<sup>22</sup> Some later versions (*e.g.*, that of Woods, Kaplan, and Nash-Webber 1972) had the parser keep all the information necessary to back up and look for a different parse if the first one found turned out to be semantically unacceptable (*see section 6.3.1*).

The status of procedural semantics itself as a theory of semantics has been a matter of considerable controversy (Woods 1975, Johnson-Laird 1977, JA Fodor 1978, Johnson-Laird 1978, JA Fodor 1979, Woods 1981, Wilks 1982a). There are many variations, but the gist of procedural semantics as a semantic theory is that the meaning of a sentence is the procedure into which the sentence is compiled, either in the computer or in the mind; the procedure itself can be seen as the intension of the sentence, and the result of the execution as the extension. (Woods himself would not agree with this; he argues that truth-conditionality must also play a role (Woods 1981).)

The notion of a procedure is so basic to computation that procedural semantics seems a very natural approach for AI, and it has been used in many systems, including the well-known LUNAR natural language system (Woods, Kaplan, and Nash-Webber 1972). Since its original incarnation it has been refined considerably and is still today perhaps the predominant approach, despite its problems and its essentially ad hoc nature. However, in its pure form as described above, procedural semantics is not adequate for AI, because the procedures themselves do not have an adequate interpretation and the items they manipulate are uninterpreted symbols. This is not a difficulty if one is just inserting or retrieving database values with little or no interpretation,<sup>23</sup> but if one is interested in maintaining and manipulating a knowledge base, performing inference, solving problems, and the like, procedural semantics suffers from the same problem as decompositional se-

<sup>21</sup>It may be possible to circumvent this problem by the use of a Prolog-like database language; such a language would have the same procedural-declarative ambiguity, but would resolve the ambiguity in context:

(i) `:- equal (x, y) .`

(ii) `equal (x, y) :- .`

<sup>22</sup>In addition, because the interpretation of the sentence itself proceeds bottom-up but not left to right, the resolution of intrasentence reference is problematic; see Hirst 1981a[1]: 36–37.

<sup>23</sup>The construction of natural language interfaces to databases whose contents have little meaning to the interface is, of course, still an important area of research. The best interfaces, such as TEAM (Archbold, Grosz, and Sagalowicz 1981; Grosz 1983; Martin, Appelt, and Pereira 1983), have a vocabulary that lets them talk about database fields with words other than the fields' literal names.

manatics: symbols have been translated into other symbols, but the new symbols are scarcely easier to deal with than the old ones.

### 2.3.2 Knowledge semantics

The AI knowledge base-centered view of semantics has been called KNOWLEDGE SEMANTICS by Tarnawsky (1982). It is Tarnawsky's view that "the meaning of a sentence depends on the knowledge of the interpreter" (1982: ii) and includes the propositions, possibly infinite in number,<sup>24</sup> entailed by the sentence with respect to that knowledge. Tarnawsky formalized a semantic theory in which a knowledge base played a central role. In his approach, semantic interpretation takes as its input surface structures enriched with Chomskyan traces (*e.g.*, Chomsky 1975) and with anaphoric references resolved. Transformational rules then map a surface structure to a statement in higher-order predicate logic, which is added to the knowledge base.<sup>25</sup>

To the extent that each assumes a database in which the system's knowledge is stored, there is an obvious similarity between knowledge semantics and procedural semantics. There are two major differences between the two. First, in knowledge semantics it is the statement added to the knowledge base and the consequences of its addition, rather than the request to add it, that is taken to be the meaning of the sentence.<sup>26</sup> The second is in the attitude toward that knowledge base. Procedural semantics places no semantic burden on the content of the knowledge base; rather, the claim is that the meaning of a sentence inheres in the database manipulation procedures into which it is translated. Knowledge semantics, on the other hand, construes the content of the database, and the inferences generated by the addition of a new item, as the entities in which meaning inheres. The problem of differing representations for declarative and interrogative forms does not arise in knowledge semantics, as the process that translates sentences into knowledge base entries is reasonably compositional—compositional at the SYMBOL-MANIPULATION level, that is; knowledge semantics does not, however, make the MEANING of the whole a systematic function of the meaning of the parts, because it ascribes no meaning to the parts. This is because the knowledge base provides a semantic model for sentences but not for the components of the sentences. For example (Tarnawsky 1982: 159–160):

<sup>24</sup>In AI, of course, the number of inferences that result from the addition of a sentence to the knowledge base is finite, and, further, systems carefully restrict the inferences that may be made. Just what should be inferred is still a matter of controversy, however; see Charniak 1976[1].

<sup>25</sup>Thus, again, the entire sentence must be present before semantic interpretation commences.

<sup>26</sup>Interestingly, questions are not taken as requests to retrieve information from the knowledge base. Rather, it is simply predicated that the sentence is interrogative; that a reply might be appropriate is a possible, but not necessary, entailment of the predication (Tarnawsky 1982: 226–230). This reflects the fact that not all questions are really requests for information.

- (2-29) The man ate the kutya.  
ate (the (man), the (kutya))

The semantic interpretation is composed of the same words as the input sentence; it has been restructured, but the symbols themselves have not in any sense been interpreted. The knowledge base may or may not know what *kutya* is, that is it may or may not contain other assertions about *kutya*, but it will nowhere contain anything that represents *kutya* per se. Likewise, constituents such as prepositional phrases cannot be represented, for if the meaning of a sentence includes the inferences it entails in the knowledge base, then what could be the meaning, the inferences entailed, from a fragment such as *with a knife*, even if mapped by the system into a representation such as *with (a (knife))*? There is no semantic object per se in the system that represents *with a knife* and that can combine compositionally with that of a constituent such as *John cut*. Tarnawsky claims (personal communication) that the system could be extended to handle subsentence-level constituents, but I am skeptical of this claim.

Because symbols have no direct interpretation in the knowledge base, the burden of resolving coreferential expressions is placed upon the inference mechanism. Consider, for example, the style often used in news reports:

- (2-30) MONTREAL—Hard-line separatist Guy Bertrand is pulling out all the stops in his bid to topple his opponents and become the new leader of the Parti Québécois.

The fiery Quebec City lawyer yesterday unveiled a detailed plan of his vision of an independent Quebec that would start with close economic ties to the United States and a new currency for Quebec—the U.S. dollar.<sup>27</sup>

If this text is to be understood, then the second sentence must be recognized as an expansion of the first, with coreference relations between *Guy Bertrand* and *the fiery Quebec City lawyer*, and between *pulling out all the stops . . .* and *unveil[ing] a detailed plan . . .*. This may be done by profligate inference. Any time something is predicated true of *the fiery Quebec City lawyer*, it is inferred to be true of *Guy Bertrand*, of *Mr Bertrand*, of *the dark horse of the PQ leadership race*, and so on. But an inference must be generated for each of the large if not infinite number of different ways an entity in the predicate can be described, an effect that is multiplicative if there is more than one entity in the predicate. This was not a concern for Tarnawsky, who was interested in creating a competence theory, but has obvious problems in any direct computer implementation.<sup>28</sup>

Knowledge semantics is also weak in dealing with case relationships, and makes an artificial distinction between cases flagged by syntactic position and those flagged by a preposition (*section 1.1.2*). Consider these examples:<sup>29</sup>

<sup>27</sup>*The Globe and Mail*, 14 August 1985.

<sup>28</sup>See also section 8.2.2 for another diatribe against uninterpreted symbols.

<sup>29</sup>The first is based on Tarnawsky's structure for *The man loves the girl* (1982: 172); the second is given by Tarnawsky (1982: 183).

(2-31) John hit the nail.  
hit (John, the (nail))

(2-32) John hit the nail with a hammer.  
with (hit) (John, the (nail), a (hammer))

The case flag *with* seems to become a function that operates on the predicate *hit* to change its expected argument structure. Syntactic position seems to be represented by argument position, although syntactic position does not in English unambiguously determine the case it flags; *e.g.*, the system will erroneously ascribe *The hammer hit the nail* a structure similar to (2-31). Tarnawsky (personal communication) believes case to be purely a surface or syntactic phenomenon, and that inference in the knowledge base, rather than case structure in the input predicate, is what counts. Again, this position may be acceptable for a competence theory, but does not seem tenable in a computer implementation.

### 2.3.3 Object-oriented semantics

We saw in the previous section that Tarnawsky's theory uses higher-order predicate logic as a representation, and thus represents sentences but not the entities to which they refer. That is, the logical statements of the knowledge base may be regarded as interpreted, in that there is a process that generates valid inferences from them, but the components of the statements are still uninterpreted symbols.

We can solve this problem by using a suitable knowledge structure, such as semantic nets or frames (*see sections 1.2 and 1.3.1*), for our representation. The uninterpreted symbols of the logical statements can then be replaced by references to the appropriate piece of the knowledge structure—let's assume it will be a frame. For example, we replace the string *kutya* in the present representation with a pointer to the frame that represents **kutya**. A lexicon maps words and phrases to their frame representations; synonyms are mapped to the same representation. The pointer for the *kutya* frame may happen to be the string *kutya*, but this is coincidental; the semantics of this pointer are quite different from those of the similar string in knowledge semantics. If the word *kutya* is represented by a pointer to the **kutya** frame, the frame then provides an interpretation for it and permits access to whatever is known about *kutya*. Similarly, the frame system gives a method for determining a unique referent for a definite reference. Thus *Nadia, the teacher*, and *my aunt's brother's chaquette* can all be mapped to, and thus interpreted by, the same frame.

A suitable knowledge-structure system thus provides a firmer basis for semantic interpretation than procedural semantics and knowledge semantics do. In particular, it provides a deeper level of interpretation and a more adequate account of reference. It is this approach that I will develop in chapter 3.

## 2.4 Psycholinguistic research on semantic interpretation

I will touch only briefly upon psycholinguistic research on semantic interpretation. Matters of psychological reality are not immediately relevant to most of the issues discussed in this chapter, and, unlike lexical disambiguation (*section 4.3*), there are as yet few psycholinguistic data to support an AI approach to semantic interpretation motivated by psychological reality.

However, one aspect of semantic interpretation in which psychological data are relevant is the relationship between syntax and semantics. Are they psychologically separate processes? If so, what is the nature of their interaction?

As we have already observed (*section 2.1*), it is intuitively obvious that people, unlike LUNAR (*section 2.3.1*) or knowledge semantics (*section 2.3.2*), do not delay interpreting a sentence until after they have finished parsing it. Rather, semantic interpretation seems to run either in tandem with or closely behind the word-by-word input of the sentence, and semantic and pragmatic information from the earlier parts of the sentence is used to help interpret the later parts (Marslen-Wilson and Tyler 1980). The general approach is “do it as early as possible”. Experiments by Marslen-Wilson and Tyler (1980; Tyler and Marslen-Wilson 1982) suggest the reality of separate lexical, syntactic and semantic, and pragmatic knowledge, though the reality of separate processes to use each knowledge source does not necessarily follow (*cf.* footnote 4 of chapter 1).<sup>30</sup> Marslen-Wilson and Tyler suggest a model with BOTTOM-UP PRIORITY, that is, a model in which the lower levels of knowledge have priority over the higher levels. “The system allows for top-down effects in the loose sense that contextual information affects the recognition process. But it does not allow for top-down effects in the more precise sense of the term,” in which likely inputs are pre-selected even before the sensory data have been received (Marslen-Wilson and Tyler 1980: 62). In a similar vein, Stanovich (1980, 1984) has argued that purely top-down or bottom-up models cannot explain experimental data on individual differences in reading skills.

The high degree of uncertainty in present psycholinguistic results makes it premature to base AI models of semantic interpretation on psychological data. However, it is clear that an AI semantic interpreter should, at least, follow the principles of “do it as early as possible” and bottom-up priority.

I always get buggered by the bottom-up approach.  
—Rogatien “Gatemouth” Cumberbatch<sup>31</sup>

<sup>30</sup>See Forster 1979 for evidence for the autonomy of lexical, syntactic, and semantic processing. Bever (1970) and Reimold (1976) suggest a different division of serial autonomous processes.

<sup>31</sup>While presenting a paper at the first national conference of the Canadian Society for the Computational Studies of Intelligence / Société canadienne pour l'étude de l'intelligence par ordinateur, Vancouver, 26 August 1976.

## 2.5 Qualities desirable in a semantic interpreter

With the discussion of the previous sections in mind, we now review exactly what it is that we desire in our own semantic interpreter (*cf. section 2.1*).

**1. Compositionality.** Compositionality is clearly a desideratum. We want each syntactically well-formed component of a sentence to correspond to a semantic object, and we want that object to retain its identity even when it forms part of a larger semantic object.

**2. Semantic objects.** We saw in section 2.3.3 the advantages of a knowledge-structure representation. If we adopt such a representation—in particular, a frame representation (*see sections 1.2 and 1.3.1*)—we can then take the elements of the system—frames, slots, statements, etc.—as our semantic objects.

**3. Not ad hoc.** One of the goals of this work is to reduce ad hoc-ness in semantic interpretation, so the next requirement is that the system be elegant and without the unnecessary power in which such messiness can purulate. The semantic rules or formalism should be able to manipulate semantic objects and build new ones, but the rules should not be able to mangle the semantic objects (jeopardizing compositionality), and each should be general and well-motivated. The rules must also be able to take into account the contribution of a sentence's syntactic structure to its meaning.

**4. Feedback for the parser.** We would like the interpreter to work in parallel with the parser, in order to be able to give it the feedback necessary for structural disambiguation, and we require that the representation of the partially interpreted sentence always be a well-formed semantic object in order that it be used in this way.

**5. Lexical ambiguity.** It must be possible for ambiguous words to be assigned a unique sense in the representation, and for this to happen as soon after the occurrence of the word as possible.

**6. Semantic complexities.** The semantic interpreter should be able to deal with all the complexities of semantics that Montague and others have dealt with. These include such things as intension, opaque contexts, generics, complex quantification, and so on.

In Chapter 3, I will describe Absity, a semantic interpreter constructed with these desiderata in mind. It will fulfill the first five of them, and provide a basis for future work on the sixth.

Like everybody who is not in love, he imagined that one chose the person whom one loved after endless deliberations . . . on the strength of various qualities and advantages.  
—Marcel Proust<sup>32</sup>

<sup>32</sup>*Remembrance of things past*. 1913.