Semantic Interpretation and Ambiguity

Graeme Hirst
Department of Computer Science, University of Toronto, Toronto, Canada M5S 1A4

Recommended by M. Palmer and J. Hobbs

ABSTRACT

A new approach to semantic interpretation in natural language understanding is described, together with mechanisms for both lexical and structural disambiguation that work in concert with the semantic interpreter.

ABSITY, the system described, is a Montague-inspired semantic interpreter. Like Montague formalisms, its semantics is compositional by design and is strongly typed, with semantic rules in one-to-one correspondence with the meaning-affecting rules of a Marcus parser. The Montague semantic objects—functors and truth conditions—are replaced with elements of the frame language FRAIL. ABSITY's partial results are always well-formed FRAIL objects.

A semantic interpreter must be able to provide feedback to the parser to help it handle structural ambiguities. In ABSITY, this is done by the "Semantic Enquiry Desk," a process that answers the parser's questions on semantic preferences. Disambiguation of word senses and of case slots is done by a set of procedures, one per word or slot, each of which determines the word or slot's correct sense, in cooperation with the other procedures.

It is from the fact that partial results are always well-formed semantic objects that the system gains much of its power. This, in turn, comes from the strict correspondence between syntax and semantics in ABSITY. The result is a foundation for semantic interpretation superior to previous approaches.

1. Introduction

Research in semantic interpretation has not kept pace with research in other subfields of natural language understanding (NLU). Parsing, knowledge representation, and pragmatics are all much better understood than they were in the early 1970s. The predominant paradigm for semantic interpretation, however, has remained procedural semantics, first devised by Woods in 1967 [67, 68], and developed in the well-known LUNAR system [69].

Artificial Intelligence **34** (1988) 131–177
0004-3702/88/$3.50

The goal of the present research is to place semantic interpretation on a firmer foundation, one that is less ad hoc and more theoretically motivated, enabling one to deal with the semantic complexities of natural language. I believe that ABSITY, the system to be presented below, provides such a foundation; in particular, I will show that ABSITY is a suitable basis for both lexical and structural disambiguation.

In Section 2, I will describe this approach to semantic interpretation. In Section 3, I describe the Polaroid Word system for lexical disambiguation, and in Section 4 the Semantic Enquiry Desk for structural disambiguation.

2. Semantic Interpretation

By *semantic interpretation* we mean the process of mapping a syntactically analyzed text of natural language to a representation of its meaning. The input to a semantic interpreter is a parse tree, but we do not require that it represent a complete sentence; we allow well-formed subtrees such as noun phrases and even single words (labeled with their syntactic category and features) as input. The output of a semantic interpreter is the literal meaning of the input text, or a suitable representation thereof.

We exclude from semantic interpretation all aspects of syntactic analysis; rather, we assume that a parser performs morphological and syntactic analysis upon an input text before it is passed to the semantic interpreter. This is not in any sense a logical necessity; systems have been built in which syntactic analysis and semantic interpretation have been completely integrated—e.g., [3, 4, 30, 31, 51–53]. However, this approach becomes very messy when complex syntactic constructions are considered. Keeping syntactic and semantic analysis separate is well motivated merely by basic computer science principles of modularity. Moreover, it is our observation that those who argue for the integration of syntactic and semantic processing are usually disparaging the role of syntax (e.g., Small [57, p. 12]), a position that I reject and which has been found to be unworkable [43, 46, 59] and, probably, psychologically unreal [44, pp. 62–66]. This is not to say that parsing is possible without semantic help, or that parsing need be finished before semantics starts; on the contrary, there are many situations, such as prepositional phrase and relative clause attachment, in which the parser must call on semantic knowledge. In this paper I will show that syntax and semantics may work together well and yet remain distinct modules.[1]

I will exclude from semantic interpretation any consideration of discourse pragmatics; rather, discourse pragmatics operate upon the output of the

[1] Mellish [46] suggests the possibility of a system in which separate syntactic and semantic modules are automatically compiled into an efficient unified system. Hendler and Phillips [27] suggest object-oriented computing to achieve this goal.

semantic interpreter. Thus, semantic interpretation does not include the resolution in context of anaphors or definite reference, or of deictic or indexical expressions, or the recognition and comprehension of speech acts, irony and sarcasm, metaphor, or other nonliteral meanings. These exclusions should not be thought of as entirely uncontroversial; while few would advocate making speech act interpretation part of semantic interpretation, Moore [48] argues that definite reference resolution, as well as certain "local" pragmatic matters, must be resolved during semantic interpretation.

An important concept that I shall consider is *compositionality*. By compositionality I mean that the meaning of the whole is a systematic function of the meaning of the parts; a semantic theory with compositionality accounts for the relationship between the meaning of a sentence and the meanings of its components. This is not as trivial as it at first sounds; in the next section we shall see an example of a noncompositional semantic theory.

2.1. Two approaches to semantics

2.1.1. *Procedural semantics*

Woods' dissertation [67, 68] 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:

AA-57 is nonstop from Boston to Chicago.
equal (numstops (aa-57, boston, chicago), 0) (1)

Every flight that leaves Boston is a jet.
(for every X1/flight: depart (X1, boston); jet (X1)) (2)

What is the departure time of AA-57 from Boston?
list (dtime (aa-57, boston)) (3)

Does AA-57 belong to American Airlines?
test (equal (owner (aa-57), american-airlines)) (4)

The interpretation was based on production 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 and not fully compositional; they looked for certain key items in the parse tree, and their output was quite unconstrained—a word's contribution to the output could

vary wildly with the presence or absence of other words or phrases:

AA-57 departs from Boston.
depart (aa-57, boston) (5)

AA-57 departs from Boston on Monday.
dday (aa-57, boston, monday) (6)

AA-57 departs from Boston at 8:00 a.m.
equal (dtime (aa-57, boston), 8:00am) (7)

This variation was possible because the rules were both very specific and very powerful. For example, the rule for sentence (1) would not apply if the verb of the sentence were changed to a synonym, even though the verb itself is not used in the output. The rule could, of course, be extended to look for other verbs as well as *be*, but the point is that the system was inherently unable to handle such synonymy except by exhaustively listing synonyms in each rule in which they might occur. And the rule was also tied to a particular sentence structure; separate rules were needed for paraphrases. Moreover, the output was not bound in any way to the input; a rule could ignore any or all of its input, or make changes that were quite inconsistent with those of other rules.

Noncompositionality was necessitated by the particular set of primitives that Woods used, selected for being "atomic" concepts in the domain of discourse [67, pp. 7-4–7-11] rather than for promoting compositionality.[2] Woods' 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.[3]

Another problem with Woods' approach was that semantic interpretation necessarily occurred after the parsing of the sentence was complete, and so the interpretation of the early part of the sentence was not yet available to aid the parser if a structural ambiguity arose later in the sentence.[4] Some later versions (e.g., that of Woods, Kaplan, and Nash-Webber [69]) 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.

Since its original incarnation, procedural semantics has been refined consid-

[2] Warren [61] points out that a first-order representation, such as Woods', is inadequate in principle if both compositionality and a suitable typing are to be maintained.

[3] 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. See [35] for discussion.

[4] In addition, because the interpretation of the sentence itself proceeds bottom-up but not left to right, the resolution of intrasentence reference is problematic [32, pp. 36–37].

erably. However, in its pure form as described above it 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, but if one is interested in maintaining and manipulating a knowledge base, performing inference, solving problems and the like, procedural semantics suffers from the problem that symbols are translated into other symbols, but the new symbols are scarcely easier to deal with than the old ones.

2.1.2. *Montague semantics*

In his well-known "PTQ" paper [47], Richard Montague presented the complete syntax and semantics for a small fragment of English. Although it was limited in vocabulary and syntactic complexity, Montague's fragment dealt with such semantic problems as intensions and opaque contexts, different types of predication with the word *be*, and the "*the temperature is ninety* problem".[5] 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 [15] 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 correspond to a state of affairs in the world [15, pp. 4–6], that is, to be *true*. 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 [15, pp. 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 (possibly the empty set) for each word of the language; the same entity could be represented by more than one word, of course. 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 in Section 3 of this paper.

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 higher- and higher-level functions. At the top level,

[5] That is, to ensure that *The temperature is ninety, and the temperature is rising* cannot lead to the inference *ninety is rising*.

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 interpreting English directly with 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, one can say that the meaning of a constituent in that category is a semantic object of such and such a type; for example, the semantic type of a proper noun is set of properties of individual concepts [15, p. 187]. 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.

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*.

Because of the strong typing, the tandem operation of the two nonbasic 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. 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 [18].[6]

The second reason we cannot 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* [60]

[6] Despite this problem, Friedman, Moran, and Warren [19, 20] 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 [19, p. 26].

in which semantic objects tend to be symbols or expressions in a declarative or procedural knowledge representation system. Moreover, truth-conditional semantics really only deals with declarative sentences [15, p. 13] (though there has been work attempting to extend Montague's work to other types of sentence); 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 [37] and Smith's objections to their approach [58]. There has also been interest in using the intensional logic itself (or something similar) as an AI representation[7] (e.g., [48]). But while it may be possible to make limited use of intensional logic expressions, 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 [48] for a useful discussion. Moreover, Gallin [21] has shown Montague's intensional logic to be incomplete.

Nevertheless, it is possible to use many aspects of Montague's approach to semantics in AI. The semantic interpreter that I will present below has several of the desirable properties of Montague semantics that I described above.

2.2. Desiderata for a semantic interpreter

Let's now consider exactly what it is that we desire in a semantic interpreter.

(1) *Compositionality*. Compositionality is clearly a desideratum. We would like 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 would like to use a conventional AI representation for semantic objects, so that they can be used for retrieval, inference, problem solving, and the like.

(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 develop. 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 to take into account the contribution of a sentence's syntactic structure to its meaning.

[7] 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.

(4) *Feedback for the parser about structural ambiguity.* We would like an 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 should 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.

2.3. The ABSITY semantic interpreter

In this section, I describe the ABSITY semantic interpreter.[8] The design of ABSITY uses ideas from Montague semantics to avoid problems of procedural semantics, and to fulfill most of the desiderata of the previous section.

ABSITY is part of the artificial intelligence research project at Brown University. It uses the project's representation language, FRAIL [11], and one of the project's parsers, PARAGRAM [9], a deterministic limited-lookahead parser based on Marcus' well-known PARSIFAL [42]. PARAGRAM differs from conventional Marcus parsers in that it distinguishes base phrase structure rules from transformational rules. ABSITY takes its input from PARAGRAM, and produces output in FRAIL to be sent to the knowledge base. It relies upon FRAIL for knowledge retrieval and inference.

FRAIL is a representation with two "faces"—it can be viewed either as a conventional frame-like system [5], or as a first-order logic representation [6]. I take the first of these views in the discussion below. Table 1 shows the kinds of object that FRAIL uses. The terms *frame* and *slot* may be understood in the conventional way; a *frame determiner* is a frame retrieval function,[9] and a *frame statement* is a complete assertion to FRAIL, which evaluates to an *instance*. Question marks denote variables. The reader should not be misled by the syntactic similarity of frames and frame determiners; the former is a static data structure and the latter is a function.

The implementation to be described is necessarily dependent upon the nature of the other project components, as are many aspects of ABSITY's design. Nevertheless, the design has been kept as independent as possible of

[8] ABSITY: A Better Semantic Interpreter Than Yours.

[9] The semantics of the various frame determiners correspond closely to those of English determiners. Thus the frame determiner the takes into account concepts of focus and uniqueness in the manner of the English singular *the*.

TABLE 1
Types in the FRAIL frame language

Basic types:	Frame (penguin ?x), (love ?x)
	Slot color, agent
	Frame determiner (the ?x), (a ?x)
Other types:	Slot/filler pair = slot + frame statement (color = red), (agent = (the ?x (fish ?x)))
	Frame descriptor = frame + zero or more slot/filler pairs (penguin ?x (owner = Nadia)), (love ?x (agent = Ross) (patient = Nadia)), (dog ?x)
	Frame statement or instance = frame determiner + frame descriptor (the ?x (penguin ?x (owner = Nadia))), (a ?x (love ?x (agent = Ross) (patient = Nadia))) (the ?x (dog ?x)) penguin87 [an instance]

the representation formalism and the parser. The main ideas in ABSITY should be usable with other representations that have a suitable notion of semantic object and also, in particular, with other parsers, deterministic or otherwise.

2.3.1. *Two helpful strategies: Strong typing and tandem processing*

In the design of ABSITY, we will make use of two features of Montague's approach: a strong typing of semantic objects, and running syntax and semantics in tandem. Like Montague, we will impose upon the semantic objects of the FRAIL representation a typing that corresponds to the categories of syntax. We can then put our syntactic and semantic construction rules in one-to-one correspondence, so that when the parser constructs a new constituent with some particular syntactic rule, the corresponding semantic rule can be invoked to make the object that will serve as that constituent's representation.

Note, however, that we cannot simply have one semantic rule for *each* syntactic rule, as Montague did. Montague's syntax was very simple, and each of its rules was in fact a *construction* rule. But we cannot assume that this property is true of parsers in general—indeed, it couldn't be true of any but the simplest parser, as constituent movement must be accounted for. We need, therefore, to determine which of our parser's rules or actions are construction rules and thus require a corresponding semantic construction rule. In the grammar of the PARAGRAM parser, the set of construction rules turns out to be exactly the set of base rules. This should not be surprising; in many transforma-

tional theories of grammar, including at least one of Chomsky's [12, p. 132], base rules are those that determine the deep structure and the sentence's meaning, while transformations can only make changes to the syntactic structure that do not affect the meaning.

A suitable typing for FRAIL is shown in Table 2. The items in the right-hand column are types of FRAIL objects, as described above. The correspondence shown in the table satisfies the requirements imposed by our application of rules in tandem. For example, a preposition corresponds to a slot, a noun phrase to a frame statement, and their combination, a prepositional phrase, corresponds to a slot/filler pair, as required. A detailed description of the types and a more complete justification for this particular typing may be found in [35].

By adopting these two strategies, we have implicitly satisfied our requirement that semantic processing be able to provide feedback to the parser. By having the semantic interpreter proceed in lockstep with the parser, rule-for-rule, and by ensuring that the representation of a partially interpreted sentence is always a well-formed semantic object, we ensure that the fullest possible amount of semantic information is always available for the parser to use.

2.3.2. *Another strategy: Turning meaningful syntax into words*

In Table 2, I showed that prepositions, which flag cases of the verb,[10]

TABLE 2
Type correspondences in ABSITY

Syntactic type	Semantic type
Sentence	Frame statement, instance
Sentence body	Frame descriptor
Proper noun phrase	Frame statement, instance
Pronoun	Frame statement, instance
Common noun	Frame
Adjective	Slot/filler pair
Determiner	Frame determiner
Noun phrase	Frame statement, instance
Preposition	Slot name
Prepositional phrase	Slot/filler pair
Conjunction	Slot name
Subordinate clause	Slot/filler pair
Verb	(Action) frame
Adverb	Slot/filler pair
Auxiliary	Slot/filler pair
Verb phrase	Frame descriptor
Clause end	Frame determiner

[10] I assume a conventional case system.

correspond to FRAIL slots.[11] But cases can also be flagged by syntactic position—by the fact of the filler being in the subject, object, or indirect object position of the sentence. Clearly, we need to have in ABSITY semantic objects—slots—that correspond to these *positional* case flags. Above we tacitly assumed that semantic objects corresponded only to syntactic objects, and including syntactic position seems to work against our goal of compositionality—the meaning of the whole would be more than a function of the meaning of the parts. Further, it seems to require us to complicate our semantic rules so that they can deal with the contribution of syntactic position, and thus appears to threaten our strong typing and tandem processing.

There is, however, an easy way around the problem—*pretend* that syntactic positions are words just like the real words of the sentence. In fact, we will carry out the pretense to the extent that we will require the parser to *insert* these pretend words into the sentence, and ABSITY to then treat them as if they had been in there all along. We will use three such words: *SUBJ*, *OBJ*, and *INDOBJ*, corresponding to the syntactic positions of subject, object, and indirect object. Further, since they are case flags, we will deem our new words to be prepositions; we will sometimes call them *pseudo-prepositions* to distinguish them from "natural" prepositions.[12]

So, for example, a sentence like (8):

> Nadia gave Ross a pewter centipede for his birthday, because she knew that he didn't have one already. (8)

will become, and will be parsed as, (9):

> SUBJ Nadia gave INDOBJ Ross OBJ a pewter centipede for his birthday, because SUBJ she knew OBJ that SUBJ he didn't have OBJ one already. (9)

2.3.3. *The semantic rules of ABSITY*

The semantic rules of ABSITY follow immediately from the discussion in the preceding sections. Tandem processing means that each rule will simply combine two or more semantic objects to make a new one. What the objects are will be given by the corresponding syntactic rule, their types will be as given by Table 2, and they will be combined in the manner appropriate for the types.

Some examples will make the semantic interpreter clearer. First, let's consider a simple noun phrase, *the book*. From Table 2, the semantic type for

[11] It is a feature of ABSITY's uniformity that prepositions flagging noun phrase modifiers also correspond to slots.

[12] The base rule for a sentence will thus be modified slightly; instead of S→NP VP and VP→V [NP] [NP] PP*, we will have S→PP VP and VP→V PP*.

the determiner *the* is a frame determiner function, in this case (the ?x), and the type for the noun *book* is a kind of frame, here (book ?y). These are combined in the canonical way—the frame name is added as an argument to the frame determiner function and the variables are bound—and the result, (the ?x (book ?x)), is a FRAIL frame statement (which evaluates to an instance) that represents the unique book referred to.[13]

Next, consider *the red book*. A descriptive adjective corresponds to a slot/filler pair; so, for example, *red* is represented by (color = red), where color is the name of a slot and red is a frame instance, the name of a frame. A slot/filler pair can be added as an argument to a frame, so *the red book* would have the semantic interpretation (the ?x (book ?x (color = red))).

Now let's consider a complete sentence:

Nadia bought the book from a store in the mall. (10)

Table 3 shows the representation for each component of the sentence; note that in the table the basic noun phrases have already been formed in the manner described above. Also, we have inserted the pseudo-prepositional subject and object markers *SUBJ* and *OBJ*, and represent the clause end with a period. For simplicity, we assume that each word is unambiguous (disambiguation procedures are discussed in Section 3); we also ignore the tense of the verb. Table 4 shows the next four stages in the interpretation. First, noun phrases and their prepositions are combined into prepositional phrases; their semantic objects form slot/filler pairs. Then, second, the prepositional phrase *in the mall* can be attached to *a store* (since a noun phrase, being a frame, can have a slot/filler pair added to it), and the prepositional phrase *from a store in the mall* is formed. The third stage shown in the table is the attachment of the

TABLE 3
ABSITY example (Part I)

Word or phrase	Semantic object
SUBJ	agent
Nadia	(the ?x (person ?x (propername = "Nadia")))
bought	(buy ?x)
OBJ	patient
the book	(the ?y (book ?y))
from	source
a store	(a ?z (store ?z))
in	location
the mall	(the ?w (mall ?w))
. [period]	(a ?u)

[13] It is the responsibility of FRAIL to determine, with the help of the discourse focus, which one of the books that it may know about is the correct one in context.

TABLE 4
ABSITY example (Part II)

SUBJ Nadia
(agent = (the ?x (person ?x (propername = "Nadia"))))

OBJ the book
(patient = (the ?y (book ?y)))

in the mall
(location = (the ?w (mall ?w)))

a store in the mall
(a ?z (store ?z (location = (the ?w (mall ?w)))))

from a store in the mall
(source = (a ?z (store ?z (location = (the ?w (mall ?w))))))

Nadia bought the book from a store in the mall
(buy ?u (agent = (the ?x (person ?x (propername = "Nadia"))))
(patient = (the ?y (book ?y)))
(source = (a ?z (store ?z (location = (the ?w (mall ?w)))))))

Nadia bought the book from a store in the mall.
(a ?u (buy ?u
(agent = (the ?x (person ?x (propername = "Nadia"))))
(patient = (the ?y (book ?y)))
(source = (a ?z (store ?z (location = (the ?w (mall ?w))))))))

slot/filler pairs for the three top-level prepositional phrases to the frame representing the verb. Finally, the period, which is translated as a frame determiner function, causes instantiation of the buy frame, and the translation is complete.

Our next examples show how ABSITY translates *yes/no* and *wh-* questions. We will use interrogative forms of the previous example:

Did Nadia buy the book from a store in the mall? (11)

What did Nadia buy from a store in the mall? (12)

Sentence (11) has almost the same parse as the previous example, and hence almost the same translation. The only difference is that the frame determiner for the clause is now (question ?x), the translation of *?*, instead of (a ?x); thus the translation is (13):

```
(question ?u
  (buy ?u
    (agent = (the ?x (person ?x (propername = "Nadia"))))
    (patient = (the ?y (book ?y)))
    (source = (a ?z (store ?z (location = (the ?w (mall ?w)))))))))
```
(13)

In a complete NLU system, it would be the responsibility of the language

generator to take the result of this FRAIL call, which will be either nil or the matching instance or instances, and turn it into a suitable English reply, such as *No, she didn't.*

For the *wh-* question, (12), we make use of the fact that (question ?x) returns the bindings of its free variables. The translation will be (14):

```
(question ?u
  (buy ?u
    (agent = (the ?x (person ?x (propername = "Nadia"))))
    (patient = ?WH)
    (source = (a ?z (store ?z (location = (the ?w (mall ?w)))))))))          (14)
```

Notice that *what* has been translated as the free variable ?WH. As before, the call will return either nil or an instance list, but in the latter case the list will include the bindings found for ?WH, i.e., the book that Nadia bought. A reply generator would use the bindings to compose a suitable answer, such as *She bought "The Joy of Socks."*

2.4. ABSITY as the fulfillment of our dreams

In Section 2.2, we listed in six categories the various qualities that we desired in a semantic interpreter. It is now time to pass judgment upon ABSITY with respect to these qualities. We will find that we have numbered the desiderata so that ABSITY meets five of the six.

(1) *Compositionality.* ABSITY is nothing if not compositional. Its semantic rules do little more than combine objects to make new ones, and have no power to ignore or modify semantic objects. In fact, as we shall see below when discussing its shortcomings, ABSITY is, if anything, a little too compositional.

(2) *Semantic objects.* The frames of FRAIL have been suitable semantic objects (but see Section 2.5). (The fact that we are using a structured knowledge base, and not just a database, will be crucial for the disambiguation methods of Sections 3 and 4.)

(3) *Not ad hoc.* The rules of ABSITY meet our requirement that they be clean and general and not mangle semantic objects (cf. point (1) above). By using pseudo-words, ABSITY allows the rules to also be sensitive to the contributions to meaning of syntax.

(4) *Feedback for the parser.* ABSITY is able to provide feedback by running in parallel—a fortiori, in tandem—with the parser, with its partial results always being well-formed semantic objects. I will show in Section 4 how this property may be used in structural disambiguation.

(5) *Lexical ambiguity.* I will show in Section 3 how ABSITY supports lexical disambiguation.

(6) *Semantic complexities*. This is the requirement that is as yet unfilled by ABSITY, which, as a new system, has not been developed to the point of perfection. In the next section, I describe a couple of the complexities of semantics that ABSITY cannot handle and show that the prospects for overcoming these defects are good.

2.5. What ABSITY cannot do yet

Although it has the virtues enumerated in the previous section, ABSITY still falls short of being "the answer to the semantics problem." It does, however, provide a base upon which "a more nearly perfect semantic interpreter" can be built, and this is why I refer to it [34] as a "foundation for semantic interpretation." In this section, I discuss a couple of the ways in which ABSITY is not satisfactory and how its inadequacies may one day be cured.

I will put some of the blame for ABSITY's deficiencies on FRAIL. FRAIL, as an extensional first-order representation, is inadequate for the representation of all of the concepts that natural languages such as English can express, for which a higher-order representation seems necessary [61]. (One such representation is the higher-order modal temporal intensional logic used by Montague [47].) ABSITY, therefore, occasionally finds itself embarrassed by not having a suitable representation available to it for certain constructs of English. However, in many cases, even if FRAIL were to be improved, it would not be straightforward to amend ABSITY to take advantage of it.

2.5.1. *Intensional contexts*

Intensions, obviously, are the first item in the list of things that cannot be represented in a purely extensional formalism. For example:

> Nadia talked about unicorns. (15)

This sentence is ambiguous: it has an extensional *de re* meaning, in which it says that there are some particular unicorns about which Nadia talked:

> "I met Belinda and Kennapod, the unicorns that live behind the laundromat, as I was walking home tonight," said Nadia. (16)

and an intensional *de dicto* meaning, in which it simply says that unicorns were the topic of discussion:

> "I wonder whether I shall ever meet a unicorn," Nadia mused. (17)

It cannot be inferred from the de dicto reading that any unicorns exist at all,

only that the idea or *intension* of a unicorn does. This distinction is lost upon FRAIL, which could only represent the extensional meaning, making the inference that there was a particular set of unicorns of which Nadia spoke.

The reason for this problem is that although FRAIL contains intensions, namely generic frames, its deductive section supports only reasoning with instances of frames; it can reason only about individuals, not abstract descriptions.

There has been little work as yet in AI on intensional representations. As soon as first-order representations are left in favor of representations such as the typed lambda-calculus, major problems, such as a lack of decidability, immediately occur [61]. For preliminary work on the topic see [40, 41, 61]. For an attempt at a first-order solution, see [45].

Since FRAIL does not provide the necessary support, ABSITY makes no attempt to handle intensions or opaque contexts. But even if intensions could be represented, they would be difficult for ABSITY, because, ironically, it is too compositional: the noun phrase rules always take an NP to refer to an extension, and once it is so construed, none of ABSITY's higher-level rules have the power to change it.

Exactly how this might be fixed would depend on the particular intensional representation. It is reasonable, however, to assume a representation like Montague's intensional logic [47] with an operator that converts an intension to an extension at a given index. ABSITY might then treat all NPs as intensional, but add to each a flag that indicates whether the extension operator should be applied to the NP's representation when it is evaluated by the frame language. This flag could be a pseudo-word, and be "disambiguated" to either the extension operator or the identity operator by the same lexical disambiguation procedures described in Section 3 for other words. An alternative that might be possible with some representations is to conflate this flag with the determiner of the NP. Thus, for example, the *a* of *a unicorn* would be regarded as an ambiguous word that could map to either an intensional or extensional frame determiner. Whether this is possible would depend on the relationship between frame determiners and intensionality in the representation.

2.5.2. *Nonrestrictive noun phrase modifiers*

ABSITY is not able to handle nonrestrictive modifiers, whether appositive or not:

> Ross, whose balloon had now deflated completely, began to cry. (18)

> Ross in a bad mood is a sight to be seen. (19)

Ideally, (18) should be translated into two separate FRAIL statements representing the two sentences from which it is composed:

Ross' balloon had now deflated completely.
Ross began to cry. (20)

but ABSITY has as yet no mechanism for this. Sentence (19) is problematic because it is unclear exactly how the NP *Ross in a bad mood* should be represented in FRAIL.

2.5.3. *Other deficiencies*

ABSITY is limited in its dealings with *habitual actions*, certain kinds of *predication*, *complex quantifiers*, *inherent vagueness*, *time and space*, *moral and contingent obligation*, *negation*, *conjunction*, and *nondescriptive noun modifiers*. Many of these are difficult unsolved problems in natural language understanding. A discussion of the problems and some suggestions for their solution may be found in [35].

2.6. Examples

Table 5 shows some of the sentences that ABSITY can handle, and some that it cannot. (Some of the examples assume the disambiguation mechanisms of Sections 3 and 4.)

3. Resolving Word and Case Ambiguities

I now turn to showing how ABSITY can handle ambiguity. This section deals with lexical and case ambiguity, and Section 4 with structural ambiguity.

3.1. What is necessary for resolving lexical ambiguity?

The problem of determining the correct sense of a lexically ambiguous word in context has often been seen as one primarily of context recognition, a word being disambiguated to the unique meaning appropriate to the frame or script representing the known or newly established context. For example, in the well-known SAM program [14, 54, 55], each script has associated with it a set of word meanings appropriate to that script; in the restaurant script, there will be unique meanings given for such words as *waiter* and *serve*, and when (21) is processed:

The waiter served the lasagna. (21)

the fact that *serve* has quite a different meaning in the tennis script will not even be noticed [54, p. 183].

In its most simple-minded form, the script approach can easily fail:

The lawyer stopped at the bar for a drink. (22)

TABLE 5
What ABSITY can and cannot do

Sentences that can be interpreted
Nadia gave Ross a marmot for his birthday.
The fish that Ross loved loved the fish that Nadia loved.
Ross promised to study astrology.
Nadia wanted the penguin to catch some fish.
What did Ross eat for breakfast?
Did a computer destroy the world?
Ross knows that Nadia assembled the plane.
A seagull landed on the beach.
What does Nadia want from Ross?
An obnoxious multi-national corporation wants to hire Nadia.
Sentences that cannot be interpreted
Nadia resembles a pika. (*de dicto reading*) *No representation of intensions.*
Ross sleeps on the floor. (*habitual reading*) *No representation of habitual activities.*
All but five of the students whose fathers like cheese gave three peaches to many of the tourists. *No complex determiners or quantification.*
Ross ought to swim home tomorrow. *No representation of contingent obligation.*
Ross in a bad mood should be avoided. *No representation for problematic NP modifier.*

If only the lawyering script is active, then *bar* as an establishment purveying alcoholic beverages by the glass will be unavailable, and its legal sense will be incorrectly chosen. If resolution is delayed until the word *drink* has had a chance to bring in a more suitable script, then there will be the problem of deciding which script to choose the sense from; as Charniak [10] points out, it is reasonable to expect in practice to have fifty or more scripts simultaneously active, each with its own lexicon, necessitating extensive search and choice among alternatives. Thus the main advantage of the script approach, having disambiguation "fall out" of context recognition, is lost.

Further, even in a single script, a word, especially a polysemous one, may still be ambiguous. In the lawyering script, the word *bar* still has about seven distinct possible meanings, including the physical bar of a courtroom or the legal profession. Moreover, choosing which script to invoke in the first place may require resolving the meaning of an ambiguous word.

In general, word sense can depend not only upon global context, but also (or only) upon local cues, such as selectional restrictions upon fillers of case slots, syntax, and the meaning of nearby words. For example, the various meanings of the verb *keep* may be distinguished by the syntactic form of the verb complement taken by each: *keep quiet; keep cats; keep singing.* In (23):

Nadia's car is a lemon-colored Subaru. (23)

the badly made car meaning of *lemon* can be rejected in favor of the citrus fruit meaning, without any consideration of global context; all one has to do is look at the word with which it has been hyphenated, knowing that color is salient and constant for only one of *lemon*'s two meanings. Often, all that seems necessary for disambiguation is that there be a "semantic association" between one sense of the ambiguous word and nearby words:

The dog's bark woke me up.
[*bark* ≠ surface covering of tree] (24)

As Hayes [25, p. 43] puts it, "an association between a sense of an ambiguous word and the context surrounding a use of that word is strong evidence that the interpretation of that word should come through that sense." The nearby disambiguating words may themselves be ambiguous; a well-known example [57] is *deep pit*. The word *deep* can mean profound or extending far down and *pit* can be fruit stone or hole in the ground; however, only one meaning of each fits with the other, so they are mutually disambiguating.

While restrictions on what may fill a case slot can provide disambiguating information, often there is an ambiguity as to which case is being flagged. For example, the preposition *with* can flag cases such as manner, accompanier, and instrument, each with its own set of restrictions. Thus the case and its filler may form a mutually disambiguating pair, just like *deep pit* above. Our strategy in ABSITY of inserting pseudo-prepositions to represent case-flagging syntactic positions allows us to regard all case flags, be they prepositions or (in the original sentence) syntactic positions as words to be disambiguated by the same process as content words.

It sometimes happens, however, that high-level inference, a relatively expensive operation, will be necessary. An example (from [25]):

Nadia swung the hammer at the nail, and the head flew off. (25)

The word *head* is interpreted by most informants as the hammer head, not the nail head (or Nadia's person head), but figuring this out requires inference about the reasons for which one head or the other might have flown off, with a knowledge of centrifugal force suggesting the head of the hammer.

Necessary for word sense disambiguation, then, are:
- a knowledge of context;
- a mechanism to find associations between nearby words;
- a mechanism to handle syntactic disambiguation cues;
- a mechanism to handle selectional restriction reconciliation negotiations between ambiguous words; and
- inference, as a last resort.

3.2. Polaroid Words

In the description of ABSITY in Section 2, I made the unrealistic assumption that each word and pseudo-word corresponds to the same unique semantic object whenever and wherever it occurs; that is, I assumed there to be no lexical ambiguity and no case flag ambiguity. I now remove this assumption, and develop a method for disambiguating words and case flags within the framework of ABSITY, finding the correct semantic object for an ambiguous lexeme.

Given ABSITY's heritage, the obvious thing to do first is see how Montague handled lexical ambiguity in his PTQ formalism. It turns out, however, that Montague had nothing to say on the matter. His PTQ fragment assumes that there is a unique semantic object for each lexeme. Nor does Montague explicitly use case flags. The verbs of the fragment are all treated as one-place or two-place functions, and syntactic position in the sentence distinguishes the arguments. Nevertheless, there is an easy opening in the formalism where we may deal with lexical ambiguity: except for a few special words, Montague's formalism does not specify where the translation of a word comes from; rather, there is just assumed to be a function g that maps a word α to its translation, or semantic object, $g(\alpha)$, and as long as $g(\alpha)$ is of the correct semantic type, it doesn't really matter how g does its mapping. This means that if we can "hide" disambiguation inside g, we need make no change to the formalism itself to deal with lexical ambiguity in PTQ.

Moreover, we can do exactly the same thing in ABSITY. ABSITY, like the PTQ formalism, does not put any constraints on the lexicon look-up process that associates a word, pseudo-word, or case flag with its corresponding semantic object. If this process could disambiguate each lexeme before returning the semantic object to ABSITY, then no change would have to be made to ABSITY itself to deal with ambiguity; disambiguation would be completely transparent to it.

There is, however, an immediate catch in this scheme: often a word cannot be disambiguated until well after its occurrence, whereas ABSITY wants its semantic object as soon as the word appears. But this is easily fixed. What we shall do is give ABSITY a *temporary* semantic object, with the promise that in due course it shall be replaced by the real thing. The temporary object can be labeled with everything that ABSITY needs to know about the object, that is, with the word itself and its FRAIL type (which is readily determined). ABSITY can build its semantic structure with this, and when the real object is available, it can just be slipped in where the temporary is. We will do this thus: the temporary objects that we shall give ABSITY will be self-developing Polaroid[14]

[14] *Polaroid* is a trademark of the Polaroid Corporation for its range of photographic and other products. It is used here to emphasize the metaphor of a self-developing semantic object, and the system described herein carries no association with, or endorsement by, the Polaroid Corporation.

"photographs" of the semantic objects, and our promise shall be that by the time the sentence is complete, the photograph will be a fully developed "picture" of the desired semantic object. And even as the picture develops, ABSITY will be able to manipulate the photograph, build it into a structure, and indeed do everything with it that it could do with a fully developed photograph, except look at the final picture. Moreover, like real Polaroid photographs, these will have the property that as development takes place, the partly developed picture will be viewable and usable in its degraded form. That is, just as one can look at a partly developed Polaroid picture and determine whether it is a picture of a person or a mountain range, but perhaps not which person or which mountain range, so it will be possible to look at our *Polaroid Words* and get a partial idea of what the semantic object it shows looks like. (This point will also be important in Section 4, when we describe the Semantic Enquiry Desk.)

3.3. Marker passing

Earlier, we saw the importance of semantic associations between words in lexical disambiguation: an association between one sense of an ambiguous word and other words in the sentence or context can be an important disambiguation cue. Psycholinguistic research on lexical disambiguation has shown that semantic priming—that is, the previous occurrence of an associated word—speeds up disambiguation in people, and may lead the retrieval process straight to the correct meaning[15] [56].

Polaroid Words therefore need a mechanism that will allow them to find semantic associations. One such mechanism was that of Hayes' CSAW system [25, 26], which imposed upon a semantic network a frame system with ISA and PART-OF hierarchies in order to detect associations. Our mechanism will be similar but more general; we will use *marker passing* in our FRAIL knowledge base.

Marker passing can be thought of as passing tags or markers along the arcs of the knowledge base, from frame to frame, from slot to filler, under the rules to be discussed below. It is a discrete computational analogue of the *spreading activation* models often used in psychological models of memory (e.g., [1]). The network of frames corresponds to the mental conceptual and lexical network, with each connection implying a semantic relationship of some kind between its two nodes. Passing a marker from one node to another corresponds to activating the receiving node. If marker passing is breadth-first from the starting point (new markers being created if a node wishes to pass to two or more other nodes simultaneously), then marker passing will "spread" much as spreading activation does.

[15] Or perhaps straight to an incorrect meaning, if the semantic prime is misleading. For example, most people find *The astronomer married the star* to be a *semantic garden-path* sentence, reading *star* as astronomical object instead of celebrity. For discussion, see [35, 36].

Marker passing was first used in AI by Quillian [49, 50], who used it to find connections between concepts in a semantic network. Marker passing is, of course, expensive when the net is interestingly large. Fahlman, who used it for deduction in his NETL system [16], proposed super-parallel hardware for marker passing. Although our scheme is much simpler than Fahlman's, we too assume that hardware of the future will, like people of the present, be able to derive connections between concepts in parallel, and that the serial implementation to be described below is only an interim measure.

The frame language FRAIL contains a built-in marker passer (MP for short) that operates upon the FRAIL knowledge base.[16] The MP is called with the name of a node (a frame, slot, or instance) as its argument, to use as a starting point. From this origin, it marks all nodes in the knowledge base that participate in assertions that also contain the origin; these can include slots, slot restrictions, and ISA relationships. For example, suppose the origin is to be the frame that describes airplanes:

```
[frame:  airplane
   isa:  vehicle
 slots:  (owner (airline))
         (type (airplane-type))
         ...]                                    (26)
```

Markers would be placed on vehicle, owner, airline, type, airplane-type, and so on.

Once all the nodes reachable in one step from the origin are marked, each node reachable from these nodes—that is, each node two steps from the origin—is marked in the same way. Thus marker passing proceeds, fanning out from the origin until all nodes whose distance is n or less from the origin have been marked, where n defaults to 5 if the programmer doesn't specify otherwise.[17]

If at any time during marker passing the MP comes upon a node already marked by a previous call, then a *path* (or *chain*) has been found between the origin node of the present call and that of a previous call. It is also possible that the origin itself has been marked by a previous call to the MP, resulting in an instantly discovered path. We call such paths *immediate paths* to distinguish them from *constructed paths* in which the intersection occurs at a third node.

When marking is finished, the MP returns a list of any paths it found. The user may, at any time, clean the markers from all nodes in the knowledge base.

3.3.1. *Lexical disambiguation with marker passing*

In this section, I give a very simple example of lexical disambiguation in which

[16] Research is proceeding in other applications for marker passing in FRAIL besides those discussed here. These include context recognition, discovering causal connections, and problem solving [7, 8, 28, 29].

[17] This is a very arbitrary threshold; see [36] for discussion of problems of threshold setting.

Polaroid Words need only the FRAIL marker passer.

The marker passer operates independently of ABSITY and in parallel with it. That is, following only basic morphological analysis, the input sentence goes to both the PARAGRAM parser and the MP, both of which separately grind away on each word as it comes in. Suppose the input is (27), with the ambiguities underlined:

Nadia's plane taxied to the terminal. (27)

As the words come in from left to right, the MP passes markers from the frames representing each known meaning of each content word in the sentence (including unambiguous ones such as *Nadia*). In (27), immediate paths will be found between the frames airplane and airport-building, which were starting points for *plane* and *terminal*, and between airplane and aircraft-ground-travel (*plane* and *taxi*), indicating that the corresponding meanings of *plane*, *terminal*, and *taxi* should be chosen. (A path will also be found between airport-building and aircraft-ground-travel, but this adds no new information.) Markers will also be passed from the frames representing the other meanings of *plane*, *taxi*, and *terminal*, namely wood-smoother, taxicab, and computer-terminal, but these paths will go off into the wilderness and never connect with any of the other paths.

3.3.2. *Constraining marker passing*

Since marker passing is a blind and mindless process, it is clear that many paths in the knowledge base will be marked besides the ones that provide useful disambiguating information. In fact, if the MP gets too carried away, it will eventually mark everything in the knowledge base (as any node can be reached from any other) and then find paths between the wrong senses of ambiguous words as well as between the right senses. For example, a connection could be found between airplane and computer-terminal simply by passing markers up the ISA chain from airplane through vehicle and the like to mechanical-object, and then down another ISA chain from there to computer-terminal. Therefore, to prevent as many "uninteresting" and misleading paths as possible, we put certain constraints on the MP and prohibit it from taking certain steps.

First, as I already mentioned, FRAIL passes markers a maximum of n arcs from the origin; one would normally choose n to be small compared to the size of the knowledge base. Second, FRAIL permits the programmer to specify restrictions on passing along various types of path. For example, by default the MP will pass markers only upwards along ISA links, not downwards—that is, markers are passed to more general concepts, but never to more particular ones (prohibiting thereby the path from mechanical-object to computer-terminal mentioned above).

Determining exactly what restrictions should be placed on marker passing is a matter for experiment (see [29]). We postulate restrictions such as an

"*anti-promiscuity*" *rule*: not allowing paths to propagate from nodes with more than c connections, for some chosen c. This is because nodes with many connections tend to be uninteresting ones near the top of the ISA hierarchy—mechanical-object, for example. We must be careful, however, not to be so restrictive that we also prevent the useful paths that we are looking for from occurring. And no matter how clever we are at blocking misleading paths, we must be prepared for the fact that they will occasionally turn up. The problem of such *false positives* is discussed by Charniak [7], who posits a *path checker* that would filter out many paths that are uninteresting or silly.

3.4. Other mechanisms of Polaroid Words

In Section 3.2, we introduced the idea of the Polaroid Word mechanism (PW to its friends), which would be responsible for disambiguating each word. We saw in Section 3.1 that there are many sources of information that can be used in disambiguation, and it would be up to each PW to use whatever information is available to it to make a decision for each word. Sometimes, as in the case of example (27), all that is required is looking at the paths found by the marker passer. At other times, MP will return nothing overwhelmingly conclusive; or, in the case of a word with several related meanings, more than one meaning may be marked. It would then be necessary for PWs to use other information and negotiation between possible meanings. In this section and the next, I will describe the operation of Polaroid Words.

3.4.1. *What Polaroid Words look like*

While it would be quite possible to operate Polaroid Words under the control of a single supervisory procedure that took the responsibility for the development of each "photograph," it seems more natural to instead put the disambiguation mechanism (and the responsibility) into each individual Polaroid Word. That is, a PW will be a procedure, running in parallel with other PWs,[18] whose job it is to disambiguate a single instance of a word. At this point, however, we must stretch our Polaroid photograph metaphor, for unlike a real self-developing photograph, a PW's development cannot be completely self-contained: the PWs will have to communicate with one another and with their environment in order to get the information necessary for their disambiguation. The idea of communicating one-per-word procedures brings to mind Small's word experts [57]. The similarity between PWs and Small's procedures is, however, only superficial; the differences will become apparent as we describe PWs in detail.

Instead of having a different PW for each word, we have but one kind of PW

[18] In the implementation described below, only one PW is active at a time, in order to simplify the implementation.

for each syntactic category; that is, there is a noun PW and a verb PW, and each noun uses the same disambiguation procedure as all the other nouns, each verb uses the same procedure as the other verbs, and similarly for other syntactic categories.[19] The knowledge about the meaning of each individual word is kept distinct from the disambiguation procedure itself, and indeed much of the knowledge used by PWs is obtained from the FRAIL knowledge base when it is necessary. When a new PW is needed, an instance of the appropriate type is cloned and is given a little packet of knowledge about the word for which it will be responsible. (Sometimes we will be sloppy and call these packets Polaroid Words as well. No confusion should result.) As far as possible, the packets contain only lexical knowledge—that is, only knowledge about how the word is used, rather than world knowledge (already available through FRAIL) about the properties of the word's denotations. The simplest packet of knowledge is that for a noun: it just contains a list of the semantic objects (frames; see Table 2) that the noun could represent. Figure 1 shows the knowledge packet for the noun *slug*. Any information needed about properties of the senses of the noun is obtained from the FRAIL knowledge base.

The packet for prepositions is a little more complicated; listed with the possible semantic objects, whose semantic type is *slot*, is a *slot restriction predicate* for each—a predicate that specifies what is required of an instance to be allowed to fill the slot. Figure 2 shows the packet for the preposition *with*; it assumes that the preposition is a case flag. (PWs for prepositions of noun-modifying PPs are discussed in [35].) A simple predicate, such as physobj ("physical object"), requires that the slot filler be under the specified node in the ISA hierarchy. A complex predicate may specify a boolean combination of

```
[slug (noun):
  gastropod-without-shell
  bullet
  metal-stamping
  shot-of-liquor]
```

FIG. 1. Packet of knowledge for *slug* for noun Polaroid Word.

```
[with (prep):
  instrument     (and physobj (not animate))
  manner         manner-quality
  accompanier    physobj]
```

FIG. 2. Packet of knowledge for *with* for preposition Polaroid Word.

[19] At this writing, PWs are implemented only for nouns, verbs, prepositions, and, in rudimentary form, noun modifiers. Determiners are straightforward, and PWs for them will exist later; see also below.

features that the filler must satisfy; thus in Fig. 2, the filler of instrument must be a physobj, but not an animate one. Predicates may also require that a property be proved of the filler; (property sharp) is an example of such a predicate.

The predicates for each slot are, in effect, the most restrictive predicate compatible with the restrictions on all instances of the slot in the knowledge base; thus, in Fig. 2, the predicate for instrument is true of all instruments (flagged by *with*) in all verbs in the system. In English, an animate entity can never be an instrument.

Verbs have the most complex knowledge packets. Figure 3 shows the packet for *operate*. For each meaning (a frame), the case slots that it takes are listed, with the preposition or prepositions that may flag each slot. Slot restriction predicates for each slot need not be specified in the packet, because they may be immediately found in the frame in the knowledge base. These predicates will, in general, be more restrictive than the predicates given in the PW for the corresponding preposition, but they must, of course, be compatible. For example, in the perform-surgery frame, the predicate on instrument may be (property sharp), which particularizes the predicate shown for instrument in Fig. 2.

3.5. Polaroid Words in action

PWs operate in parallel with ABSITY and the parser. As each word comes in to the parser and its syntactic category is assigned, a PW process is created for it. I first describe the way the process works and then give an example.

```
[operate (verb):
  [cause-to-function
    agent          SUBJ
    patient        SUBJ, OBJ
    instrument     SUBJ, with
    method         by
    manner         with
    accompanier    with]
  [perform-surgery
    agent          SUBJ
    patient        upon, on
    instrument     with
    method         by
    manner         with
    accompanier    with]]
```

FIG. 3. Packet of knowledge for *operate* for verb Polaroid Word.

3.5.1. *How Polaroid Words operate*

There are two easy cases. The first, obviously, is that the word is unambiguous. If this is the case, the PW process merely announces the meaning and uninterestingly hibernates—PWs always announce the fact and knock off work as soon as they have narrowed their possible meanings to just one. The second easy case is that the marker passer, which has been looking for paths between senses of the new word and unrejected senses of those already seen, finds a (single) nice connection that permits one alternative to be chosen. This was the case with example (27) of Section 3.3.1. I discuss in [35] exactly what makes a marker passing path "nice"; in general, short constructed paths are nice, and immediate paths are nicer.

If neither of these cases obtains, then the PW has to find out some more about the context in which its word occurs and see which of its alternatives fits best. To do this, it looks at certain preceding PWs to see if they can provide disambiguating information; we will describe this process in a moment. Using the information gathered, the PW will eliminate as many of its alternatives as possible. If this leaves just one possibility, it will announce this fact and terminate itself; if still undecided, it will announce the remaining possibilities and then sleep until a new word, possibly the bearer of helpful information, comes along.

Communication between PWs is restricted. The only information that a PW may ask of another is what its remaining possibilities are; that is, each may see other partly or fully developed photographs. In addition, a PW is restricted in two ways as to the other PWs it is allowed to communicate with. First, since a sentence is processed from left to right, when it is initially invoked a PW will be the rightmost word in the sentence so far and may only look to PWs on its left. As new words come in, the PW will be able to see them, subject to the second constraint, which is that each PW may only look at its *friends*.[20] Friendships among PWs are defined as follows: Verbs are friends with the prepositions and nouns they dominate; prepositions are friends with the nouns of their prepositional phrase and with other prepositions; and noun modifiers are friends with the noun they modify. In addition, if a prepositional phrase is a candidate for attachment to a noun phrase, then the preposition is a friend of the head noun of the NP to which it may be attached (see Section 4.3). The intent of the friendship constraints is to restrict the amount of searching for information that a PW has to do; the constraints reflect the intuition that a word has only a very limited sphere of influence with regard to selectional restrictions and the like.

An "announcement" of its meaning possibilities by a PW takes the form of a list of the one or more alternatives from its knowledge packet (with their slot restriction predicates and so on if they are included in the packet) that the PW

[20] Note that friendship constraints do not apply to the marker passer.

has not yet eliminated. An announcement is made by posting a notice in an area that all PWs can read; when a PW asks another for its possibilities, what it is actually doing is reading this notice. (PWs read only their friends' notices, of course.)

From the information that the notices provide, the PW eliminates any of its meanings that don't suit its friends. For example, each case slot may occur at most once in a sentence (but see [35, Section 1.1.2]), so if one preposition PW has already decided that it is an AGENT, say, any new preposition PW could cross AGENT off its own list. A preposition PW will also eliminate from its list any cases that its dominating verb does not allow it to flag, and any whose predicates are incompatible with its noun complement. Its friends may still be only partly developed, of course, in which case the number of eliminations it can make may be limited. However, if, say, one of its cases requires a filler that is hanim (a "higher animate entity," e.g., a human or corporation) but none of the alternatives in the partly developed noun is hanim, then it can confidently cross that case off its list. The PW may use FRAIL to determine whether a particular sense has the required properties. What is happening here is, of course, very like the use of selectional restriction cues for disambiguation; however, the restrictions are based on world knowledge rather than just symbols in the lexicon, and may, in principle, apply, all the inference power of FRAIL.

Similarly, nouns and verbs can strike from their lists anything that doesn't fit their prepositional friends, and nouns and noun modifiers can make themselves compatible with each other by ensuring that the sense selected for the noun is a frame in which the slot/filler pair of the adjective sense will fit. (If a PW finds that this leaves it with no alternatives at all, then it is in trouble; see below.)

When a PW has done all it can, it announces the result, a fully or partly developed picture, and goes to sleep. The announcement wakes up any of its friends that have not yet made their final decision, and each sees whether the new information—both the new word's announcement and any MP chain between the old word and the new—helps it make up its mind. If so, it too makes an announcement of its new possibilities list, in turn awakening its own friends (which will include the first PW again, if it is as yet undecided). This continues until none can do any more and quiescence is achieved. Then the next word in the sentence comes in, its PW is created, and the sequence is repeated.

3.5.2. *An example of Polaroid Words in action*

Let's consider this example, concentrating on the subordinate clause:

Ross found that the slug would operate the vending machine. (28)

SUBJ the slug operate OBJ the vending machine. (29)

Note the insertion of the pseudo-prepositions *SUBJ* and *OBJ*. We want to work out that *the slug* is a metal stamping, not a gastropod, a bullet, or a shot of whiskey; that the frame that *operate* refers to is cause-to-function, not perform-surgery; and that *SUBJ* and *OBJ* indicate the slots instrument and patient respectively. *Vending machine*, we will say, is unambiguous. For simplicity, we will ignore the tense and modality of the verb. The PWs for *slug* and *operate* were shown in Figs. 1 and 3; those for the other words are shown in Fig. 4.

Disambiguation proceeds as follows. The first words are *SUBJ* and *slug*; their PWs, when created, have not yet enough information to do anything interesting, nor has marker passing from the senses of *slug* produced anything (since there are no other words with which a connection might be found yet). Then *operate* comes along, and tells the others that it could mean either cause-to-function or perform-surgery. It too has no way yet of deciding upon its meaning. However, the *SUBJ* PW notices that neither meaning of *operate* uses *SUBJ* to flag the source or destination case, so it can cross these off its list. It also sees that while both meanings can flag their agent with *SUBJ*, both require that the agent be hanim. None of the possibilities for *slug* has this property, so the *SUBJ* PW can also cross agent off its list, and announce that it means either instrument or patient.

This wakes up the *operate* PW, which notices that only one of its meanings, cause-to-function, can take either an instrument or a patient flagged by *SUBJ*, so it too announces its meaning. The *slug* PW is also woken up, but it is unable to use any of this information.

Next comes the word *OBJ*. It could be patient or transferee, but the verb *operate* doesn't permit the latter, so it announces the former. Note that if *operate* had not already been disambiguated from previous information, this would happen now, as the *operate* PW would notice that only one of its senses takes any case flagged by *OBJ*. Upon hearing that *OBJ* is going to be patient, the

```
[SUBJ (prep):
  agent          animate
  patient        thing
  instrument     physobj
  source         physobj
  destination    physobj]

[OBJ (prep):
  patient        thing
  transferee     physobj]

[vending machine (noun):
  vending-machine]
```

FIG. 4. Packets of knowledge for *SUBJ*, *OBJ*, and *vending machine* PWs.

PW for *SUBJ* now crosses patient from its own list, since a case slot can appear but once in a sentence; this leaves it with instrument as its meaning. The PW for *slug* is not a friend of that for *OBJ*, so *OBJ*'s announcement does not awaken it.

The noun phrase *vending machine* now arrives, and we assume that it is recognized as a canned phrase representing a single concept (cf. [63–65]). The marker passer constructs a chain that, depending on the exact organization of the frames, might be (30):

vending-machine → coin → metal-stamping (30)

since a fact on the vending-machine frame would be that they use coins, and a coin ISA metal-stamping. This is enough for the *slug* PW to favor metal-stamping as its meaning, and all words are now disambiguated. Now that processing is complete, all markers in the knowledge base are cleared away.

3.5.3. *Recovery from doubt*

Now let's consider this example, in which marker passing is not used at all:

The crook operated a pizza parlor. (31)

This proceeds as example (29) did, until *operate* arrives. Since *crook* can either be something that is hanim, namely a criminal, or not, namely a shepherd's-staff, *SUBJ* is unable to make the move that in the previous example disambiguated both it and *operate*, though it can cross patient off its list. Still, when *OBJ* comes along, the *operate* PW can immediately eliminate perform-surgery. Let us assume that *pizza parlor* is an unambiguous canned phrase, as *vending machine* was. However, after it is processed, the PWs reach a standstill with *SUBJ* and *crook* still undisambiguated, as MP, being unaware of current trends in organized crime, finds no connection between *crook* and *pizza parlor*.

If it happens that at the end of the sentence one or more words are not fully disambiguated, then there are three ways that they may yet be resolved. The first is to invoke knowledge of a *preferred* or *deprecated meaning* for them. Preferred and deprecated meanings are indicated as an additional part of the knowledge packet for each word; a word can have zero or more of each. For example, the meaning female-swan of *pen* is deprecated, and should never be chosen unless there is positive evidence for it; the meaning writing-instrument is preferred, and the meaning enclosure is neither preferred nor deprecated. The possibilities that remain are ranked accordingly, and the top one or ones are chosen. In the present example, therefore, the two unfinished PWs look for their preferred meanings. It is clear that in English agent is far more common for *SUBJ* than the other remaining possibility, instrument, and so the *SUBJ* PW should prefer that. This, in turn will wake up the *crook* PW, which now finds

the requirement that its meaning fit *operate*'s agent, and therefore chooses criminal, completing disambiguation of the sentence.[21]

The second possibility at the end of the sentence is the use of "weak" marker passing chains. It may be the case that during processing of the sentence, MP found a path that was considered too weak to be conclusive evidence for a choice. However, now that all the evidence available has been examined and no conclusion has been reached, the weak path is taken as being better than nothing. In particular, a weak path that runs to a deprecated meaning is used as evidence in support of that meaning. In the present implementation, the trade-off between weak chains and preferred meanings is accomplished by "magic numbers" [36].

If neither preferred meanings nor weak chains help to resolve all the remaining ambiguities, then inference and discourse pragmatics may be invoked. It should be clear that Polaroid Words with marker passing are not a replacement for inference and pragmatics in word sense and case disambiguation; rather, they serve to reduce substantially the number of times that these must be employed. When a president tells us (32):

I am not a crook.[22] (32)

neither MP nor PWs will help us discover that he or she is not denying being a shepherd's staff, even though we may readily determine that shepherd's staff he or she indeed is not.[23]

Throughout this process, however, it should be kept in mind that some sentences are genuinely ambiguous to people, and it is therefore inappropriate to take extraordinary measures to resolve residual problems. If reasonable efforts fail, PWs can always ask the user what he or she really meant.

3.5.4. *Cues unused*

In Section 3.1, when we listed several lexical disambiguation cues that a system should be sensitive to, we included a sensitivity to syntax and mentioned that the verb *keep* could be disambiguated by looking at the syntactic type of its complement. At present, PWs do not have this sensitivity, nor would the flow of information between PWs and PARAGRAM support it even if they did. I do not anticipate major difficulties in adding this in future versions of Polaroid Words.

[21] It is possible that the results will vary depending on which PW applies its preferred meaning first. It is unlikely that there is a single "correct" order for such situations. If a sentence is really so delicately balanced, people probably interpret it as randomly as Polaroid Words do.

[22] Nixon, R.M., 11 November 1973.

[23] The present implementation does not have such an inference or pragmatics system available to it.

Because PWs are not sensitive to syntax yet, they cannot yet handle passive sentences. To be able to handle passive sentences, the PWs for verbs and for the prepositions *SUBJ* and *by* will have to be able to look at the verb's auxiliary and, if it is marked as being passive, then adjust their case flag mappings accordingly. Again, this is straightforward. Note that we are assuming a lexical analysis of passive sentences in the style of Bresnan [2]. Also awaiting a sensitivity to syntax are PWs for determiners. For example, the word *the* translates as either the or the-pl, depending on whether its NP is marked as singular or plural. Determiner PWs would be particularly simple, as they do not have to deal with marker passing or provide feedback to any other PWs.

A second unused disambiguation cue is global context. Marker passing is used as a mechanism for local (intrasentence) context cues, but our system has at present no representation of global context. It is my conjecture that it will *not* work simply to extend MP so that paths may be constructed between words of a sentence and the one before it. Rather, there should be a representation of context as a node or area in the knowledge base; this may include nodes that were senses of words of previous sentences, instances created by the semantic representations of the sentences, nodes that participated in inferences made as sentences were read, and so forth. (Such representations of context are also motivated by the need to analyze reference and connections in discourse [32, 33].) Marker passing may then be extended to include this representation of context.

The last unused cue is the requirement made by some verbs that certain of their cases must be present or that certain combinations of cases are prohibited. Adding this would allow preposition PWs to rule out possibilities in which a required case would not be present. In English, however, required cases are only a very weak cue, for English has few compulsory cases, and an assumption of well-formedness serves to enforce most of them.

3.6. What Polaroid Words cannot do

It is possible, as I mentioned above, that a PW could cross all its meanings off its list and suddenly find itself embarrassed. One possible reason for such an occurrence is that the word, or one nearby, is being used metaphorically, metonymically, or synecdochically. It is not possible in such cases to determine which word was actually responsible for the failure. Research by Gentner [22, 23] suggests that if the system is looking for a possible metaphor, it should consider the verb first, because verbs are inherently more "adjustable" than nouns; nouns tend to refer to fixed entities, while verb meanings bend more readily to fit the context. Thus, if a noun PW and a verb PW have irreconcilable differences, the noun should take precedence over the verb (regardless of which occurred first in the sentence [22, p. 165]).[24]

[24] There are, of course, exceptions to this general strategy; see [35, Section 5.4].

Note that these problems occur only in cases where slot restrictions are tested. In the case of conflicting unambiguous words, one or both being used in a new, metaphoric, metonymic, or synecdochic sense, the conflict will not be noticed until the final ABSITY output is sent to FRAIL, since there is no reason to have checked for consistency. This will also be the case when strong MP paths have caused a meaning to be chosen without checking slot restrictions. In [36], I show that from the perspective of psychological reality this is a desirable state of affairs.

3.7. Lexical disambiguation: Conclusion

Polaroid Words are cooperating mechanisms that both disambiguate word senses and determine case slots by finding connections between concepts in a network of frames and by negotiating with one another to find a set of mutually satisfactory meanings. In contrast to Hayes' CSAW system [25, 26], PW processes work in parallel with a parser, PARAGRAM, and a semantic interpreter, ABSITY, permitting them to deal with ambiguous words as if their semantic object were assigned immediately. (We shall see in Section 4 that PWs also help in structural disambiguation.) Also unlike CSAW, a similar PW control structure may be used for all syntactic categories. Polaroid Words minimize the need for separate, ill-motivated, purely linguistic knowledge; PWs have access to the NLU system's world knowledge and use it wherever possible.

Polaroid Words are implemented as processes that interpret LISP data structures containing purely lexical information that each word has in its

TABLE 6
What Polaroid Words can and cannot do

Words that can be disambiguated
SUBJ the slug operated OBJ the vending machine.
SUBJ the crook operated OBJ the pizza parlor.
SUBJ the crook wanted to kidnap OBJ Ross.
SUBJ Nadia's plane taxied to the terminal.
SUBJ Ross sold OBJ the lemming to Nadia.
SUBJ the man walked on the deck.
SUBJ the deep philosopher threw OBJ the peach pit into the deep pit.
Words that cannot be disambiguated
The astronomer married the star. *Marker passing is misled.*
Nadia swung the hammer at the nail, and the head flew off. *Requires inference.*
I want to eliminate some moles. *No disambiguating information.*
SUBJ the vending machine was operated by the slug. *No passives yet.*

dictionary entry. This is in contrast to approaches such as Small's [57], where the meaning of a word is represented as a large, barely constrained procedure, different for every word, which parses and performs semantic interpretation as well as lexical disambiguation. Rather, we let the parser, ABSITY, and the marker passer do much of the work that Small requires his "word experts" to perform. We thereby capture generalizations in disambiguation, needing only one type of PW for each syntactic category and relying almost exclusively on general world knowledge.

Table 6 shows some examples of sentences that the system can handle and some that it cannot yet.

In the next section, I will show that if we increase slightly the power of Polaroid Words, they can both provide substantial help for and be helped by the structural disambiguation process.

4. Structural Disambiguation

It remains to deal with ambiguities of syntactic structure. I now present a mechanism for this, the Semantic Enquiry Desk (SED). There are many types of structural ambiguity (see [35] for a long list); the SED handles two important kinds—prepositional phrase attachment and problems of gap finding in relative clauses—and provides a foundation for the development of methods for dealing with other kinds. In this paper, we will look at prepositional phrase attachment, in which a PP may be attached to either the verb phrase of the clause as a case slot filler, or to a noun phrase as a modifier.

Recall that we are using PARAGRAM, a Marcus-type limited-lookahead deterministic parser. A Marcus parser requires the assistance of a process that will tell it what is semantically preferred whenever it finds more than one syntactically admissible path and doesn't know which way to go. In such cases, the parser will execute a rule of the following form:

> If semantics prefers that X fill slot A {much more | somewhat more | no more} than Y filling slot A, then . . . , else . . . . (33)

The Semantic Enquiry Desk answers questions like this for the parser. While intended primarily for Marcus parsers, the approach could, however, be adapted to other types of parser, provided only that they are able to give the SED the information it requires.

4.1. Two theories of structural disambiguation

The SED synthesizes two rather different theories of structural disambiguation: the lexical preference theory of Ford, Bresnan, and Kaplan [17] and the presupposition minimization theory of Crain and Steedman [13]. We explain each briefly.

Ford, Bresnan, and Kaplan (FBK) show that disambiguation strategies that are based solely on syntactic preferences are inadequate to account for the resolution preferences that people exhibit in experiments. Such strategies included *Minimal Attachment* [39]: a new constituent should be attached to the parse tree using as few new nonterminal nodes as possible. FBK found that, rather, the preferred structure can change with the verb:

The women discussed the dogs on the beach.
(*i.e., NP attachment:*
The dogs on the beach were discussed by the women.) (34)

The women kept the dogs on the beach.
(*i.e., VP attachment:*
On the beach was where the women kept the dogs.) (35)

FBK propose a theory of *lexical preferences*, in which each verb is marked with the cases that are generally used with it. Each PP is assumed to be one of these *expected cases*, to be attached to the VP, and is interpreted as such if at all possible, until the last expected case is filled; subsequent PPs are assumed to be NP modifiers of the final expected case. These assumptions are dropped if an anomalous interpretation would result, or if pragmatics overrule them. FBK show that this principle accounts for some other kinds of structural ambiguity as well as PP attachment.

A very different theory of structural disambiguation has been proposed by Crain and Steedman [13], who claim that discourse context and, in particular, *presupposition* and *plausibility*, are paramount in structural disambiguation. The presuppositions of a sentence are the facts that a sentence assumes to be true and the entities that it assumes to exist. If a sentence presupposes information that the reader does not have, she has to detect and invoke these *unsatisfied* presuppositions. People have no trouble doing this, though there is evidence that it increases comprehension time [24]; Weischedel [62] has shown how presuppositions may be determined as the sentence is parsed.

Crain and Steedman hypothesize the *Principle of Parsimony:* the reading that leaves the fewest presuppositions unsatisfied is the one to be favored, other things being equal. This is a particular case of the *Principle of A Priori Plausibility:* prefer the reading that is more plausible with regard to either general knowledge about the world or specific knowledge about the universe of discourse, other things being equal. These principles can explain well-known garden-path sentences such as (36):

The horse raced past the barn fell.
(*i.e., the horse that was raced past the barn fell.*) (36)

The correct parse presupposes both the existence of a particular horse and that this horse is known to have raced past a barn, presuppositions unsatisfied in the null context. The incorrect parse, the one that garden-paths, presupposes only the first of these; the other is taken as new information that the sentence is conveying. The Principle of Parsimony claims that the garden-path parse is chosen just because it makes fewer unsatisfied presuppositions. Experiments by Crain and Steedman support this analysis, and suggest that Ford, Bresnan, and Kaplan's results are just artifacts of their use of the null context without controlling for unsatisfied presuppositions. Nevertheless, FBK's experiments found ambiguities whose preferred resolutions do seem to require an explanation in terms of lexical preference rather than presupposition or plausibility [35]. A more detailed discussion of the two approaches may be found in [35].

4.2. Prepositional phrase attachment

Many easy cases of prepositional phrase attachment can be handled by simple and absolute lexical and syntactic knowledge about allowed attachment. For example, few verbs will admit the attachment of a PP whose preposition is *of*, and such knowledge may be included in the lexical entry for each verb.

For those cases where deeper consideration is necessary, the SED's approach to PP attachment is to synthesize the two approaches of the previous section. There are four things needed for this:

- an annotation on each verb sense as to which of its cases are "expected";
- a method for determining the presuppositions that would be engendered by a particular PP attachment, and for testing whether they are satisfied or not;
- a method for deciding on the relative plausibility of a PP attachment;
- a method for resolving the matter when the preceding strategies give contradictory recommendations.

4.2.1. *Verb annotations*

The first requirement, annotating verbs for what they expect, is straightforward once we have data on verb preferences. These data should come from formal experiments on people's preferences, such as the one Ford, Bresnan, and Kaplan [17] ran, or from textual analysis; however, for a small, experimental system such as ours, the intuitions of the author and his friendly informants suffice. We classify cases as either compulsory, preferred, or unlikely.

4.2.2. *Testing for presupposition satisfaction*

The next requirement is a method for deciding whether a particular PP attachment would result in an unsatisfied presupposition. Now, there is a simple trick, first used by Winograd [66], for determining many PP attachments: try each possibility and see if it describes something that is known to exist. For example, sentence (37):

Put the block in the box on the table. (37)

could be asking that the block be placed in the box on the table, or that the block in the box be placed on the table. The first reading can be rejected if *the block* does not in context uniquely identify a particular block, or if there is no box on the table, or if *the box on the table* does not uniquely identify a particular box. Similar considerations may be applied to the second reading. (If neither reading is rejected, or if both are, the sentence is ambiguous, and Winograd's program would seek clarification from the user.) Crain and Steedman have called this technique the *Principle of Referential Success:* a reading that succeeds in referring to an entity already established in the hearer's mental model of the domain of the discourse is favored over one that does not.

An analysis of all possible cases [35] shows that we can "factor out" most of the presupposition testing: the candidate attachments will always score equally for unsatisfied presuppositions, except that VP attachment wins if the NP candidate is definite but NP attachment would result in reference to an unknown entity. On the other hand, if NP attachment would result in a felicitous definite reference, the number of unsatisfied presuppositions will remain the same for both attachments, but by the Principle of Referential Success we will prefer the NP attachment.[25]

Testing for this is easy for the SED because of the property of ABSITY that the semantic objects associated with the syntactic constituents are all well-formed FRAIL objects. The SED puts them into a call to FRAIL to see whether the mooted NP attachment entity exists in the knowledge base or not. (The entity may be there explicitly, or its existence may be inferred; that is up to FRAIL.) If the entity is found, the presupposition is satisfied, and the PP should be attached to the NP; otherwise, if the presupposition is unsatisfied, or if no presupposition was made, the VP is favored for the PP.

As an example, let's suppose the SED needs to decide on the attachment of the PP in (38):

Ross saw the man with the telescope . (38)

It will have the semantic objects for *see*, *the man*, and *with the telescope*, the last having two possibilities, one for each attachment mooted. It constructs the FRAIL statement (39) for the NP attachment:

(the ?x (man ?x (attr = (the ?y (telescope ?y))))) (39)

[25] A corollary of this is that a PP is never attached to an indefinite NP if VP attachment is at all possible, except if the NP is the final expected argument. This seems too strong, and the rule will probably need toning down. This corollary is not completely out of line, however, as definiteness does influence attachment; see [35].

If this returns an instance, man349 say, then the SED knows that presupposition considerations favor NP attachment: if it returns nil, then it knows they favor VP attachment.

4.2.3. *Plausibility*

Now let's consider the use of plausibility to evaluate the possible PP attachments. In the most general case, deciding whether something is plausible is extremely difficult, and I make no claims to having solved the problem. In the best of all possible worlds, FRAIL would be able to answer most questions on plausibility, and the slot restriction predicates on frames would be *defined* to guarantee plausibility; but, of course, we don't know how to do that.

However, there are two easy methods of testing plausibility that we can use that, though nondefinitive, will suffice in many cases. The first of these, used in many previous systems, is selectional restrictions. In the present system, these are applied as slot restriction predicates in Polaroid Words even before the SED becomes involved, and are often adequate by themselves. While satisfying the predicates does not guarantee plausibility, failing the predicates indicates almost certain implausibility.

The second method is what we shall call the *Exemplar Principle* (a weak form of the Principle of Referential Success): an object or action should be considered plausible if the knowledge base contains an instance of such an object or action, or an instance of something similar. Again, the SED can easily construct from the semantic objects supplied to it the FRAIL call to determine this. For example, if the SED wants to test the plausibility of a cake with candles or operate with a slug, it looks in the knowledge base to see if it has run across such a thing before:

```
(a ?x (cake ?x (attr = (some ?y (candle ?y)))))        (40)

(a ?x (operate ?x (instrument = (a ?y (slug ?y)))))   (41)
```

If it finds an instance, it takes the attachment to be plausible. If no such item is found, the matter is unresolved.[26] Thus the results of plausibility testing by the SED will be either exemplar exists or can't tell.[27]

4.2.4. *Making the attachment decision*

The SED's last requirement is a method for deciding on the PP attachment, given the results of verb expectation and presupposition and plausibility

[26] Various recovery strategies suggest themselves; see [35].

[27] With a large knowledge base, it may be possible to assign ratings based on the number of exemplars found; an item that has a hundred exemplars would be considered more plausible than one with only one exemplar, other things being equal. See [35] for discussion.

testing. If all agree on how the attachment should be made, then everything is fine. However, as Ford, Bresnan, and Kaplan make clear [17], verb expectations are only biases, not absolutes, and can be overridden by conflicting context and pragmatic considerations. Therefore, the SED needs to know when overriding should occur. Table 7 shows a decision algorithm for this that assumes that one VP and one NP are available for attaching the PP to. (An algorithm for the case of several available NPs is presented in [35].) The algorithm gives priority to ruling out implausible readings, and favors NP attachments that give referential success (referential success is tried first, since it is the stronger condition); if these tests don't resolve matters, it tries to use verb expectations.[28] If these don't help either, it goes for VP attachment (i.e., Minimal Attachment), since that is where structural biases seem to lie, but it is more confident in its result if an unsatisfied presupposition contraindicates NP attachment.

Some sentences for which the algorithm gives the correct answer are shown in Table 8. I also show a couple of sentences on which the algorithm fails. The fault in these cases seems to be not in the algorithm but rather in the system's inability to use world knowledge as well as people do. I cannot believe that people have some sophisticated mental algorithm that tells them how to attach PPs in those awkward cases where several different possibilities all rate approximately the same; rather, they use a simple algorithm and lots of knowledge, and in the rare awkward (and, probably, artificial) case, either ask

TABLE 7
Decision algorithm for restrictive PP attachment (one VP and one NP)

```
[Referential success]
  if NP attachment gives referential success
    then attach to NP

[Plausibility]
  else if an exemplar is found for exactly one attachment
    then make that attachment

[Verb expectations]
  else if verb expects a case that the preposition could be flagging
    then attach to VP
  else if the last expected case is open
    then attach to NP

[Avoid failure of reference]
  else if NP attachment makes unsuccessful reference
    then attach to VP
  else sentence is ambiguous, but prefer VP attachment anyway.
```

[28] There are sentences in which verb expectations prevail over plausibility [35]. Ideally, the SED would react to these sentences the way people do; however, the procedure I present errs on the side of common sense.

TABLE 8
PPs that are and are not correctly attached

PPs that are correctly attached
The women discussed the dogs on the beach. *NP-attached.*
The women discussed the tigers on the beach. *NP-attached if there are tigers on the beach, but VP-attached if no examples of* tigers on the beach *are found.*
Ross bought the book for Nadia. *VP-attached unless there is a book for Nadia available for reference.*
Ross included the book for Nadia. *NP-attached, as per FBK's preference data.*
PPs that are not correctly attached
The women discussed the dogs on the beach. *NP-attached because* dogs on the beach *is plausible and doesn't fail referentially, though VP attachment seems to be preferred by informants.*
The women discussed the dogs at breakfast. *NP-attached like* the dogs on the beach, *because the subtle unusualness of* the dogs at breakfast *is not detected.*

for clarification, choose an attachment almost at random, or use conscious higher-level inference (perhaps the kind used when trying to figure out garden paths) to work out what is meant.

4.3. Muffling combinatorial explosions

The preceding discussion assumed that while the meaning of the preposition of the PP may be unresolved, the potential attachment heads (i.e., the noun of the NP and the verb of the VP) and the remainder of the PP were all either lexically unambiguous or already disambiguated by their Polaroid Words. Now let's consider what happens if they are not, that is, if the words that must be used by the SED to decide on an attachment still have more than one possible meaning. We will see that the SED's decision will often as a side effect allow the words to be disambiguated as well.

In principle, the number of combinations of meanings of the words that are not yet disambiguated could be large. For example, if the two potential attachment heads, the preposition, and the prepositional complement all have three uneliminated senses, then 81 (i.e., 3^4) combinations of meanings could be constructed. In practice, however, many combinations will not be semantically possible, as one choice will constrain another—the choice for the verb will restrict the choices for the nouns, for example. Moreover, such multiple ambiguities are probably extremely rare. (I was unable to construct an example that didn't sound artificial.) It is my intuition that verbs are almost always

disambiguated by the NP or PP that immediately follows them, before any PP attachment questions can arise. In addition, the SED could use the strategy that if the verb remains ambiguous when PP attachment is being considered and combinatorial explosion seems imminent, the verb PW is *required* by the SED to disambiguate itself forthwith, even if it has to guess.[29] (This is in accord with Just and Carpenter's model of human reading processes [38], in which combinatorial explosion is avoided by judiciously early choice of word senses.)

Given, then, a manageably small number of lexical ambiguity combinations, structural disambiguation by the SED may proceed as before. Now, however, each attachment must be tried for each combination. The type of attachment that scores best for some combination is then chosen, thereby also choosing that combination as the resolution of the lexical ambiguities. For example, if combination *A* suggests NP attachment on the basis of referential success, thus beating combination *B*'s suggestion of VP attachment on the basis of plausibility, then both NP attachment and the word senses in combination *A* are declared winners. Ties are, of course, possible, and may indicate genuine ambiguity; see [35] for discussion.

4.4. Other structural ambiguities

In [35], I show how similar techniques may be used for gap finding in relative clauses, and give some preliminary suggestions on how the SED may also handle particle detection, relative clause attachment, and adverb attachment.

4.5. Structural disambiguation: Conclusion

Like Polaroid Words, the Semantic Enquiry Desk gains much of its power from the property of ABSITY that its partial results, the constituents with which the SED works, are always well-formed FRAIL objects, enabling it to use the full power of a frame and inference system. Even if the correct choice of object for an ambiguous word is not yet known, the alternatives will be well-formed and easily accessible.

5. Conclusion

5.1. What has been achieved

In this paper, I have presented a semantic interpreter and two disambiguation systems: one for structural ambiguity and one for lexical ambiguity. The systems have been designed to work closely with one another and with an existing parser and knowledge representation system.

The semantic interpreter, ABSITY, adapts to AI several aspects of Montague's way of thinking about semantics [47]: it is compositional; it has a strong

[29] This strategy is not actually implemented in the SED.

notion of "semantic object"; it operates in tandem with a parser; its partial results are always well-formed semantic objects; and it imposes a strong typing upon its semantic objects. The semantic objects are (unlike Montague's) objects of the knowledge representation, and the types are the types that the representation permits (which, I showed, correspond to the syntactic categories of English).

The structural disambiguator is the Semantic Enquiry Desk. It tells the parser what to do whenever the parser needs semantic help to decide between two or more alternative structures. The SED makes its decisions by looking at the semantic objects in the partial results of the semantic interpreter, and, if necessary, by querying the knowledge base for information on plausibility and referential success.

Polaroid Words are the individual, one-per-word processes for lexical disambiguation, one type for each syntactic category. Each process figures out the meaning of the word for which it is responsible through negotiation with its "friends" (certain nearby Polaroid Words), using the knowledge base to find simple properties of semantic objects when necessary. Even when "undeveloped," Polaroid Words can be regarded by ABSITY as semantic objects that it can manipulate, and both the SED and other Polaroid Words can obtain well-formed partial information from a PW that is still undecided.

Implicit in this work is the idea that an NLU system should be composed of interacting processes in which everything looks as well-formed as possible to everything else. Thus ABSITY always keeps the semantic objects that it is building well-formed, even when they are not final, so that the SED and the FRAIL representation can use them. Similarly, a PW represents a semantic object, and even if it cannot decide which particular object it is, it will have a set of well-formed possibilities visible for other processes to make use of, but still being manipulable as a single object by those processes that don't care about its eventual meaning.

A second implicit idea is that, while not compromising the previous principle, things should generally happen in their own sweet time but as soon as possible. Thus, a Polaroid Word is not obliged to decide upon its final answer either immediately upon its creation or at any particular time thereafter (except perhaps in certain special circumstances; see Section 4.3). Nevertheless, a PW is expected not to dawdle, but to announce its answer as soon as it possibly can (cf. [38]).

A third aspect of the approach is that the design of each of the interacting processes must accede to the demands of the design of the other processes, and the design and development of each must therefore be coordinated. Thus, the parser demands of the structural disambiguator that it be able to make semantic decisions halfway through the parse. The disambiguator in turn therefore requires of the semantic interpreter that it be incremental and have well-formed partial results that can be used for inference in and retrieval from

the knowledge base. And the semantic interpreter thus requires that the knowledge representation be compositional and support the concept of typed semantic objects. It also requires of the parser (thereby completing a circle of demands) that it support tandem processing; this requires that lexical disambiguation appear to be immediate. Lexical disambiguation requires that the parser to able to insert pseudo-prepositions where necessary.

Nevertheless, many of the main ideas embodied in the system and its components should be adaptable to other systems. ABSITY, for example, should be able to work with other parsers and with other knowledge representations, provided only that they support its requirements of compositionality and support of semantic objects; most frame-based representations should meet these requirements.

There is insufficient space here to present comparison of this work with that of other researchers; the interested reader should see [35, Chapter 8].

5.2. The future

To adequately substantiate the claim that ABSITY is a suitable "foundation for semantic interpretation," it is necessary to see if it can be extended to handle more of the complexities of natural language, as described in Section 2.5. Extensions to Polaroid Words and the Semantic Enquiry Desk are also planned.

ACKNOWLEDGMENT

This paper is based on my dissertation (published as [35]) at the Department of Computer Science, Brown University, Providence, Rhode Island. An earlier version of Section 4 was presented at the National Conference on Artificial Intelligence (AAAI-84), Austin, August 1984, and appears in the *Proceedings* thereof.

Financial support was provided in part by the U.S. Office of Naval Research under contract number N00014-79-C-0592. Preparation of this paper was supported by grants from the University of Toronto and the Natural Sciences and Engineering Research Council of Canada.

I am grateful to Eugene Charniak for discussions throughout the course of this work, and to the many others who influenced its development, including James Allen, Robert Amsler, Trina Avery, Emmon Bach, Carole Chaski, Gennaro Chierchia, Robin Cohen, Stephen Crain, Mike Gavin, Tom Grossi, Phil Hayes, Jim Hendler, Polly Jacobson, Tony Maida, Amy Rakowsky, Stu Shieber, Nadia Talent, and Doug Wong. Yawar Ali, Jean-Pierre Corriveau, Brenda Fawcett, and Diane Horton made helpful comments on an earlier draft of this paper.

REFERENCES

1. Anderson, J.R., A spreading activation theory of memory, *J. Verbal Learning and Verbal Behavior* **22** (3) (1983) 261–295.
2. Bresnan, J.W., The passive in lexical theory, in: J.W. Bresnan (Ed.), *The Mental Representation of Grammatical Relations* (MIT Press, Cambridge, MA, 1982) 3–86.
3. Cater, A.W.S., Analysis and inference for English, Doctoral Dissertation, Tech. Rept. 19, Computer Laboratory, University of Cambridge, 1981.

4. Cater, A.W.S., Request-based parsing with low-level syntactic recognition, in: K. Sparck Jones and Y.A. Wilks (Eds.), *Automatic Natural Language Parsing* (Ellis Horwood/Wiley, Chichester, 1983).
5. Charniak, E., Inference and knowledge, in: E. Charniak and Y.A. Wilks (Eds.), *Computational Semantics: An Introduction to Artificial Intelligence and Natural Language Comprehension* (North-Holland, Amsterdam, 1976) 1–21, 129–154; also: Organization and inference in a frame-like system of common sense knowledge, in: *Proceedings, Theoretical Issues in Natural Language Processing*, Cambridge, MA (1975) 46–55; also: Inference and knowledge in language comprehension, in: *Machine Intelligence* **8** (1977) 541–574.
6. Charniak, E., A common representation for problem-solving and language-comprehension information, *Artificial Intelligence* **16** (3) (1981) 225–255; also: Tech. Rept. CS-59, Department of Computer Science, Brown University, Providence, RI, 1980.
7. Charniak, E. Passing markers: A theory of contextual influence in language comprehension, Tech. Rept. CS-80, Department of Computer Science, Brown University, Providence, RI, 1981; also: *Cognitive Sci.* **7** (3) (1983) 171–190.
8. Charniak, E. Context recognition in language comprehension, in: W.G. Lehnert and M.H. Ringle (Eds.), *Strategies for Natural Language Processing* (Erlbaum, Hillsdale, NJ, 1982) 435–454.
9. Charniak, E., A parser with something for everyone, in: M. King (Ed.), *Parsing Natural Language* (Academic Press, London, 1983) 117–149; also: Tech. Rept. CS-70, Department of Computer Science, Brown University, Providence, RI, 1981.
10. Charniak, E., Cognitive science is methodologically fine, in: W. Kintsch, J.R. Miller and P.G. Polson (Eds.), *Methods and Tactics in Cognitive Science* (Erlbaum, Hillsdale, NJ, 1983).
11. Charniak, E., Gavin, M.K. and Hendler, J.A., The FRAIL/NASL reference manual, Tech. Rept. CS-83-06, Department of Computer Science, Brown University, Providence, RI, 1983.
12. Chomsky, N., *Aspects of the Theory of Syntax* (MIT Press, Cambridge, MA, 1965).
13. Crain, S. and Steedman, M., On not being led up the garden path: The use of context by the psychological parser, in: D.R. Dowty, L.J. Karttunen and A.M. Zwicky (Eds.), *Natural Language Processing—Psychological, Computational, and Theoretical Perspectives* (Cambridge University Press, 1984).
14. Cullingford, R.E., Script application: Computer understanding of newspaper stories, Doctoral Dissertation, Research Rept. 116, Department of Computer Science, Yale University, New Haven, CT, 1978.
15. Dowty, D.R., Wall, R.E. and Peters, S., *Introduction to Montague Semantics* (Reidel, Dordrecht, 1981).
16. Fahlman, S.E., *NETL: A System for Representing and Using Real-World Knowledge* (MIT Press, Cambridge, MA, 1979).
17. Ford, M., Bresnan, J.W. and Kaplan, R.M., A competence-based theory of syntactic closure, in: J.W. Bresnan (Ed.), *The Mental Representation of Grammatical Relations* (MIT Press, Cambridge, MA, 1982) 727–796.
18. Friedman, J., Moran, D.B. and Warren, D.S., Explicit finite intensional models for PTQ, *Am. J. Comput. Linguistics* 1978:1, microfiche 74 (1978) 3–22; also: Paper N-3, Computer Studies in Formal Linguistics, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI, 1978.
19. Friedman, J., Moran, D.B. and Warren, D.S., An interpretation system for Montague grammar, *Am. J. Comput. Linguistics* 1978:1, microfiche 74 (1978) 23–96; also: Paper N-4, Computer Studies in Formal Linguistics, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI, 1978.
20. Friedman, J., Moran, D.B. and Warren, D.S., Evaluating English sentences in a logical model: A process version of Montague grammar, in: *Proceedings 7th International Conference on Computational Linguistics*, Bergen, Norway, 1978; also: Paper N-5, Computer Studies in

Formal Linguistics, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI, 1978.
21. Gallin, D., *Intensional and Higher-Order Modal Logic with Applications to Montague Semantics* (North-Holland, Amsterdam, 1975); also: Doctoral Dissertation, Department of Mathematics, University of California, Berkeley, CA, 1972.
22. Gentner, D., Some interesting differences between nouns and verbs, *Cognition and Brain Theory* **4** (2) (1981) 161–178.
23. Gentner, D., Integrating verb meanings into context, *Discourse Processes* **4** (4) (1981) 349–375.
24. Haviland, S.E. and Clark, H.H., What's new? Acquiring new information as a process in comprehension, *J. Verbal Learning and Verbal Behavior* **13** (5) (1974) 512–521.
25. Hayes, P.J., Some association-based techniques for lexical disambiguation by machine, Tech. Rept. 25, Department of Computer Science, University of Rochester, Rochester, NY, 1977.
26. Hayes, P.J., On semantic nets, frames and associations, in: *Proceedings IJCAI-77*, Cambridge, MA (1977) 99–107.
27. Hendler, J.A. and Phillips, B., A flexible control structure for the conceptual analysis of natural language using message-passing, Tech. Rept. TR-08-81-03, Computer Science Laboratory, Texas Instruments Incorporated, 1981.
28. Hendler, J.A., Integrating marker-passing and problem solving, in: *Proceedings Seventh Annual Conference of the Cognitive Science Society*, Irvine, CA (1985) 130–139.
29. Hendler, J.A., Integrating marker-passing and problem-solving: A spreading-activation approach to improved choice in planning, Doctoral Dissertation, Tech. Rept. CS-86-01, Department of Computer Science, Brown University, Providence, RI, 1986.
30. Hendrix, G.G., The LIFER manual: A guide to building practical natural language interfaces, Tech. Note 138, Artificial Intelligence Center, SRI International, Menlo Park, CA, 1977.
31. Hendrix, G.G., Human engineering for applied natural language research, in: *Proceedings IJCAI-77*, Cambridge, MA (1977) 181–191.
32. Hirst, G., *Anaphora in Natural Language Understanding: A Survey*, Lecture Notes in Computer Science **119** (Springer, New York, 1981).
33. Hirst, G., Discourse-oriented anaphora resolution in natural language understanding: A review, *Am. J. Comput. Linguistics* **7** (2) (1981) 85–98.
34. Hirst, G., A foundation for semantic interpretation, in: *Proceedings 21st Annual Meeting of the Association for Computational Linguistics*, Cambridge, MA (1983) 64–73; also: Tech. Rept. CS-83-03, Department of Computer Science, Brown University, Providence, RI, 1983.
35. Hirst, G., *Semantic Interpretation and the Resolution of Ambiguity* (Cambridge University Press, 1987).
36. Hirst, G., Jumping to conclusions: Psychological reality and unreality in a word disambiguation program, in: *Proceedings Sixth Annual Conference of the Cognitive Science Society*, Boulder, CO (1984) 179–182.
37. Hobbs, J.R. and Rosenschein, S.J., Making computational sense of Montague's intensional logic, *Artificial Intelligence* **9** (3) (1977) 287–306.
38. Just, M.A. and Carpenter, P.A., A theory of reading: From eye fixations to comprehension, *Psychol. Rev.* **87** (4) (1980) 329–354.
39. Kimball, J., Seven principles of surface structure parsing in natural language, *Cognition* **2** (1) (1973) 15–47.
40. Maida, A.S., Using lambda abstraction to encode structural information in semantic networks, Rept. 1982-9-1, Center for Cognitive Science, Brown University, Providence, RI, 1982.
41. Maida, A.S. and Shapiro, S.C., Intensional concepts in semantic networks, *Cognitive Sci.* **6** (4) (1982) 291–330.
42. Marcus, M.P., *A Theory of Syntactic Recognition for Natural Language* (MIT Press, Cambridge, MA, 1980).

43. Marcus, M.P., On some inadequate theories of human language processing, in: T.G. Bever, J.M. Carroll and L.A. Miller (Eds.), *Talking Minds: The Study of Language in Cognitive Science* (MIT Press, Cambridge, MA, 1984) 253–278.
44. Marslen-Wilson, W.D. and Tyler, L.K., The temporal structure of spoken language understanding, *Cognition* **8** (1) (1980) 1–71.
45. McCarthy, J. First order theories of individual concepts and propositions, in: J.E. Hayes, D. Michie and L.I. Mikulich (Eds.), *Machine Intelligence* **9** (Ellis Horwood, Chichester, 1979) 129–147.
46. Mellish, C.S., Incremental semantic interpretation in a modular parsing system, in: K. Sparck Jones and Y.A. Wilks (Eds.), *Automatic Natural Language Parsing* (Ellis Horwood/Wiley, Chichester, 1983).
47. Montague, R., The proper treatment of quantification in ordinary English, in: K.J.J. Hintikka, J.M.E. Moravcsik and P.C. Suppes (Eds.,), *Approaches to Natural Language: Proceedings of the 1970 Stanford Workshop on Grammar and Semantics* (Reidel, Dordrecht, 1973) 221–242; also in: R.H. Thomason (Ed.), *Formal Philosophy: Selected Papers of Richard Montague* (Yale University Press, New Haven, CT, 1974) 247–270.
48. Moore, R.C., Problems in logical form, in: *Proceedings 19th Annual Meeting of the Association for Computational Linguistics*, Stanford, CA (1981) 117–124; also: Tech. Note 241, Artificial Intelligence Center, SRI International, Menlo Park, CA, 1981.
49. Quillian, M.R., Semantic memory, in: M.L. Minsky (Ed.), *Semantic Information Processing* (MIT Press, Cambridge, MA, 1968) 227–270.
50. Quillian, M.R., The teachable language comprehender: A simulation program and theory of language, *Commun. ACM* **12** (8) (1969) 459–476.
51. Riesbeck, C.K., Computational understanding: Analysis of sentences and context, Doctoral Dissertation, Memo AIM-238 Artificial Intelligence Laboratory, Computer Science Department, Stanford University, Stanford, CA, 1974.
52. Riesbeck, C.K., Conceptual analysis, in: R.C. Schank (Ed.), *Conceptual Information Processing* (North-Holland, Amsterdam, 1975) 83–156.
53. Riesbeck, C.K. and Schank, R.C., Comprehension by computer: Expectation-based analysis of sentences in context, in: W.J.M. Levelt and G.B. Flores D'Arcais (Eds.), *Studies in the Perception of Language* (Wiley, New York, 1978); also: Research Rept. 78, Department of Computer Science, Yale University, New Haven, CT, 1976.
54. Schank, R.C. and Abelson, R.P., *Scripts, Plans, Goals and Understanding: An Enquiry into Human Knowledge Structures* (Erlbaum, Hillsdale, NJ, 1977).
55. Schank, R.C. and The Yale AI Project, SAM—A story understander, Research Rept. 43, Department of Computer Science, Yale University, New Haven, CT, 1975.
56. Seidenberg, M.S., Tanenhaus, M.K., Leiman, J.M. and Bienkowski, M.A., Automatic access of the meanings of ambiguous words in context: Some limitations of knowledge-based processing, *Cognitive Psychol.* **14** (4) (1982) 489–537; also: Tech. Rept. 240, Center for the Study of Reading, University of Illinois at Urbana-Champaign, IL, 1982.
57. Small, S.L., Word expert parsing: A theory of distributed word-based natural language understanding, Doctoral Dissertation, Tech. Rept. 954, Department of Computer Science, University of Maryland, College Park, MD, 1980.
58. Smith, B.C., Intensionality in computational contexts, Unpublished manuscript, Artificial Intelligence Laboratory, MIT, Cambridge, MA, 1979.
59. Tait, J.I., Semantic parsing and syntactic constraints, in: K. Sparck Jones and Y.A. Wilks (Eds.), *Automatic Natural Language Parsing* (Ellis Horwood/Wiley, Chichester, 1983).
60. Tarnawsky, G.O., Knowledge semantics, Doctoral Dissertation, Department of Linguistics, New York University, 1982.
61. Warren, D.S., Using λ-calculus to represent meanings in logic grammars, in: *Proceedings 21st Annual Meeting of the Association for Computational Linguistics*, Cambridge, MA (1983)

51–56; also: Tech Rept. 83/045, Department of Computer Science, State University of New York at Stony Brook, NY, 1983.
62. Weischedel, R.M., A new semantic computation while parsing: Presupposition and entailment, in: C.K. Oh and D.A. Dinneen (Eds.), *Syntax and Semantics* **11**: *Presupposition* (Academic Press, New York, 1979) 155–182.
63. Wilensky, R. and Arens, Y., PHRAN: A knowledge-based approach to natural language analysis, Memo UCB/ERL M80/34, Electronics Research Laboratory, University of California, Berkeley, CA, 1980.
64. Wilensky, R. and Arens, Y., PHRAN: A knowledge-based natural language understander, in: *Proceedings 18th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, PA (1980) 117–121.
65. Wilensky, R., Arens, Y. and Chin, D., Talking to UNIX in English: An overview of UC, *Commun. ACM* **27** (6) (1984) 574–593.
66. Winograd, T. Understanding natural language, *Cognitive Psychol.* **3** (1) (1972) 1–191; also: *Understanding Natural Language* (Academic Press, New York, 1972).
67. Woods, W.A., *Semantics for a Question-Answering System* (Garland Publishing, New York, 1979); also: Doctoral Dissertation, Harvard University, Cambridge, MA, 1967.
68. Woods, W.A., Procedural semantics for a question-answering machine, *AFIPS Conference Proceedings* **33** (1968) 457–471.
69. Woods, W.A., Kaplan, R.M. and Nash-Webber, B.L., The Lunar Sciences Natural Language Information System: Final report, Rept. 2378, Bolt, Beranek and Newman, Cambridge, MA, 1972.

Received August 1984; revised version received January 1987[30]

[30] The editors apologize to the author and to the readers for the long delay in refereeing this paper.