# DYNAMICALLY COMBINING SYNTAX AND SEMANTICS IN NATURAL LANGUAGE PROCESSING

Steven L. Lytinen

Department of Computer Science
Yale University
Box 2158 Yale Station
New Haven, CT 06520, USA

## ABSTRACT

A controversy has existed over the interaction of syntax and semantics in natural language understanding systems. According to theories of *integrated* parsing, syntactic and semantic processing should take place simultaneously, with the parsing process driven by a single rule base which contains both syntactic and semantic knowledge. This is in sharp contrast to traditional linguistic approaches to language analysis, in which syntactic and semantic processing are performed separately from one another, driven by completely separate sets of syntactic and semantic rules.

This paper presents an approach to natural language understanding which is a compromise between these two views. It is an integrated approach, in the sense that syntactic and semantic processing take place at the same time. However, unlike previous integrated systems, the approach described here uses largely separate bodies of syntactic and semantic knowledge, which are combined only at the time of processing.

## I. INTRODUCTION

A controversy exists among researchers in natural language processing over the way in which syntax and semantics** should interact with each other. A *modular* approach to syntactic and semantic processing was argued for in [2]. In this approach, syntactic analysis is performed on an input text, producing a syntactic parse tree, which is then operated on by semantic interpretation rules. This sort of modular approach, or variations in which a limited amount of interaction is permitted between syntactic and semantic components, has been used in many natural language understanding systems, including LUNAR [14], Winograd's [12] system, and PARSIFAL [5].

In contrast, others have argued for an *integrated* approach to natural language processing. According to this argument, since semantic information often can be of use in making decisions about the syntactic structure of a text, semantics should be utilized as early as possible in the parsing process. Proponents of the integrated approach have argued that there should be no discernable stages in the language understanding process. Syntactic and semantic processing are performed simultaneously in integrated systems, usually by parsing rules that contain a mixture of both syntactic and semantic information. Also in contrast to the modular approach, no separate syntactic representation is built during language understanding. Instead, a "conceptual" representation, or representation of the meaning of the input text, is built directly during processing of the input. Examples of natural language systems which use the integrated approach to parsing include Wilks' parser [10] [11], ELI [7], the Integrated Partial Parser (IPP) [3], and the Word Expert Parser [9].

In this paper, I will argue that both sides of the syntax-semantics controversy are too extreme. Although semantic information should be brought to bear as quickly as possible so as to resolve syntactic ambiguities, I will argue that the way in which this has been accomplished in previous integrated parsers, by statically combining syntactic and semantic knowledge together in parsing rules, is representationally inefficient. As an alternative, I will present an approach to integrated parsing in which syntactic and semantic knowledge are *dynamically* combined at parse-time. In this approach, an explicit syntactic grammar is used by the system, encoded in syntactic rules similar to those used in PARSIFAL [5]. However, the application of these rules is quite different from syntax-first parsers, in that semantic information is used to determine when to apply particular syntactic rules.

This approach to integrated parsing has been implemented in a machine translation system called MOPTRANS, which parses short (1-3 sentences) newspaper stories about terrorism and crime, in English, Spanish, French, German, and Chinese. Translations are produced for these stories in English and/or German. Enough vocabulary, linguistic knowledge, and semantic knowledge have been encoded in the parser to enable it to parse 25-50 stories for each input language.

This paper will not include a discussion of MOPTRANS' semantic analyzer. For a detailed description, see [4]. Instead, this paper will focus on the way in which the semantics of the system is integrated with syntactic processing, and why this integration is desirable.

## II. WHY SYNTAX NEEDS SEMANTICS

Consider the following sentences:

> The cleaners dry-cleaned the coat that Mary found at the rummage sale for $10.
>
> The cleaners dry-cleaned the coat that Mary found in the garbage for $10.

The decision as to where to attach the prepositional phrase "for $10" in these two sentences cannot be made on the basis of syntactic information alone. However, due to the differences in meaning of the word "found," their syntactic structures are not the same. In the first example, since "found" refers to a purchase, it is appropriate to attach "for $10" to it.

*This research was done at the Computer Science Department of Yale University. It was supported in part by the Advanced Research Projects Agency of the Department of Defense and monitored by the Office of Naval Research under contract No. N00014-82K-0149.

**By semantics, I mean the traditional linguistic concept of semantics, or knowledge about the meanings of words, as well as *pragmatics*, or knowledge about the world and about how language is used.

However, in the second sentence, since it does not make sense to find something in the garbage for $10, the prepositional phrase must attach to "dry-cleaned."

In syntax-first parsing, then, the resolution of some syntactic ambiguities must be delayed until semantic interpretation. There is a computational price to pay for this, because often an unresolved syntactic ambiguity can affect the complexity of subsequent syntactic analysis. For example:

> The cleaners dry-cleaned the coat that Mary found in the garbage for $10 while she was away in New York.

If semantics is used immediately to resolve the attachment of "for $10" to the verb "dry-cleaned," then there is no ambiguity as to where to attach the clause "while she was away in New York." However, if the PP attachment is not resolved immediately, then it is also possible to attach this clause to "found." Thus, putting off the resolution of the first ambiguity would result in a syntax-first parser finding this sentence to be 3-way ambiguous. The third interpretation would not even have to be considered in an integrated parser.

Carrying forward ambiguities in syntactic analysis that could be resolved in an integrated parser can cause a combinatorial explosion in the number of syntactic ambiguities that must be considered as the parse continues. For example, consider the following sentence:

> The stock cars raced by the spectators crowded into the stands at over 200 mph on the track at Indy.

This sentence is highly ambiguous syntactically, due to the fact that either "raced" or "crowded" could be the sentence's main verb, and the prepositional phrases in the sentence could be attached in many different ways. In a syntax-first parser, these ambiguities would cascade, resulting in an increasingly large number of interpretations that would have to be considered during the course of the parse.*** However, the use of semantics drastically reduces the number of syntactic ambiguities that would have to be considered. Semantics can tell us that "raced" in this sentence must be active, because it is unlikely that spectators would race stock cars. This fact also resolves the syntactic ambiguity of "crowded," since both verbs cannot be active. This, in turn, eliminates many prepositional phrase attachments from consideration.

As this example demonstrates, the price for separating syntactic and semantic processing can be quite expensive computationally. Unresolved syntactic ambiguities can build on each other, resulting in the need to consider many syntactic attachments which would be eliminated if semantic processing were done in parallel.

## III. WHAT'S WRONG WITH PREVIOUS INTEGRATED PARSERS?

In order to bring semantics into the language understanding process as early as possible, previous integrated systems have compiled syntactic and semantic knowledge together into one set of rules. Although this sort of integration accomplishes the goal of utilizing semantic information early on in the parsing process, storing parsing knowledge in this form is highly inefficient.

First, the combination of syntactic and semantic information in the parser's rule base results in the inability to write parsing rules which apply to syntactic categories in general. Consider some of the parsing rules used in the Conceptual Analyzer (CA) [1], a descendant of ELI. Like ELI, much of CA's parsing knowledge was encoded in the form of *requests* [6], or test-action pairs, which were stored mainly in the parser's lexicon. Requests were used to build a conceptual representation of an input text as the parse proceeded. One of the tasks in building a representation was to fill the slots of a representational structure with the appropriate fillers. For example, to parse the sentence "Fred gave Sally a book," CA built the representation (ATRANS ACTOR FRED OBJECT BOOK RECIPIENT SALLY), where ATRANS was the Conceptual Dependency (CD) [8] primitive meaning "transfer of control of an object." To fill in the ACTOR of this action with FRED, CA used the following request:

> "Gave" request: Look back for a noun group which has the semantic property ANIMATE, which is not the object of a preposition, or the object of a verb, or attached syntactically to anything before it. Place the conceptualization in the ACTOR slot of the ATRANS.

Most other verbs in CA had similar requests, looking for a noun group of a certain semantic type before the verb, with the same syntactic restrictions on this noun group, to fill a slot in the conceptualization built by the verb. This slot was not always the ACTOR slot, as it was for "gave." For example, the RECIPIENT of an ATRANS preceded the verb "received." However, the request for "received" still shared much of the same information:

> "Received" request: Look back for a noun group which has the property ANIMATE, which is not attached syntactically to anything before it. Place the conceptualization in the RECIPIENT slot of the ATRANS built by "received."

These requests, as well as similar requests stored in the dictionary definition of every verb in CA's dictionary, all shared common syntactic information: namely, that the subjects of verbs precede them, and are not syntactically attached to anything before them. Thus, it would be much more economical to store this common information in only one rule, rather than duplicate it in countless verb-specific rules. However, because this syntactic information was combined with semantic information about the particular slot filled by the subject for each particular verb, and the semantic constraints on what the subject could be, this syntactic information had to be duplicated over and over again.

Another problem with previous integrated systems and the lack of autonomy of syntax in these systems is evident if we examine the way in which these parsers attempted to resolve certain types of syntactic ambiguities. Consider the way in which CA resolved the syntactic ambiguity in the following sentence:

> A small plane stuffed with 1500 pounds of marijuana crashed.

The word "stuffed" can function as either a past participle or a past active verb. To resolve this ambiguity, CA used a request which looked for the word "with" appearing after "stuffed." If it was found, "stuffed" was treated as passive, and the NP to the left of the verb (in this case

***For instance, in a left-to-right syntactic parse of this sentence, there would be 13 ways to attach the PP "at over 200 mph." Considering only syntax, the 2 verbs could be parsed in 4 ways (either one active, both part of unmarked relative clauses attaching to "cars," or the second relative clause attaching to "spectators"). Then, for each of these 4 interpretations, there are 5 possible places to attach the PP: to "cars," "raced," "spectators," "crowded," or "stands." This makes 20 possible attachments, 7 of which could be eliminated by various constraints on where PP's can be attached. The combinatorics are even worse for the subsequent PP's in the sentence.

"plane") was the OBJECT being stuffed. This request, if it fired, also activated another request which looked for another verb further on in the sentence, marking the end of the relative clause.

This request, and other requests used to resolve other types of syntactic ambiguities, used *local* syntactic information in order to perform disambiguation. By this, I mean that only words in the immediate neighborhood of the ambiguity were checked for particular syntactic properties, or for their presence or absence. In this case, the presence of the word "with" immediately after the verb was the local information. The advantage of this was that it was not necessary for the parser to keep track of a separate syntactic analysis. Syntactic ambiguities were resolved by examining short-term memory to see what words were there or what *semantic* constituents had been built.

However, it is not always the case that these sorts of local checks are enough. Consider the following examples:

The soldier called to his sergeant.
I saw the soldier called to his sergeant.

The slave boy traded for a sack of grain.
I saw the slave boy traded for a sack of grain.

In these cases, the appearance of a preposition after the verbs "called" and "traded" does not guarantee that the verbs are passive. This is because both verbs can be used either transitively or intransitively. Instead, the information that must be used to determine whether the verbs are active or passive is whether or not there is another verb in the sentence which functions as the main verb. However, since CA did not keep track of more global syntactic information such as whether a particular verb functioned as the main verb of the sentence, it would be much more difficult to write requests for these examples.

In general, then, it appears that some syntactic ambiguities cannot always be resolved by using only local syntactic checks. This is because the resolution of syntactic ambiguities sometimes requires more global knowledge about the syntax of a sentence, such as whether a particular verb functions as the main clause verb. Information like this cannot be determined so easily by rules which examine only immediate context. Thus, although we would like for syntactic and semantic processing to be integrated, it seems that a separate syntactic representation must be built during the analysis process in order to resolve some types of ambiguities.

## IV. A PARSER WHICH SATISFIES BOTH CONSTRAINTS

The MOPTRANS parser overcomes the difficulties that I have outlined in the last two sections. MOPTRANS is an integrated parser, in the sense that syntactic and semantic processing take place in tandem. However, it is different from previous integrated parsers, in that it uses a largely autonomous set of syntactic rules, and a syntactic representation of the input text is built during parsing. MOPTRANS uses PARSIFAL-like parsing rules [5], which specify how sequences of syntactic constituents in the input text can be attached to each other. However, unlike PARSIFAL and other syntactic parsers, syntax rules in MOPTRANS are only considered and applied if the syntactic attachments that they make are judged by the parser's semantic analyzer to be semantically appropriate. In this way, syntactic and semantic processing are completely integrated.

As MOPTRANS parses a piece of text, the semantic and syntactic representations that it builds are kept in its active memory. During parsing, new constituents are added to active

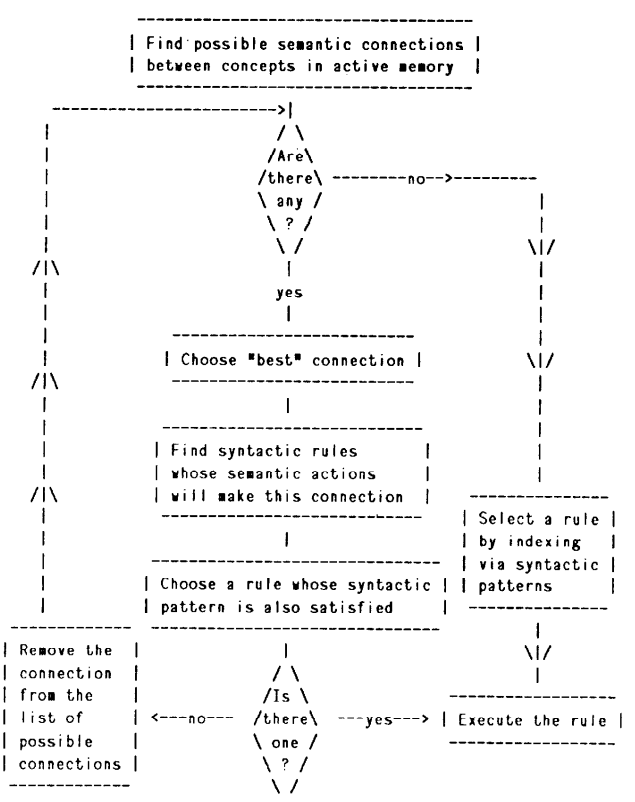


**Figure 1:** Interaction Between Syntax and Semantics in MOPTRANS

memory as each new word is read. As new constituents are added, semantics is asked if anything in active memory "fits together" well; that is, if there are any semantic attachments that could be made between the elements in active memory. If so, MOPTRANS' syntactic rules are consulted to see if any of these semantic attachments are syntactically legal. In other words, semantics proposes various attachments, and syntax acts as a filter, choosing which of these attachments makes sense according to the syntax of the input. The interaction between syntax and semantics is displayed graphically in Figure 1.

To make this more clear, consider how the following simple sentence is parsed by MOPTRANS:

John gave Mary a book.

MOPTRANS' dictionary definitions contain information about what semantic representation the parser should build when it encounters a particular word. Thus, "John" causes the representation PERSON to appear in the parser's active memory. At the same time, since "John" is a proper noun, the syntactic class NP is also activated.

When the word "gave" is processed, MOPTRANS' definition of this word causes the CD representation ATRANS to be placed in active memory. At this point, MOPTRANS considers the two semantic representations in active memory, PERSON and ATRANS. The semantic analyzer tries to combine these representations in whatever way it can. It concludes that the PERSON could be either the ACTOR or the RECIPIENT of the ATRANS, since the constraints on these roles are that they must be ANIMATE. It also concludes that the PERSON could be the OBJECT of the ATRANS (that is, the thing whose control or possession is being transferred). However, since this role is expected to be a PHYSICAL-OBJECT rather than an ANIMATE, the match is not as good

as with the ACTOR or RECIPIENT roles.****

This is the point at which the MOPTRANS parser utilizes its syntactic rules. Semantics has determined that 2 possible attachments are preferred. Now the parser examines its syntactic rules to see if any of them could yield either of these attachments. Indeed, the parser's Subject Rule will assign the PERSON to be the ACTOR of the ATRANS. The Subject Rule looks like this:

Subject Rule

| | |
|---|---|
| Syntactic pattern: | NP, V (active) |
| Additional restrictions: | NP is not already attached syntactically |
| Syntactic assignment: | NP is SUBJECT of V, V is indicative (V-IND) |
| Semantic action: | NP is ACTOR of V (or another slot, if specified by V) |
| Result: | V-IND |

This rule applies when an NP is followed by a V, and when the NP can fill the ACTOR slot of the semantic representation of the V. The NP is marked as the SUBJECT of the V, and the V is marked as indicative (V-IND). As dictated by the RESULT of the rule, the V-IND is left in active memory, but the NP is removed, since its role as subject prevents many subsequent attachments to it, such as PP attachments. In addition to these syntactic assignments, the semantic representation of the NP "John" is placed in the ACTOR slot of the ATRANS representing the verb.

The rest of the sentence is parsed in a similar fashion. To determine how "Mary" should be attached to "gave," semantics is asked for its preference. Just as with "John," "Mary" fits well into the ACTOR or RECIPIENT slots. Since "John" has already been selected as the ACTOR, semantics chooses the RECIPIENT slot for "Mary." Syntax is consulted to see if any syntactic rules can make this attachment. This time, the Dative Movement rule is found:

Dative Movement Rule

| | |
|---|---|
| Syntactic pattern: | V-IND, NP |
| Additional restrictions: | V-IND allows dative movement |
| Syntactic assignment: | NP is INDIRECT OBJECT of V-IND |
| Semantic action: | NP is (semantic) RECIPIENT of V-IND (or another slot, if specified by V-IND) |
| Result: | V-IND, NP |

When applied, this rule assigns "Mary" as the indirect object of "gave," and places the PERSON concept which represents "Mary" into the RECIPIENT slot of the ATRANS.

The final NP in the sentence, "the book," is attached to "gave" in a similar way. Semantics is asked to determine the best attachment of "book," which is represented as a PHYSICAL-OBJECT, to other concepts in active memory, which at this point contains the ATRANS as well as the person representing "Mary." Semantics determines that the best attachment is to the OBJECT role of the ATRANS. The syntactic rule which can perform this attachment is the Direct Object rule, which is similar in form to the Dative Movement rule above. This rule is applied, yielding the final semantic representation (ATRANS ACTOR PERSON OBJECT PHYSICAL-OBJECT RECIPIENT PERSON), and the syntactic markings of "John" as the subject of "gave," "book" as its direct object, and "Mary" as its indirect object.

One important thing to note about the parsing process on this sentence is that although the Direct Object Rule could have applied syntactically when "Mary" was found after the verb, it was never even considered. This is because the semantic analyzer preferred to place "Mary" in the RECIPIENT slot of the ATRANS. Since a syntactic rule was found which accomodated this attachment, namely the Dative Movement rule, the parser never tried to apply the Direct Object rule.

The MOPTRANS parser is able to resolve syntactic ambiguities that proved difficult for past integrated parsers. For the sentence discussed earlier, "I saw the soldier called to his sergeant," MOPTRANS has no trouble determining that "called" is an unmarked passive, because according to its syntax rules, another indicative verb at this point is not possible. The rule which is applied instead is the Unmarked Passive rule:

Unmarked Passive Rule

| | |
|---|---|
| Syntactic pattern: | NP, VPP |
| Syntactic assignment: | NP is (syntactic) SUBJECT of VPP, VPP is PASSIVE, VPP is a RELATIVE CLAUSE of NP |
| Semantic action: | NP is (semantic) OBJECT of S (or another slot, if specified by VPP) |
| Result: | NP, VPP |

"Called" is represented by the Conceptual Dependency primitive MTRANS, which is used to represent any form of communication. Since "soldier" can be attached as either the ACTOR or the OBJECT of an MTRANS, semantics would be happy with either of these attachments. However, the Subject Rule cannot apply at this point, since "soldier" is already attached as the syntactic direct object of "saw." Thus, this restriction on the Subject Rule prevents this attachment from being made. Instead, the Unmarked Passive Rule applies, since it semantically attaches "soldier" as the OBJECT of the MTRANS, and since "called" is marked as potentially being a past participle (VPP).

Unlike syntax-first parsers, the MOPTRANS parser can immediately resolve syntactic ambiguities on the basis of semantic analysis, thereby cutting down on the number of syntactic attachments that it must consider. We have already seen this in the example, "John gave Mary the book," in which the parser does not even consider if "Mary" is the direct object of "gave." Let us return now to two examples discussed earlier:

> The cleaners dry-cleaned the coat that Mary found in the garbage for $10.
>
> The cleaners dry-cleaned the coat that Mary found at the rummage sale for $10.

MOPTRANS parses the relative clause "that Mary found" with the following rule:

Clause Rule for Gap After the Verb (CGAV Rule)

| | |
|---|---|
| Syntactic pattern: | NP, RP (relative pronoun) (optional), V-IND |
| Additional restrictions: | V-IND is not followed by a NP |
| Syntactic assignment: | V-IND is a RELATIVE CLAUSE of NP |
| Semantic action: | NP is the semantic OBJECT of the V-IND |

****The way in which the semantic analyzer reaches these conclusions will not be discussed in this paper. For more details, see [4].

Result: NP, V-IND (changed to CLAUSE-VERB)

The Subject Rule assigns "Mary" to be the subject of "found," since "Mary" is not yet attached syntactically to anything before it. Then, since no NP follows "found," and since the attachment of "coat" (a PHYSICAL-OBJECT) as the OBJECT of the ATRANS is semantically acceptable, the CGAV rule applies, assigning "that Mary found" as a relative clause.

When the parser reaches "for $10" in the first example above, the representations of "dry-cleaned" and "found" are both still in active memory. The NP "$10" is represented as MONEY. The preposition "for" also has a semantic representation, which describes the possible semantic roles that a PP beginning with "for" can fill. One of these roles is called IN-EXCHANGE-FOR. "Dry-cleaned" is represented by the concept PROFESSIONAL-SERVICE, which expects to have its IN-EXCHANGE-FOR role filled with MONEY, since most professional services are done for money. ATRANS, on the other hand, does not explicitly expect an IN-EXCHANGE-FOR role. Thus, semantics prefers to attach the PP "for $10" to PROFESSIONAL-SERVICE and the verb "dry-cleaned."

In the second example, on the other hand, when the PP "at the rummage sale" is attached to "found," this triggers an inference rule that the ATRANS representing "found" must actually be the concept BUY, since "rummage sale" is a likely setting for this action. BUY, like PROFESSIONAL-SERVICE, expects the role IN-EXCHANGE-FOR to be filled with MONEY. Thus, semantics has no preference as to which verb to attach "for $10" to. To resolve the ambiguity, a syntactic recency preference is used, thereby attaching "for $10" to "found."

Because of this resolution of ambiguity, the MOPTRANS parser does not have to consider ambiguities further on in the sentence that it might otherwise have to. For example, in the sentence, "The cleaners dry-cleaned the coat Mary found in the garbage for $10 while she was away in New York," the PP attachment rule which MOPTRANS uses removes the representation of "found" from active memory, since the PP attaches to something before the clause containing "found." Therefore, when the parser reads "while she was away in New York," there is only one possible verb, "dry-cleaned," to which this clause can be attached.

## V. CONCLUSION

In this paper I have argued that semantic and syntactic analysis should be integrated. By this, I mean that syntactic and semantic processing must proceed at the same time, relying on each other to provide information necessary to resolve both syntactic and semantic ambiguities. Non-integrated, syntax-first parsers must leave some syntactic ambiguities unresolved until the semantic analysis stage. This can result in a highly inefficient syntactic analysis, because the failure to resolve one syntactic ambiguity can lead to other, "artificial" syntactic ambiguities which would not have to be considered had the original ambiguity been resolved with semantics. These new ambiguities may also be unresolvable using only syntax. If several of these ambiguities are encountered in one sentence, the combinatorics of the situation can get out of hand.

Previous integrated parsers have avoided these inefficiencies, but have suffered from problems of their own. Because of the lack of a separate representation of the input text's syntactic structure, these parser must rely on "local" syntax-checking rules to resolve syntactic ambiguities. Some types of ambiguities cannot easily be resolved with local checks.

To solve both of these problems at the same time, the MOPTRANS parser is integrated, in that syntactic and semantic processing proceed in parallel, but MOPTRANS has a separate body of syntactic knowledge, and builds a representation of the syntactic structure of input sentences. This enables it to use semantics to resolve syntactic ambiguities, and to easily resolve ambiguities that cause difficulties for local syntax-checking rules.

## REFERENCES


**1.** Birnbaum, L., and Selfridge, M. Problems in Conceptual Analysis of Natural Language. Tech. Rept. 168, Yale University Department of Computer Science, October, 1979.

**2.** Chomsky, N.. *Aspects of the Theory of Syntax.* MIT Press, Cambridge, Mass., 1965.

**3.** Lebowitz, M. *Generalization and Memory in an Integrated Understanding System.* Ph.D. Th., Yale University, October 1980. Research Report #186

**4.** Lytinen, S. *The Organization of Knowledge in a Multilingual, Integrated Parser.* Ph.D. Th., Yale University, Department of Computer Science, November 1984.

**5.** Marcus, M. *A Theory of Syntactic Recognition for Natural Language.* Ph.D. Th., Massachusetts Institute of Technology, February 1978.

**6.** Riesbeck, C. Conceptual Analysis. In *Conceptual Information Processing*, North-Holland, Amsterdam, 1975.

**7.** Riesbeck, C., and Schank, R.C. Comprehension by Computer: Expectation-based Analysis of Sentences in Context. Tech. Rept. 78, Yale University Department of Computer Science, October, 1976.

**8.** Schank, R.C. "Conceptual Dependency: A Theory of Natural Language Understanding." *Cognitive Psychology 3*, 4 (1972), 552-631.

**9.** Small, Steven. *Word Expert Parsing: A Theory of Distributed Word-based Natural Language Understanding.* Ph.D. Th., Department of Computer Science, University of Maryland, 1980.

**10.** Wilks, Y. An Artificial Intelligence Approach to Machine Translation. In *Computer Models of Thought and Language*, Schank, R., and Colby, K., Ed.,W.H. Freeman and Co., San Francisco, 1973, ch. 3, pp. 114-151.

**11.** Wilks, Y. "A Preferential, Pattern-matching, Semantics for Natural Language Understanding." *Artificial Intelligence 6*, 1 (1975).

**12.** Winograd, T.. *Understanding Natural Language.* Academic Press, New York, 1972.

**13.** Woods, W., Kaplan, R., and Nash-Webber, B. The Lunar Sciences Natural Language Information System: Final Report. Tech. Rept. 2378, Bolt, Beranek and Newman, Inc., 1972.