

We have begun work at evaluating the overall effectiveness of techniques for limiting the consideration of syntactic possibilities using semantic information. Our initial results, using a combination of fast segmentation (bracketing) of text and conceptual structure sharing, have shown a speedup of a factor of five⁴² in the Dow Jones domain, with improvements, though somewhat less significant, in other domains. This improvement does not come without some expense—it relies on the assumption that only certain semantic information is “important” in a domain and that intermediate results can often be sacrificed. Nevertheless, the practice of “skimming” by avoiding parsing, where a semantic model deems it unnecessary, seems to be a new and useful method that we are currently extending and testing.

B. The Analysis Process

The previous discussion emphasized the role of multiple interpretations in language analysis. The derivation of specific interpretations from general structures is equally important. The algorithm that TRUMP uses to perform linguistic analysis consists of the following mechanisms: 1) the grammatical analyzer, which identifies linguistic constituents and instantiates linguistic relations that are tied to matched structures, 2) a mapping mechanism, which adds conceptual structures to the chart by following VIEW and REF links from these instantiated linguistic structures, and 3), the *concretion* mechanism, which finds the most specific concept suggested by other information and fills out the roles of the concept.

Figure 10 is a sketch of the operation of the TRUMP system, showing the results of the syntactic, mapping, and concretion processes. Each of the three phases produces instantiations of new structures within the system's knowledge hierarchy. Grammatical analysis creates new linguistic structures, reflected in the chart described earlier, while mapping and concretion instantiate conceptual structures. This discussion will emphasize the concretion process, since this is the unusual element of analysis in TRUMP.

1. Grammatical Analysis

Grammatical analysis matches the input text to TRUMP's grammar, which comprises a hierarchy of linguistic patterns with associated relations. This analysis adds to the chart segments of parse trees that come together in the final parse, as well as instantiating linguistic relations. The input *Send Jones the message* shown in Figure 10 would add the *dat ive-VP* structure to the chart.

2. Mapping

Each time TRUMP completes the analysis of a linguistic structure, such as a noun phrase or verb phrase, it maps the linguistic relations it has produced into pieces of conceptual structure. This mapping process consists of applying

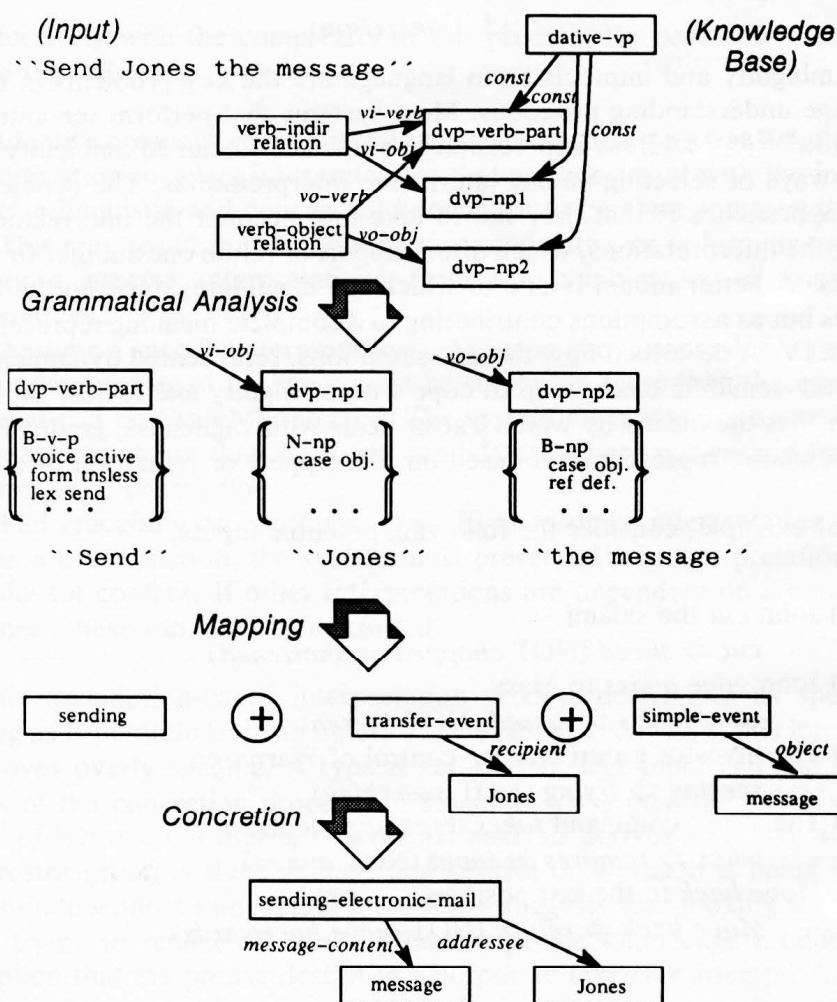


Figure 10. Operation of TRUMP.

VIEW and REF associations in the hierarchy to derive concepts. Since VIEW and REF have structure associated with them, the mapping process also starts to fill some conceptual roles, such as *recipient*. The program connects these new conceptual structures to the chart as shown in Section IV-A.

Except for VIEWS, which apply only when the context provides a "trigger" or information supporting a particular VIEW, the mapping process tries to produce as many new pieces of conceptual information as possible. The final meaning of an input is a synthesis from these pieces of conceptual structure. Based on knowledge of grammar as well as semantic constraints, the analyzer must know how the various concepts must fit together, and also when there is enough information to produce a more specific concept. This topic, at the heart of the semantic interpretation problem, is next.

3. Concretion

Ambiguity and imprecision in language are the key problems in building language understanding programs. Most systems that perform semantic interpretation^{26,27,34,43} address imprecision in a manner similar to ambiguity, by offering ways of selecting among alternative interpretations. The problem with these approaches is that they fail to take into account the interrelationships among the interpretations, which often support or refute one another to various degrees. A better model is one in which the candidates exist not as distinct choices but as assumptions contributing to a complete meaning representation. Section IV-A described how these assumptions, represented dynamically in a syntactic-semantic chart, help to cope with ambiguity and lexical gaps. Concretion^{44*} is the means by which TRUMP deals with vagueness, gradually refining a semantic representation based on the support or refutation of each element.

For example, consider the following potential inputs:

- (1) John *cut* the salami
cut $\Rightarrow$ *sliced* (NOT *chopped* or *shortened*)
- (2) John *gave a kiss* to Mary
gave a kiss $\Rightarrow$ *kissed* (NOT *presented*)
- (3) The investor group *seeking* control of Warnazco. . .
seeking $\Rightarrow$ *trying* (NOT *searching*)
- (4) The "rm" command *takes* three arguments.
takes $\Rightarrow$ *requires as input* (NOT *moves*)
- (5) *Move back* to the last position.
Move back $\Rightarrow$ *return* (NOT *move backwards*)

The examples above represent potential inputs from several disparate real and "toy" domains. The idea of TRUMP is to use a combination of "core" knowledge about language and certain specialized knowledge to produce a complete semantic interpretation. In each of the examples, the italicized word or phrase represents a vague, ambiguous, or metaphorical verb sense. The problem for a good semantic interpreter is to derive the real or intended sense of each phrase without excessive computation or specialized knowledge. *It is simply not practical to treat this process as one of discriminating among a large set of distinct interpretations.* The space of intended meanings is too large, and there are too many common characteristics of various senses. To

*This term was originally proposed by Joe Faletti. The problem of concretion was initially defined in a series of seminars conducted by Robert Wilensky at the University of California at Berkeley. In addition to Wilensky, Faletti, and the author, participants in these seminars included Yigal Arens, Margaret Butler, David Chin, Chuck Fillmore, Paul Kay, Marc Luria, Jim Martin, Jim Mayfield, Peter Norvig, Lisa Rau, and Nigel Ward.

deal effectively with the complexity of this process, the semantic interpreter must do the following:

- Identify prospective interpretations—The system must use linguistic information to select interpretations that are consistent with the input.
- Use linguistic and conceptual knowledge to combine interpretations—This may result in ruling out certain candidates, or in forming new and more precise interpretations from the combination of knowledge sources.
- Assume a specific interpretation—As in the above examples, a practical understanding of the input must be somewhat more than the maximum that can be “safely” inferred. The system must produce some knowledge structures that are likely candidates but are not certain from the linguistic description.
- Fail gracefully on contradictions—If an assumed interpretation results in a contradiction, the system must preserve those interpretations that do not conflict. If other interpretations are dependent on a conflicting one, these too must be discarded.

This assumption-based interpretation process determines as specific a meaning as is possible from an input, while enabling recovery if this interpretation proves overly specific. A typical natural language input can test several aspects of the concretion process. In example 3, *the investor group seeking control of Warnaco*, a first-pass semantic analysis derives a *seeking* action—the investor group is the searcher, and control of Warnaco is being sought. Domain-independent conceptual knowledge suggests that looking for a state means trying to realize the state. Domain-specific knowledge produces the assumption that the phrase describes a corporate takeover attempt. An interpretation of this specificity is necessary to drive inferences and cooperative responses.

The concretion process is illustrated in Figure 11. Each stage described above, and each knowledge source, must be distinct. The surface level semantic analysis is essential because it derives conceptual relations that would apply also to “looking for,” “searching for,” and even “pursuing”; this analysis thus avoids the redundant representation of each construct. This intermediate analysis also makes it possible to use abstract conceptual roles (such as *actor* and *recipient*) to determine specific underlying roles (such as the *target* and *suitor* of a corporate takeover) (cf. Ref. 45). The second aspect of concretion, applying conceptual knowledge to produce more specific interpretations, is necessary to refine vague terms and identify metaphorical or other nonliteral constructs. The third component, using domain-specific knowledge, separates this general conceptual knowledge from assumptions that depend on an implied context, the domain of corporate takeovers in this example.

Types of concretion. Concretion is the specialization of an abstract concept to a more precise interpretation in a given context. As the examples illustrate, concretion can involve a combination of linguistic and conceptual

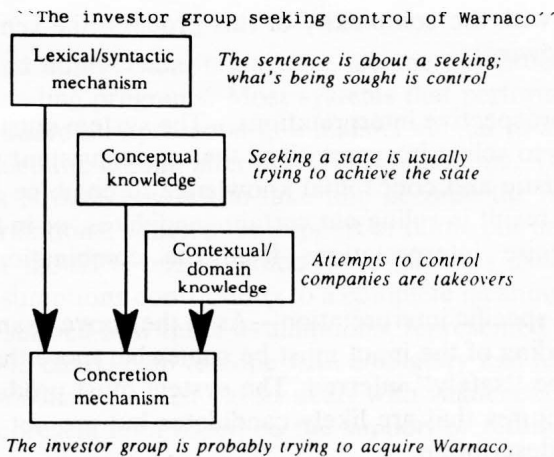


Figure 11. Concretion integrates knowledge sources.

knowledge, and can result in either a direct specialization or a metaphorical extension. In all cases, concretion requires four ingredients:

- An instantiated concept to be specialized
- A linguistic or conceptual “trigger”
- A target concept type
- A conceptual relation between source and target

For example, in *John cut the salami*, the concept to be concreted is *cutting*, the trigger is the combination of *cutting* with *edible-obloid* or some such, the target concept is *slicing* (indicated by the trigger), and the relation is subcategorization, or DOMINATES.

Concretion is often triggered by a linguistic structure, such as a particular combination of lexical items or the use of a certain phrase structure. Figure 12 shows the concretion of the concept *cutting* to the concept *severing* in the phrase *cut the tip off*. In this case, the knowledge base contains the information that *severing* is DOMINATED by *cutting*. The lexical relation *v-part-cut-off*, as described in Section III-B.2, serves as a trigger for this specialization of *cutting*.

TRUMP often reaches a specific interpretation through a series of concretions, as in the *seeking control* example. In the case of *cut*, the assumed interpretation can be more specific if the conceptual object is of a particular type, such as *body-part*, as shown in Figure 13. In this example, the concept of *amputating* results from a combination of linguistic and conceptual clues.

An intended meaning is not necessarily a subcategory of an intermediate interpretation, as shown by the *seeking control* example. Associations between concepts that are analogous or metaphorically related are represented as VIEWS (cf. Section III-C.2). Figure 14 illustrates the application of a VIEW in

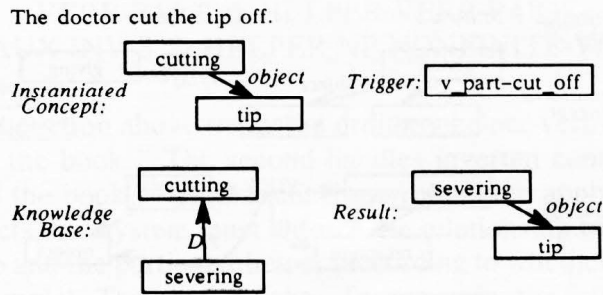


Figure 12. Concretion with linguistic trigger.

the concretion process. In this example, the use of the noun *kiss* to describe the conceptual object of a *giving* serves as a trigger. A general VIEW of *action* as *transfer-event* relates *giving* (which is a *transfer-event* to *kissing* (which is an *action*), and also the *recipient* of the *giving* to the conceptual object of the action. When used in this concretion, the VIEW derives a *kissing* concept with Mary as the *kissee*. There are two conceptual objects in a *kiss*—the person being kissed and the surface being kissed—and the concretion mechanism must use constraints on these roles to determine the correct role.

The above examples show several ways in which concretion results in the creation of a specific interpretation using a combination of linguistic and conceptual relationships. The examples also demonstrate that different types of concretion can combine, as in *cut the leg off*. In any of these cases, a slight variation in the input can negate the resulting interpretation. When this happens, the concretion mechanism retreats to intermediate structures, thus preserving as much of the semantic result as possible. In *the doctor cut the leg off accidentally* or *the doctor cut the leg off the table*, the system will preserve the *severing* interpretation. Each concretion, therefore, is an assumption that explicitly depends on its trigger and one other consequential relationships.

Concretion produces a specific interpretation and can help to avoid ambiguity, but TRUMP still must generally select a final meaning interpretation from a set of candidates. The next section covers this process.

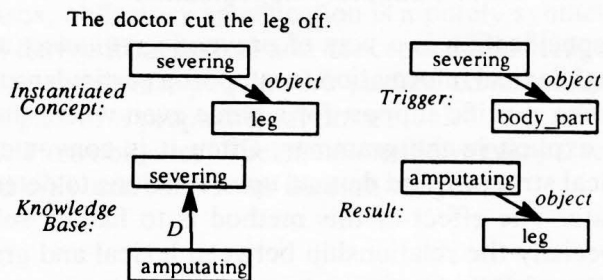


Figure 13. Concretion with conceptual trigger.

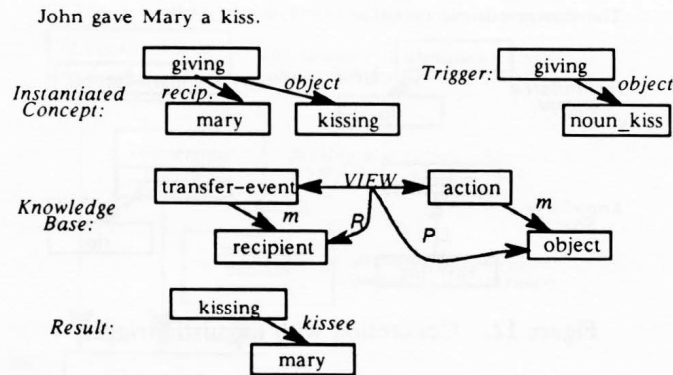


Figure 14. Concretion using VIEWS.

C. Techniques for Choosing the Best Interpretation

Concretion, described in the previous section, puts together a specific interpretation of a linguistic input, taking into account the relationships among conceptual structures. It does not, however, handle the interaction between linguistic and conceptual possibilities described in Section IV-A. Concretion implicitly assumes a preference for the most specific conceptual interpretation; the selection of the "best" interpretation of an input must combine the results of concretion with other linguistic and conceptual preferences. This section describes these preferences, including two ways of reducing TRUMP's chart, the general method for scoring conceptual possibilities, and the selection among possible conceptual roles.

TRUMP relies on two processing strategies that make use of linguistic relations in pruning the syntactic-semantic chart. The first, *obligatory specification*, is a method of eliminating classes of parses when they invoke insufficient linguistic relations. The second, *obligatory concretion*, eliminates structures that are semantically inconsistent. Both strategies apply the core grammar of TRUMP but use specific information derived from the input to control this application.

1. Obligatory Specification

Obligatory specification is a way of preventing runaway ambiguity when there is not enough lexical information to support a particular parse. It requires that the parser have specific support for a parse even where the details of that support are not explicit in the grammar. Often it is convenient to specify a simple grammatical structure and depend upon relations to determine when the rule is appropriate. The effect of this method is to handle subcategorization phenomena, especially the relationship between lexical and grammatical constraints, without requiring a larger grammar. A common example of this is the *helper-verb* relation, expressed by the following two rules:

VERB-PART $\Rightarrow$ HELPER VERB-PART
 AUX-INV-S $\Rightarrow$ HELPER NP NONFINITE-VP

The first production above treats the ordinary helper-verb construct, as in "John has sold the book." The second handles inverted constructs, such as "Has John sold the book?" To prevent these rules from applying to ungrammatical constructs, the system must enforce the relationship between the form of the main verb and the particular helper (according to whether it is a modal or passive, for example). This distinguishes, for example, the following two sentences:³⁹

- (3a) Have the students take the exam.
- (3b) Have the students taken the exam?

In (3a) the system should discard the possibility that "have" is an auxiliary for "take" and therefore consider (3a) as an imperative sentence. The augmentation discussed above could rule out this possibility by asserting that the helper "have" requires a perfect verb. A cleaner and more general way of expressing this augmentation is to dictate that the *helper-verb* relation has obligatory specification; that is, any grammar rule containing the relation must be abandoned unless the system produces a more specific relation, such as *helper-passive* or *helper-perfect*. This approach is preferable because the same constraints that enforce the grammaticality of the surface structure must determine other syntactic and semantic properties.

TRUMP invokes obligatory specification frequently. Even in the newswire texts, where there are few inverted constructs, the requirement prevents considering impossible auxiliary-verb combinations, as in the following segment from October 5, where it needs no special grammar rule to eliminate the possibility that "has" is an auxiliary verb for "leading":

...
 CASTLE & COOKE HAS LEADING U. S. MARKET POSITIONS IN SEVERAL FRESH
 AND PROCESSED FRUIT AND VEGETABLE CATEGORIES.
 ...

In these cases, obligatory specification is a purely syntactic phenomenon; however, it can derive from lexical and conceptual knowledge as well. Simple lexical knowledge comes into play in verb-particle combinations, where a preposition must be part of a prepositional phrase unless the *verb-particle* relocation is specified. Thus obligatory specification allows two parses for "John took on the job" and only one for "John died on the job."[†]

[†]Semantic interpretation must later rule out the invalid parse, [John][took][on the job].

2. Obligatory Concretion

Obligatory concretion dictates that the conceptual interpretation of a relation must help to specify the meaning of the structure in which the relation occurs. This tool is especially useful in resolving vague and ambiguous verb senses while attaching phrases to the verbs. For example, when prepositional phrases attach to verbs, it is obligatory that the prepositional phrase further specify the verb. Obligatory concretion applies to most constructs with a verb and a complement. "They elected John president" and "They elected to go" are both grammatically correct, but their grammaticality depends on the sense of the verb (*voted* in one case, *decided* in the other). TRUMP enforces this by assuring that its parse successfully contributes to its conceptual interpretation.

While many cases of obligatory concretion might be handled using lexical subcategorization or other linguistic constraints, this is more difficult in other cases, especially where the constraint or preference is more a feature of the complement or modifying phrase than of the verb. For example, certain noun phrases describing spatial or temporal relationships can attach to verbs almost regardless of the complement structure of the verb. In "the stock rose the day the deal was announced," for example, the noun phrase "the day. . ." serves as a temporal modifier of the verb "rose." Obligatory concretion permits the parser to allow noun phrases to appear in these adverbial constructs, provided they fill a temporal or spatial role. This constraint is easier to express than a purely lexical or syntactic constraint on such noun phrases.

3. Scoring

TRUMP uses a fairly intricate scoring scheme (developed empirically while attempting to parse various inputs) to determine an order for candidate interpretations. The scoring assigns a number to each grammatical structure, and assigns to each conceptual interpretation the sum of the scores of linguistic structures that contribute to that interpretation. Some of the heuristic features of this scheme are the following:

- (1) Every specification of a relation improves the score of the grammatical structure to which it belongs (e.g., "was sold" improves the score of the *helper-verb-part* structure)
- (2) Compound nominals, verb particles, and certain other strongly-binding relations are automatically marked up (20 times as much as other relations).
- (3) Every concretion (further semantic specification) improves the score of the structures with which it is associated, by an amount proportional to the semantic depth of the concept produced (measured by distance from the top of the conceptual hierarchy).
- (4) In other cases, TRUMP generally prefers "deeply nested" concepts; that is, those where meanings of phrases are associated with embedded

phrases. Otherwise, concepts with more roles specified are preferred (This enforces some syntactic attachment preferences as well).

No theory motivates these rules, but they are presented because they have evolved in exercising a real system. We are in the process of constructing a more principled model of scoring preferences based on our experience with TRUMP.

4. Role Resolution

The determination of conceptual role fillers influences both the selection of a parse and the final semantic interpretation. Often a linguistic construct can fill multiple conceptual roles, but one role is favored because that role is later filled by another construct. This is important because (1) Role filling cannot be indelible, and therefore the parser often cannot rule out many paths due to selectional restrictions, and (2) Linguistic preferences can always be overruled by semantic constraints. The following two sentences, where the role of "Acme" is the role *not* chosen for "Ace," illustrate this type of role resolution:

- (6a) A court approved Acme's purchase of Ace.
- (6b) A court approved Acme's purchase by Ace.

This section has described TRUMP's dynamic consideration of multiple syntactic and semantic interpretations, the derivation of specific conceptual interpretations using concretion, and the selection of a "best" interpretation. The strength of these methods is that they use general linguistic and conceptual preferences, thus limiting the amount of domain-specific knowledge engineering required to produce valid interpretations within a domain.

V. System Status and Current Projects

TRUMP currently consists of about 10,000 lines of Common Lisp code, including the parser, semantic interpreter, knowledge base manipulation, and lexical acquisition mechanisms. The "core" Ace knowledge base consists of several hundred linguistic entries with about a thousand concepts, but this is not large enough to select a new domain without coding a fair amount of new knowledge. We have augmented this knowledge in a transparent manner with the new lexicon and thesaurus of over 10,000 roots. The system is currently being used, experimentally, in a number of very different natural language projects, from research and aerospace to financial applications. Direct funding from GE's businesses has been the principal source of support for the project, and over 100 trained users in these businesses have used TRUMP. While we are far from having a program that can fit seamlessly into a new application area, the exercise is proving useful in identifying natural language tools that do apply

across domains. Our main thrust for the 1990's is to cultivate these domain-independent technologies to produce generic semantic interpretation methods.

VI. SUMMARY AND CONCLUSION

TRUMP is an experimental natural language processing program that applies common "core" knowledge about language to parsing and semantic interpretation in various domains. The language analyzer combines this generic knowledge with limited domain-specific information in disambiguating word senses, determining conceptual relationships, and selecting among possible interpretations. The project has been successful in developing techniques that combine generic and specialized knowledge in language analysis. The program is still in use and partially maintained by groups other than its initial designers; this makes it one of very few natural language tools that can be applied in this manner.

References

1. T. Winograd, *Understanding Natural Language*, Academic Press, New York, 1972.
2. L. Harris, "Experience with INTELLECT: Artificial intelligence technology transfer," *AI Magazine*, 5(2), 43-50 (1984).
3. J. Ginsparg, "A robust portable natural language data base interface," In *Proceedings of the Conference on Applied Natural Language Processing*, Santa Monica, CA, 1983.
4. B. Ballard and J. Lusth, "The design of DOMINO: A knowledge-based retrieval module for transportable natural language access to personal databases," In *Proceedings of the Workshop on Expert Database Systems*, Kiawah Island, SC, 1984.
5. B. Grosz, D. Appelt, P. Martin, and F. Pereira, *TEAM: An Experiment in the Design of Transportable Natural Language Interfaces*, Technical Report 356, SRI International, 1985.
6. P.S. Jacobs, "Language analysis in not-so-limited domains," In *Proceedings of the Fall Joint Computer Conference*, Dallas, TX, 1986.
7. P.S. Jacobs, "Knowledge-intensive natural language generation," *Artificial Intelligence*, 33(3), 325-378 (1987).
8. L.F. Rau, "Knowledge organization and access in a conceptual information system," *Information Processing and Management, Special Issue on Artificial Intelligence for Information Retrieval*, 23(4), 269-283, (1987).
9. L.F. Rau, "Information retrieval in never-ending stores," In *Proceedings of the Sixth National Conference on Artificial Intelligence*, Seattle, WA, July 1987, Morgan Kaufmann Inc., pp. 317-321.
10. P. Jacobs and L. Rau, "SCISOR: A system for extracting information from financial news," *Communications of the Association for Computing Machinery*, 34(11) 88-97 (1990).
11. I. Sider and P. Jacobs, *A Cooperative Natural Language Directory System*, Technical Report 88CRD025, GE R & D Center, 1988.
12. M. Bates and R. J. Bobrow, "A transportable natural language interface for information retrieval," In *Proceedings of the 6th Annual International ACM SIGIR Conference, ACM Special Interest Group on Information Retrieval and American Society for Information Science*, Washington, D.C., 1983.

13. B. S. ...
por...
14. P.S. ...
van...
15. R. V. ...
ence...
16. R. E. ...
sent...
17. D. E. ...
lang...
18. S. S. ...
for t...
19. J. B. ...
cess...
20. R. V. ...
ders...
21. P.S. ...
tion...
22. I. Ste ...
sema...
23. M.G. ...
phra...
24. D. L. ...
Jova...
25. M.A. ...
MD,
26. N. S. ...
KL-
27. G. H. ...
Univ...
28. P.S. ...
sis, U...
29. S. L. ...
proce...
30. J. Kin ...
15-47
31. L. F. ...
mode...
32. M. F. ...
closu...
33. S. Sh ...
Proc...
34. R. B. ...
proce...

13. B. Sundheim, *Second Message Understanding Conference (MUCK-II) Test Report*, Technical Report 1328, Naval Ocean Systems Center, San Diego, CA, 1990.
14. P.S. Jacobs and L.F. Rau, "Ace: Associating language with meaning," In *Advances in Artificial Intelligence*, T. O'Shea, (Ed.), North Holland, Amsterdam, 1985, pp. 295-304.
15. R. Wilensky, "Knowledge representation—A critique and a proposal," In *Experience, Memory, and Reasoning*, J. Kolodner and C. Riesbeck (Ed.), Lawrence Erlbaum Associates, Hillsdale, NJ, 1986, pp. 15-28.
16. R. Brachman and J. Schmolze, "An overview of the KL-ONE knowledge representation system," *Cognitive Science*, 9(2), 171-216 (1985).
17. D. Bobrow and T. Winograd, "An overview of KRL, a knowledge representation language," *Cognitive Science*, 1(1), 3-46 (1977).
18. S. Shieber, *An Introduction to Unification-based Approaches to Grammar*, Center for the Study of Language and Information, Palo Alto, CA, 1986.
19. J. Becker, "The phrasal lexicon," In *Theoretical Issues in Natural Language Processing*, Cambridge, MA, 1975.
20. R. Wilensky and Y. Arens, "PHRAN—A knowledge-based natural language understander," In *Proceedings of the 18th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, 1980.
21. P.S. Jacobs, "PHRED: A generator for natural language interfaces," *Computational Linguistics*, 11(4), 219-242 (1985).
22. I. Steinacker and E. Buchberger, "Relating syntax and semantics: The syntactico-semantic lexicon of the system VIE-LANG," In *Proceedings of the First European Meeting of the ACL*, Pisa, Italy, 1983.
23. M.G. Dyer and U. Zernik, "Encoding and acquiring meanings for figurative phrases," In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, New York, 1986.
24. D. Lockwood, *Introduction to Stratificational Linguistics*, Harcourt, Brace, and Jovanovich, New York, 1972.
25. M.A.K. Halliday, *Language as Social Semiotic*, University Park Press, Baltimore, MD, 1978.
26. N. Sondheimer, R. Weischedel, and R. Bobrow, "Semantic interpretation using KL-ONE," In *Proceedings of the Tenth International Conference on Computational Linguistics*, Palo Alto, CA, 1984.
27. G. Hirst, *Semantic Interpretation and the Resolution of Ambiguity*, Cambridge University Press, Cambridge, England, 1987.
28. P.S. Jacobs, *A Knowledge-Based Approach to Language Production*, Ph.D. Thesis, University of California, Berkeley, 1985, Computer Science Division Report UCB/CSD86/254.
29. S. Lytinen, "Dynamically combining syntax and semantics in natural language processing," In *Proceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia, 1986.
30. J. Kimball, "Seven principles of surface parsing in natural language," *Cognition*, 2, 15-47 (1973).
31. L. Frazier and J.D. Fodor, "The sausage machine: A new two-stage parsing model," *Cognition*, 6, 291-325 (1978).
32. M. Ford, J. Bresnan, and R. Kaplan, "A competence-based theory of syntactic closure," In *The Mental Representation of Grammatical Relations*, MIT Press, Cambridge, MA, 1982.
33. S. Shieber, "Sentence disambiguation by a shift-reduce parsing technique," In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, Karlsruhe, Germany, 1983.
34. R. Bobrow and B. Webber, "Knowledge representation for syntactic/semantic processing," In *Proceedings of the National Conference on Artificial Intelligence*, Palo Alto, CA, 1980.

35. Y. Wilks, "A preferential, pattern-matching semantics for natural language understanding," *Artificial Intelligence*, 6, 53-74 (1975).
36. L. Schubert, "Are there preference trade-offs in attachment decisions?" In *Proceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia, 1986.
37. C. Riesbeck, "Conceptual analysis," In *Conceptual Information Processing*, R.C. Schank (Ed.), American Elsevier Publishing Company, New York, 1975.
38. M. Lebowitz, "Memory-based parsing," *Artificial Intelligence*, 21(4), 363-404 (1983).
39. M. Marcus, *A Theory of Syntactic Recognition of Natural Language*, MIT Press, Cambridge, 1980.
40. P.S. Jacobs and U. Zernik, "Acquiring lexical knowledge from text: A case study," In *Proceedings of the Seventh National Conference on Artificial Intelligence*, Palo Alto, CA, August 1988, Morgan Kaufmann, Inc., pp. 739-744.
41. E. Rich, J. Barnett, K. Wittenburg, and D. Wroblewski, "Ambiguity procrastination," In *Proceedings of the Sixth National Conference on Artificial Intelligence*, Seattle, WA, 1987.
42. P. Jacobs, "To parse or not to parse: Relation-driven text skimming," In *Proceedings of the Thirteenth International Conference on Computational Linguistics*, Helsinki, Finland, 1990.
43. S. Lytinen, *The Organization of Knowledge in a Multi-lingual, Integrated Parser*, Ph.D. Thesis, Yale University, 1984.
44. R. Wilensky, "Memory and inference," In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, Karlsruhe, Germany, 1983.
45. P.S. Jacobs, "A knowledge framework for natural language analysis," In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, Milan, Italy, 1987.