Generic Text Processing: A Progress Report

Paul S. Jacobs, George R. Krupka, Susan W. McRoy
Lisa F. Rau, Norman K. Sondheimer and Uri Zernik

Artificial Intelligence Program
GE Research and Development Center
Schenectady, NY 12301

Introduction

A generic natural language system, without modification, can effectively analyze an arbitrary input at least to the level of word sense tagging. Considerable research has addressed the *transportability* of natural language systems, but not *generic* text processing capabilities. For example, previous DARPA-sponsored work [1, 2] produced transportable interfaces to database systems. Each new application of these interfaces generally required modifications to lexicons, new semantic knowledge bases, and other specialized features. The most that natural language text processing systems have accomplished has been the parsing of arbitrary text, without any real semantic analysis.

Our text processing work at GE, like the earlier efforts in interfaces, has emphasized transportability. We have applied the same core of natural language tools, including the grammar, lexicon, parser, and semantic interpreter (collectively known as the NLToolset [3]), to domains ranging from personnel directories to aircraft engine service reports. Some other applications of the NLToolset are the SCISOR news story reader [4] and the GE effort in the MUCK-II evaluation [5]. These prototypes process texts at a rate of several hundred paragraphs per hour, synthesize a structured representation for each text, and answer English questions about the contents.

This transportable text processing effort has provided successful proof-of-concept demonstrations for applications with only a few person-months of effort, often with only minor assistance from the NLToolset developers. Unfortunately, even this amount of customization is far too much for most potential text processing applications, which continue to use more robust but far more superficial methods, such as keyword indexing, pattern matching, and statistical retrieval.

Our effort at GE currently aims at developing *generic* text processing capabilities. Building on the techniques previously used for a wide variety of applications, we are extending our methods to the semantic analysis of unrestricted text. This has placed the greatest stress on enhanced lexical processing. Most importantly, the system must now apply lexical information that is rich enough to cover the content words of arbitrary text, while providing enough information about words in context to control ambiguity and produce "preferred" interpretations. In the near term, we expect at least to reduce future customization efforts while experimenting with the effects of "text coding" or word sense-tagging without customization.

We will report on some preliminary results in generic text processing, including the development of a 10,000-root core sense-disambiguated lexicon, the use of text preprocessing (including tagging and bracketing) to help control parsing, and the design of a module for collecting sense preferences from different knowledge sources in text analysis. The program currently discriminates word senses (albeit somewhat roughly) in arbitrary text. We will report on the current state of this effort and describe the key unresolved issues.

Lexical inadequacy

Most problems in natural language processing—including information retrieval, database generation, and machine translation—hinge on relating words to other words that are similar in meaning. Because of the extreme difficulty of producing any accurate deep-level analysis of text, many of these strategies are inherently word-based. In the case of information retrieval, current methods match words in a query with words in documents, with the degree of match weighted according to the frequency of words in the texts. In database generation, programs map individual words into names of frames or database records. In language translation, systems use mappings between words in a "source" language and words in a "target" language to guide lexical choice (word choice). In all these applications, current methods are limited in their accuracy by the fact that many words have multiple senses, although different words often have similar meanings.

We will refer to this as the problem of *lexical inadequacy*, including the issue of genuinely ambiguous words as well as vague terms and derivative words (words that have a common root but vary slightly in meaning). Previous approaches to the problem of lexical inadequacy fall into two basic categories–word-based approaches and deep-level approaches. Word-based approaches have ad-

dressed the problem in several general ways, including using co-occurrence and other contextual information as an indicator of text content to try to filter out inaccuracies, using word roots rather than words by stripping affixes, and using a thesaurus or synonym list that matches words to other words. Deep-level approaches use a knowledge representation, or interlingua, to reflect text content, thereby separating text representation from the individual words. Deep-level approaches can be more accurate than word-based approaches, but have not been sufficiently robust to perform any practical text processing task. This lack of robustness is generally due to the difficulty in building knowledge bases that are sufficient for broad-scale processing.

Text coding

Our approach tries to take advantage of the main benefit of word-based analysis, i.e. the explicit recognition in text representation of the relationship between words that derive from a common root, while overcoming the main limitation, i.e. the loss of information between complete text and word roots. The idea is that the first step in text processing should be to develop a "recoding" or sense-tagging of the source text that captures word-level knowledge as well as any deeper information the system is able to produce. In theory, this approach would lead to capabilities that are no worse, with zero customization, than word-based methods for information retrieval, but allow for easy improvement using more knowledge-intensive methods. This theory is still untested, although there is some preliminary evidence that word sense tagging could improve information retrieval system performance [6].

As an example of the input and output of the system, the following is an arbitrary segment of Wall Street Journal text with the resulting coding. Each word is tagged with its part of speech and sense code (a number suffix), along with a parent concept. For example, the tag `[changing verb_3 (c-replacing)]` shows that the input word is "changing", the preferred sense is number 3 of the verb, and that this sense falls under the concept `c-replacing` in the hierarchy. The program produces this output by analyzing the text in several stages, from stochastic tagging through semantic interpretation, and collecting results in a matrix of word senses and preference scores.

> The network also is changing its halftime show to include viewer participation, in an attempt to hold on to its audience through halftime and into the second halves of games. One show will ask viewers to vote on their favorite all-time players through telephone polls.

```
[the det_1 (c-definite-qual) ]
[network noun_2 (c-entertainment-obj
         c-business-org c-system) ]
[also adv_1 (c-numeric-qual) ]
[is *aux* ]
[changing verb_3 (c-replacing) ]
[its ppnoun_1 (c-obj) ]
[halftime noun_1 (c-entity) ]
[show c-act-of-verb_show1 (c-manifesting) ]
[to *infl* ]
[include verb_2 (c-grouping) ]
[viewer c-verb_view2-er (c-entity) ]
[participation c-result-of-being-verb_participate1
   (c-causal-state) ]
[*comma* *punct* ]
[in prep_27 (c-group-part) ]
[an det_1 (c-definite-qual) ]
[attempt c-act-of-verb_attempt1 (c-attempting) ]
[to *infl* ]
[hold verb_4 (c-positioning) ]
[on adv_1 (c-range-qual c-continuity-qual) ]
[to prep_1 (c-destination-rel) ]
[its ppnoun_1 (c-obj) ]
[audience noun_1 (c-human-group) ]
[through prep_1 (c-course-rel) ]
[halftime noun_1 (c-entity) ]
[and coordconj_1 (c-conjunction) ]
[into prep_5 (c-engage-in) ]
[the det_1 (c-definite-qual) ]
[second c-numword_two1-th (c-order-qual) ]
[halves noun_1 (c-portion-part) ]
[of prep_8 (c-stateobject-rel) ]
[games noun_1 (c-recreation-obj) ]
[*period* *punct* ]

[one noun_1 (c-entity) ]
[show c-act-of-verb_show1 (c-manifesting) ]
[will *aux* ]
[ask verb_2 (c-asking) ]
[viewers c-verb_view2-er (c-entity) ]
[to *infl* ]
[vote verb_1 (c-selecting) ]
[on prep_4 (c-temporal-proximity-rel) ]
[their ppnoun_1 (c-obj) ]
[favorite adj_1 (c-importance-qual
                 c-superiority-qual) ]
[all det_1 (c-quantifier) ]
[*hyphen* *punct* ]
[time noun_1 (c-indef-time-period) ]
[players c-verb_play1-er (c-entity) ]
[through prep_1 (c-course-rel) ]
[telephone noun_1 (c-machine) ]
[polls c-act-of-verb_poll1 (c-asking) ]
[*period* *punct* ]
```

Naturally, this text recoding assumes a representation scheme for the enhanced text, a large lexicon including word senses and knowledge for discriminating them, and a robust parser and semantic interpreter. The rest of this paper will give a progress report on the development of these resources so far.

The core lexicon

GE's core lexicon development followed an experiment with a moderate-sized (10,000-unique root), commercially-available lexicon, which demonstrated

many substantive problems in applying lexical resources to text processing. The lexicon had good morphological and grammatical coverage, as well as a thesaurus-based semantic representation of word meanings. However, it was inadequate for producing semantic representations of text because it did not provide reasonable information for discriminating word senses. Building from this effort, we designed and constructed a lexicon of roughly the same size that included much more word-sense information, as well as constraining the number of senses of each entry to avoid spurious semantic interpretations.

The main features of the new lexicon are a hierarchy of 1,000 parent concepts for encoding semantic preferences and restrictions, sense-based morphology and subcategorization, a distinction between primary and secondary senses and senses that require particular "triggers" or appear only in specific contexts, and a broad range of collocational information. For example, the following are the lexical entries for the word "issue":

```
( issue
   :POS noun
   :SENSES
   (( issue1
      :EXAMPLE (address important issues)
      :TYPE p
      :PAR (c-concern)
      :ASSOC (subject)   )
    ( issue2
      :EXAMPLE (is that the october issue?)
      :TYPE s
      :PAR (c-published-document)
      :ASSOC (edition)   )))
( issue
   :POS verb
   :G-DERIV nil
   :SENSES
   (( issue1
      :SYNTAX (one-obj io-rec)
      :EXAMPLE (the stockroom issues supplies)
      :TYPE p
      :PAR (c-giving)
      :ASSOC (supply)
      :S-DERIV ((-able adj tr_ability)
                (-ance noun tr_act)
                (-er noun tr_actor))   )
    ( issue2
      :SYNTAX (one-obj io-rec)
      :EXAMPLE (I issued instructions)
      :TYPE p
      :PAR (c-informing)
      :ASSOC (produce)
      :S-DERIV ((-ance noun tr_act))   )
    ( issue3
      :SYNTAX (one-obj no-obj)
      :EXAMPLE (good smells issue from the cake)
      :TYPE s
      :PAR (c-passive-moving)   )))
```

The lexicon, by design, includes only the coarsest distinctions among word senses; thus the financial sense of "issue" (e. g., a new security) falls under the same core sense as the latest "issue" of a magazine. This means that, for a task like database generation, task-specific processing or inference must augment the core lexical knowledge, but avoids many of the problems with considering many nuances of meaning or low-frequency senses. For example, the "progeny" sense of issue, as well as the "exit" sense, are ommitted from our lexicon. The idea is to preserve in the core lexicon the common, coarsest distinctions among senses.

Each entry has a part of speech `:POS` and a set of set of core `:SENSES`. Each core sense has a `:TYPE` field that indicates `p` for all primary senses and `s` for secondary senses. While we are in the process of enriching the information contained in this field, the general rule is that the semantic interpreter should not consider secondary senses without specific contextual information. For example, the word "yard" can mean an enclosed area, a workplace, or a unit of measure, but only the enclosed area sense is considered in the zero-context.

The `:PAR` field links each word sense to its immediate parent in the semantic hierarchy. Without going through the entire hierarchy, it is difficult to convey the semantics of each sense, but the parents and siblings of the two senses of the noun "issue" can give an idea of the coverage of the lexicon. In the output below, word senses are given by a root followed by a sense number, with conceptual categories designated by any atom beginning with `c-`. Explicit derivations are shown by roots followed by endings and additional type specifiers:

```
NOUN_ISSUE1:
PARENT CHAIN:  c-concern c-mental-obj c-obj
               c-entity something

SIBLINGS: (all nouns) regard1      realm2
puzzle1   province2   premonition1 pity1
pet2      parameter1  ground3      goodwill1
feeling2  enigma1     draw2        department2
concern1  cause2      care1        business3
baby2     apprehend-ion-x

NOUN_ISSUE2:
parent chain:  c-published-document c-document
c-phys-obj c-obj c-entity something

SIBLINGS: (all nouns): week-ly-x volume1
transcript1 tragedy2   tome1     thesaurus1
supplement2 strip4     source2   software1
serial1     scripture1 romance2  publication2
profile2    digest1    bible1    paperback1
paper3      paper2     pamphlet1 omnibus1
obituary1   novel1     notice2   month-ly-x
memoir1     map1       manual1   magazine1
library1    journal1   handbook1 anthology1
guide1      grammar1   gazette1  dissertation1
feature4    facsimile1 epic1     encyclopedia1
fiction1    column1    book1     period-ic-al-x
```

```
directory1  copy2       atlas1     dictionary1
comic1      column2     blurb1     catalogue1
calendar1   bulletin1   brochure1  biography1
article1    bibliography1 constitute-ion-x1
```

The basic semantic hierarchy acts as a sense-disambiguated thesaurus [7], under the assumption that in the absence of more specific knowledge word senses will tend to share semantic constraints with the most closely related words. Note that derivative lexical entries, such as `week-ly-x` above, do "double duty" in the lexicon, so that an application program can use the derivation as well as the semantics of the derivative form.

The `:ASSOC` field, not currently used in processing, includes the lexicographer's choice of synonym or closely related words for each sense.

The `:SYNTAX` field encodes syntactic constraints and subcategorizations for each sense. Where senses share constraints (not the case in this example), these can be encoded at the level of the word entry. When the syntactic constraints (such as `io-rec`, `one-obj`, and `no-obj`) influence semantic preferences, these are attached to the sense entry. For example, in this case "issue" used as an intransitive verb would favor "passive moving" even though it is a secondary sense, while the `io-rec` subcategorization in the first two senses means that the ditransitive form will fill the recipient conceptual role. The grammatical knowledge base of the system relates these subcategories to semantic roles.

The `:G-DERIV` and `:S-DERIV` fields mark morphological derivations. `G-DERIV` (NIL in this case to indicate no derivations) encodes these derivations at the word root level, while `S-DERIV` encodes derivations at the sense preference level. We have been gradually moving more of the derivations to the sense level on the basis of corpus analysis. For example, the `S-DERIV` constraint allows "issuance" to derive from either of the first two senses of the verb, with "issuer" and "issuable" deriving only from the "giving" sense.

The derivation triples (such as `(-er noun tr_actor)`) encode the form of each affix, the resulting syntactic category (usually redundant), and the "semantic transformation" that applies between the core sense and the resulting sense. For example, the "issuer" in this case would play the actor role of sense one of the verb *issue*. Because derivations often apply to multiple senses and often result in different semantic transformations (for example, the ending `-ion` can indicate the act of performing some action, the object of the action, or the result of the action), the lexicon often contains strongly "preferred" interpretations, to help control the ambiguity.

The lexicon currently contains 8,775 roots (with noun and verb roots separated) and 13,415 senses. In addition, there are about 10,000 explicit derivations.

In applying the lexicon, the most obvious errors arise from collocational expressions, so the lexicon now includes a substantial number of (currently several hundred) common collocations, such as verb-particles and verb-complement combinations. These expressions are often semantically productive, but the representation of common expressions helps the semantic interpreter to apply preferences. For example, the following is one set of entries for expressions with *take*:

```
( take
   :POS verb
   :SPECIAL
   (( take50
      :S-COMPOUNDS
      ((vc (or (member c-verb_advise2-obj
                c-act-of-verb_blame1
                c-act-of-verb_lose1 noun_profit2)
        c-giving)))
      :EXAMPLE (take delivery)
      :PAR (c-receiving)   )
    ( take51
      :S-COMPOUNDS ((vc (or (member noun_effort1)
                            c-temporal-obj c-energy)))
      :EXAMPLE (the job takes up time))
      :PAR (c-require-rel)   )
    ( take52
      :S-COMPOUNDS ((vc (member noun_news1
        noun_burden1   noun_load2   noun_pressure3
        noun_pressure2 noun_stress1 noun_stress2
        c-act-of-verb_strain1)))
      :PAR (c-managing)   )
     . . .
    ( take58
      :S-COMPOUNDS ((vc (or (member noun_office2
                noun_advantage1 noun_charge1
                c-act-of-verb_control1 noun_command2
                noun_responsibility1) c-structure-rel
        c-shape-rel)))
      :PAR (c-contracting)   )
    ( take59
      :S-COMPOUNDS ((vc (member noun_effect1)))
      :PAR (c-transpire)   )
    ( take60
      :S-COMPOUNDS ((vc (or c-task)))
      :PAR (c-deciding)   ))
```

The above entries contain only the verb-complement (*vc*) relations for "take". Whether these expressions are productive or not, the lexicon can include explicit word sense pairings (such as *take52* with *noun_pressure2*), in which case the collocation helps to discriminate the senses of both verb and complement, or a pairing with a conceptual category (such as *take51* with *c-temporal-obj*), in which case the pairing is more likely to conflict with another but will cover a much broader class of expressions (from *take one's time* to *take years*).

The descriptions above cover most of the central components of the GE lexicon, especially those that allow for general sense preferences in text. The natural question is how to make use of all this knowledge in semantic interpretation. The next section describes the general problem of applying lexical knowledge.

A little knowledge: a dangerous thing

Success with the new lexicon came quickly in retrofitting the knowledge base to domain-specific database genera-

tion systems. By allowing systems to specify key portions of the knowledge base for certain domains (about 300 word senses for MUCK-II), we derived a benefit from the core lexicon (i. e. reducing lexicon-building effort) without introducing many spurious senses. However, the main goal of the effort was to achieve some sort of useful semantic results without *any* customization. The same algorithms applying the core lexicon to arbitrary samples of the Wall Street Journal, not surprisingly, caused some serious problems. After three "easy" sentences, the program encountered the following typical input:

> A form of asbestos once used to make Kent cigarette filters has caused a high percentage of cancer deaths among a group of workers exposed to it more than 30 years ago, researchers reported.

While producing over 100 parses for the above sentence, the program did quite poorly at first in determining the "preferred" sense of each word and even at distinguishing noun forms from verb forms without any "domain" knowledge. This seemed to be a practical example of the "I see a cow" problem (i.e. a large dictionary greatly increases the degree of ambiguity by introducing low-frequency possibilities). After analyzing the program's performance on many examples of this type, we found that we had already reached diminishing returns in adding words and word senses to the lexicon, and that most of the problems with the sense tagging task broke down into three categories: (1) local syntactic preferences, such as that "make" is a verb (as opposed to "a make of car") and "filters" is a noun, (2) simple attachment preferences (such as minimal attachment), and (3) recognition of "senseless" parsing distinctions, such as the multiple attachments of "once used", "among a group...", and "exposed to it" (e. g. whether "among a group" modifies "deaths", "percentage", or "caused" does not affect semantic preferences).

Word sense preferences are only loosely coupled to "traditional" syntactic and semantic preferences. Because many preferences depend on local constraints and lexical relations, we chose to handle some of these issues by using text pre-processing to limit parsing and lexical ambiguity. The next section briefly describes the division of effort between pre-processing and traditional parsing.

Processing and pre-processing

The current style of processing separates both syntactic and semantic analysis into two stages. The pre-processing stage performs a morphological analysis and quick lexical lookup of the text, followed by tagging and bracketing (including stochastic tagging with correction) and collocational analysis. The result of this pre-processing is a pre-filtered version of the text, with some of the lexical ambiguity eliminated as well as baseline semantic preferences. In the second stage, a more complete analysis parses the text in a chart style [8], applying semantic constraints as selectional restrictions and pruning off paths with low semantic "scores".

The motivation for pre-processing, of course, is that ambiguity at the word level is such a problem. Even with our moderate-size lexicon, almost any English content word has more than one part of speech. "Obvious" nouns such as *table* and *case* can appear as verbs, and almost all verbs can appear as nouns. Tagging has emerged as an essential component in corpus-processing systems, removing or reducing this part-of-speech ambiguity. Our effort uses a tagger to perform dynamic part-of-speech tagging based on lexical and morphological analysis. The tagger is in some ways similar to Church's method [9], but is lexicon-dependent (Church's system does not use morphology) and combines statistics with heuristics (such as knowing that words following determiners are nouns, rather than relying only on specific rules). The tagger tags only content words, and processes input at a rate of 500,000 words per hour.

While the tagger uses a simple lexical lookup, we found it useful to use another pre-processing procedure to correct some grammar-specific tagging problems as well as to fetch collocations from the lexicon before parsing. In a few cases, this post-tagging procedure re-introduces ambiguity where the tagger gives a specific choice. While the accuracy of the tagger is highest when it is heavily biased toward nouns, the accuracy of the parser depends on having *some* verbs.

After pre-processing, the TRUMP parser and semantic interpreter [8] go to work on the tagged text, collecting relations (such as modification and verb-complement) at the level of each clause. The sense preference mechanism assigns a total score to each relation to help with attachment, as well as maintaining a weighted vector of preference information that determines the final sense tagging of the text. The final sense tag thus does not depend on having a single final correct parse.

The following list describes the vector of five preference scores for each word sense, along with the five contributing preferences for each role filler:

Sense preferences:
- frequency preference (zero-context)
- morphological pref. (primary or secondary)
- cluster preference (based on topic, etc.)
- collocation (phrase) preference
- syntactic preference of sense

Roel-filler preferences:
- preference for fller, independent of attachment
- role pref. (how well filler fills role)
- rel pref. (how well frame "likes" role)
- base preference (how well role likes frame)
- syntactic preference of filler

Since each word sense can have roles, the relations between head and filler reflect the interactions among sense preferences in the text. The final choice of sense tags thus indirectly reflects correct attachment and syntactic analysis.

Certainly, the bulk of work that remains is in "filling out" these sparse preference vectors. The next section comments on the results produced using the still-sparse preference information.

Current status and interim results

It is still difficult to evaluate the results of this sort of effort. The new lexicon and sense preference mechanism at least shows strong evidence of improved transportability, since the task of customizing the lexicon to a limited domain now takes no more than a day or two. The system also shows excellent lexical and morphological coverage, with well over 90% of non-proper-noun word occurrences covered. When tested on the Wall Street Journal texts (for which there has been no adaptation or customization aside from a company-name recognizer), it rarely produces a single correct parse; however, the partial parses produced generally cover most of the text at the clause level.

The examples of sense-coded text shown earlier are produced by combining clause-level fragments produced by the analyzer. Since most semantic preferences appear at this level (and those that do not, do not depend on syntactic analysis), the results of this sense-coding are encouraging. At least, we have never before been able to produce any partial semantic results from processing arbitrary text.

We made an unsuccessful attempt at evaluating the accuracy of sense-tagging over a corpus. First, we discovered that a human "expert" had great difficulty identifying each sense, and that this task was far more tedious than manual part-of-speech tagging or bracketing. Second, we questioned what we would learn from the evaluation of these partial results, and have since turned our attention back to task-oriented evaluation.

Our next step is to evaluate the effect of text coding on an information retrieval task, by applying traditional term-weighted statistical retrieval methods to the recoded text. One intriguing aspect of this approach is that errors in distinguishing sense preferences should not be too costly in this task, so long as the program is fairly consistent in its disambiguation of terms in both the source texts and the input queries. At the same time, we will be applying the system to much broader database generation projects than those of SCISOR and MUCK-II.

Conclusion

We have developed a substantial knowledge base for text processing, especially a word-sense-based lexicon, and applying this new lexicon to semantic interpretation and database generation. In database generation, the new knowledge base has proven successful in reducing the amount of lexicon engineering required, although current database generation tasks are still too small for this effect to be more than marginal. In generic text processing, there are some encouraging results from applying the system to sense tagging of arbitrary text. We expect to evaluate these results on tasks in information retrieval, and, later, machine translation, to determine the likelihood of achieving substantive improvements through sense-based semantic analysis.

References

[1] 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.

[2] Madeleine Bates and Robert 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.

[3] Paul S. Jacobs and Lisa F. Rau. The GE NLToolset: A software foundation for intelligent text processing. In *Proceedings of the Thirteenth International Conference on Computational Linguistics*, Helsinki, Finland, 1990.

[4] Lisa F. Rau and Paul S. Jacobs. Integrating top-down and bottom-up strategies in a text processing system. In *Proceedings of Second Conference on Applied Natural Language Processing*, pages 129–135, Morristown, NJ, Feb 1988. ACL.

[5] Beth Sundheim. Second message understanding conference (MUCK-II) test report. Technical Report 1328, Naval Ocean Systems Center, San Diego, CA, 1990.

[6] R. Krovetz. Lexical acquisition and information retrieval. In U. Zernik, editor, *First International Lexical Acquisition Workshop*. 1989.

[7] E. Fox, J. Nutter, T. Ahlswede, M. Evens, and J. Markowitz. Building a large thesaurus for information retrieval. In *Proceedings of Second Conference on Applied Natural Language Processing*. Association for Computational Linguistics, February 1988.

[8] P. Jacobs. TRUMP: A transportable language understanding program. Technical Report CRD89/181, General Electric Corporate Research and Development, Schenectady, NY, 1989.

[9] K. Church, W. Gale, P. Hanks, and D. Hindle. Parsing, word associations, and predicate-argument relations. In *Proceedings of the International Workshop on Parsing Technologies*, Carnegie Mellon University, 1989.