Speech understanding systems: summary of results of the five-year research effort at Carnegie-Mellon University.

Carnegie-Mellon University, Computer Science Dept.

Follow this and additional works at: http://repository.cmu.edu/compsci

This Technical Report is brought to you for free and open access by the School of Computer Science at Research Showcase @ CMU. It has been accepted for inclusion in Computer Science Department by an authorized administrator of Research Showcase @ CMU. For more information, please contact research-showcase@andrew.cmu.edu.
NOTICE WARNING CONCERNING COPYRIGHT RESTRICTIONS:
The copyright law of the United States (title 17, U.S. Code) governs the making of photocopies or other reproductions of copyrighted material. Any copying of this document without permission of its author may be prohibited by law.
SPEECH UNDERSTANDING SYSTEMS

Summary of Results of the Five-Year Research Effort at Carnegie-Mellon University

Carnegie-Mellon University
Department of Computer Science
Pittsburgh, Pennsylvania 15213

First Version printed September 1976
Present Version printed August 1977

This research was supported by the Defense Advanced Research Projects Agency under contract no. F44620-73-C-0074 and monitored by the Air Force Office of Scientific Research.
PREFACE

This report is an augmented version of a report originally issued in September of 1976, during the demonstration at the end of the five-year speech effort. The first section reports on the various speech understanding systems developed at CMU during the five year period and highlights their individual contributions. Section II contains a brief description of several techniques and knowledge sources that contributed to the success of the final systems. Section III gives detailed performance results of the Harpy and Hearsay-II systems. Results include the performance of the systems not only for the 1000 word task but for several simpler tasks. Section IV contains reprints of papers presented at various conferences since September 1976. Section V contains a list of publications of the CMU speech group.

The CMU Speech Group gratefully acknowledges the following contributions which have been instrumental to the successful conclusion of the five-year speech understanding systems research effort at Carnegie-Mellon University:

Howard Wactlar, Director of our Computer Facility, for his untiring efforts in providing a smoothly working real-time computing environment for speech understanding systems research.

Carolyn Council, Mark Faust, Bill Haffey, John Paulson, and other members of the operations staff for providing a highly cooperative and reliable operating environment.

Bill Broadley, Stan Kriz, Rich Lang, Paul Newbury, Mike Powell, Brian Rosen, and Jim Teter of the engineering group who designed and maintained the special-purpose systems needed for this research. A special thanks to Mark Firley and Ken Stupak for their superb engineering support.

Allen Newell for giving freely of his time and ideas to foster this research.

Joe Traub and the Faculty of the Department of Computer Science for their help in facilitating this research.

Other individuals and groups working in this area for providing a stimulating, intellectual atmosphere in which to solve this difficult problem.

Dave Carlstrom, Steve Crocker, Cordell Green, Lick Licklider, and Larry Roberts for providing a research management environment which makes breakthroughs possible.
# TABLE OF CONTENTS

## I. Multi-system Approach to Speech Understanding

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>Systems</td>
<td>4</td>
</tr>
<tr>
<td>Hearsay-I</td>
<td></td>
</tr>
<tr>
<td>Dragon</td>
<td></td>
</tr>
<tr>
<td>Harpy</td>
<td></td>
</tr>
<tr>
<td>Hearsay-II</td>
<td></td>
</tr>
<tr>
<td>Locust</td>
<td></td>
</tr>
<tr>
<td>Parallel Systems</td>
<td>8</td>
</tr>
<tr>
<td>Discussion</td>
<td></td>
</tr>
</tbody>
</table>

## II. Knowledge Sources and Techniques

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>The ZAPDASH Parameters, Feature Extraction, Segmentation, and Labeling</td>
<td>10</td>
</tr>
<tr>
<td>for Speech Understanding Systems</td>
<td></td>
</tr>
<tr>
<td>A Syllable Based Word Hypothesizer for Hearsay-II</td>
<td>11</td>
</tr>
<tr>
<td>WIZARD: A Word Verifier for Hearsay-II</td>
<td>12</td>
</tr>
<tr>
<td>Word Pair Adjacency Acceptance Procedure in Hearsay-II</td>
<td>15</td>
</tr>
<tr>
<td>Syntactic Processing in Hearsay-II</td>
<td>16</td>
</tr>
<tr>
<td>Focus and Control in Hearsay-II</td>
<td>18</td>
</tr>
<tr>
<td>Policies for Rating Hypotheses, Halting, and Selecting a Solution</td>
<td>19</td>
</tr>
<tr>
<td>in Hearsay-II</td>
<td></td>
</tr>
<tr>
<td>Semantics and Pragmatics in Hearsay-II</td>
<td>22</td>
</tr>
<tr>
<td>Discourse Analysis and Task Performance in Hearsay-II</td>
<td>24</td>
</tr>
<tr>
<td>Parallel Processing in Speech Understanding Systems</td>
<td>28</td>
</tr>
<tr>
<td>Parallelism in Artificial Intelligence Problem-Solving</td>
<td></td>
</tr>
<tr>
<td>The HSIIC.mmp System</td>
<td></td>
</tr>
<tr>
<td>A Parallel Production System for Speech Understanding</td>
<td></td>
</tr>
</tbody>
</table>

## III. Performance Measurement

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance of the Harpy and Hearsay-II Systems</td>
<td>32</td>
</tr>
<tr>
<td>Connected Digit Recognition using Symbolic Representation</td>
<td></td>
</tr>
<tr>
<td>of Pronunciation Variability</td>
<td>37</td>
</tr>
<tr>
<td>Effects of Branching Factor and Vocabulary Size on Performance</td>
<td>39</td>
</tr>
<tr>
<td>Appendix III-A: Test sentences</td>
<td>41</td>
</tr>
<tr>
<td>Appendix III-B: AI Retrieval Language Dictionary</td>
<td>46</td>
</tr>
<tr>
<td>Appendix III-C: AI Retrieval Language Grammars</td>
<td>63</td>
</tr>
</tbody>
</table>
IV. Collected papers

A Functional Description of the Hearsay-II Speech Understanding System... 131
Selection of Word Islands in the Hearsay-II Speech Understanding System.. 135
Word Verification in the Hearsay-II Speech Understanding System........ 139
The d’ Model of Signal Detection Applied to Speech Segmentation........ 143
An Application of Connected Speech to the Cartography Task............... 146
Dynamic Speaker Adaptation in the Harpy Speech Recognition System....... 150
Use of Segmentation and Labeling in Analysis-Synthesis of Speech......... 153
A Halting Condition and Related Pruning Heuristic for Combinatorial Search 158

V. CMU Computer Science Department Speech Publications

An annotated bibliography of speech group publications.
INTRODUCTION

In 1971, a group of scientists recommended the initiation of a five-year research program towards the demonstration of a large-vocabulary connected speech understanding system (Newell et al., 1971). Instead of setting vague objectives, the group proposed a set of specific performance goals (see Fig. 1.1 of Newell et al., 1971). The system was required to accept connected speech from many speakers based on a 1000 word vocabulary task-oriented grammar, within a constrained task. The system was expected to perform with less than 10\% semantic errors, using about 300 million instructions per second of speech (MIPSS)** and to be operational within a five year period. The proposed research was a highly ambitious undertaking, given the almost total lack of experience with connected speech systems at that time.

The Harpy and Hearsay-II systems developed at Carnegie-Mellon University had the best overall performance at the end of the five year period. Figure 1 illustrates the performance of the Harpy system relative to the original specifications. It not only satisfies the original goals, but exceeds some of the stated objectives. It recognizes speech from male and female speakers using a 1011-word-vocabulary document retrieval task. Semantic error is 5\% and response is an order of magnitude faster than expected. The Hearsay-II system achieves similar accuracy and runs about 2 to 20 times slower than Harpy.

Of the many factors that led to the final successful demonstration of these systems, perhaps the most important was the systems development methodology that evolved. Faced with prospects of developing systems with a large number of unknowns, we opted to develop several intermediate “throw-away” systems rather than work towards a single carefully designed ultimate system. Many dimensions of these intermediate systems were deliberately finessed or ignored so as to gain deeper understanding of some aspect of the overall system. The purpose of this paper is to

<table>
<thead>
<tr>
<th>GOAL (Nov. 1971)</th>
<th>HARPY (Nov. 1976)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept connected speech from many cooperative speakers in a quiet room using a good microphone with slight tuning/speaker accepting 1000 words using an artificial syntax in a constraining task yielding &lt; 10% semantic error requiring approx. 300 MIPSS**</td>
<td>Yes 5 (3 male, 2 female) yes computer terminal room close-talking microphone 20-30 sentences/talker 1011 word vocabulary avg. branching factor = 33 document retrieval 5/ requiring 28 MIPSS using 256k of 36 bit words costing $5 per sentence processed</td>
</tr>
</tbody>
</table>

Figure 1. Harpy performance compared to desired goals.

** The actual specifications stated “a few times real-time” on a 100 MIPS (Million instructions per second) machine.
<table>
<thead>
<tr>
<th>Task characteristics</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>speakers; number, male/female, dialect</td>
<td>vocabulary and syntax</td>
</tr>
<tr>
<td>response desired</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Signal gathering environment</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>room noise level</td>
<td>transducer characteristics</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Signal transformations</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>digitization speed and accuracy</td>
<td>special-purpose hardware required</td>
</tr>
<tr>
<td>parametric representation</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Signal-to-symbol transformation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>segmentation?</td>
<td>level transformation occurs</td>
</tr>
<tr>
<td>label selection technique</td>
<td>amount of training required</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Matching and searching</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>relaxation: breadth-first</td>
<td>blackboard: best-first, island driven</td>
</tr>
<tr>
<td>productions: best-first</td>
<td>Locus: beam search</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Knowledge source representation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>networks</td>
<td>procedures</td>
</tr>
<tr>
<td>frames</td>
<td>productions</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System organization</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>levels of representation</td>
<td>single processor / multi-processor</td>
</tr>
</tbody>
</table>

Figure 2. Design choices for speech understanding systems.

illustrate the incremental understanding of the solution space provided by the various intermediate systems developed at CMU.

Figure 2 illustrates the large number of design decisions which confront a speech understanding system designer*. For each of these 10 to 15 design decisions, we have 3 to 10 feasible alternative choices. Thus the solution space for speech systems seems to contain $10^6$ to $10^8$ possible system designs. Given the interactions between design choices, it is not possible to evaluate each design choice in isolation outside the framework of the total system.

* Further discussion of many of these design choices can be found in Reddy (1976).
Figure 3. CMU Speech Understanding Systems Genealogy
Figure 3 shows the genealogy of the speech understanding systems developed at CMU. In this section we will briefly outline the interesting aspects of each of these systems and discuss their contributions towards the development of speech understanding systems technology. More complete descriptions of these systems can be found in the references listed at the end.

The Hearsay-I System (Erman, Fennell, Lowerre, Neely, and Reddy)*

Hearsay-I (Reddy, Erman and Neely 1973; Reddy, Erman, Fennell and Neely 1973), the first speech understanding system developed at Carnegie-Mellon University, was demonstrated in June of 1972. This system was one of the first connected speech understanding systems to use task dependent knowledge to achieve reduction of the search space. Recognition uses a best-first search strategy.

Model

Hearsay-I was the first system to utilize independent, cooperating knowledge sources and the concept of a global data base, or "blackboard", through which all knowledge sources communicate. Knowledge sources consist of the acoustic-phonetic, syntactic, and semantic modules. Each module operates in the "hypothesize-and-test" mode. Synchronous activation of the modules leads to a best-first search strategy. Several other systems have used this strategy (Forgie 1974). This system was one of the first to use syntactically derived word diagrams and trigrams, as anti-productions (Neely 1973), to predict forward and backward from "islands of reliability". Task dependent knowledge, such as a board position in the chess task, is used by the semantic module (Neely 1973) to reject meaningless partial parses early in the recognition process. The acoustic-phonetic module uses amplitude and zero-crossing parameters to obtain a multilevel segmentation into syllable-size and phoneme-size units (Erman, 1974).

Performance

Over a wide range of tasks, the average sentence error rate was 0.697 with a word error rate of 457. Speed varied between 3 and 15 MIPSS over 162 utterances containing 578 words. Hearsay-I yields much higher accuracies on tasks with which it is carefully trained. For the chess task, for instance, average sentence and word error rates were 21 and 7 percent, respectively, with an average speed of 2 MIPSS.

Discussion

Hearsay-I, as a successful connected-speech understanding system, served to clarify the nature and necessary interaction of several sources of knowledge. Its flexibility provided a means for testing and evaluating competing theories, allowing the better theories to be chosen as a basis for later systems. In retrospect, we believe this system organization would have been adequate for the ARPA specifications given present acoustic-phonetic knowledge.

* The principle contributors towards the development of each of these systems are listed within parentheses.
The Dragon System (Baker)

Baker formulated the recognition process as a dynamic programming problem. The Dragon recognition system (Baker, 1975), based on this model was first demonstrated in April of 1974. The system was motivated by a desire to use a general abstract model to represent knowledge sources. The model, that of a probabilistic function of a Markov process, is flexible and leads to features which allow it to function despite high error rates. Recognition accuracy was greater with Dragon than with Hearsay-I, but the system ran significantly slower.

Model

Dragon was the first system to demonstrate the use of a Markov model and dynamic programming in a connected speech understanding system. It included several interesting features, such as delayed decisions and integrated representation, and is based on a general theoretical framework. The general framework allows acoustic-phonetic, syntactic, and semantic knowledge to be embodied in a finite-state network. Each path through this precompiled network represents an allowed pronunciation of a syntactically acceptable sentence. Recognition proceeds left-to-right through the network, searching all possible paths in parallel to determine the globally optimal path (i.e., the path which best matches the spoken utterance). Acoustic inputs are peak-to-peak amplitudes and zero-crossings from overlapping, one-third octave filters, sampled every centi-second.

Performance

Recognition accuracy was greater with Dragon than that obtained with Hearsay-I, but at a cost of speed, Dragon being approximately 5 to 10 times slower. Over a wide variety of tasks, the average sentence error rate was 517. Speed ranged from 14 to 50 MIPSS. The computation is essentially linear with the number of states in the Markov network. Performance was later improved by Lowerre (Lowerre, 1976).

Discussion

Dragon, with more accurate performance than Hearsay-I, served to stimulate further research into factors that led to its improved performance. Many of the ideas motivating its design were important in the development of subsequent connected-speech understanding systems. Although later systems do not use the Markov Model and do not guarantee finding the globally optimal path, the concepts of integrated representation of knowledge sources and delayed decisions proved to be very valuable.

The Harpy System (Lowerre and Reddy)

The Harpy system (Lowerre 1976) was the first connected speech system to satisfy the original specifications given in the Newell report and was first demonstrated in September of 1976. System design was motivated by an investigation of the important design choices contributing to the success of the Dragon and Hearsay-I systems. The result was a combination of the "best" features of these two systems with additional heuristics to give high speed and accuracy.

Model

The Harpy system uses the locus model of search. The locus model of search, a very successful search technique in speech understanding research, is a graph-searching technique in which all except a beam of near-miss alternatives around the
best path are pruned from the search tree at each segmental decision point, thus containing the exponential growth without requiring backtracking. This technique was instrumental in making Harpy the most successful connected speech understanding system to date. Harpy represents syntactic, lexical, and juncture knowledge in a unified network as in Dragon, but without the a-priori transition probabilities. Phonetic classification is accomplished by a set of speaker-dependent acoustic-phonetic templates based on LPC parameters which represent the acoustic realizations of the phones in the lexical portion of the network.

Performance

The system was tested on several different tasks with different vocabularies and branching factors. On the 1011-word task using the AIX05 grammar (see Appendix III-C), the system word error rate was 37 and the semantic error rate was 57 (see fig. 1). The system was also tested with connected digits recognition attaining a 27 word error rate. Using speaker-independent templates, error rate increases to 77 over 20 speaker including 10 new speakers. Using telephone input increases the error rate to 77 to 117 depending on the noise characteristics of the telephone system.

Discussion

Backtracking and redundant computation have always been problematic in AI systems. The Harpy system eliminates these in an elegant way, using the beam search technique. By compiling knowledge ahead of time, Harpy achieves a level of efficiency that is unattainable by systems that dynamically interpret their knowledge. This permits Harpy to consider many more alternatives and deal with error and uncertainty in a graceful manner.

The Hearsay-II System (Erman, Hayes-Roth, Lesser, and Reddy)

Hearsay-II has been the major research effort of the CMU speech group over the last three years. During this period, solutions were devised to many difficult conceptual problems that arose during the implementation of Hearsay-I and other earlier efforts. The result represents not only an interesting system design for speech understanding but also an experiment in the area of knowledge-based systems architecture. Attempts are being made by other AI groups to use this type of architecture in image processing and other knowledge-intensive systems.

Hearsay-II is similar to Hearsay-I in that it is based on the hypothesize-and-test paradigm, using cooperating independent knowledge sources communicating through a global data structure (blackboard). It differs in the sense that many of the limitations and shortcomings of Hearsay-I are resolved in Hearsay-II.

Hearsay-II differs from the Harpy system in that it views knowledge sources as different and independent and thus cannot always be integrated into a single representation. Further, it has as a design goal the ability to recognize, understand, and respond even in situations where sentences cannot be guaranteed to agree with some predefined, restricted language model as is the case with the Harpy system.

Model

The main features of the Hearsay-II system structure are: 1) the representation of knowledge as self-activating, asynchronous, parallel processes, 2) the representation of the partial analysis in a generalized three-dimensional network; the dimensions being level of representation (e.g., parametric, segmental, syllabic, lexical, syntactic), time, and alternatives, with contextual and structural support connections explicitly specified, 3) a modular structure for incorporating new knowledge into the system at any level, and 4) a system structure suitable for execution on a parallel processing system.
Performance

The present system has been tested using about 100 utterances of the training data for the 1011-word vocabulary task. For a grammar with simple syntax (AIX05, the same one used by Harpy), the sentence error rate is about 16% (semantic error 16%). For a grammar with more complex syntax (AIX15, see appendix III-C), the sentence error rate is about 42% (semantic error 26%). The system runs about 2 to 20 times slower than Harpy.

Discussion

Hearsay-II represents an important and continuing development in the pursuit of large-vocabulary speech understanding systems. The system is designed to respond in a semantically correct way even when the information is fuzzy and only partial recognition is achieved. Independent knowledge sources are easily written and added to Hearsay-II; knowledge sources may also be removed in order to test their effectiveness. The Hearsay-II system architecture offers great potential for exploiting parallelism to decrease recognition times and is capable of application to other knowledge-intensive AI problems dealing with errorful domains. Many more years of intensive research would be necessary in order to evaluate the full potential of this system.

The Locust System (Bisiani, Greer, Lowerre, and Reddy)

Present knowledge representation and search used in Harpy tend to require much memory and are not easily extendable to very large languages (vocabularies of over 10,000 words and more complex syntax). But we do not view this as an insurmountable limitation. Modified knowledge representation designed for use with secondary memories and specialized paging should overcome this difficulty. In addition, it appears larger-vocabulary speech understanding systems can be implemented on mini-computers without significant degradation in performance. Locust is designed to demonstrate the feasibility of these ideas.

Model

The model is essentially the same as the Harpy system except, given the limitations of storage capacity of main memory, the knowledge representation has to be reorganized significantly. The network is assumed to be larger than main memory, stored on secondary memory, and retrieved using a specialized paging mechanism. The choice of the file structure representation and clustering of the states into pages of uniform size are the main technical problems associated with the development of this system.

Discussion

A paging system for the 1011 word vocabulary is currently operational on a PDP-11/40E and has speed and accuracy performance comparable to Harpy on a PDP-10 (KA10). Simulation of various paging models is currently in progress. As memories with decreased access times become available, this class of systems is expected to perform as accurately and nearly as fast as systems requiring no secondary memory.

Parallel Systems (Feiler, Fennell, Lesser, McCracken, and Oleinick)

Response time for the present systems is usually greater than real-time, with indications that larger vocabularies and more complex syntax will require more time for search. One method of achieving greater speed is to use parallel processing. Several systems designed and developed at CMU exploit multi-processor hardware such as C.mmp and Cm*.
Models

Several systems are currently under development as part of multi-processor research projects which attempt to explore potential parallelism of Hearsay and Harpy-like systems. Fennell and Lesser (1977) studied the expected performance of parallel Hearsay systems and issues of algorithm decomposition. McCracken (1977) is studying a production system implementation of the Hearsay model. Oleinick (1977) and Feiler (1977) are studying parallel decompositions of the Harpy algorithm. Several of these studies are not yet complete, but preliminary performance results are very encouraging. Oleinick has demonstrated a version of Harpy that runs faster than real-time on C.mmp for several tasks.

Discussion

The main contribution of these system studies (when completed) will be to show the degree of parallelism which can reasonably be expected in complex speech understanding tasks. Attempts to produce reliable and cost-effective speech understanding systems would require extensive studies in this direction.

DISCUSSION

In the previous section we have briefly outlined the structure and contributions of various speech systems developed at CMU. In retrospect, it is clear that the slow rate of progress in this field is directly attributable to the large combinatorial space of design decisions involved. Thus, one might reasonably ask whether the human research strategy in solving this and other similar problems can benefit from search reduction heuristics that are commonly used in AI programs. Indeed, as we look around, it is not uncommon to find research paradigms analogous to depth-first exploration, breadth-first with shallow cut-off, backtracking, "jumping-to-conclusions", thrashing, and so on.

Our own research has been dominated by two such paradigms. First is a variant of best-first search: find the weakest link (and thus the potential for most improvement) in the system and attempt to improve it. Second is a variant of the beam search: when several alternative approaches look promising, we use limited parallel search with feed-forward. The systems shown in Figure 3 are examples of this type of system iteration and multi-systems approach.

Many system design decisions require an operational total systems framework to conduct experiments. However, it is not necessary to have a single system that permits all possible variations of system designs. Given enough working components, with well-designed interfaces, one can construct new system variants without excessive effort.

The success of the speech understanding research effort is all the more interesting because it is one of the few examples in AI research of a five year prediction that was in fact realized on time and within budget. It is also one of the few examples in AI where adding additional knowledge can be shown to lead to system speed-up as well as improved accuracy.

We note in conclusion that speech understanding research, in spite of the many superficial differences, raises many of the same issues that are central to other areas of AI. Faced with the problem of reasoning in the presence of error and uncertainty, we generate and search alternatives which have associated with them a likelihood value representing the degree of uncertainty. Faced with the problem of finding the most plausible symbolic description of the utterance in a large combinatorial space, we use techniques similar to those used in least-cost graph searching methods in problem
solving. Given the problems of acquisition and representation of knowledge, and control of search, techniques used in speech are similar to most other knowledge intensive systems. The main difference is that given human performance the criteria for success, in terms of accuracy and response time, far exceed the performance requirements of other AI tasks except perhaps vision.

ACKNOWLEDGMENT

I would like to thank Gary Goodman and Lee Erman for their help and comments in the preparation of this paper.

References


II. KNOWLEDGE SOURCES AND TECHNIQUES

The Zapdash Parameters, Feature Extraction, Segmentation, and Labeling for Speech Understanding Systems (Goldberg, Reddy, and Gill)

Introduction

In spite of early success with very simple parametric representations of speech (see Reddy 1966 and Erman 1974), recent emphasis has been on highly accurate but computationally expensive parameter extraction techniques such as LPC spectral analysis, formant tracking, etc. We feel that simpler, more efficient methods must first be applied to reduce the amount of input data before more expensive analysis is performed. The uniform application of LPC analysis to all the input produces accurate but very redundant results, and at high cost. (see Goldberg 1975)

Our approach involves two levels of parameter extraction and analysis. The first level produces an accurate segmentation with strong clues as to manner of articulation and phonetic identity of the segments. For this purpose, we have developed the ZAPDASH parameters, described below. They provide a highly efficient basis for an accurate, robust segmenter and broad classifier. After the phonetic elements are isolated, a uniform LPC labeling stage is applied only where it is needed to further refine the segment identification. Preliminary evaluations show significant computational savings is possible with no sacrifice of segmentation or labeling accuracy.

The ZAPDASH Parametric Representation

As digital processing of speech becomes commonplace, it becomes desirable to have a parametric representation of speech which is simple, fast, accurate, and directly obtainable from the PCM representation of speech. The ZAPDASH representation of speech (Zero-crossings And Peaks of Differenced And Smooth waveforms) is of this nature. An important means of reducing computational cost in much of the low level processing of speech is to reduce the quantity of data in the input representation to the minimum necessary for accurate analysis of the phonetic content of the speech signal. Our past experience shows that very simple measures of activity in the low and the high frequency bands (approximately: <1kHz. and >1kHz.) would suffice for all but the fine labeling stage. Peak-to-peak amplitudes and zero-crossing counts provide simple measures of the amount of activity within each particular band. In ZAPDASH, the PCM data is used to generate a differenced waveform and a down-sampled, smoothed waveform (for 10KHz sampling rate, the smoothing FIR filter coefficients were -1 0 1 2 4 4 2 1 0 -1, used every 4th point). Peak-to-peak distances and number of zero-crossings are calculated each 10 ms, resulting in 400 8-bit parameters per second of speech. ZAPDASH can be calculated in 15 to 20 computer instructions per sample and, therefore, can be extracted in less than a 1/3 real time on minicomputers with 2 micro-sec. instruction time. A simple parametric representation like ZAPDASH appears to provide sufficient information for accurate phone segmentation, thus sharply reducing the amount of more detailed spectral analysis required by many other methods. The resulting four parametric measurements (Smoothed Peak-to-peak, Smoothed Zero-crossing, Differenced Peak-to-peak, and Differenced Zero-crossing) are sufficient to detect, with reasonable accuracy, a set of 10 features, described below, which are quite useful for both segmentation and initial broad labeling. The ZAPDASH parameters are used by the first stage segmenter to make decisions on manner of articulation. The resulting segmentation and broad classification is accurate yet inexpensive. Further refinement of the segment labels using spectral analysis is then much more economical.
Segmentation and Broad Classification

The first stage of the program contains an hierarchical, feature-extraction based segmenter and classifier. A number of features relating to manner of articulation are extracted. Silence, voicing, frication, front-back placement, high-low placement, consonant-like, flap-like, aspiration-like, nasal, and sibilant decisions are made using the ZAPDASH parameters. In the processing of an utterance, a set of segments is chosen, with broad classification, for the entire utterance. These identify regions of the signal such as SIL-silence, SON-sonorant, UFR-unvoiced fricative, VBK-back vowel, etc. Further sub-segmentation and/or reclassification is conditional upon segment class type, context, and feature values. There are 59 classes currently used internally, although many overlap one another in the acoustic space.

Modified LPC Labeling

At the second stage, where no further refinement is possible using the ZAPDASH information, a fine labeler is applied at the mid-points of all segments. The original PCM signal is compared against stored templates by a modified LPC distance metric. Itakura's minimum prediction residual metric (Itakura 1975) is used to compare the segment mid-point to a set of speaker-specific trained templates. The segment class is used to provide a sub-set of the approximately 100 templates, or a set of a priori weights to be added to the metric values for all templates. In this way, the manner-of-articulation and the contextual information provided by the earlier feature extraction improve the labeling.

Results

The highly efficient segmentation procedures in the first level segmenter and the limitation upon the need for LPC analysis provide a factor of 5 speedup over the uniform procedures used by HARPY and Hearsay-II. Preliminary tests with this program indicate that results for HARPY using this parameterization will be just as accurate and will be computed faster than the results obtained with the more redundant parameterization it now uses. Present performance of ZAPDASH can be summarized as follows: Segmentation -- less than 20% extra segments, less than 2% missed segments, and boundary placement within an average of 10 ms. of the manually defined location. Labeling (broad classes) -- 90% correct, (finer labeling) -- correct template in first place 50% of the time, in the first five places 75% of the time. A more detailed evaluation will be available shortly.

References


A Syllable Based Word Hypothesizer for Hearsay-II (Smith)

Problem and Motivation

A central problem for speech understanding systems is efficiently and accurately determining what words are implied at the lexical level by the data at lower levels. One solution to the problem is to map each word hypothesized by syntactic and semantic information to the lower level representation, then match and rate the word.
But as speech systems permit larger vocabularies and languages with less restricted syntax and semantics, they must depend more on bottom-up methods to limit the search space of possible word sequences. The effectiveness of a hypothesizer can be measured by the percent of the correct words and the number of competing words it hypothesizes. One method of bottom up word hypothesis is to go directly from the phone sequences found for the utterance to word hypotheses as in the BBN HWIM speech system (Klovstad, 1976). The solution used in Hearsay-II uses an intermediate level of syllables between the words and phone segments.

Solution

The word hypothesizer uses equivalence classes of syllables (called Syltypes) to support word hypotheses (Smith, 1976). These Syltypes were defined so that syllables which were likely to be given similar segments and labels by the speech system would have the same Syltype. No attempt is made by the word hypothesizer to distinguish between words which have the same sequence of Syltypes. The word verifier later makes this distinction as it rates the words.

The Syltypes we now use are defined by a sequence of states corresponding to phoneme equivalence classes. A Markov probability model relates the state sequence of a Syltype to the segment labels hypothesised by the segmenter and labeler. A word may be hypothesised by the following sequence of events: For each syllable nucleus in the utterance (defined by a heuristic using segment labels and an amplitude function), the most likely Syltype state sequences are found by searching the segments from the nucleus out to adjacent nuclei, or perhaps the utterance boundaries. For each Syltype hypothesized with a "good" rating the set of words containing syllables mapping to the Syltype, are retrieved using an inverted lexicon. A multi-syllabic word in the set is rejected if it matches poorly with adjacent Syltype hypotheses. The word verifier is then called to rate each word. Those with a poor rating are rejected.

Results

Since the word hypothesizer's ratings for words are used only to determine whether to reject the word or to verify the word, it is used as a filter for the word verifier. The performance relevant to this task is the percentage of the spoken words correctly hypothesized and the fraction of the vocabulary hypothesized per spoken word. The results from twenty test sentences indicate that, for a 1011 word vocabulary, 67% of the correct words are hypothesized when 80 words are hypothesized per spoken word (8% of the vocabulary). Of course these numbers can be varied by changing thresholds. If the speech system can function with only 57% of the correct words hypothesized bottom-up, then only 51 words need to be hypothesized per spoken word (5% of the vocabulary). Similarly, higher accuracy can be obtained with a greater number of competing word hypotheses.

References


Wizard: A Word Verifier for Hearsay-II (McKeown)

Problem and Motivation

A key problem for speech understanding systems is the verification of word hypotheses generated by various knowledge sources in the system. The verifier must assign a likelihood score which is commensurate with the match between the
underlying acoustic data and the phonetic description of the word. The goodness of a score may be only temporally significant; the scores should rank order competitive words in any time area such that the correct word is high in the ordering. In addition to this acceptance criteria, it is also necessary for the verifier to reject absolutely a large percentage of the hypothesized words, without rejecting a significant number of correct words, in order to constrain the combinatorics at higher levels.

Solution

In HEARSAY II, words may be generated bottom-up by the word hypothesizer (POMOW) or predicted top-down by the syntax and semantics module (SASS). Each uses a very different strategy for verification since bottom-up hypothesis have a known approximate begin/end time while top-down hypotheses use a verified word to predict words to the left or right, and thus only one time is known.

The word verifier, WIZARD, uses a general Markov model for speech recognition (BAKER, 1975; LOWERRE, 1976). The acoustic information is a segmentation of the utterance where each segment is represented as a vector of phoneme probabilities. Each word in the lexicon is represented by a statically defined network which embodies alternate pronunciations of the word. This model finds the optimal path through the word network and assigns as the word score a normalized sum of all the log-probabilities for states (phonemes) on that path. Networks do not take into account word junctures but do handle internal phoneme junctures. Thus WIZARD attempts to verify words as if they exist in isolation.

Wizard handles bottom-up words in the following manner: The predicted begin/end times are mapped into their respective begin/end segments: bseg/eseg. All paths which begin at bseg-1/bseg/bseg+1 and end at eseg-1/eseg/eseg+1 are explored in parallel. Each of the nine possible optimal mappings is examined and the best of these is chosen as the mapping of the word network over the segmented acoustic data. This possible time shifting allows the verifier to recover from incorrect times due to differences in representation of the acoustic data between knowledge sources. As a result, the verifier may change times on word hypotheses as well as rate them.

Words which are hypothesized top-down pose a different problem in terms of verification, since only the begin or end time is known. In this mode it is necessary for WIZARD to predict the missing time as well as to return a rating. A major problem is bounding the number of segments considered in a prediction. Currently several heuristics are employed. Since all states on the optimal path must be mapped to at least one segment, the lower bound on the number of segments is the minimal number of network transitions (mintran). An upper bound was experimentally determined to be 4*mintran, thus on the average no more than 4 segments are mapped into any one state. This number is a function of the segmentation, which tends to over-segment, and the network descriptions, which allow reduced spellings. The POMOW word hypothesizer generates an upper bound based on the expected number of vowel nuclei in the word and their position relative to the beginning of the prediction. The smaller of these upper bounds is used. WIZARD iteratively maps each of the segments from the given begin segment to the upper bound. It considers those mappings which fall between the lower and upper bounds and picks the best after appropriate normalization. The time of the best end segment is returned along with the rating.

Results and Conclusions

The results summarized in Table 1 are for five data sets, containing 100 utterances, in which 332 correct words were hypothesized bottom-up by POMOW. In addition, 13053 incorrect words were generated. The vocabulary size for POMOW and WIZARD was approximately 550 words. WIZARD rated each of the words using begin/end times generated bottom-up. Each verification took, on the average, 100ms of CPU time on a DEC PDP-10 (KA). For each rating threshold (15,10) the number of correct and incorrect words that were accepted or rejected is tabulated. From this
data the number of words hypothesized per word position and the percent of the vocabulary hypothesized per word position can be calculated. These numbers give a vocabulary independent measure of performance, allowing comparisons between various system configurations. An average rank order of the correct word is provided which measures, at each threshold, the number of words in each word position that must be examined in order to include the correct word. The range of rank orders between the data sets (20 utterances/set) is also indicated.

### Table I

<table>
<thead>
<tr>
<th>THR</th>
<th># HYPED BY POMOW</th>
<th>ACCEPTED</th>
<th>REJECTED</th>
<th>5.6 RANK ORDER</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>332</td>
<td>328 (98%)</td>
<td>6 (2%)</td>
<td>(3.6 - 7.1)</td>
</tr>
<tr>
<td></td>
<td>13053</td>
<td>10426 (80%)</td>
<td>2627 (20%)</td>
<td></td>
</tr>
<tr>
<td>TOTAL</td>
<td>13385</td>
<td>10752 (80%)</td>
<td>2633 (20%)</td>
<td></td>
</tr>
<tr>
<td>#/WORD POS</td>
<td>40 (8%)</td>
<td>32 (6%)</td>
<td>8 (2%)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>THR</th>
<th># HYPED BY POMOW</th>
<th>ACCEPTED</th>
<th>REJECTED</th>
<th>4.5 RANK ORDER</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>332</td>
<td>312 (94%)</td>
<td>20 (6%)</td>
<td>(3.4 - 5.6)</td>
</tr>
<tr>
<td></td>
<td>13053</td>
<td>6462 (49%)</td>
<td>6591 (51%)</td>
<td></td>
</tr>
<tr>
<td>TOTAL</td>
<td>13385</td>
<td>6774 (51%)</td>
<td>6611 (49%)</td>
<td></td>
</tr>
<tr>
<td>#/WORD POS</td>
<td>40 (8%)</td>
<td>28 (4%)</td>
<td>20 (4%)</td>
<td></td>
</tr>
</tbody>
</table>

Sample results of verification in the prediction mode are presented in Table II. In this mode it is important that the best rating for the predicted word comes from a mapping that closely approximates the actual time in which the word appears. If this is not the case there is the danger that a correct word, which is highly rated, will be hypothesized with times which will disrupt the recognition of word sequences by top end knowledge sources. Small errors in the determination of the missing time can propagate time errors which may cause whole words to be missed. Table II summarizes the results of an experiment to predict begin/end times of 529 words where both times were actually known. The distance, in segments, is calculated from the known word bound and its predicted word bound. The table also shows the distribution of distances for the best mapping. Given that the average segment duration is 3.2cs, a distance of 2 would correspond to a range of predicted bounds 6.5cs about the actual bound. Each prediction takes, on the average, 180ms of CPU time.

### Table II

<table>
<thead>
<tr>
<th>DIST</th>
<th>FREQ</th>
<th>%</th>
<th>CUM %</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>125</td>
<td>24%</td>
<td>24%</td>
</tr>
<tr>
<td>1</td>
<td>289</td>
<td>40%</td>
<td>64%</td>
</tr>
<tr>
<td>2</td>
<td>183</td>
<td>19%</td>
<td>83%</td>
</tr>
<tr>
<td>3</td>
<td>41</td>
<td>8%</td>
<td>91%</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>4%</td>
<td>95%</td>
</tr>
<tr>
<td>5</td>
<td>17</td>
<td>3%</td>
<td>98%</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>1%</td>
<td>99%</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>1%</td>
<td>100%</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>0%</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>0%</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Areas of further research involve dynamic generation of multiple word networks.
using static networks and word juncture rules, alternate score normalization schemes, and improvement in the effectiveness of bounding predictions using vowel nuclei.

References


Word Pair Adjacency Acceptance Procedure in Hearsay-II (Robert Cronk)

Introduction

In the Hearsay-II speech understanding system, several knowledge sources attempt to construct sequences of words from the word candidates hypothesized on the blackboard. Pairs of words which are approximately time-contiguous and syntactically adjacent (may be paired in the grammar) are considered for extending word sequences. To avoid the combinatorial explosion which occurs in a grammar with a large branching factor, a procedure is required which will constrain the number of word pairs to those which have a high probability of being the correct ones.

Such a procedure must be computationally inexpensive, since it must make decisions on hundreds of pairs of hypothesized words. It must rely upon knowledge of word junctures and upon the information contained in the segmental transcription of the spoken utterance. And it must reject as many incorrect pairs (word pairs not actually spoken) as possible, without rejecting any of the correct pairs.

This paper describes the word pair adjacency acceptance procedure (JUNCT) developed for Hearsay-II, the knowledge it uses, and the current results.

Description

Input to the JUNCT procedure is a pair of word hypotheses. If it determines that the words are adjacent, based upon the times associated with the hypotheses, the juncture rules contained in the procedure, and the blackboard segmental description of the spoken utterance the pair is accepted as a valid sequence; otherwise it is rejected.

Word junctures which JUNCT must use to make its decisions fall within three distinct cases:
(1) Time-contiguous hypotheses: Words which are time contiguous in the blackboard are immediately accepted by JUNCT as a possible sequence. No further tests for adjacency are performed.
(2) Overlapping hypotheses: When two words overlap in time, juncture rules are applied in the context of the blackboard segmental transcription of the utterance to determine if such a juncture is allowable for the word pair.
(3) Separated hypotheses: When the words are separated by some interval of time, rules are applied, as in the overlap case, to determine whether the pair can be accepted as a valid sequence in the utterance.

The juncture rules used by JUNCT are of two types: (1) allowable overlaps of word end-phoneme and begin-phoneme, and (2) tests for disallowed segments within the word juncture. A bit matrix of allowable overlaps is precompiled into the procedure, and is indexed by the end-phoneme and begin-phoneme of the word pair. Any overlap juncture involving phonemes which are not allowed to share segments is rejected by JUNCT. In the separation case, as in allowed overlaps, the blackboard segmental description of the spoken utterance is examined in the context of the end-phoneme and begin-phoneme of the word pair to determine if any disallowed segments are present in the juncture gap. If such segments are found, the word pair is rejected. Only when a word pair passes all rule tests which apply in the segmental context of its juncture is it accepted as a valid sequence.
Current Results

Stand-alone performance evaluation runs were made over 60 utterances using words generated from files produced by the Hearsay-II word hypothesizer. Syntactically adjacent pairs of words whose ratings were 40 and above (on a scale from 0 to 100) and whose times (left-word end time and right-word begin time) were within a 200 millisecond interval were considered. All of the words used for testing the procedure were hypothesized "bottom-up" in Hearsay-II; no predictions were used in the evaluation runs. The following table summarizes the performance of the JUNCT procedure.

<table>
<thead>
<tr>
<th></th>
<th>CORRECT WORD PAIRS</th>
<th>INCORRECT WORD PAIRS</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCEPTED</td>
<td>188 (95%)</td>
<td>2891 (41%)</td>
<td>3079 (42%)</td>
</tr>
<tr>
<td>REJECTED</td>
<td>5 (5%)</td>
<td>4224 (59%)</td>
<td>4233 (58%)</td>
</tr>
<tr>
<td>TOTAL</td>
<td>197</td>
<td>7115</td>
<td>7312</td>
</tr>
</tbody>
</table>

It is expected that, as lower-level sources of knowledge provide more accurate times for word hypotheses, the rules for acceptance of valid word pairs may be tightened, further increasing the speed and performance of Hearsay-II.

Syntactic Processing in Hearsay-II (Hayes-Roth, Erman, Fox, and Mostow)

The basic tasks facing the three syntactic knowledge sources in Hearsay-II are: to parse syntactically acceptable sequences of words; to predict words that can be (syntactically) adjacent to the ends of a word sequence; and to construct larger sequences when predicted words are verified. The chief obstacle is finding all possible syntactic structures that can produce a given sequence of words. Of the traditional parsing mechanisms, only bottom-up Kay-type parsers have addressed the problem of building phrase-structure trees which are not necessarily anchored at the start (or end) of a sentence. But these methods are still inadequate for parsing in the current environment because of their requirement that all constituents of a phrase be present in order for a phrase to be recognized. In Hearsay-II, a general method for such partial parsing of incomplete phrase structures has been developed and is used to parse grammatical word sequences, to predict extensions, and to join up to three sequences of words together in a new syntactic structure.

The details of the method are now briefly described. To minimize redundant computing, the syntactic (context-free) grammar is converted to an equivalent template normal form grammar in which all sequential productions have binary derivations (e.g., $A \rightarrow B C D$ is replaced by $A \rightarrow B X$ and $X \rightarrow C D$). Thus, frequently occurring grammatical subsequences are replaced by a common higher-order non-terminal.
thereby minimizing recomputation of common subexpressions (Hayes-Roth and Mostow, 1975).

The word-sequence hypothesizer, WOSEQ, generates the initial word sequences that are partial-parsed. Given a word sequence \( w_1 \ldots w_n \), the RECOGNIZE parser knowledge source works in a conventional bottom-up manner, with the exception that any words or phrases (non-terminals) that are required by a grammar rule to precede (follow) a constituent at the first (last) position of the sequence are pseudo-recognized; that is, if the word sequence \( w_1 \ldots w_n \) can be derived from the productions
\[
S \rightarrow A \ T, 
T \rightarrow w_1 \ V, 
V \rightarrow U \ X, 
U \rightarrow \ldots \ w_n, 
A \rightarrow w_0, \text{ and } X \rightarrow w(n+1),
\]
then the non-terminals A and X will be pseudo-recognized and the sequence \( w_1 \ldots w_n \) will be parsed as an instance of S, with closest left-missing constituent A and closest right-missing constituent X. Bottom-up parsing continues until all of the words in the input sequence are subsumed by each highest-order phrase or until no further rewrites are possible. The highest-order phrases constructed that derive the entire word sequence are referred to as spanning phrases. Because parsing is discontinued on spanning phrases, the partial-parse technique essentially identifies minimal (lowest-order) parses of each sequence. Each distinct parse of a sequence specifies a spanning phrase and the pseudo-recognized closest missing constituents. There may, of course, be several distinct parses of any word sequence. If no parse of a sequence is found, it is rejected. Whenever a sequence hypothesized by the word-sequence hypothesizer is rejected, that knowledge source wakes up, decomposes the rejected sequence into maximal subsequences, and then hypothesizes any sufficiently rated new word sequences.

Given a spanning parse of a sequence \( w_1 \ldots w_n \) with closest left and right-missing constituents A and X, the words that can be adjacent to \( <w_1 \text{ or } w_n> \) are all rightmost derivatives of A or leftmost derivatives of X. If a spanning phrase has no closest left-missing (right-missing) constituent, the possible adjacent words are found by "going up-and-over": the rightmost (leftmost) derivatives are computed for each constituent that can be directly adjacent to this left-complete (right-complete) phrase in some higher-level spanning phrase. Predictions of words are made by the PREDICT knowledge source whenever the extension of a previously parsed word sequence is scheduled and executed. Predictions may be made to both sides or to only one side depending on the relative and absolute numbers of grammatically possible words on the two sides. In any case, if none of the predicted words on one side is verified, the word-sequence hypothesis, although syntactically valid, is deactivated. No further processing of that sequence can occur unless it is retrieved by another sequence extension colliding with it on the side that failed the extension effort. Such a salutary collision results in the reactivation of the sequence.

When predicted words are verified, the CONCAT knowledge source may extend the parse by concatenating the verified words to the predicting word reference. Given the sequence \( <w_1 \ldots w_n> \) and verified preceding predicted words \( a_1, a_2, \ldots, a_k \) and verified succeeding predicted words \( b_1, b_2, \ldots, b_m \), an attempt is made to partial-parse all sequences \( <a_i \ldots w_l b_j> \) as well as all sequences \( <x_1 x_2 \ldots x_p a_i \ldots w_l \ldots w_n b_j y_1 y_2 \ldots y_q> \) where \( <x_1 x_2 \ldots x_p a_i> \) \( <b_j y_1 y_2 \ldots y_q> \) is a previously parsed sequence of words on the blackboard that is time-adjacent to and precedes (succeeds) \( <w_1 \ldots w_n> \). All successfully parsed sequences generate phrasal hypotheses. Thus, in addition to simply extending sequences a-word-at-a-time in each direction, finding a predicted word as the terminus of an existing adjacent sequence can trigger the concatenation of three sequences at once.

Conclusion

Because the words that are hypothesized from other knowledge sources form arbitrary sequences that usually do not completely satisfy constituent structures of phrase rewriting rules, a general mechanism for partial-parsing is needed. The current implementation generates minimal spanning phrases and retains at most one closest
missing constituent on each side of each phrase. Partial-parsing times average about 50 msec on the KL10 for a 1000 word vocabulary with a 15 branching-factor grammar. Extensions of sequences are quickly computed by running down the right or left sons of the binary sequence nodes of the closest missing constituents. Three adjacent sequences are syntactically concatenated by partial-parsing the concatenated word sequences. The current implementation provides an efficient solution to essential problems of syntactic processing. In addition, the three related knowledge sources decompose this processing into natural components with a grain-size that is attractive for focusing and control.

References

Focus and Control in Hearsay-II (Hayes-Roth and Lesser)

The Hearsay-II speech understanding system currently comprises 13 knowledge sources (KSs), 11 of which are data-directed. Each data-directed KS is invoked whenever new or modified blackboard data configurations matching patterns of interest are found. Monitoring for potentially relevant data changes is performed in two steps: changes in hypotheses or links at particular levels are collected in change sets specific to each KS; procedures called preconditions then closely examine each accumulated change and its blackboard context to determine if the exact pattern of interest is present. Once such a pattern is detected, the relevant KS is invoked (scheduled) to operate upon it. The basic control problem is to execute first those preconditions and KSs that are most likely to lead to successful recognition of the utterance. The two chief subgoals are: (1) to find the best interpretation as quickly as possible and (2) to reduce the number of incorrect hypotheses that are generated and tested. In fact, if too many incorrect hypotheses are examined, working storage capacity of the system may be exceeded, thus precluding eventual correct recognition of the utterance.

The current approach to the control problem follows closely the design of the focus of attention mechanism described in detail in Hayes-Roth and Lesser (1976). The basic concepts of that paper are quickly reviewed here: (1) The Competition Principle: the best of several alternatives should be performed first; (2) The Validity Principle: more processing should be given to KSs operating on more valid data; (3) The Significance Principle: more processing should be given to KSs whose expected results are more significant; (4) The Efficiency Principle: more processing should be given to KSs that perform most reliably and inexpensively; (5) The Goal Satisfaction Principle: more processing should be given to KSs whose responses are most likely to satisfy processing goals.

The degree to which a precondition or KS satisfies these principles is reflected by its desirability, an increasing function of its validity, duration, level of analysis, importance, concordance with control thresholds (goals), (relative and absolute) expected superiority over the best competing alternative in the same time area, and the time elapsed since an improved degree of recognition was achieved (stagnation) in that time area. While the desirability of a KS instantiation awaiting execution is determined directly from only one data pattern and the declarative control knowledge about the direction (on the blackboard) and relative effectiveness of its actions, the desirability of a precondition is taken to be the maximum of such values over all hypotheses in its change set.

Using this general scheme, we have implemented one particular control strategy by setting particular processing goals on the blackboard. Initially the
segmenter/labeller is executed and is forced to run to completion. This insures that bottom-up syllable hypothesization will have the benefit of complete segmental contexts. The syllable hypothesizer is executed in turn, and for a similar reason is also forced to run to completion. At this point the syllable-to-word KS responds to new syllables and generates all potentially plausible words. The strategy module then establishes thresholds governing which of these words is hypothesized. It attempts to have several highly rated words hypothesized in each area of the utterance. After this processing is completed, the word-sequence hypothesizer examines all words in parallel and identifies promising connected sequences of time-adjacent syntactically possible pairs of words (seeds). The best of these in each time are then hypothesized. From this point on, a complex sequence of data-directed preconditions and KSs is invoked, scheduled, and executed to control syntactic parsing, hypothesization of plausible words to extend syntactic sequences, concatenation of verified words or phrases with adjacent phrases, and the generation of further seeds when the system is stagnating. Whenever any new complete parse is found, a special KS is invoked to determine which remaining hypotheses and KS instantiations are insufficiently attractive to preserve. These are either rejected or deleted. Processing then continues until a quiescence occurs reflecting that the remaining alternatives are insufficiently credible to continue. If a sufficiently plausible sentence has been recognized, the stopping condition KS decides to terminate the analysis; or if no complete sentence has been formed, an attempt is made to interpret the best partial sequences by the syntax and semantics knowledge source.

Conclusion

Each precondition and KS is regarded as a [condition-action] schema, with known inputs (blackboard hypotheses and links), a known direction of action (bottom-up, top-down, or same-level and forwards, backwards, or same-time), known reliability and efficiency, and therefore, a known expected result. By comparing the expected results of all scheduled activities to the current state of recognition and desired areas of activity, the best pending instantiation can be executed first. As a result of tuning the various weighting factors, we seem to have achieved a desirable balance of breadth- and depth-first search (in a global sense) with effective suppression of sub-optimal (in a local sense) activities. Further, by separating expensive searches into two or more successive steps (e.g., change sets and preconditions do gross filtering and only subsequent KSs do fine, expensive processing; or, before expensive syntactic searches are performed, inexpensive searches are made for plausible sequences of syntactic word pairs), it appears that we have achieved some efficiency in the overall organization and control of the search process.

Reference


Policies for Rating Hypotheses, Halting, and Selecting a Solution in Hearsay-II (Hayes-Roth, Lesser, Mostow, and Erman)

Purpose of hypothesis validity ratings

The rating policy module (RPOL) in Hearsay-II provides a uniform basis for comparing the plausibility of different hypotheses. The hypotheses may be competing alternative interpretations of the same portion of the utterance at some level of the blackboard, in which case the hypothesis whose validity rating is higher is considered
more likely to be the correct interpretation. However, the hypotheses may describe
different portions of the utterance, or provide representations at different levels of
the blackboard. Having a uniform rating policy means that such hypotheses may
nonetheless be meaningfully compared on the basis of their validity ratings. This
information is used in three ways by Hearsay-II:

1. to focus attention in promising directions by considering higher-rated (more
likely correct) hypotheses before lower-rated hypotheses. This is implemented by
making the priority of a scheduled action an increasing function of the validity ratings
of the hypotheses which are being acted upon (Hayes-Roth and Lesser, 1976). Also,
certain types of actions are not even scheduled on hypotheses which fail minimum
plausibility tests specified by knowledge source modules. These tests use validity
ratings as a measure of plausibility.

2. to select the most likely correct interpretation of the utterance if there is
more than one phrasal hypothesis spanning the utterance. The highest-rated such
hypothesis is then the chosen interpretation.

3. to prune the search once a solution (i.e., an utterance-spanning phrasal
hypothesis) has been found. This is done by restricting further processing to those
actions which are capable of leading to a better (higher-rated) solution.

Computation of hypothesis validity ratings

Hypotheses in Hearsay-II represent interpretations of the speech signal at
various levels of representation: segmental (lowest level), syllabic, lexical, word-
sequential, and phrasal (highest level). An hypothesis may be either conjunctive,
representing a logical product, or temporal sequence, of lower level hypotheses or
disjunctive, representing a logical summation of lower level alternative hypotheses.
The degree to which each lower level hypothesis supports the upper hypothesis is
indicated by an implication between -100 (maximally disconfirming) and +100
(maximally confirming). This number is attached to a link in the blackboard from the
lower to the upper hypothesis.

The validity rating \(VLD(H)\) of an hypothesis \(H\) is a measure of the extent to
which that hypothesis is supported, ultimately, from the acoustic data. The lowest
level hypotheses are rated by the bottom-end processor. The rating of a higher level
hypothesis \(H\) is computed from the validities of the hypotheses which support \(H\)
directly from below, and is stored on the blackboard as part of \(H\). The validity rating
of \(H\) need only be recomputed when the validity or implication of its support changes,
or when \(H\) receives new support. In such cases, RPOL immediately propagates
resultant validity changes up through the blackboard. Storing the ratings on the
blackboard avoids the expense of recomputing them recursively whenever they are
used.

The validity rating \(VLD(H)\) of a disjunctive hypothesis \(H\) supported by \(n\) lower
level hypotheses \(H_1, \ldots, H_n\) via respective links \(L_1, \ldots, L_n\) is given by

\[
\text{Max } VLD(H_i) \times \text{IMPLICATION(L_i)}/100, (1 \leq i \leq n).
\]

Similarly, the validity rating of a conjunctive hypothesis at the word level or
below is given by

\[
(1 + (n-1)/10) \times (\text{Sum } VLD(H_i) \times \text{IMPLICATION(L_i)}/100), (1 \leq i \leq n).
\]

The weighting factor \((1 + (n-1)/10)\) reflects the increased plausibility of an
hypothesis which has many conjunctive supports.

Above the word level, a somewhat different function is used to rate conjunctive
hypotheses. The validity \(VLD(H)\) of a phrasal or word sequence hypothesis \(H\) is given
by the duration-weighted average validity of its \(n\) underlying words \(W_i\), where
duration is measured in number of syllables. \(i.e.,\)
VLD(H) = (Sum VLD(Wi)*length(Wi)) / Sum length(Wi), (1≤i≤n),

where length(Wi) = length (in syllables) of the word hypothesis Wi. This formula is based on the empirical observation that the longer a word Wi, the greater the correlation between its correctness and the correctness of H.

Halting conditions and heuristic pruning

A phrasal hypothesis can be thought of as a subpath through a flow graph whose arcs are word hypotheses, and whose source and sink are respectively the beginning and end of the utterance. A solution (utterance-spanning phrase) then corresponds to a complete path through the graph. The validity rating of a subpath (hypothesis) is given by the average arc (word hypothesis) validity along the subpath, weighted by arc (word) length measured in syllables.

There is a qualitative difference between the task of searching for a solution (complete path) and the task of deciding when to stop searching and accept the current best solution. The former task can efficiently be done best-first, i.e., by extending the most promising path at each step in the search. In contrast, the latter task inherently involves searching all possible paths in order to guarantee that no path is better than the best one found so far. Once a path has been found, the goal of processing should be to enable such a guarantee to be made as quickly as possible. In order to accelerate the attainment of this goal, two heuristics for pruning the search are used.

The first heuristic consists of rejecting every word, word sequence, and phrase hypothesis which, due to its low rating, cannot be extended into a better solution than the best already found. This heuristic can be thought of as a form of alpha-beta pruning, simplified for the case of a one-player game. Rejecting a subpath (hypothesis) amounts to abandoning certain nodes in the search tree which correspond to extensions of that subpath. In operation, an hypothesis is rejected if, when it is extended into an utterance-spanning path using the highest-rated word hypotheses currently on the blackboard, the resulting (not necessarily syntactically legal) path is rated lower than the best existing solution. Further processing on rejected hypotheses is cancelled. This operationalization is imperfect in that it ignores the possibility of "missing arcs," i.e., words which may subsequently be predicted by the syntax module (added as arcs in the graph) and be rated high enough to invalidate previous decisions to reject earlier hypotheses.

The second heuristic is based on the observation that, if a better solution than the current best solution exists, it must be possible to construct it by extending some existing subpath (hypothesis) which is rated higher than the subpath of the existing solution spanning the same time interval. (Once again, the missing arc problem is ignored.) All hypotheses (subpaths) which do not have this property are deactivated, i.e., incapacitated as active stimuli. Any scheduled inferential action based on a stimulus set of hypotheses is cancelled if all the hypotheses in the set are deactivated. This heuristic can be thought of as another form of alpha-beta pruning, modified to allow sharing of common subtrees in the search tree. Deactivating a subpath (hypothesis) amounts to deferring expansion of certain search tree nodes which correspond to extensions of that subpath.

The observed effect of these two heuristics is to cancel a large amount of scheduled processing once a solution is found, and to focus attention on those activities which are capable of leading to a better solution. When no such activities are left to pursue, RPOL halts processing, selects the highest-rated solution, and passes it to the semantics module to be interpreted.
Solutions and partial solutions

RPOL also halts processing when Hearsay-II exceeds predefined limits on size or execution time. In this case, RPOL chooses the highest-rated utterance-spanning phrasal hypothesis as its solution. If no such hypothesis has been generated, RPOL tries to extract a maximum of information from the blackboard by selecting the best partial parses (phrasal hypotheses) and passing them to the semantics module for further interpretation (Hayes-Roth, Fox, Gill, and Mostow, 1976). Here, the "best" phrase hypothesis H at time t is considered to be the hypothesis whose time interval includes t and which has the highest information content, defined by VLD(H) * length(H). RPOL finds the best hypothesis at each time t (measured in syllables from the beginning of the utterance), and passes the (typically small) set of such hypotheses to the semantics module. Thus even when Hearsay-II fails to find a complete solution, the best partial solution (set of partial interpretations) is found, and this information is used in determining the system's response to the utterance (Hayes-Roth, Gill, and Mostow, 1976).

Conclusions

The task of rating hypotheses in Hearsay-II is handled by the system policy module RPOL. The role of knowledge source modules in this task is limited to linking together hypotheses and specifying the implications with which lower hypotheses support upper hypotheses. Thus the effects of hypothesis rating changes due to new information are automatically propagated throughout the blackboard without requiring the help of the knowledge source modules. The centralized implementation of rating computation and propagation has made it easy to experiment with different rating formulas. It has also simplified the task of developing new knowledge source modules.

The uniform rating scheme employed permits the meaningful comparison of the plausibility of any two hypotheses. Validity ratings are used by Hearsay-II to focus processing, to prune the search, and to select the best solution or partial solution. In addition, hypothesis validity ratings are used by the knowledge source modules for plausibility tests which must be satisfied in order for various inferencing rules to be applied. Thus validity ratings help to guide processing in a best-first direction until a solution is found, and to validate it quickly thereafter as the best possible solution.

References


Semantics and Pragmatics in Hearsay-II (Hayes-Roth, Fox, Gill, and Mostow)

A speech understanding system differs from a recognition system in two principal ways. First, an understanding system verifies that the sentences it hears are meaningful and plausible. This requires use of semantic knowledge. Second, the understanding system expects particular types of communication to occur in specific discourse contexts and interprets the sentences it recognizes accordingly. Such expectation and contextual interpretation requires pragmatic knowledge. The purpose of semantics and pragmatics knowledge sources is to convert this knowledge about meanings, intentions, and communication conventions into effective action. The most significant type of action is one that constrains the recognition process, a search for a plausible parse of the spoken utterance. The second most important type of action is to hypothesize what was intended, when what was said cannot fully be recognized. The last type of effective action needed is to interpret (deduce the intention) of a successfully parsed utterance.
The complexity of artificial spoken languages may be constrained by restricting either the way ideas are expressed (syntax) or the number of ideas that can be expressed (semantics). Our approach, in the news retrieval and computer science abstract retrieval tasks, has been to develop one comprehensive semantic grammar (average branching factor 50) used for interpretation of recognized word sequences and to vary systematically the syntactic constraint of the languages used for speech recognition per se (branching factors 5, 15, 25). Regardless of the particular syntax used for recognition, the same general semantic grammar is used for semantic analysis. This grammar is a template grammar like those developed for Parry by Colby, with distinct templates for each unique type of semantic form (Colby, 1974; Hayes-Roth and Mostow, 1975). Semantic interpretation is accomplished by extracting from the (parse) tree of instantiated templates the particular words or expressions filling the various functional "slots."

Partially recognized sentences are also easily interpreted in this framework. When the attempt to recognize a complete sentence has failed, the best (longest and most highly rated) syntactic word sequences in each time area of the utterance are passed to semantic analysis. All templates fully or partially satisfied by word sequences are instantiated. The most fully matched semantic pattern is then chosen as the interpretation of the utterance. Thus, the recognized sequence "Newell or Simon" would be interpreted effectively as if "List all abstracts by Newell or Simon from any journal from any date" had been recognized.

The capacity to provide semantic constraint during recognition is determined primarily by the reliability of predictions regarding what the speaker is likely to say. We have implemented a discourse knowledge source including a conversation model that prompts the speaker with questions, provides information about using the system and the organization of the data base, and predicts the (semantic and syntactic) type of utterance next expected. Earlier versions of the syntax and semantics knowledge source biased recognition actions in favor of predicted communication forms. However, both because any valid sentence is permitted at any time and because the system is usually employed for isolated sentence understanding, no direct semantic bias is currently used. The basic scheme for such bias is, however, conceptually simple: given an expected type of utterance (a highest-level semantic template), recursively compute the expected lower-order subtemplates and, ultimately, the words and phrases that would instantiate the expected meaning templates. During recognition, priority is given to actions based on expected forms, at the expense of delayed processing of unexpected word sequences.

Conclusions

We have identified three types of actions to be performed by semantics and pragmatics knowledge sources: (1) bias recognition in favor of expected forms; (2) interpret semantically plausible, partial sequences; and (3) correctly interpret the intention of the speaker when a sentence is fully recognized. These actions are effected in Hearsay-II by combining semantic template grammars with a conversational model that anticipates the speaker's general intention and can enumerate its manner of expression. The realization of such actions, at least in restricted domains of discourse, can now be considered a well-understood technology.

References


F. Hayes-Roth, G. Gill and D. J. Mostow (1976). "Discourse analysis and task


Discourse Analysis and Task Performance in Hearsay-II (Hayes-Roth, Gill, and Mostow)

The discourse analysis module (DISCO) in Hearsay-II uses knowledge about the state of the conversation to interpret the speaker's intention and to direct the appropriate actions within the task program. Usually, the intention of the speaker is to establish a general area of interest, to retrieve articles by keyword expression, to further qualify a keyword expression, to print selected articles, or to request certain information about the retrieved articles, such as title, date, author, author's affiliation, or publisher. The speaker can also ask for help or complain about the system's response.

The state of discourse is represented by the contents of several semantic registers, one of which points to a node in a finite state automaton discourse model. (See Figure 1.) Each node in the model corresponds to a general sentence pattern or template (Hayes-Roth, Fox, Gill, and Mostow, 1976). (See Figure 2.) Other registers hold the current menu (general area of interest), the most recent keyword expression, the article most recently referred to, the most recently retrieved articles, and the subset of retrieved articles which satisfy further qualifications specified by the speaker. The finite state model is used to interpret yes-or-no responses and partially-recognized utterances, and to make predictions about what the speaker is likely to say next. All possible transitions between nodes in the model are permitted; the arcs in the model indicate the transitions which are considered likely.

Figure 3 shows a sample interaction between DISCO and a speaker. Utterances enclosed in square brackets denote recognized spoken utterances. In the example shown, the first utterance

\[ \text{WE'RE INTERESTED IN LEARNING} \]

is recognized by the semantics module as an instance of the $\text{SELECTION}$ template, and the semantic feature $\text{LEARNING}$ (indicated area of interest, or menu) is extracted. This semantic interpretation of the utterance is passed to DISCO, which records the indicated area of interest, LEARNING, in the MENU register, and sets the NODE register to point at the $\text{SELECTION}$ node in the finite state model. DISCO then predicts that the next utterance will be an instance of the $\text{REQUEST}$ template and will concern the area of LEARNING. These predictions can be used to bias subsequent processing to favor recognition of keywords in the LEARNING menu and function words characteristic of a $\text{REQUEST}$ (Hayes-Roth, Fox, Gill, and Mostow, 1976). Such predictions can also be used to respond gracefully in the case of a partially-recognized utterance (Hayes-Roth, Lesser, Mostow, and Erman, 1976). In the example, if the speaker's second utterance

\[ \text{WERE ANY ARTICLES ON LEARNING WRITTEN IN MAY 1974} \]

were not fully recognized, DISCO would assume that the speaker had REQUESTed some articles about LEARNING and could ask him to repeat the request. If the utterance fragment "LEARNING WRITTEN IN MAY 1974" were recognized and interpreted by the semantics module, DISCO could retrieve articles on learning dated May, 1974.
Figure 1: Semantic registers and finite state discourse model. 
labels Y and N indicate YES and NO responses; 
0 indicates empty retrieval set.
$SELECTION [ WE'RE INTERESTED IN LEARNING ]
Specifies a menu. DISCO responds by printing keywords and phrases from the menu.

$REQUEST [ WERE ANY ARTICLES ON LEARNING WRITTEN IN MAY 1974 ]
Specifies a set of articles. DISCO retrieves the articles and asks for further directions.

$PRUNE !LIST [ WHICH OF THESE MENTION ROBOTS ]
Further specifies a set of articles. DISCO removes articles from the currently retrieved set which don't satisfy the new restrictions.

$GET!INFO [ WHO WROTE THESE ]
Requests information about the retrieved articles. DISCO prints the requested information.

$LIST!THEM [ PLEASE LIST THEM ]
Requests output of a set of articles. DISCO prints all the articles in the currently retrieved set.

$LIST!IT [ PRINT IT ]
Requests output of a single article. DISCO prints the article most recently referred to.

$STOP!LISTING [ STOP LISTING PLEASE ]
Requests cessation of output. DISCO aborts the current output operation.

$MAKE!FILE [ MAKE A FILE OF THESE PIECES ]
Requests file output. DISCO creates a file containing the retrieved articles.

$CONTENTS!MENU [ WHAT ARE THE KEYWORDS FOR LEARNING ]
Requests the contents of a menu. DISCO lists the keywords and phrases of the menu.

$HELP [ WHAT CAN I ASK ]
Requests assistance or information about the system. DISCO attempts to fulfill the request.

$GRIPE [ WHY ARE YOU SO SLOW ]
Complaint. DISCO responds with a pacifying message.

The following three nodes represent responses to yes-or-no questions asked by DISCO.

$WANT!SEE
DISCO has printed the title of an article and asked, "Do you want to see this article?"

$WANT!MORE
DISCO has printed some of the articles in the currently retrieved set and asked, "Do you want to see another of these articles?"

$MORE!ON!MENU
DISCO has exhausted the currently retrieved set and asked, "Do you want to ask more about <current menu>?" where <current menu> is the most recent area of interest.

Figure 2. Templates corresponding to nodes in the finite state model.
Instances of templates are shown in square brackets.
Text describes the intention of each utterance template and DISCO's response to it.
OK, WHAT SUBJECT DO YOU WANT TO ASK ABOUT?

[ WE'RE INTERESTED IN LEARNING ]

THE KEYWORD PHRASES WHICH I KNOW ABOUT ARE:
- LEARNING
- HYPOTHESIS FORMATION
- CELL ASSEMBLY THEORY
- BRAIN THEORY
- CAUSAL REASONING
- COMMON SENSE THEORY FORMATION
- CONCEPTUAL INFERENCE
- CYBERNETICS
- GRAMMATICAL INFERENCE
- NEURAL NETWORKS
- PERCEPTRONS
- RULE ACQUISITION CAPABILITIES
- ABstraction
- ACQUISITION OF KNOWLEDGE

OK, ASK ME ABOUT LEARNING

[ WERE ANY ARTICLES ON LEARNING WRITTEN IN MAY 1974 ]

THERE ARE 2 STORIES
THE FIRST ONE IS:

TITLE:
DESIGN OF AN INTERACTIVE ENVIRONMENT TO STUDY THE BEHAVIOR OF SEVERAL
ROBOTS WHICH CAN LEARN, PLAN THEIR ACTIONS, AND COEXIST

DO YOU WANT TO SEE THIS ARTICLE?

[ WHO WROTE THESE ]

DESIGN OF AN INTERACTIVE ENVIRONMENT TO STUDY THE BEHAVIOR OF SEVERAL
ROBOTS WHICH CAN LEARN, PLAN THEIR ACTIONS, AND COEXIST
NICHOLAS V. FINDLER

PROCEDURAL LEARNING IN WORLDS OF ROBOTS
LAURENT SIKLOSY

Figure 3. Sample interaction between DISCO and speaker.
Spoken utterances are enclosed in square brackets.
PARALLEL PROCESSING IN SPEECH UNDERSTANDING SYSTEMS

Parallelism in Artificial Intelligence Problem-solving (Lesser and Fennell)

Background

Many artificial intelligence problem-solving tasks require large amounts of processing power because of the size of the search space that needs to be examined during the course of problem solution. This is especially true for tasks that involve the interpretation of real-world perceptual data which is generally very noisy (i.e., speech and image understanding systems). For example, a speech-understanding system capable of reliably understanding connected speech involving a large vocabulary is likely to require from 10 to 100 million instructions per second of computing power, if the recognition is to be performed in real time. Recent trends in technology suggest that raw computing power of this magnitude can be economically obtained through a closely-coupled network of asynchronous "simple" processors. The major problem with using a network multiprocessor is in specifying the various problem-solving algorithms in such a way as to exhibit a structure appropriate for exploiting the available parallelism.

This restructuring of an artificial intelligence task for parallel processing may not be as difficult as might be expected. The basic problem-solving paradigm that is used to resolve ambiguities resulting from the error in input data and the imprecise and errorful nature of knowledge sources implicitly involve parallel activity. This parallel activity arises because many weakly supported alternative hypotheses must be "simultaneously" evaluated in order to locate a consistent hypothesis which is a solution to the problem. These problem-solving techniques are implemented through sophisticated control structures that (1) permit the selective searching (usually heuristic) of a large part of the state-space of possibilities and (2) allow the combining of multiple, diverse sources of knowledge (e.g., in the speech domain, acoustics, syntax, semantics, prosodies) so as to cooperate in resolving ambiguity [Reddy 76, Woods 74, and Lesser 75A]. The state-space searching in existing systems is implemented through backtracking control structures; these are basically sequential implementations of non-deterministic control structures. Thus, a large potential for parallelism arises from implementing these non-deterministic control structures in a parallel manner, i.e., searching different parts of the state space in parallel. In addition, if these diverse knowledge sources (KS's) can be made independent, there exists the potential for a proportional speed-up in the recognition process by executing them in parallel. Finally, there is the possibility of decomposing each knowledge source into separate parallel processes.

Summary of Current Research

In order to test the ease and effectiveness with which an artificial intelligence task could be structured for and executed on a multiprocessor, an organization for a knowledge-based artificial intelligence problem-solving system was developed which takes maximum advantage of any separability of the processing or date components available within that organization. Knowledge sources are intended to be largely

References


independent and capable of asynchronous execution in the form of knowledge source processes. Overall system control is distributed and primarily data-directed, being based on events occurring in a globally shared data base. Such a problem-solving organization is believed to be particularly amenable to implementation in the hardware environment of a network of closely-coupled asynchronous processors which share a common memory. The Hearsay II speech-understanding system (HSII) [Lesser 75, Fennell 77, Erman 75], which has been developed using the techniques for system organization described above, has provided a context for evaluating the multiprocessing aspects of this system architecture.

Based on multiprocess simulations and implementation of these systems on the C.mmp multiprocessor, the following results were obtained [Fennell 75]:

1. There does exist extensive parallelism in the speech understanding task (e.g., given a small configuration of knowledge sources, between 4-14 processors could be effectively utilized).
2. The overheads involved in supporting the multiprocessing and synchronization primitives are quite high (e.g., over 100%).
3. The locking structures had to be very carefully tailored to the particular set of knowledge sources; otherwise, the effective parallelism would be significantly degraded.

In trying to understand the implications of the last two results, some tentative observations were made. The first and somewhat surprising observation was that the basic self-correcting nature of the information flow in the HSII system, which comes from knowledge source cooperation through a hypothesize-and-test paradigm, may obviate the need for most uses of explicit synchronization techniques to maintain data integrity. To elaborate on this point, one knowledge source can correct the mistake of another knowledge source whether the error arises from a mistake in the theory behind the knowledge source or from incorrect synchronization (i.e., working on partially invalid data). Another example of this self-correcting type of computation structure is the relaxation method (iterative refinement) used to solve partial differential equations. This type of computational structure, when put on asynchronous multiprocessors, can be decomposed so as to avoid a lot of explicit synchronization at the expense of more cycles for convergence. This type of decomposition is accomplished by not requiring each point to be calculated based on the most up-to-date values of its neighboring points. The iterative refinement nature of computation will correct (within a certain range) for this lack of synchronization. It is felt the feed-forward/feed-backward data-directed problem-solving paradigm of HSII has similar properties. The other observation was that a drastic decrease in the cost of certain types of synchronization primitives could be accomplished if their implementation is tailored to their (statistical) usage.

References


The HSII/C.mmp System (Lesser, Buchalter, McCracken, Robertson, and Suslick)

The HSII/C.mmp system has been developed to test whether an asynchronous multiprocess architecture such as C.mmp (16 PDP-11 processors sharing a common memory) can be effectively applied to speed up the higher level processing of a speech understanding system. Extensive simulation studies were done on a PDP-10 using a multiprocess version of Hearsay-II to test the feasibility of the idea before embarking on the actual implementation (Fennell and Lesser 1977).

A prototype version of this system written in L*, a system building language developed by Newell et al. 1970-71, was constructed and running in February of 1976. In addition, an algebraic-language interpreter, SL*, was constructed for executing knowledge sources written in an Algol dialect. However, the knowledge source modules were very primitive, and no substantial results were obtained except the measurement of the overhead of certain Hearsay-II primitives. As a result of these measurements, a reimplementation was begun in order to significantly speed up the system (especially those system primitives which deal with synchronization operations), and to make it possible to run large knowledge source modules in the small address space environment that the PDP-11 provides. This reimplementation is now almost complete, with preliminary results indicating a speed-up of approximately 10 over the original version. In addition, a translator has been developed which takes most PDP-10 statements written in SAIL and translates them into equivalent SL* statements. Thus, it should be possible in the next few months to run, without major code conversion, the knowledge source modules of the PDP-10 Hearsay-II system on the HSII/C.mmp system.

References
A Parallel Production System for Speech Understanding (McCracken)

The question addressed by this thesis (McCracken 1977) is whether or not a production system architecture can remedy some of the chronic problems of knowledge representation and system organization in large knowledge-based artificial intelligence systems, particularly speech understanding systems. Of particular interest is the problem of exploiting parallel machine architectures to obtain near real-time response. To explore this question, a production system version of the Hearsay-II speech understanding system, called HSP, for HearSay Production system, is being implemented on C.mmp, the CMU multi-mini-processor. A large fraction of the Hearsay-II speech knowledge has been translated into productions for HSP, specifically: POMOW (word recognizer), POSSE-WOMOS (word verifier) and SASS (syntax and semantics).

Expected results come under two main categories: comparisons between the way knowledge is encoded in HSP versus Hearsay-II, and comparisons in the use of parallelism. The major differences between the HSP and Hearsay-II architectures are: (1) the basic knowledge unit in HSP, a production, is considerably smaller than a Hearsay-II Knowledge Source; (2) HSP encodes knowledge in a more formal and simple, but less expressive, language than Hearsay-II; (3) HSP totally segregates condition from action (i.e., read from write), while Hearsay-II allows a mixture; and (4) there is virtually no use of local working memory in HSP (only a single shared working memory), whereas Hearsay-II knowledge sources make use of rather large local data contexts in addition to the shared Blackboard. It is expected that these architectural differences will yield an improvement for HSP in effective parallelism, in clarity of knowledge, in ease of augmentation, and in other problem areas, such as handling of error, directionality control, and performance analysis.

1. A production system encodes all long-term knowledge as simple condition-action rules which operate from a shared working memory. For entry into the subject see: R. Davis and J. King, An Overview of Production Systems, Computer Science Department, Stanford University, Oct. 1975.
2. POSSE, WOMOS, and the version of SASS used are from an earlier version of Hearsay-II used in the Spring of 1972.

References
III. PERFORMANCE MEASUREMENT

In this section we present the detailed performance results obtained for the Harpy and Hearsay-II systems in September of 1976. Since then both systems have been improved; future papers will provide results of improved performance. The purpose of this section is to provide a record of system performance as measured on September 8, 1976.

In addition to the performance of the systems on the 1011-word tasks, this section also contains results of experiments on connected digit recognition, effect of telephone on accuracy, effect of multiple speakers (using speaker independent templates) on accuracy, and effects of branching factor and vocabulary size on the performance of the Harpy system.

Performance of the Harpy and Hearsay-II Systems

Figure 1 gives the performance of the Harpy system on the 1011-word AI abstract retrieval task. The vocabulary used in this task and the phone dictionary associated with the vocabulary is given in Appendix III-B.

Given the vocabulary and protocols taken of humans interacting with a mock system, Hayes-Roth generated a set of typical sentences that are likely to be useful in the abstract retrieval task. No attempt was made to restrict these to any specific grammar. However, care was taken to see that each word in the vocabulary occurred at least once in these sentences. These sentences (a total of 496) served two purposes: 1) as a set of training sentences (spoken by Lee Erman), and 2) for the design of a family of languages with varying branching factors that accept at least the training sentences and possibly many more.

Goodman designed many such languages. Two extreme examples are a language where any word (of the 1011) could follow any other word, permitting many nonsense sentences, and another in which only the 496 training sentences were legal. Of the several languages chosen for the experimentation, three specific ones—AIX05, AIX15, and AIXF—were given in Appendix III-C (an earlier version of AIXF was developed by Hayes-Roth).

The grammar that allowed Harpy to reach the performance goals of the ARPA program was AIX05, with a static branching factor of 9.53 and an average dynamic fanout of 33.4. The others were too large to fit within the memory of the PDP-10 system. However, it was possible to study the performance of AIX15 and AIXF using variants which used smaller vocabularies, created by eliminating some of the proper nouns.

The training sets for the other four speakers (two male and two female) consisted of a small subset of the original training sentences. These were used to generate speaker-dependent phone templates for each of the speakers (see the paper by Lowerre in Section IV on speaker adaptation).

A completely new set of 100 test sentences was created by Hayes-Roth which were not part of the training set. These are given in Appendix III-A. Erman recorded all the 100 test sentences and the other four speakers recorded a subset of twenty one sentences each. These sentences were used only for testing the performance of the system; the system was not tuned in any way in response to errors in this set.

The Harpy system achieved an aggregate 91% sentence accuracy and 95% semantic accuracy over all the 5 speakers and required 27.9 million instructions per second of speech processed (Fig. 1). Hearsay-II (Fig. 3) was tested on only twenty two sentences for lack of time and achieved 91% semantic accuracy and required about 85 mips. Figures 2 and 4 give the performance of the two systems on test sentences recorded live in the classroom on September 8. The Harpy system recognized four of
the five sentences recorded by two male and one female speaker correctly. The
Hearsay-II system recognized three of the five. These sentences were generated by
the observers who were given copies of the grammar; the sentences were in no way
preselected. The classroom environment was somewhat more noisy than the terminal
room environment normally used to collect training data.
**TASK**

Recognition of AI information retrieval task
- Vocabulary size: 1011
- Branching factor: 9.53
- Average fanout: 33.4

**DATA**

Number of speakers: 5
- 3 male
- 2 female

Training set for speaker LE
- 496 sentences
- 4049 words
- 24.7 minutes of speech

Training set for speakers DS, KP, BH, CW
- 256 sentences
- 1444 words
- 10.1 minutes of speech

Test set for all speakers
- 184 sentences
- 1138 words
- 6.5 minutes of speech

**PERFORMANCE ON THE TEST DATA**
- 97% word accuracy
- 91% sentence accuracy
- 95% semantic accuracy
- 27.9 Mipss

Figure 1. Harpy results for the AI retrieval task test data.
RESULTS OF LIVE SENTENCES

<table>
<thead>
<tr>
<th>UTT</th>
<th>TIME</th>
<th>WORDS</th>
<th>OUT</th>
<th>#COR</th>
<th>%COR</th>
<th>TIME</th>
<th>TIMEOUT</th>
<th>STATES</th>
<th>SEGMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.2</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>100.0</td>
<td>71.3</td>
<td>32.2</td>
<td>261</td>
<td>82</td>
</tr>
<tr>
<td>2</td>
<td>2.1</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>100.0</td>
<td>69.9</td>
<td>33.0</td>
<td>355</td>
<td>79</td>
</tr>
<tr>
<td>3</td>
<td>3.8</td>
<td>9</td>
<td>11</td>
<td>5</td>
<td>55.6</td>
<td>311.1</td>
<td>80.3</td>
<td>396</td>
<td>138</td>
</tr>
<tr>
<td>4</td>
<td>2.1</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>100.0</td>
<td>96.1</td>
<td>45.6</td>
<td>432</td>
<td>81</td>
</tr>
<tr>
<td>5</td>
<td>1.5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>100.0</td>
<td>53.8</td>
<td>36.6</td>
<td>352</td>
<td>53</td>
</tr>
</tbody>
</table>

Correct utts = 4/5 = 80.0%

RESULTS OF LIVE SENTENCES

<table>
<thead>
<tr>
<th>UTT</th>
<th>HARPY VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>UTT 1</td>
<td>&quot;ARE ANY PAPERS ABOUT SEMANTIC NETWORKS&quot;</td>
</tr>
<tr>
<td>REC</td>
<td>&quot;ARE ANY PAPERS ABOUT SEMANTIC NETWORKS&quot;</td>
</tr>
<tr>
<td>CORRECT</td>
<td>6/6 AVE. PRB.=-.4954988</td>
</tr>
</tbody>
</table>

| UTT 2| "DOES SEMANTIC NETS GET MENTIONED ANYWHERE" |
| REC  | "DOES SEMANTIC NETS GET MENTIONED ANYWHERE" |
| CORRECT | 6/6 AVE. PRB.=-.5610788 |

| UTT 3| "WHICH PAPERS ON REGION ANALYSIS ALSO DISCUSS LANGUAGE UNDERSTANDING" |
| REC  | "WHICH PAPERS ON A REGION ANALYSIS SUBSYSTEM AND DESIGN MENTION UNDERSTANDING" |
| CORRECT | 5/9 AVE. PRB.=-.6636969 |

| UTT 4| "HOW MANY ARTICLES ON CHESS AND LEARNING ARE THERE" |
| REC  | "HOW MANY ARTICLES ON CHESS AND LEARNING ARE THERE" |
| CORRECT | 9/9 AVE. PRB.=-.5521564 |

| UTT 5| "WE'RE INTERESTED IN HEARSAY" |
| REC  | "WE'RE INTERESTED IN HEARSAY" |
| CORRECT | 4/4 AVE. PRB.=-.6638372 |

Figure 2. Harpy results for the live demonstration, 8 September 1976.
TASK  Recognition of AI information retrieval task
   Vocabulary size: 1011
   Branching factor: 9.53
   Average fanout: 33.4

DATA  Number of speakers: 1 male speaker

Training set for word hypothesizer
   60 sentences
   340 words
   2.2 minutes of speech

Training set for word verifier
   747 sentences
   4049 words
   24.7 minutes of speech

Test set for all speakers
   22 sentences
   154 words
   1.0 minute of speech

PERFORMANCE ON THE TEST DATA
   86%  word accuracy
   73%  sentence accuracy
   91%  semantic accuracy
   85.0 Mipps

Figure 3. Hearsay-II results for the AI retrieval task test data.

RESULTS OF LIVE SENTENCES: HEARSAY-II

UTT 1: UTT="I AM INTERESTED IN ENGLISH"
   REC="I AM INTERESTED IN ENGLISH"

UTT 2: UTT="ARE ANY PAPERS ABOUT SEMANTIC NETWORKS"
   REC="ARE ANY PAPERS ABOUT A SEMANTIC NETWORK"

UTT 3: UTT="DOES SEMANTIC NETS GET MENTIONED ANYWHERE"
   TIMEOUT - 2 best partial parses are:
   [DO SIMULTANEOUS ACTIONS........]
   [.....DESIGN AND SYNTAX MENTIONED ANYWHERE]

UTT 4: UTT="HOW MANY ARTICLES ON CHESS AND LEARNING ARE THERE"
   TIMEOUT

UTT 5: UTT="WE'RE INTERESTED IN HEARSAY"
   REC="WE'RE INTERESTED IN HEARSAY"

48% SENTENCE ACCURACY
68% SEMANTIC ACCURACY

Figure 4. Hearsay-II results for the live demonstration, 8 September 1976.
Connected Digit Recognition using Symbolic Representation of Pronunciation Variability (Goodman, Lowerre, Reddy, and Scelza)

Most connected speech recognition systems, such as Harpy and Hearsay-II, use some form of symbolic representation to represent alternative pronunciations of the vocabulary, whereas most isolated word recognition systems use word templates. In an attempt to compare relative performance of systems that use symbolic representations of words, the Harpy system was run on four tasks requiring the recognition of random sequences of digits. Recording was in a computer terminal room environment (approximately 60 dBA) with speakers recording one session per day in order to include as much intra-speaker variability as possible. Both male and female speakers were used.

3-Digits Task
This task was selected as a typical numerical data input task. Sentences are connected sequences of three digits, such as "zero three eight". Each of ten speakers spoke thirty training sentences and 100 test sentences over a period of three weeks. Using speaker-specific phoneme templates, the word error rate over all ten speakers was about 27.

7-Digits Task
This task, sometimes referred to as the "telephone number task", consists of connected seven digit sequences such as "seven three nine six one seven three". This task was selected as a benchmark. Error rate for the single speaker was 17.

Telephone Input Task
Sentences are three digit connected sequences, as in the 3-digits task. Recordings were taken over telephone lines in order to determine the effects of restricted frequency response, distortion, envelope delay, etc. The error rate under these conditions was 77.

Speaker Independent Task
This task is similar to the 3-digits task. However, recognition is performed using speaker-independent phoneme templates computed from the training data for all speakers. The word error rate was about 77 on test data of 1200 random three-digit sequences from twenty speakers, including ten new speakers.
A summary of the results for these tasks is shown in the accompanying tables. The total test data are 2700 sentences, representing more than an hour of recorded speech. While this is already a large amount of data, a more extensive and thorough study is to be initiated.

<table>
<thead>
<tr>
<th>TASK</th>
<th>3-Digit</th>
<th>7-Digit</th>
<th>Telephone</th>
<th>Speaker-Independent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vocabulary Size</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Branching Factor</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>No. of Speakers</td>
<td>10</td>
<td>1</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>Male</td>
<td>7</td>
<td>1</td>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>Female</td>
<td>3</td>
<td></td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>Training Set</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of Sentences</td>
<td>300</td>
<td>30</td>
<td>120</td>
<td>300</td>
</tr>
<tr>
<td>No. of Words</td>
<td>900</td>
<td>210</td>
<td>360</td>
<td>900</td>
</tr>
<tr>
<td>Mins. of Speech</td>
<td>7.5</td>
<td>1.4</td>
<td>3.1</td>
<td>7.6</td>
</tr>
<tr>
<td>Words/minute</td>
<td>120</td>
<td>150</td>
<td>116</td>
<td>118</td>
</tr>
<tr>
<td>Test Set</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of Sentences</td>
<td>1000</td>
<td>100</td>
<td>400</td>
<td>1200</td>
</tr>
<tr>
<td>No. of Words</td>
<td>3000</td>
<td>700</td>
<td>1200</td>
<td>3600</td>
</tr>
<tr>
<td>Mins. of Speech</td>
<td>25.1</td>
<td>4.8</td>
<td>10.3</td>
<td>33.0</td>
</tr>
<tr>
<td>Words/minute</td>
<td>120</td>
<td>146</td>
<td>117</td>
<td>109</td>
</tr>
<tr>
<td>Performance on Test Data</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>%Word Accuracy</td>
<td>98</td>
<td>99</td>
<td>93</td>
<td>93</td>
</tr>
<tr>
<td>%Sent. Accuracy</td>
<td>96</td>
<td>96</td>
<td>82</td>
<td>83</td>
</tr>
<tr>
<td>Mipps</td>
<td>3.5</td>
<td>3.5</td>
<td>3.5</td>
<td>3.5</td>
</tr>
</tbody>
</table>
Effects of Branching Factor and Vocabulary Size on Performance (Goodman, Lowerre, and Reddy)

Analysis

Analysis of the languages of a given set of recognition tasks permits the comparison of the relative difficulties of the tasks. We have developed notions of equivalent vocabulary size, branching factor, effective branching factor, search space size, and search space reduction (Goodman 1976). All of these are useful as relative comparison measure.

Design

A family of languages having varying characteristics is required in order to be able to compare language measures with actual performance data. Such a family has been generated for the AI abstract retrieval task by interactive grammatical inference. There are four subfamilies for each of the (approx.) vocabulary sizes 250, 500, 750, and 1000 words. Several grammars representing differing branching factors exist within each subfamily. With the 250 word grammar, for instance, the available branching factors are 1.23, 3.87, 4.6, 8.2, 8.8, 11.9, 33.3, and 39.5.

Results

The relationships between accuracy and speed versus branching factor and vocabulary size are summarized in the accompanying tables. As expected, there is positive correlation in all cases. In the case of speed versus branching factor, the relationship is almost linear. A more comprehensive study of measures for grammatical complexity and their predictive abilities is necessary before any significance can be attached to these preliminary results.

Table I. Effects of branching factor on error rates of the Harpy system within the 250 word family of grammars.

<table>
<thead>
<tr>
<th>GRAMMAR</th>
<th>MIPSS</th>
<th>STATIC BRANCHING ERROR</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIS06</td>
<td>6.63</td>
<td>4.6 8%</td>
</tr>
<tr>
<td>AIS10</td>
<td>9.36</td>
<td>8.2 4%</td>
</tr>
<tr>
<td>AIS15</td>
<td>13.65</td>
<td>11.9 6%</td>
</tr>
<tr>
<td>AIS30</td>
<td>44.72</td>
<td>33.3 16%</td>
</tr>
<tr>
<td>AIS40</td>
<td>59.15</td>
<td>39.5 16%</td>
</tr>
</tbody>
</table>

Table II. Speed versus vocabulary size for Harpy when branching factor is held constant (approx. 10).

<table>
<thead>
<tr>
<th>GRAMMAR</th>
<th>MIPSS</th>
<th>BRANCHING</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIS10</td>
<td>9.36</td>
<td>8.2 250</td>
</tr>
<tr>
<td>AIM12</td>
<td>16.77</td>
<td>10.5 500</td>
</tr>
<tr>
<td>AIX05</td>
<td>26.00</td>
<td>9.5 1000</td>
</tr>
</tbody>
</table>

References

APPENDICES for Section III

Appendix III-A lists the 100 test sentences used by the Harpy and Hearsay-II systems, along with characteristics measuring their complexity relative to several grammars.

Appendix III-B is the phonetic dictionary for the 1011 words used in the AI retrieval language.

Appendix III-C contains the complete definition of three of the grammars (AIXF, AIX15, and AIX05) used in testing the systems. These grammars have become standards for future development and testing. AIXF was not used to test Harpy because the network was too large to be generated.
Appendix III-A. Characteristics of the AI Retrieval Task sentences

Below is a description of the test sentences used for the Harpy and Hearsay-II systems. The September Hearsay-II results used 22 of the sentences randomly selected from the 100. The entire set of 100 was used for the 100 single-speaker test sentences for Harpy, and 21 of them were used for the other four speakers tested on Harpy.

CMU Test Sentences

The branching factors previously given for the languages used by the CMU speech understanding systems (Harpy and Hearsay-II) are "static" branching factors (SBF) (as derived by Gary Goodman and described in his recent thesis). Intuitively, they can be thought of as being derived by doing a Monte Carlo probing of a network describing all acceptable word sequences and taking the average of the number of words possible following any legal initial sequence. Other groups have generated somewhat similar numbers.

What we present here is a characterization of the lexical fanout allowed by our grammars for the particular test sentences. The notion is to calculate the average fanout for each sentence-initial sequence of words (i.e., going left-to-right).

The method used here is the following: For any sequence of words, denote by Word Branches (WB) the number of words that may legally follow that sequence in the given language. Consider a sentence of length N-1 words to have N WB's -- each is calculated from the initial sequence of i words, i=0,1,...N. (I.e., the first WB for any sentence is always the same -- the number of legal first words.) Then, for any sentence or collection of sentences, the Average Fanout (AF) is the arithmetic mean of the WB's of the sentence(s).

The languages used (all defined using the same 1011-word vocabulary) are called AIX05, AIX15, and AIXF. The first two have static branching factors of 10 and 28, respectively. This summary is over 100 test sentences containing a total of 683 words.

<table>
<thead>
<tr>
<th>AF</th>
<th>AIX05</th>
<th>AIX15</th>
<th>AIXF</th>
<th>sents</th>
<th>words/sent</th>
</tr>
</thead>
<tbody>
<tr>
<td>33.4</td>
<td>46.5</td>
<td>68.0</td>
<td>100</td>
<td>6.83</td>
<td>(average over all)</td>
</tr>
<tr>
<td>17.3</td>
<td>26.0</td>
<td>33.4</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>31.3</td>
<td>45.4</td>
<td>84.0</td>
<td>10</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>36.1</td>
<td>50.7</td>
<td>73.0</td>
<td>11</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>29.7</td>
<td>41.5</td>
<td>60.3</td>
<td>21</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>33.6</td>
<td>47.0</td>
<td>70.2</td>
<td>24</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>37.2</td>
<td>51.1</td>
<td>70.3</td>
<td>15</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>30.1</td>
<td>40.5</td>
<td>63.0</td>
<td>9</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>42.3</td>
<td>61.5</td>
<td>70.8</td>
<td>3</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>42.8</td>
<td>57.9</td>
<td>76.3</td>
<td>3</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>21.2</td>
<td>29.9</td>
<td>53.4</td>
<td>2</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>
The 100 sentences, presented with fanouts according to the AIX05 language.

[66] DO 6 ANY 6 OF 3 THESE 3 MENTION 192 PSYCHOLOGY 3 words=6 AF=39.857
[66] WHICH 21 COGNITIVE 1 PSYCHOLOGY 2 CONTAINED 192 WINOGRAD'S 1 ARTICLE 1 words=6 AF=40.571
[66] WHAT 26 TOPICS 1 ARE 1 RELATED 1 TO 192 SEMANTIC 2 NETWORKS 3 . words=7 AF=36.500
[66] DOES 16 PATTERN 3 DIRECTED 1 FUNCTION 1 INVOCATION 3 GET 2 DISCUSSED 1 ANYWHERE 1 words=8 AF=30.444
[66] WHICH 21 TITLES 1 CONTAIN 1 THE 1 PHRASE 192 TIME 2 COMPLEXITY 3 words=7 AF=35.875
[66] DOES 16 THAT 1 ARTICLE 1 MENTION 192 TIME 2 OR 1 SPACE 1 BOUNDS 3 words=8 AF=51.444
[66] WHICH 21 OF 2 THEM 1 DISCUSSES 192 EVALUATION 1 FUNCTIONS 3 words=6 AF=38.857
[66] ARE 292 THERE 2 ANY 5 ABSTRACTS 1 WHICH 1 REFER 1 TO 192 PAPERS 1 BY 96 NEWELL 3 words=10 AF=60.000
[66] WHERE 5 IS 192 PREDICATE 1 CALCULUS 3 MENTIONED 1 words=5 AF=44.667
[66] WHAT 26 ARE 3 SOME 1 OF 1 THE 1 AREAS 1 OF 192 ARTIFICIAL 1 INTELLIGENCE 3 words=9 AF=29.500
[66] WHAT 26 WAS 1 ITS 1 TITLE 1 words=4 AF=19.000
[66] WHO 5 WAS 2 THE 1 AUTHOR 1 words=4 AF=15.000
[66] WHERE 5 DOES 1 HE 1 WORK 1 words=4 AF=14.800
[66] WHAT 26 IS 4 HER 1 AFFILIATION 1 words=4 AF=19.600
[66] WHAT 26 ADDRESS 1 IS 1 GIVEN 1 FOR 1 THE 1 AUTHORS 1 words=7 AF=12.258
[66] HOW 4 MANY 8 REFERENCES 1 ARE 1 GIVEN 1 words=5 AF=13.500
[66] PLEASE 4 LIST 1 THE 1 AUTHORS 1 words=4 AF=14.600
[66] PLEASE 4 MAKE 1 ME 1 A 1 FILE 1 OF 1 THOSE 1 words=7 AF=9.500
[66] CAN 2 1 HAVE 1 THESE 1 ABSTRACTS 1 LISTED 1 words=6 AF=10.429
[66] ARE 292 ANY 6 ARTICLES 2 ABOUT 192 STRUCTURED 1 PATTERN 1 RECOGNITION 3 words=7 AF=78.375
[66] DO 6 ANY 6 OF 3 THESE 1 ALSO 1 DISCUSS 192 ABSTRACTION 3 words=7 AF=34.750
[66] WHICH 21 PAPERS 7 ON 192 LANGUAGE 6 UNDERSTANDING 4 ARE 1 ABOUT 192 ENGLISH 3 words=8 AF=54.667
[66] DO 6 ANY 6 PAPERS 5 ON 192 AUTOMATIC 7 PROGRAMMING 3 EXIST 1 words=7 AF=35.750
[66] WHAT 26 ABOUT 288 PROGRAM 1 VERIFICATION 3 words=4 AF=76.800
[66] I 2 AM 2 INTERESTED 1 IN 192 ARTIFICIAL 1 INTELLIGENCE 3 words=6 AF=38.143
[66] THE 3 AREA 2 1 AM 1 INTERESTED 1 IN 1 IS 192 UNDERSTANDING 3 words=8 AF=30.000
[66] DON'T 1 GET 1 ME 1 ANY 1 ARTICLES 1 WHICH 1 MENTION 192 GAME 2 PLAYING 3 words=9 AF=26.900
[66] I 2 AM 2 ONLY 1 INTERESTED 1 IN 1 PAPERS 1 ON 192 CHESS 4

42
LET'S RESTRICT OUR ATTENTION TO PAPERS SINCE 1974.

DO ANY PAPERS THIS YEAR CITE ROSENFIELD?

ARE COMPUTER NETWORKS MENTIONED ANYWHERE?

ARE ANY ARTICLES BY ROSENFIELD MENTIONED ANYWHERE?

ARE ANY ARTICLES BY ROSENFELD 3 ARE THERE ANY ABSTRACTS WHICH REFER TO PAPERS BY HOLLAND?

ARE THERE ANY PAPERS ON PROGRAM VERIFICATION MENTIONED ANYWHERE?

DO ANY OF THESE 3 ALSO MENTION PLANNER-LIKE LANGUAGES?

DOES PROBLEM SOLVING GET MENTIONED ANYWHERE?

WHICH 21 PAPERS CITE NEWELL AND SIMON?

ANY 1 ABSTRACTS 1 REFERRING 1 TO AI 4 OR ARTIFICIAL INTELLIGENCE 1 VERIFICATION 1 PROGRAM 1?

ARE ASSOCIATIVE 2 MEMORIES DISCUSSED IN RECENT JOURNALS?

ARE LEARNING 4 AND NEURAL NETWORKS MENTIONED ANYWHERE?

DID REddy PRESENT A PAPER AT IJCAI?

DIDN'T THAT PAPER QUOTE DREYFUS?

GET ME 2 EVERYTHING ON DYNAMIC CLUSTERING?

GENERATE A COPY OF THOSE ABSTRACTS?

HOW CAN I USE THE SYSTEM EFFICIENTLY?

I'D LIKE TO SEE THE MENUS ON GAME PLAYING?

WHAT ADDRESSES ARE GIVEN FOR THE AUTHORS?

WHAT PAPERS ON PREFERENTIAL SEMANTICS ARE THERE?

WHEN WAS A SEMANTIC NETWORK LAST REFERRED TO?
WHO 5 HAS 1 WRITTEN 1 ABOUT 192 AUTOMATIC 7 PROGRAMMING 3
WHO 5 WAS 2 QUOTED 1 IN 1 THAT 1 ARTICLE 1
WHICH 21 IS 1 THE 1 OLDEST 1
ARE 292 ANY 6 NEW 1 BOOKS 1 BY 76 TERRY 1 WINOGRAD 3
CAN 2 HAVE 1 THESE 1 ABSTRACTS 1 LISTED 1
DID 99 CARL 1 HEWITT 5 PRESENT 2 A 1 PAPER 1 AT 2 THE 1 IFIP 1
MEETINGS 1 IN 1 SEPTEMBER 1
DID 99 ANY 4 ACL 1 PAPERS 1 CITE 96 RICK 1 HAYES-ROTH 3
DO 6 ANY 6 OF 3 THOSE 1 PAPERS 1 MENTION 192 AXIOMATIC 1
SEMATICS 3
DURING 1 WHAT 1 MONTHS 1 WERE 1 THEY 1 PUBLISHED 1
HOW 4 MANY 8 RECENT 1 ISSUES 1 CONCERN 192 INVARIANCE 1 FOR 1
PROBLEM 1 SOLVING 3
HOW 4 MANY 8 SUMMARIES 1 DISCUSS 192 KNOWLEDGE 2 BASED 1 SYSTEMS 3
HAVE 97 ANY 2 NEW 1 PAPERS 1 BY 96 LEE 1 ERMAN 3
I'D 1 LIKE 1 TO 2 KNOW 1 THE 1 PUBLISHERS 1 OF 1 THAT 1 STORY 1
IS 290 HUMAN 3 BEHAVIOR 5 OR 191 HUMAN 3 MEMORY 3 DISCUSSED 2 IN 1
ONE RECENT 1 SUMMARY 1
LIST 2 THE 2 ABSTRACTS 1 BY 96 HERB 1 SIMON 3
WHAT 26 ABOUT 288 ALLEN 2 COLLINS 3
WHERE 5 DID 1 THAT 1 ARTICLE 1 APPEAR 1
WHO 5 HAS 1 WRITTEN 1 ABOUT 192 LANGUAGE 6 COMPREHENSION 3 AND 191 LANGUAGE 6 DESIGN 1
QUIT 1 LISTING 1 PLEASE 1
WEREN'T 1 SOME 1 ARTICLES 1 PUBLISHED 1 ON 192 GOAL 1 SEEKING 1
COMPONENTS 3
WHAT 26 SORTS 1 OF 192 LANGUAGE 6 PRIMITIVES 3 ARE 1 WRITTEN 1 UP 1
HASN'T 192 A 21 CURRENT 1 REPORT 1 ON 192 PRODUCTION 1 SYSTEMS 3 BEEN 1 RELEASED 1
ARE 292 THERE 2 ANY 5 ISSUES 1 ABOUT 192 COOPERATING 1 SOURCES 1
OF 1 KNOWLEDGE 3
DID 99 VIC 1 LESSER 5 PRESENT 2 PAPERS 1 AT 2 IFIP 1
DID 99 ANYONE 1 PUBLISH 1 ABOUT 192 LARGE 1 DATA 1 BASES 3 IN 1
COMMUNICATIONS 1 OF 1 THE 1 ACM 1
DO 6 ANY 6 AUTHORS 1 DESCRIBE 192 DRAGON 3
DOES 196 HE 1 WORK 1 AT 1 CMU 1
DO 6 ANY 6 RECENT 4 ACM 1 CONFERENCES 1 CONSIDER 192 SEMANTIC 2
NETS 3 OR 191 SEMANTIC 2 NETWORKS 1
6G DO 6 RESPONSES 1 EVER 1 COME 1 FASTER 1 words=5 AF=12.667
6G HAS 96 LEE 1 ERMAN 4 BEEN 1 REFERENCED 1 IN 1 ANY 1 OF 1 THOSE 1 words=9 AF=17.300
6G HAS 96 ALLEN 2 NEWELL 4 PUBLISHED 2 ANYTHING 1 RECENTLY 1 words=6 AF=24.571
6G HAVE 97 ANY 2 NEW 1 PAPERS 1 BY 96 TERRY 1 WINograd 3 APPEARED 1 words=8 AF=29.778
6G HOW 4 BIG 1 IS 1 THE 1 DATA 1 BASE 1 words=6 AF=10.714
6G HOW 4 MANY 8 OF 1 THESE 1 ALSO 1 DISCUSS 192 DYNAMIC 3 BINDING 3 words=8 AF=31.000
6G HOW 4 MANY 8 RECENT 1 ISSUES 1 CONCERN 192 DISPLAY 1 TERMINALS 3 words=7 AF=34.500
6G KILL 1 THE 1 LISTING 1 words=3 AF=17.250
6G PLEASE 4 MAKE 1 ME 1 A 1 FILE 1 OF 1 THOSE 1 words=7 AF=9.500
6G WHAT 26 IS 4 HIS 1 AFFILIATION 1 words=4 AF=19.600
6G WHICH 21 OF 2 THESE 5 CITES 96 PERRY 1 THORNDYKE 3 words=6 AF=27.714
6G WHICH 21 PAPERS 7 ON 192 DESIGN 6 IN 1 THE 1 ARTS 4 ALSO 2 DISCUSS 192 DESIGN 5 AUTOMATION 3 words=11 AF=41.667
6G WHO 5 WAS 2 QUOTED 1 IN 1 THAT 1 ARTICLE 1 words=6 AF=11.000
6G WHICH 21 PAPERS 7 WERE 1 WRITTEN 2 AT 1 NRL 1 OR 1 AT 1 SMC 1 words=9 AF=10.200
Appendix III-B. AI Retrieval Language Dictionary

A (-,0) (AX1, UH4, EH4, EYL, EVC1, EYR)
ABOUT (-,0) (AH2, AX, EH3, (0,+) (-,0), (-4), (B,0) (AWL,0) AWC1 (AWR,0) ((-,-), (-4)) (T,0), DX)
ABSTRACT (-,0) AE3 (+, (-0), -) S (-,0) (DR, R, T) AE21 ((-,0), (-4)) (T,0), DX
ABSTRACTION (-,0) AE3 (+, (-0), -) S (-,0) (DR, R, T) AE21 ((-,0), -) SH IH5 N
ABSTRACTIONS (-,0) AE3 (+, (-0), -) S (-,0) (DR, R, T) AE21 ((-,0), -) S (HH,0)
ACL (-,0) (EYL,0) EYC1 (EYR,0) S IY (EH EL, EL2)
ACM (-,0) (EYL,0) EYC (EYR,0) S IY AH2 M
ACQUISITION (-,0) AE5 (+, (-0), -) WH IH1 (Z[4],(Z,0) S) IH2 SH IH5 N
ACQUISITIONS (-,0) AE5 (+, (-0), -) SH IH5 N (Z[4],(Z,0) S)
ACTIVE (-,0) AE (+, (-0), -) T IH2 (F,0)
ACYCLIC (-,0) (EYL,0) EYC (EYR,0) S IH3 (I,0) (K,0) L UH2 (<- <-,0),-{4}) (K,0)
ADAPTATION (-,0) AE4 (+, (-0), -) (D,0) AE5 (+, (-0), -) T (EYL,0) EYC1 (EYR,0) SH IH5 N
ADAPTIVE (-,0) AE4 (-,0) (D,0) AE5 (+, (-0), -) T IH2 (F,0)
ADDRESS (-,0) (AE, IX), (UH) (EYL,0) EYC (EYR,0) S IH3! (I,0) (K,0)
ADDRESSES (-,0) (AE, IX), (UH) (EYL,0) EYC (EYR,0) S IH3! (I,0) (K,0)
ADVISING (-,0) AE5 (+, (-0), -) (D,0) AE (+, (-0), -) T IH2 (F,0)
AESTHETICS (-,0) (IX, UH) (EYL,0) EYC (EYR,0) S IH3! (I,0) (K,0)
AFFILIATION (-,0) (EYL,0) EYC (EYR,0) S IH3! (I,0) (K,0)
AFFILIATIONS (-,0) (EYL,0) EYC (EYR,0) S IH3! (I,0) (K,0)
AFTER (-,0) AE (+, (-0), -) T, DX) ER
AI (-,0) (EYL,0) EYC (EYR,0) (AYL,0) AYC (AYR,0)
ALGEBRAIC (-,0) AE3 EL (*- (-,0),-) SH (,8} IH (4- (-,0),-{4}) (B,0) (EYL,0) EYC (EYR,0) IH2 (I,0)
ALGOL (-,0) AE4EL (4- (-,0),-{4}) (G,0) 0W3 EL3
ALGORITHM (-,0) AE EL (+, (-0), -) (G,0) (AA, OW) R! IH (TH, DH) (IH, IX, 0) M
ALGORITHMIC (-,0) AE EL (+, (-0), -) (G,0) (AA, OW) R! IH (TH, DH) M IH (+, (-0), (-4)) (K,0)
ALL (-,0) 0W4! EL
ALL-OR-NONE (-,0) 0W4! EL (-,0) (AA4,0) ER21[7,14] (-,0) N UH (N, DX)
ALLEN (-,0) AE1! F (+, (-0), -) T, DX) ER
AM (-,0) ((EH2', AH4) M, EM»)
AMONG (-,0) (IX, AX), M UH2 NX
AN (-,0) AE5! (EN, N)
ANALOGY (-,0) AE5 (N, EN) AE4! (EL, L) 0W4 (-,0), -) SH, IH, (0) IH, (I, 0) 0W (EH2, UH4)
ANALYSIS (-,0) UH4 N AE EL3 (UH2, [2,6], IH5,0) S IH6 S (HH,0)
ANALYZER (-,0) AE5 N EL2 (AVL,0) AYC1 (AYR,0) (Z[4],(Z,0) S) ER21
AND (-,0) AE5! (5, 10) N (+, (-0), -) (D, 0)
ANN (-,0) AE41 (N, DX)
ANOTHER (-,0) AH N AA21 (DH, TH) (ER, AA2)
ANWER (-,0) AE5! N S ER
ANSWERING (-,0) AE4 N S! (R, ER) IH5 NX
ANTHONY (-,0) AE4 (N, +) (+, -0), -) TH IH4 N IY
ANY (-,0) (EH3, EH) (N[2], DX) IY! (IY3, 0)
ANYONE (-,0) (EH3, EH) (N[2], DX) IY! (IY3, 0) (-,0) W AH (N, DX)
ANYTHING (-,0) (EH3, EH) (N[2], DX) IY! (IY3, 0) (-,0) W (EH3,0) ER
APPEAR (-,0) (AH3, UH2) (+, (-0), -) (P, PH) JY21 ER[1, 18]
APPEARED (-,0) (AH3, UH2) (+, (-0), -) (P, PH) JY21 ER[1, 18] (+, (-0), -) (D, DH, 0)
APPLICATION (-,0) AE3 (-, (-0), -) (P LPL, (L,0)) IH6 (-, -0), -) (K,0) (EYL,0) EYC1 (EYR,0) SH IH5 N
APPRENTICE (-,0) AE3 (-,0) (-0), -) (P, PR, (R, 0)) UH2 N (+, -0), -) T IH4 S (HH,0)
APPROACH (-,0) UH2 (+, (-0), -) (P, PR, PR, R) 0W2 (+, -0), -) SH[I, 8]
APRIL (-,0) (EYL,0) EYC1 (EYR,0) (+, (-0), -) (P, PR, (R, 0)) IH EL, EL2
ARE (-,0) AA R (+, (-0), (-4)) (B,0) IY (+, -0), -) (B,0)
AREA (-,0) (AA3[1]) (ER2, ER21)
AREA (-,0) 1H2 ER 1Y2 UH
Dictionary
<table>
<thead>
<tr>
<th>EXPLANATION</th>
<th>(-,0) EH (+ (-,0),-) S - (P, L, I, L, L, I) IH N (ELY,0) EVC! (EVR,0) SH IH5 N</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXPRESSIONS</td>
<td>(-,0) IH3 (+ (-,0),-) S - (F, R, PR, R, 0, 0) EH3! SH (IH5 N, 0, 0) (Z[4], [2, 0]) S</td>
</tr>
<tr>
<td>FABLES</td>
<td>(-,0) F (ELY,0) EVC! (EVR,0) (+ (-,0),-[4]) R, 0, 0) AX (L, 0) (Z[4], [2, 0]) S</td>
</tr>
<tr>
<td>FACES</td>
<td>(-,0) F (ELY,0) EVC! (EVR,0) S IH4 (Z[4], [2, 0]) S</td>
</tr>
<tr>
<td>FACTS</td>
<td>(-,0) F AE! (+ (-,0),-) S (HH, 0)</td>
</tr>
<tr>
<td>FAIRY</td>
<td>(-,0) F EH! (ER, R) IV</td>
</tr>
<tr>
<td>FASTER</td>
<td>(-,0) F AE! (+ (-,0),-) S (HH, 0)</td>
</tr>
<tr>
<td>FEATURE-DRIVEN</td>
<td>(-,0) F IV (+ (-,0),-) SH, 10) ER! (+ (-,0),-) DR R IH V (((IH, 0), N), EN)</td>
</tr>
<tr>
<td>FEBRUARY</td>
<td>(-,0) F EH! (+ (-,0),-[4]) R, 0, 0) (LW, L, 0, 0) AA (ER, R) IV</td>
</tr>
<tr>
<td>FEDERAL</td>
<td>(-,0) F EH! (+ (-,0),-) D, DX) ER2 EL3</td>
</tr>
<tr>
<td>FEIGENBAUM</td>
<td>(-,0) F (ELY,0) AV3! (EYR, 0) (+ (-,0),-) (G, 0) IH5 N (+ (-,0),-) 0, (AWL, 0) AWC! (AWR, 0) M</td>
</tr>
<tr>
<td>FELDMAN</td>
<td>(-,0) F EH! EL (+ (-,0),-) M IH6 (N, DX)</td>
</tr>
<tr>
<td>FICTION</td>
<td>(-,0) F IH2! (+ (-,0),-) SH! IH5 N</td>
</tr>
<tr>
<td>FIFTEEN</td>
<td>(-,0) F IH2! F (+ (-,0),-) TD, DX) IV (N, DX)</td>
</tr>
<tr>
<td>FIFTY</td>
<td>(-,0) F IH2! F (+ (-,0),-) TD, DX) IV</td>
</tr>
<tr>
<td>FIKES</td>
<td>(-,0) F (ELY,0) AY3! (AYR, 0) (+ (-,0),-) S (HH, 0)</td>
</tr>
<tr>
<td>FILE</td>
<td>(-,0) F (ELY,0) AY3! (AYR, 0) EL3</td>
</tr>
<tr>
<td>FINISH</td>
<td>(-,0) F IH3! N IH5 SH (HH, 0)</td>
</tr>
<tr>
<td>FINISHED</td>
<td>(-,0) F IH3! N IH5 SH +[4] (T, 0)</td>
</tr>
<tr>
<td>FIRST</td>
<td>(-,0) F (AA3 ER, ER) S +[4] (T, 0)</td>
</tr>
<tr>
<td>FIVE</td>
<td>(-,0) F (ELY,0) AY3! (AYR, 0) (V, F)</td>
</tr>
<tr>
<td>FOR</td>
<td>(-,0) F (AA2, 0) ER!</td>
</tr>
<tr>
<td>FORESTS</td>
<td>(-,0) F AA2 ER! IH S - S (HH, 0)</td>
</tr>
<tr>
<td>FORMAL</td>
<td>(-,0) F AA2! ER2 (M, 0) EL2</td>
</tr>
<tr>
<td>FORMATION</td>
<td>(-,0) F AA2! ER M (ELY, 0) EVC! (EVR, 0) SH IH5 N</td>
</tr>
<tr>
<td>FORTY</td>
<td>(-,0) F AA2! (ER, ER) (+ (-,0),-) M IH6 (N, DX)</td>
</tr>
<tr>
<td>FOUR</td>
<td>(-,0) F AA2! ER</td>
</tr>
<tr>
<td>FOURTEEN</td>
<td>(-,0) F AA2! ER (+ (-,0),-) TD, DX) IV (N, DX)</td>
</tr>
<tr>
<td>FRAME</td>
<td>(-,0) F R (ELY, 0) EVC! (EVR, 0) M</td>
</tr>
<tr>
<td>FRAMES</td>
<td>(-,0) F R (ELY, 0) EVC! (EVR, 0) M (Z[4], [2, 0]) S</td>
</tr>
<tr>
<td>FROM</td>
<td>(-,0) F AA2! M</td>
</tr>
<tr>
<td>FU</td>
<td>(-,0) F LW2!</td>
</tr>
<tr>
<td>FUNCTION</td>
<td>(-,0) F AA2! NX (+ (-,0),-) SH IH5 N</td>
</tr>
<tr>
<td>FUNCTIONS</td>
<td>(-,0) F AA2! NX (+ (-,0),-) SH IH5 N (Z[4], [2, 0]) S</td>
</tr>
<tr>
<td>FUZZY</td>
<td>(-,0) F LW2! (Z[4], [2, 0]) S IV</td>
</tr>
<tr>
<td>GAME</td>
<td>(+ (-,0),-) (G, 0) (ELY, 0) EVC! (EVR, 0) M</td>
</tr>
<tr>
<td>GAMES</td>
<td>(+ (-,0),-) (G, 0) (ELY, 0) EVC! (EVR, 0) M (Z[4], [2, 0]) S</td>
</tr>
<tr>
<td>GARY</td>
<td>(+ (-,0),-) (G, 0) AE2! ER IV</td>
</tr>
<tr>
<td>GASCHNIG</td>
<td>(+ (-,0),-) (G, 0) AE2! SH N IH3 (+ (-,0),-[4]) (K, 0)</td>
</tr>
<tr>
<td>GENERAL</td>
<td>(+ (-,0),-) SH, 10) EH2 N ER2! EL3</td>
</tr>
<tr>
<td>GENERATE</td>
<td>(+ (-,0),-) SH, 10) EH2 N ER (ELY, 0) EVC! (EVR, 0) ((+ (-,0),-[4]) (T, 0), DX)</td>
</tr>
<tr>
<td>GENERATION</td>
<td>(+ (-,0),-) SH, 10) EH2 N ER (ELY, 0) EVC! (EVR, 0) SH IH5 N</td>
</tr>
<tr>
<td>GEOMETRIC</td>
<td>(+ (-,0),-) SH, 10) IV IH M EH2! (+ (-,0),-) DR R IH8 (+ (-,0),-[4]) (K, 0)</td>
</tr>
<tr>
<td>GEORGE</td>
<td>(+ (-,0),-) SH, 10) LW4 ER! (+ (-,0),-) SH, 10)</td>
</tr>
<tr>
<td>GET</td>
<td>(+ (-,0),-) (G, 0) (EYR, 0) IH3! (+ (-,0),-[4]) (T, 0), DX)</td>
</tr>
<tr>
<td>GIPS</td>
<td>(+ (-,0),-) (G, 0) IH3! (+ (-,0),-) S (HH, 0)</td>
</tr>
<tr>
<td>GIVE</td>
<td>(+ (-,0),-) (G, 0) IH3! (F, V (F, 0))</td>
</tr>
<tr>
<td>GIVEN</td>
<td>(+ (-,0),-) (G, 0) IH3! V IH!4 (N, DX)</td>
</tr>
<tr>
<td>GM</td>
<td>(+ (-,0),-) SH, 10) IV! EH2 M</td>
</tr>
<tr>
<td>GO</td>
<td>(+ (-,0),-) (G, 0) OW3! 36!</td>
</tr>
<tr>
<td>GO-MOKU</td>
<td>(+ (-,0),-) (G, 0) OW! M OW! (+ (-,0),-) (K, 0) UW</td>
</tr>
<tr>
<td>GOAL</td>
<td>(+ (-,0),-) (G, 0) OW! EL</td>
</tr>
<tr>
<td>GOALS</td>
<td>(+ (-,0),-) (G, 0) OW3! EL (Z[4], [2, 0]) S</td>
</tr>
<tr>
<td>GRAIN</td>
<td>(+ (-,0),-) (G, 0) R (ELY, 0) EVC! (EVR, 0) (N, DX)</td>
</tr>
<tr>
<td>GRAMMARS</td>
<td>(+ (-,0),-) (G, 0) R AE5! M ER! (Z[4], [2, 0]) S</td>
</tr>
<tr>
<td>GRAMMATICAL</td>
<td>(+ (-,0),-) (G, 0) ER M AE! (+ (-,0),-) TD, DX) IH8 (+ (-,0),-[4]) (K, 0) EL</td>
</tr>
<tr>
<td>GRAPH</td>
<td>(+ (-,0),-) (G, 0) R AE3! F (HH, 0)</td>
</tr>
</tbody>
</table>

51
Dictionary

GRAPHICS

HAMBURG

HANS

HAPPEN

HARRY

HAS

HASN'T

HAVE

HAVEN'T

HAYES-ROTH

HE

HEARSAY

HELP

HER

HERB

HERBERT

HEILEMABLE

HEP

HEIR

HEPB

HEURISTIC

HEWITT

HILARY

HILL

HIS

HISTORY

HOLLAND

HOW

HUGH

HUMAN

HUNDRED

HUNGRY

HUNT

HYPOTHESIS

I

I'D

I'M

IEEE

IFIP

ILLINOIS

IMAGE

IMPROVING

IN

INDUCTIVE

INDUSTRIAL

INEXACT

INFERENCES

INFERENTIAL

INFORMATION

INHERITANCE

INSANITY

INSTITUTE

INTELLIGENCE

INTENSITY

INTENTIONS

ILLEGICAL

I'VE
SIKLOSSY  (-,0) S IH2 (->[<-,-,-]) (K,0) L AA2! S IV
SIMON    (-,0) S (AYL,0) AYC! (AYR,0) M (UH2,1H3) (N,DX)
SIMULATION (-,0) S IH3 M Y UH L (EYL,0) EVC! (EVR,0) SH (1H5 N,EN)
SIMULTANEOUS (-,0) S (AYL,0) AYC! (AYR,0) M EL3 (->[<-,-,-]) T (EYL,0) EVC! (EVR,0) N IV IH3 S (HH,0)
SIMULTANEOUSLY (-,0) S (AYL,0) AYC! (AYR,0) M EL3 (->[<-,-,-]) T (EYL,0) EVC! (EVR,0) N IV IH3 S L IV
SINCE   (-,0) S (IH3,1H6) N! S (HH,0)
SIX      (-,0) S IH3! (->[<-,-,-]) S (HH,0)
SIXTEEN  (-,0) S IH3! (->[<-,-,-]) S - T IV (N,DX)
SIXTY    (-,0) S IH3! (->[<-,-,-]) S - T IV
SIZE     (-,0) S (AYL,0) AYC! (AYR,0) M (Z[4],Z,0) (Z,0) (Z,0) S)
SLAGLE   (-,0) S L (EYL,0) EVC! (EVR,0) (->[<-,-,-]) (G,0) EL
SLOW     (-,0) S L OW!
SMC      (-,0) S EH4 S EH2! M S IV
SMITH    (-,0) S M IH3! TH (HH,0)
SNARING  (-,0) S N (EYL,0) EVC! (EVR,0) ER (IH3,IV) NX
SO       (-,0) S OW3!
SOBEL    (-,0) S OW3! (->[<-,-,-]) (B,0) EL
SOFTWARE (-,0) S AGF F (->[<,>] (T,0) W ER2
SOLOWAY  (-,0) S AO EL3 UW2 W (EYL,0) EVC! (EVR,0)
SOLUTIONS (-,0) S OW! L! UWH SH IH5 N (Z[4],Z,0) S)
SOLVING  (-,0) S AA! EL2 V (IH5 NX
SOME     (-,0) S AA! M
SOMETHING (-,0) S AA! M TH (IH3,1H3,IV) NX
SOMETHEN  (-,0) S AA! M W EH3 ER
SORT     (-,0) S LW4! ER (->[<-,-,-]) (T,0,DX)
SORTS    (-,0) S LW4! ER (->[<-,-,-]) S (HH,0)
SOURCES  (-,0) S LW4! ER S IH4! (Z[4],Z,0) S)
SPACE    (-,0) S - (P,0) (EYL,0) EVC! (EVR,0) S (HH,0)
SPANNING (-,0) S - (P,0) AE5 (N,DX) (IH3,1H3,IV) NX
SPEECH   (-,0) S - (P,0) IV! (->[<-,-,-]) SH (HH,0)
SPEED    (-,0) S - (P,0) IV! (->[<-,-,-]) (D,0)
SPROULL  (-,0) S - (P R,PR <R,0>) AO EL3
SRI      (-,0) S EI4! S AA2 ER2 (AYL,0) AYC! (AYR,0)
STANFORD (-,0) S - T AE5 N F ER (->[<-,-,-]) (D,0)
STATE    (-,0) S - T (EYL,0) EVC! (EVR,0) (->[<-,-,-]) (T,0,DX)
STEREO   (-,0) S - T IH3! ER IV2 OW
STEVE    (-,0) S - T IV! V (F,0)
STOCHASTIC (-,0) S - T IH3! (->[<-,-,-]) (K,0) AE4! S - T IH3! (->[<-,-,-]) (K,0)
STOCK    (-,0) S - T AO! (->[<-,-,-]) (K,0)
STOP     (-,0) S - T AA! (->[<-,-,-]) (P,0)
STORAGE  (-,0) S - T LW4! ER IH2! (->[<-,-,-]) (ZH (SH,0), SH)
STORED   (-,0) S - T (AA4,AO) EI! (->[<-,-,-]) (D,0)
STORIES  (-,0) S - T (AA4,AO) EI! (Z[4],Z,0) S)
STORY    (-,0) S - T AO! ER IV
STRUCTURE (-,0) S - DR R EH3! (->[<-,-,-]) SH,8) ER
STRUCTURED (-,0) S - DR R EH3! (->[<-,-,-]) SH,8) ER (->[<-,-,-]) (D,SH,0)
STRUCTURES (-,0) S - DR R EH3! (->[<-,-,-]) SH,8) ER (Z[4],Z,0) S)
STUDIES   (-,0) S - T UH4! (->[<-,-,-]) D,DX) IV (Z[4],Z,0) S)
SUBJECTS (-,0) S AA! (->[<-,-,-]) SH IH3! (->[<-,-,-]) (T,0,DX)
SUBJECTS (-,0) S AA! (->[<-,-,-]) SH IH3! (->[<-,-,-]) S (HH,0)
SUBPROBLEMS (-,0) S UH2! (->[<-,-,-]) (P R,PR <R,0>) AO! (->[<-,-,-]) (B,0) EL2 M (Z[4],Z,0) S)
SUBSELECT (-,0) S AA! (->[<-,-,-]) (B,0) S AX EL EL (->[<-,-,-]) (T,0,DX)
SUBSYSTEM (-,0) S UH2! (->[<-,-,-]) (R,0) S IH4! S - T (IH6 M,EM)
SUMEX    (-,0) S UH2! M EH2 (->[<-,-,-]) S (HH,0)
SUMMARIES (-,0) S UH4! M R IV2 (Z[4],Z,0) S)
SUMMARY  (-,0) S UH4! M R IV2
SUN       (-,0) S UH2! M EH2 (->[<-,-,-]) S (HH,0)
SUNSHINE  (-,0) S UH2 N SH (AYL,0) AYC! (AYR,0) (N,DX)
<table>
<thead>
<tr>
<th><strong>Dictionary</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SURE</strong></td>
</tr>
<tr>
<td><strong>SURKNOTES</strong></td>
</tr>
<tr>
<td><strong>SURVEY</strong></td>
</tr>
<tr>
<td><strong>SURVEYS</strong></td>
</tr>
<tr>
<td><strong>SUSSX</strong></td>
</tr>
<tr>
<td><strong>SUZUKI</strong></td>
</tr>
<tr>
<td><strong>SYKES</strong></td>
</tr>
<tr>
<td><strong>SYMBOL</strong></td>
</tr>
<tr>
<td><strong>SYNCHRONIZATION</strong></td>
</tr>
<tr>
<td><strong>SYNTACTIC</strong></td>
</tr>
<tr>
<td><strong>SYNTAX</strong></td>
</tr>
<tr>
<td><strong>SYNTHESIS</strong></td>
</tr>
<tr>
<td><strong>SYNTESIZER</strong></td>
</tr>
<tr>
<td><strong>SYSTEM</strong></td>
</tr>
<tr>
<td><strong>SYSTEMS</strong></td>
</tr>
<tr>
<td><strong>TAKE</strong></td>
</tr>
<tr>
<td><strong>TALES</strong></td>
</tr>
<tr>
<td><strong>TASK</strong></td>
</tr>
<tr>
<td><strong>TECH-II</strong></td>
</tr>
<tr>
<td><strong>TECHNICAL</strong></td>
</tr>
<tr>
<td><strong>TECHNIQUES</strong></td>
</tr>
<tr>
<td><strong>TECHNOLOGY</strong></td>
</tr>
<tr>
<td><strong>TED</strong></td>
</tr>
<tr>
<td><strong>TELELOGICAL</strong></td>
</tr>
<tr>
<td><strong>TELL</strong></td>
</tr>
<tr>
<td><strong>TEMPLAR</strong></td>
</tr>
<tr>
<td><strong>TEN</strong></td>
</tr>
<tr>
<td><strong>TERMINAL</strong></td>
</tr>
<tr>
<td><strong>TERMINALS</strong></td>
</tr>
<tr>
<td><strong>TERMINATE</strong></td>
</tr>
<tr>
<td><strong>TERMINATION</strong></td>
</tr>
<tr>
<td><strong>TERRY</strong></td>
</tr>
<tr>
<td><strong>THAUMATURGIST</strong></td>
</tr>
<tr>
<td><strong>THICK</strong></td>
</tr>
<tr>
<td><strong>THEIR</strong></td>
</tr>
<tr>
<td><strong>THEM</strong></td>
</tr>
<tr>
<td><strong>THEOREM</strong></td>
</tr>
<tr>
<td><strong>THERE</strong></td>
</tr>
<tr>
<td><strong>THESE</strong></td>
</tr>
<tr>
<td><strong>THEY</strong></td>
</tr>
<tr>
<td><strong>THIRTEEN</strong></td>
</tr>
<tr>
<td><strong>THIRTY</strong></td>
</tr>
<tr>
<td><strong>THROUGH</strong></td>
</tr>
<tr>
<td><strong>TILL</strong></td>
</tr>
<tr>
<td><strong>TIME</strong></td>
</tr>
<tr>
<td><strong>TIMES</strong></td>
</tr>
</tbody>
</table>
Appendix III-C-1. AI Retrieval Language Grammar: AIXF

<SUTTERANCE> = [ <$SENTENCE> ]
<SSENTENCE> = $THE
$A
$AN
$ACQUIRE = HAVE
SEE
KNOW
GET
$AFFILIATION = $ADDRESS/S
$ADDRESS/S = ADDRESSES
ADDRESS
$AFFILIATION/S = AFFILIATIONS
AFFILIATION
$AI = AI
ARTIFICIAL INTELLIGENCE
$ALSO = ALSO
IN ADDITION
$ALSO/MENTION/TOPICS = $MENTION $TOPICS
$MENTION = CITE
REFER TO
$BE - $REF1
$DISCUSS/S
CONCERN
CONTAIN THE PHRASE
DESCRIBE
RELATE TO
$HAVE - $MENTIONED/HAVE
CONSIDER
$MENTION/S
$TOPICS = $TOPIC
$TOPIC = $CONJUNCTION $TOPIC
$ALWAYS = ALWAYS
USUALLY
REGULARLY
$ANY/NO/Pieces = $PIECE/S
$SOMETHING = $SOME $PIECE/S
$PIECE/S = $ARTICLE/S
$BOOK/S
$PAPER/S
$ABSTRACT/S
$PROCEEDING/S
$REPORT/S
$ISSUE/S
$JOURNAL/S
$NOTES
$REVIEW/S
$VOLUME/S
$PIECE
$SURVEY/S
$SUMMARY/S
$TECHNICAL PAPERS
<$PIECES1 2> = <STORY/S 2>
ARTICLE/S 2
BOOK/S 2
PAPER/S 2
ABSTRACT/S 2
PROCEEDINGS/S 2
REPORT/S 2
ISSUE/S 2
JOURNAL/S 2
NOTES
REVIEW/S 2
VOLUME/S 2
PIECE
SURVEY/S 2
SUMMARY/S 2
TECHNICAL PAPERS

<$SOMETHING> = ANYTHING
SOMETHING
EVERYTHING

<$SOME1> = <$SOME>

<$ANY!PIECES> = <$PIECES>
<$SOMEPieces> = <SOMEPieces>
<$SOME1> OF THE <$PIECES>
<$SOMETHING> = <$RECENT>

<$PIECES1> = <$PIECES1>
<$DATE> = <$PIECES1>
<$PIECES1> = <$WHENDATE> <$PIECES1> = <$WHENDATE>
<$RECENT> = <$PIECES1>

<$SOMEPIECES> = <$SOMETHING>
<$A> = <$PIECES>
<$SOME1> = <$THATPIECES>
<$SOME1> = <$PIECES>

<$SOME1> = ALL
MANY
ANY
ANY MORE
MORE
SOME
ANOTHER
SOME MORE

<$RECENT> = LATEST
RECENT
NEW
CURRENT

<$ANY!SOURCEPIECES> = <$SOURCEPIECES>
<$SOME1> = <$SOURCEPIECES>
<$SOME1> = <$RECENT> = <$SOURCEPIECES>$RECENT
<$SOME1> = <$PIECES> = <$FROM> = <$SOURCE>
<$RECENT> = <$SOURCEPIECES>$RECENT
<$PIECES> = <$FROM> = <$SOURCE>

<$SOURCEPIECES> = <$CONFERENCE>
$SOURCE = <$PIECES1 2>
$PROCEEDINGS/S = <$FROM> = <$A> = <$CONFERENCE>
$CONFERENCE = <$PIECES1 2>
$PIECES = <$FROM> = <$SOURCE>
$SOURCEPIECES$RECENT = <$CONFERENCE>

64
HUMAN PROBLEM SOLVING
THOUGHT AND LANGUAGE

ASK
REQUEST
DEMAND
SAY

AUTHORS:
REDDY
DREYFUSS
ANN RUBIN
ANTHONY MARTELLI
BERNARD MELTZER
BERT RAPHAEL
RONNIE NAGH-WEBSBER
CHRISTOPHER RIESBECK
CHUCK RIEGER
DAVE RUMI HART
DAVID MARR
DAVID MICHIE
DICK SELTZER
DONALD NORMAN
DOUG LÉNAT
DREW McDERMOTT
EARL HUNT
EARL SACHERDOT)
ED RISEMAN
ELLIOI T SOLOWAY
ERIK SANDEWALL
EUGENE CHARNIAK
GARY HENDRIX
GEORGE ERNST
HERBERT BLOCK
HILARY PUTNAM
HUGH NAGEL
IRV SOBEL
JACK MINKER
JACK MOSTOW
JAMES SLAGLE
JEAN SAMMET
JEFFREY ULLMAN
JOHN GASCHNIG
JOHN MCCARTHY
JOHN NEWCOMER
JOSEPH WEIZENBAUM
JUDEA PEARL
KARL PINGLE
KEITH PRICE
KEN RALSTON
KING SUNG FU
LAURENT SIKLOSSY
LINDA MASINTER
LES EARNEST
MADLINE BATES
MARY NEWBORN
AIXF

<$BE[PAST]>: WAS
WERE.
<$BE[THEIR]>: <$BE>
<$DO$: <$SHEARSAY> HAVE
<$SBE[THEIR]>
<$SHEARSAY>: YOU
THE DATA BANK
THE DATA BASE
HEARSAY
THE SYSTEM
<$SBE[THEIR]$: <$BE> THERE
<$HAVE$: THERE BEEN
<$BE[THEIR]$: <$BE> ANY PIECES
<$BE[THEIR]$: <$ANY> PIECES
DO YOU HAPPEN TO HAVE <$ANY> PIECES
<$SHOW$: <$ANY> PIECES
<$SHOW$: <$ANY> <$PIECES>
<$CHOOSE$: CHESS
GAME PLAYING
<$CHOOSE$: GET
CHOOSE
SELECT
SUBSELECT
RETRIEVE
<$CITE$: <$CITE/S>
REFERENCE
QUOTE
REFER TO
<$HAVE$: <$CITED>
<$CITE/S>: CITES
CITE
<$CITED>: CITED
QUOTED
REFERENCED
REFERRED TO
<$COMMAND$: TRY TO GET <$WHAT>
<$WHAT$: <$WHAT2>
<$WHAT2$: <$CONJUNCTION> <$WHAT1>
<$CONFERENCE$: <$A> <$CONFERENCE>
<$CONFERENCE$: <$CONFERENCE1> <$CONFERENCE2>
<$CONFERENCE2$: <$MEETING/S>
<$CONFERENCE/S>
<$SESSION/S>
<$CONVENTION/S>
<$MEETING/S$: MEETINGS
MEETING
<$CONFERENCE/S$: CONFERENCES
CONFERENCE
<$SESSION/S$: SESSIONS
SESSION
<$CONVENTION/S$: CONVENTIONS
CONVENTION

68
CONTAINED
CONTAINS
CONTAIN
CONTENTS MENU: THE KEYWORDS
WHAT WHICH RELATE TO SUPER MENU
WHAT WHICH MAY USE FOR RETRIEVAL
WHAT BE THE KEYWORDS
GIMME: I WANNA ACQUIRE
LEMMME: ACQUIRE HEARSAY RETRIEVE
WOULD: HEARSAY FOR ME
LIST: FOR ME, TO ACQUIRE TRY TO GET
KEYWORDS: KEY WORD/S
KEY PHRASE/S
RETRIEVAL KEY/S
WHAT WHICH
RELATE TO BE RELATED TO
SUPER MENU: GAME PLAYING LEARNING INFERENC SEMANTIC NETWORKS COMPUTATIONAL LINGUISTICS UNDERSTANDING ADAPTATION INTERACTIVE DESIGN DESIGN AUTOMATIC PROGRAMMING HYPOTHESIS FORMATION DEDUCTIVE RETRIEVAL GEOMETRIC MODELING INTERACTIVE KNOWLEDGE SYSTEMS COGNITIVE SCIENCE COGNITION AUTOMATION DATA STRUCTURES FORMAL SEMANTICS LANGUAGE UNDERSTANDING
MENUS MENU: TOPIC/S
TOPIC MENUS
MENUS
SUBJECT/S
AREA/S
MAY: CAN COULD SHOULD MUST MAY
<$WHAT> : = <$WHAT|WHICH> <$RE> 
WHAT'S <$DATE> : = <$DATE1> 
THE LAST <$NUMBER> <$TIMES> <$DATE1> <$DATE1> <$CONJUNCTION> <$DATE1> <$DATE1> <$THROUGH> <$DATE1> 
<$DATE1> : = <$YEAR> 
The <$MONTHS> OF <$MONTH> <$MONTH> <$YEAR> 
<$NUMBER> : = <$HUNDREDS> <$NUMBER1> <$HUNDREDS> <$NUMBER1> <$HUNDREDS> <$NUMBER1> <$HUNDREDS> <$NUMBER1> 
<$TIMES> + <MONTHS> 
ISSUES VOLUMES YEARS TIMES <$THROUGH> : = TO THROUGH TILL <$YEAR> : = NINETEEN <$NUMBER1> <$MONTH> : = MAY 
JANUARY FEBRUARY MARCH APRIL JUNE JULY AUGUST SEPTEMBER OCTOBER NOVEMBER DECEMBER <$MONTHS> : = <MONTHS> 
MONTH <$DESIRE> + <$WANT> WOULD LIKE DESIRE <$WANT> : = DESIRE SEEK WANT WISH <$DIGITS> : = ONE TWO THREE FOUR FIVE SIX SEVEN EIGHT NINE <$DO> : = DO DOES DID <$DON'T> : = DON'T
<ONE/S>:: ONE
<$I/WANNA> = <$ID> LIKE
<$I> <$DESIREE>
<$LEMME> = LET <$ME>
LET'S
<$MAY> <$I>
<$WOULD> = WOULD
CAN
COULD
<List>:: LIST
PRINT
TRANSMIT
WRITE
<$GIVE>:: <$GIVE1>
GET FOR
TELL
<$ME>:: ME
US
<$GIVE1>:: GET
GIVE
SHOW
<$GRIPE>:: <$BE> <$HEARSAY> <$ALWAYS> <$SLOW>
HAVEN'T YOU FINISHED
WHY <$BE> <$HEARSAY> SO SLOW
DO RESPONSES EVER COME FASTER
HOW <$MAY> <$I> <$IMPROVEHS>
DO ALL QUERIES TAKE THIS LONG
HOW LONG DOES IT TAKE
WHEN WILL <$HEARSAY> HAVE THE ANSWER
DOES IT ALWAYS TAKE THIS LONG TO ANSWER <$ME>
WHAT <$MAY> <$I> DO TO <$IMPROVEHS>
<$SLOW>:: <$BE> <$HEARSAY> <$ALWAYS> <$SLOW>
SLOW
THIS SLOW
<$IMPROVEHS>:: HELP
SPEED <$HEARSAY> UP
HELP <$HEARSAY>
USE <$HEARSAY> EFFICIENTLY
<$HELP>:: HELP
HOW BIG IS THE DATA BASE
<$WHATSORTS>OF <RETRIEVAL/S> CAN <$HEARSAY> DO
TELL <$ME> WHAT TO DO
<$WHAT WHICH> <$MENU MENU> <$MAY> <$I> <$SEEK>
<$WHATSORTS>OF <RETRIEVAL> <$KEY/S> <$MAY> <$I> <$SEEK>
<$WHATSORTS>OF <$PIECES1> <$BE[PRES]> AVAILABLE
<$WHATSORTS>OF <$MENU MENU> <$BE> STORED
WHAT IS KNOWN <$RT> EVERY <$PIECES1>
WHAT DO <$I> HAVE TO DO
CAN YOU HELP
<$WHATSORTS>OF <$MENU MENU> <$BE THERE>
CAN YOU HELP <$ME>
HELP <$ME>
<$PROVIDE> <$A> <$MENU MENU>
<$WHATIS> <$SAME> <$MENU MENU> <$FROM> <$AI>
<$WHAT WHICH> FACTS ARE STORED
<$WHATIS> THE SIZE OF <$HEARSAY>
WHAT <$MAY> <$I> <$ASK>
WHAT CAN <$HEARSAY> DO
<$WHATSORTSOF>:• <$WHATWHICH> <$SORTS> OF
<$RETRIEVALS>:• RETRIEVAL
<$SEEK>:• REQUEST
CHOOSE
SEEK
<$KEYS>:• KEYS
KEY
<$RE>:• <$REF>
<$WHICH>: <$MENTION>
<$WHATIS>:• WHATS
WHAT <$BE[S]>
<$SOME>:• <$SOME1>
<$SOME1> OF THE
<$SHOWMANYAUTHORS>:• <$SHOWMANY> <$AUTHORS>
<$DO$:• <$AUTHORS>
<$SHOWMANYBE>:• <$SHOWMANYOFTHEM> <$BE>
<$BE1>:• <$SOMEOFTHEM>:
<$SHOWMANYOFTHEM>:• <$SHOWMANY>
<$SHOWMANY> OF <$THATPIECES>
<$SOMEOFTHEM>:• <$THATPIECES>
<$SOME1> OF <$THATPIECES>
<$THATPIECES>:• <$THATPIECES2>
<$THATPIECES2>
<$THATPIECES3>:• <$THATPIECES4>
<$THATPIECES4> <THEM>
<$SHOWMANYPieces>:• <$SHOWMANYPieces2>
<$SHOWMANYPieces2>
<$SHOWMANYSOURCEPIECES>:• <$DO$: <$SOURCEPIECES>
<$DO$: <$SOURCEPIECES>
<$SHOWMANYSOURCEPIECES>:• <$FROM$: <$SOURCE2>
<$FROM$: <$SOURCE2>
<$WHATWHICH>: <$SOURCE>
<$SHOWMANY>: <$SOURCEPIECES>
<$SOURCE>:• <$AI> JOURNAL
ASSOCIATION FOR COMPUTATIONAL LINGUISTICS
COGNITIVE PSYCHOLOGY
ACL
AI TEXT
ARPA SURNOTES
SIGART NEWSLETTER
COMMUNICATIONS OF THE ACM
CACM
COMPUTING <$SURVEYS>
COMPUTING REVIEWS
INFORMATION AND CONTROL
IEEE <$TRANSACTIONS>
IJCAI <$PROCEEDINGS>
IFIP <$PROCEEDING/S>
JOURNAL OF THE ACM
<$HUNDREDS>:• <$NUMBER1> HUNDRED
A HUNDRED
<$NUMBER1>:• <$DIGITS>
<$NUMBER2>
<$TEENS>
<$HBE>:• <$STMs>
<$STMs>:• I AM
I'M
WE'RE
WE ARE
<$STVE>:• I HAVE
WE'VE
WE HAVE
I'D
WE'D
THE
THEIR
ITS
OF THAT PIECE
FROM THAT PIECE
PUBLISHER
REFERENCE
REFERENCES
TITLES
TITLE
WORDS
WORD
PHRASES
PHRASE
LAST
MOST RECENTLY
LEARNING
GRAMMATICAL INFERENCE
NEURAL NETWORKS
ABSTRACTION
DYNAMIC CLUSTERING
CELL ASSEMBLY THEORY
THE NEXT
THE NEXT
THE NEXT
THE FIRST
UP TO
BETWEEN
OF
MORE
THE FIRST
LISTED
LISTED
LISTED
PRINTED
WRITTEN
LISTING
PRINTING
TRANSMITTING
WRITING
MAKE
COPY
WRITE
MAKE
PRODUCE
GENERATE
MAKE A FILE
MAKE A FILE OF THE
MAKE A FILE OF
MAKE A FILE FROM
OF THAT PIECE
PLEASE THANKS
THANK YOU

PLEASE LET ME RESTRICT OURSELVES TO ANY WRITTEN PIECES

I'M ONLY INTERESTED IN THAT PIECE TOPICS

I'M ONLY INTERESTED IN SOME OF THEM

I'M ONLY INTERESTED IN SOME OF THEM CITED IN THAT PIECE

I'M ONLY INTERESTED IN SOME OF THEM CITED IN SOME OF THEM

I'M ONLY INTERESTED IN SOME OF THEM CITED IN SOME OF THEM

I'M ONLY INTERESTED IN WHAT WHICH TOPICS

SHOW MANY OF THEM ALSO MENTION TOPICS

SEEMS ONLY INTERESTED IN WHAT

SEEMS THAT PIECE CITOPICS

SEEMS SOME OF THEM ALSO CITOPICS

SHOW MANY OF THEM ALSO MENTION TOPICS

SHOW MANY OF THEM WRITTEN BY AUTHORS

SHOW MANY WRITTEN BY AUTHORS

WHERE

HOW MANY OF THEM CITE

HOW MANY OF THEM CITED

HOW MANY OF THEM HAVE BEEN CITED

HOW MANY OF THEM

SHOW MANY THEY WRITTEN WHEN DATE

SHOW MANY THEY BY AUTHORS

SHOW MANY THEY CITE IN THAT PIECE

SHOW MANY THEY CITED IN SOME OF THEM

SHOW MANY THEY BEEN CITED IN SOME OF THEM

SHOW MANY THEY RECENTLY IN SOURCE

SHOW MANY OF THAT PIECE APPEARED RECENTLY IN A SOURCE

SHOW MANY OF THEM FROM SOURCE

SHOW MANY OF THEM FROM ANY SOURCE PIECES

SHOW MANY OF THEM FROM ANY SOURCE PIECES WHEN DATE

SHOW MANY THEY WHERE THEY WRITTEN SHOW MANY THEY WHERE THEY WRITTEN

SHOW MANY THEY BY

SHOW WRITTEN 2 BY

RECENTLY

RECENTLY LATELY IN RECENT TIMES

RECENTLY LATELY IN RECENT TIMES

78
WHAT SORTS OF TOPICS ARE WRITTEN WHEN TOPICS ARE LAST MENTIONED
WHERE RE TOPICS MENTIONED
RE TOPICS MENTIONED SOMEWHERE 2
RE TOPICS MENTIONED SOMEWHERE 2 RECENTLY
HAVE ANY PIECES RE TOPICS BEEN WRITTEN
RE TOPICS
RE TOPICS MENTIONED
HAVE ANY PIECES
SHAVE ANY PIECES APPEARED WHICH MENTION TOPICS
SHOW MANY PIECES2 RE TOPICS THERE
SHOW MANY PIECES2 RE TOPICS SOMEWHERE 2
SHAVE ANY PIECES BEEN CITED SIN ANY PIECES
DO ANY PIECES RE TOPICS EXIST
QUERY CITATION SHOW MANY PIECES CITE AUTHORS
WHERE SOME MENTIONEDPP SOMEWHERE
WHAT WHICH PIECES1 WRITTEN BY AUTHORS BE CITED
RE TOPICS MENTIONED WHICH MENTION ANY PIECES WRITTEN BY AUTHORS
RE THERE ANY PIECES CITED AUTHORS
SHAVE ANY PIECES CITED AUTHORS
SHAVE AUTHORS BEEN CITED SIN ANY PIECES
WHAT WHICH PIECES1 WRITTEN BY AUTHORS CITED
BE THERE ANY PIECES WHEN ANY PIECES WRITTEN
SHAVE ANY PIECESEXIST
QUERY CITATION SHOW MANY PIECES CITE AUTHORS
WHERE SOME MENTIONEDPP SOMEWHERE
WHAT WHICH PIECES1 WRITTEN BY AUTHORS BE CITED
RE TOPICS MENTIONED WHICH MENTION ANY PIECES WRITTEN BY AUTHORS
RE THERE ANY PIECES CITED AUTHORS
SHAVE ANY PIECES CITED AUTHORS
SHAVE AUTHORS BEEN CITED SIN ANY PIECES
WHAT WHICH PIECES1 WRITTEN BY AUTHORS CITED
BE THERE ANY PIECES WHEN ANY PIECES WRITTEN
SHAVE ANY PIECES CITED AUTHORS
DO ANY PIECES CONFERENCE PUBLISH PROCEEDINGS
QUERY SOURCE AUTHOR SHOW MANY SOURCE PIECES CONTAINED ANY PIECES WRITTEN BY AUTHORS
SHOW MANY SOURCE PIECES CONTAIN WINograd'S ARTICLE
DID AUTHORS PRESENT ANYNODATE PIECES AT CONFERENCE3
DID AUTHORS PRESENT ANYNODATE PIECES AT CONFERENCE3 WHEN DATE
QUERY SOURCE CITATION SHOW MANY SOURCE PIECES CITE AUTHORS
QUERY SOURCE DATE DID SOURCE2 PUBLISH SOMETHING WHEN DATE
BE THERE ANY SOURCE PIECES WHEN DATE
QUERY SOURCE REFERENCED IS ANY SOURCE PIECES CITED BY AUTHORS
QUERY SOURCE TOPIC SHOW MANY ANY PIECES FROM SOURCE2 RE TOPICS
DID ANYONE PUBLISH RE TOPICS IN SOURCE2
SHOW MANY SOURCE PIECES MENTION TOPICS
RE TOPICS MENTIONED IN SOURCE2
QUERY TITLE SOURCE SHOW BE THE TITLE SOURCE FROM SOURCE2
PROVIDE TITLE SOURCE FROM SOURCE2
QUERY TOPIC DATE SHOW MANY PIECES1 WHEN DATE MENTIONED TOPICS
RELATES RELATE
REQUEST SCOMMAND
NEGSTATEMENT QUERY STATEMENT
STATEMENT GIMME WHAT TELL ME RE TOPICS
$SELECTION$ > $WHAT/S$ $SOME$ $MENU/MENU$ > $FROM$ $SUPER/MENU$
$BE$ INTERESTED IN $SUPER/MENU$
$WHAT/WHICH$ $MENU/MENU$ $BE$ RELATED TO $WHAT/MENU$
$BE$ ONLY INTERESTED IN $PIECES/S$ $RE$ $WHAT/MENU$
THE $MENU/MENU$ $IM$ INTERESTED IN $BE$ $WHAT/MENU$
$IM$ INTERESTED IN $WHAT/MENU$
$CHOICE$ $FROM$ $WHAT/MENU$
$WHAT/MENU$ $SUPER/MENU$
$ANY$ $RE$ $SUPER/MENU$
$WHAT/MENU$ $CONJUNCTION$ $WHAT/MENU$
$SEMANTIC/NETS$ + $UNDERSTANDING$
SEMANTIC NETWORKS
A SEMANTIC NETWORK SEMANTIC NETS
$UNDERSTANDING$ + HEARSAY
LANGUAGE UNDERSTANDING
ENGLISH
NATURAL LANGUAGE UNDERSTANDING
SPEECH UNDERSTANDING
SYNTACTIC
$SENTENCE$ + $CONTENTS/MENU$
$GET/INFO$
$HELP$
$LIST/THEM$
$MAKE/FILE$
$NO$
$PRUNE/LIST$
$REQUEST$
$SELECTION$
$YES$
$STOP/LISTING$
$YES$ + YES
OK
SURE
$STOP/LISTING$ = $IM$ $FINISHED$
NO MORE
$YES$ $FINISHED$
$STOP$ $LISTING$
$STOP$ THE $LISTING$
$SENTENCE$ = $SENTENCE$
$POLITENESS$ $SENTENCE$
$SENTENCE$ $POLITENESS$
$THAT/PIECES$ = $THAT$ $PIECES/S$ $THEM$
$SORTS$ = $SORT/S$
$KIND/S$
$TYPE/S$
$VARIETY/S$
$SORT/S$ + SORTS
SORT
$KIND/S$ + KINDS
KIND
$TYPE/S$ + TYPES
$VARIETY/S$ + VARIETY
$TRANSACTION/S$ + $TRANSACTIONS$
$TRANSACTION$
<STOP> - STOP
CEASE
TERMINATE
KILL
FINISH
QUIT
< THAT > - THIS
THAT
THESE
THOSE
< THESE > - IT
< THAT >
THEY
EACH
< THAT PIECE > - < THAT > < PIECES >
< THAT > < ONE / S >
< THEM > - < THAT >
THEM
< TIME / SPACE > - TIME
SPACE
TIME < CONJUNCTION > SPACE
SPACE < CONJUNCTION > TIME
< TOPIC > - < SAI >
PROBLEM SOLVING
GIPS
< CHESS >
< LEARNING >
INFERENCE
< SEMANTIC NETS >
CYBERNETICS
COMPUTATIONAL LINGUISTICS
PSYCHOLOGY
CONTROL
ADAPTATION
INTERACTIVE DESIGN
DESIGN
AUTOMATIC PROGRAMMING
HYPOTHESIS FORMATION
DEDUCTIVE RETRIEVAL
GEOMETRIC MODELING
INTERACTIVE KNOWLEDGE SYSTEMS
KNOWLEDGE SYSTEMS
COGNITIVE SCIENCE
COGNITION
AUTOMATION
DATA STRUCTURES
FORMAL SEMANTICS
A TASK ORIENTED DIALOGUE
THE TECH-II CHESS PROGRAM
SYNTHESIS OF LINE DRAWINGS
TELEOLOGICAL REASONING
TEMPORAL SCENE ANALYSIS
TEXTURE ANALYSIS
A THAUMATURGIST
SHAPE TOPOLOGY
THREE DIMENSIONAL MODELS
A TUTOR OR TUTORING ON TV
THE WEAK LOGIC OF PROGRAMS
DYNAMIC BINDING
DYNAMIC PROGRAMMING
ELECTRONIC CIRCUITS
ELECTRONICS
THE ENVIRONMENT
EXPERT SYSTEMS
EXPLANATION CAPABILITIES
FABLES OR FAIRY TALES
FEATURE-DRIVEN SYSTEMS
THE FEDERAL JUDICIAL SYSTEM
FIRST ORDER LOGIC
FRAMES
FRAMES AND THE ENVIRONMENT
FUZZY KNOWLEDGE
FUZZY PROBLEM SOLVING
A GAME MODEL
GENERAL PURPOSE MODELS
GENERATION OF NATURAL LANGUAGE
GO OR GO-MOKU
GOAL SEEKING COMPONENTS
GRAPH INTERPRETABLE GAMES
HETEROSTATIC THEORY
HEURISTIC PROGRAMMING
HEURISTIC TECHNIQUES
HUMAN BEHAVIOR
HUMAN MEMORY
HUMAN VISION
IMPROVING PROGRAMS
INDUCTIVE ASSERTIONS
INDUSTRIAL APPLICATION
INEXACT REPRESENTATION
INFERENCES
INFERENTIAL QUESTION ANSWERING
INFORMATION PROCESSING UNIVERSALS
INHERITANCE OF PROPERTIES
INTELLIGENT MACHINES
INTENTIONS
INTERACTIVE PROGRAM SYNTHESIS
INTERPRETIVE SEMANTICS
INTONATION
INvariance FOR PROBLEM SOLVING
INVESTMENT ANALYSIS
ITERATION
KNOWLEDGE BASED SYSTEMS
LAMBDA CALCULUS
LANGUAGE DESIGN
LANGUAGE PRIMITIVES
LARGE DATA BASES
THE BAY AREA CIRCLE
THE BERKELEY DEBATE
THE DREFSUS DEBATE
THE HISTORY OF AI
THE HUNGRY MONKEY
THE INSANE HEURISTIC
AXIOMS FOR GO
COMPUTER BASED CONSULTANT
IMAGE INTENSITY UNDERSTANDING
TROUBLE SHOOTING
LANGUAGE COMPREHENSION
"TIME'SPACE" BOUNDS
PERCEPTRONS
COMPUTER NETWORKS
GRAPH MATCHING
ASSOCIATIVE MEMORY/IS
UNIFORM PROOF PROCEDURES
PLANNER-LIKE LANGUAGES
HILL CLIMBING
"TIME'SPACE": COMPLEXITY
EVALUATION FUNCTIONS
PROGRAM VERIFICATION
FRAME THEORY
PREDICATE CALCULUS
GRAIN OF COMPUTATION
PATTERN MATCHING
RECOGNITION DEVICES
PATTERN RECOGNITION
STRUCTURED PATTERN RECOGNITION
PATTERN DIRECTED FUNCTION INVOCATION
RESOLUTION THEOREM PROVING
MEDICAL CONSULTATION
VISUAL COMMUNICATION
A PARTIAL EVALUATOR
THE LANGUAGE PASCAL
PHOTOGRAMMETRY
PICTURE RECOGNITION
VISUAL PLANES IN THE RECOGNITION OF POLYHEDRA
PREFERENCE SEMANTICS
THE GAME OF POKER
PROCEDURAL EVENTS
PRICE'S TUTORIAL
PRODUCTIVITY TECHNOLOGY
A REGION ANALYSIS SUBSYSTEM
REPRESENTATION OF REAL-WORLD KNOWLEDGE IN RELATIONAL PRODUCTION SYSTEMS
ROBOTICS COOPERATION AND RESOURCE LIMITED PROCESSES
USING S-L-GRAPHS
RULE ACQUISITION CAPABILITIES
SCENE SEGMENTATION
SERIAL PATTERN ACQUISITION
THE SIX SEVEN EIGHT NINE GAME
SNARING DRAGONS
SENTENCE MEANING IN CONTEXT
SOFTWARE INTERRUPTS
SEVERAL GOALS SIMULTANEOUSLY
SHAPE GRAMMARS
SIMULTANEOUS ACTIONS
STATE DESCRIPTION MODELS
STOCHASTIC MODELING
A STEREO PAIR OF VIEWS
STORAGE REDUCTION
SYNTACTIC METHODS
SYNCHRONIZATION OF CONCURRENT PROCESSES
AI LECTURES
THE COMPUTERS AND THOUGHT AWARD
"MEMORY" = MEMORY
MEMORIES
"WHAT2" = "ANYPIECES" "RETOPICS"
WHAT1: - RE!TOPICS.
WHAT2
WHERE1: - AT WORKPLACE
       IN WORKPLACE2
       WITH SUMEX
WORKPLACE: - CMU
           GM RESEARCH LABS
           THE INSTITUTE FOR SEMANTIC AND COGNITIVE STUDIES
           MASSACHUSETTS
           NRL
           NIH
           ROCHESTER
           RUTGERS
           SMC
           SRI
           STANFORD
           SUSSEX
           WATSON RESEARCH
           ILLINOIS
           HAMBURG
           EDINBURGH
WORKPLACE2: - THE SUNSHINE STATE
           THE US
           THE USSR
Appendix III-C-2. AI Retrieval Language Grammar: AIX15

<SENT> = [ <SS> ]
<SS> = <ANY.PAPERS> <ABOUT TOPIC>
<SS> = <ANY.JOURNALS> <ABOUT TOPIC>
<SS> = <ANY.PAPERS> IN <JOURNAL>
<SS> = <ANY.PAPERS> SINCE <DATE>
<SS> = <ANY.PAPERS> THAT MENTION THE <DATES OF THE CONFERENCE>
<SS> = <ANY.PAPERS> WHICH <CITE AUTHOR>
<SS> = <ANY.PAPERS> <ABOUT TOPIC>
<SS> = <ANY.JOURNALS> <ABOUT TOPIC> BUT NOT <TOPICS>
<SS> = <ANY.PAPERS> <ABOUT TOPIC> <ALSO ABOUT TOPIC>
<SS> = <ANY.PAPERS> FROM <A CONFERENCE>
<SS> = <ANY.PAPERS> FROM <JOURNAL>
<SS> = <ANY.PAPERS> FROM <THE CONFERENCE> IN THE MONTH OF <DATE>
<SS> = <AUTHORS> CITED BY <ANY.PAPERS>
<SS> = <TOPICS> <MENTIONED> ANYWHERE IN <A.PAPERS>
<SS> = <TOPICS> <MENTIONED> IN <A.PAPER>
<SS> = <TOPICS> <MENTIONED> IN <JOURNAL>
<SS> = ANY <BY AUTHOR>
<SS> = YOU <ALWAYS> <THIS SLOW>
<SS> = <THE CONFERENCE> <PUBLISH> <JOURNAL>
<SS> = <ANY.PAPERS> <ABOUT TOPIC> <ALSO MENTION TOPIC>
<SS> = <ANY.PAPERS> <ABOUT TOPIC> <MENTION TOPIC>
<SS> = <ANY.PAPERS> <ABOUT TOPIC> EXIST
<SS> = <ANY.PAPERS> <CITE AUTHOR>
<SS> = <ANY.PAPERS> <MENTION TOPIC> BUT NOT <TOPICS>
<SS> = <ANY.PAPERS> <THIS YEAR> <CITE AUTHOR>
<SS> = <AUTHORS> PRESENT <A.PAPER> AT <THE CONFERENCE>
<SS> = <AUTHORS> PRESENT <A.PAPER> AT <THE CONFERENCE> IN <DATE>
<SS> = <AUTHORS> PRESENT <PAPERS> AT <THE CONFERENCE>
<SS> = <AUTHORS> PUBLISH <A.PAPER>
<SS> = <AUTHORS> WRITE <A.PAPER>
<SS> = <AUTHORS> WRITE <A.PAPER> <LATELY>
<SS> = <AUTHORS> WRITE <A.PAPER> <THIS YEAR>
<SS> = <THE AUTHORS> <MENTION TOPIC>
<SS> = <THE JOURNAL> PUBLISH ANYTHING IN <DATE> OR <DATE>
<SS> = <THE PAPER> <CITE AUTHOR>
<SS> = <THE JOURNAL> ALL QUERIES TAKE THIS LONG
<SS> = ANYONE PUBLISH <ABOUT TOPIC> IN <THE JOURNAL>
<SS> = RESPONSES EVER COME FASTER
<SS> = THEY WORK AT <WORKPLACE>
<SS> = <THE PAPER> <MENTION TOPIC>
<SS> = <THE PAPER> REFERENCE <A.JOURNAL>
<SS> = <TOPICS> <MENTIONED> ANYWHERE
<SS> = IT <ALWAYS> TAKE THIS LONG TO ANSWER ME
<SS> = <ANY.PAPERS> <ABOUT TOPIC>
<SS> = <A.JOURNAL> REFERENCED <BY AUTHOR>
<SS> = <A.PAPER> AFTER <DATE>
<SS> = <ANY.PAPERS> <ABOUT TOPIC>
<GET.ME> <ANY.PAPERS> <ABOUT.TOPIC> BUT NOT <TOPICS>
<GET.ME> <ANY.PAPERS> <ABOUT.TOPICS> FROM <DATE> TILL <DATE>
<GET.ME> <ANY.PAPERS> <BY.AUTHOR>
<GET.ME> <PAPERS> PRINTED IN <TIME.PERIOD>
<GET.ME> <QUANTITY> <PAPERS> <ABOUT.TOPIC>
<GET.ME> <QUANTITY> MORE PLEASE
<GET.ME> EVERYTHING <ABOUT.TOPIC>
<GET.ME> SOME REVIEWS <ABOUT.TOPIC>
<GET.ME> SOMETHING <ABOUT.TOPIC>
<GET.ME> SOMETHING FROM <JOURNAL> <ABOUT.TOPIC>
<GET.ME> THE <AND.OR.AUTHOR.DATE.TITLE> FROM <THE.JOURNAL>
<GET.ME> THE <AND.OR.AUTHOR.DATE.TITLE> OF <THE.PAPER>
<GET.ME> THE <AND.OR.AUTHOR.DATE.TITLE> OF EACH
<GET.ME> THE MENUS
<SHOW.MANY.PAPERS> <ALSO.MENTION.TOPIC>
<SHOW.MANY.PAPERS> <MENTION.TOPIC>
<SHOW.MANY.PAPERS> <THIS.YEAR> <MENTION.TOPIC>
<SHOW.MANY.PAPERS> <WERE> <BY.AUTHOR>
<SHOW.MANY.PAPERS> <WERE> <BY.AUTHOR> AND NOT <AUTHORS>
<SHOW.MANY.PAPERS> <WERE> <WRITTEN.PUBLISHED> FROM <DATE> TO <DATE>
<SHOW.MANY.PAPERS> FROM <DATE> THROUGH <DATE> <MENTIONED.TOPIC>
<SHOW.MANY.PAPERS> HAVE <AUTHORS> <WRITTEN.PUBLISHED> SINCE <DATE>
<SHOW.MANY> <JOURNALS> <MENTION.TOPIC>
<SHOW.MANY> REFERENCES <ARE> GIVEN
<IS.THERE> A CONFERENCE IN <GEOPLACE>
<IS.THERE> A JOURNAL FROM <DATE> OR <DATE>
<IS.THERE> A PAPER <ABOUT.TOPIC>
<IS.THERE> ANYTHING NEW <ABOUT.TOPIC>
<IS> ANY AUTHORS BUT NOT <AUTHORS> CITED IN <ANY.PAPERS>
<IS> ANY AUTHORS CITED BY <ANY.PAPERS> ISSUED IN <TIME.PERIOD>
<IS> ANY AUTHORS CITED BY THESE PAPERS
<IS> ANY AUTHORS CITED IN <ANY.PAPERS>
<IS> ANY AUTHORS CITED IN <THE.PAPER>
<IS> ANY TOPICS <MENTIONED>
<IS> ANY TOPICS <MENTIONED> LATELY
<IS> ANY TOPICS <MENTIONED> ANYWHERE
<IS> ANY TOPICS <MENTIONED> IN A PAPER
<IS> ANY TOPICS <MENTIONED> SOMEWHERE
<IS> ANY TOPICS <MENTIONED> SOMEWHERE LATELY
<IS> IT <WRITTEN.PUBLISHED> BY THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS
<IS> IT <WRITTEN.PUBLISHED> BY <THE.JOURNAL>
<IS> THAT <ABOUT.TOPIC>
<SKILL>
<QUANTITY>
<QUANTITY> PLEASE
<WE'RE.INTERESTED.IN> <JOURNALS> AFTER <DATE>
<WE'RE.INTERESTED.IN> <PAPERS> <ABOUT.TOPIC>
<WE'RE.INTERESTED.IN> <PAPERS> <WRITTEN.PUBLISHED> IN <TIME.PERIOD>
<WE'RE.INTERESTED.IN> <PAPERS> ISSUED SINCE <DATE>
<WE'RE.INTERESTED.IN> <PAPERS> SINCE <DATE>
<WE'RE.INTERESTED.IN> TOPICS
<WE'RE> ANY PAPERS <BY.AUTHOR>
<WE'RE> ANY PAPERS <WRITTEN.PUBLISHED> <ABOUT.TOPIC>
<WE'RE> ANY PAPERS <WRITTEN.PUBLISHED> IN <GEOPLACE> OR IN <GEOPLACE>
<WE'RE> ANY <WRITTEN.PUBLISHED> AFTER <DATE>
<WHAT.ABOUT> <AUTHORS>
<WHAT.ARE> SOME OF THE AREAS OF <TOPICS>
<WHAT.ARE> THE <AND.OR.AUTHOR.DATE.TITLE> OF THE RECENT <JOURNAL>
AIX15

<$WHAT.ARE> THE KEY PHRASES
<$WHAT.ARE> THEIR AFFILIATIONS
<$WHAT.HAS> <$AUTHORS/> <$WRITTEN/PUBLISHED> <$LATELY>
<$WHAT.IS> <$AUTHOR>’S AFFILIATION
<$WHAT.IS> KNOWN ABOUT EVERY ARTICLE
<$WHAT.IS> THE <$AUTHORS> <$DATE/TITLE> OF <$PAPERS>
<$WHAT.IS> THE <$AUTHORS> <$DATE/TITLE> OF <$THE.PAPER>
<$WHAT.IS> THE <$AUTHORS> <$DATE/TITLE> OF THAT PIECE
<$WHAT.IS> THE SIZE OF THE DATA BANK
<$WHEN.WAS> <$HUMAN.PROBLEM.SOLVING> <$WRITTEN/PUBLISHED>
<$WHEN.WAS> <$THE.PAPER> <$WRITTEN/PUBLISHED>
<$WHEN.WAS> <$TOPICS> <$MENTIONED>
<$WHEN.WAS> IT <$WRITTEN/PUBLISHED>
<$WHEN.WAS> THE LAST PAPER <$BY.AUTHOR> <$WRITTEN/PUBLISHED>
<$WHERE.IS> <$TOPICS> <$MENTIONED>
<$WHICH.AUTHORS> WORK AT <$PLACE> OR AT <$PLACE>
<$WHICH.AUTHORS> WORK AT <$WORKPLACE> OR AT <$WORKPLACE>
<$WHICH.AUTHORS> WORK WITH <$WORKPLACE> OR AT <$PLACE>
<$WHICH.OF.THIS> <$TOPIC> <$MENTION.TOPIC>
<$WHICH.OF.THIS> <$TOPIC> <$MENTION.TOPIC>
<$WHICH.OF.THIS> <$TOPIC> <$WERE> <$WRITTEN/PUBLISHED> <$LATELY>
<$WHICH.OF.THIS> <$ARE> <$BY.AUTHOR>
<$WHICH.OF.THIS> <$BY.AUTHOR> <$ARE> REFERENCED
<$WHICH.OF.THIS> <$CITE.AUTHOR>
<$WHICH.OF.THIS> <$CONTAINED> <$THE.PAPER> <$BY.AUTHOR>
<$WHICH.OF.THIS> <$CONTAINED> <$TOPICS>
<$WHICH.OF.THIS> <$MENTION.TOPIC>
<$WHICH.OF.THIS> <$MENTIONED.TOPIC>
<$WHICH.OF.THIS> <$WERE> <$BY.AUTHOR>
<$WHICH.OF.THIS> <$WERE> <$BY.AUTHOR> SINCE LAST YEAR
<$WHICH.OF.THIS> <$WERE> <$WRITTEN/PUBLISHED> AT <$WORKPLACE> OR AT <$WORKPLACE>
<$WHICH.OF.THIS> APPEARED <$LATELY> IN <$THE.JOURNAL>
<$WHICH.OF.THIS> CITES <$AUTHORS>
<$WHICH.OF.THIS> MENTIONS <$TOPICS>
<$WHICH.OF.THIS> REFER TO THESE
<$WHICH.OF.THIS> WAS <$BY.AUTHOR>
CAN I HAVE <$THESE>PAPERS> LISTED
CAN YOU HELP ME

CHOOSE AMONG <$JOURNALS> BEFORE <$DATE>
DURING WHAT MONTHS <$WERE> THEY <$WRITTEN/PUBLISHED>
GENERATE A COPY OF THOSE
HAS <$AUTHORS> <$WRITTEN/PUBLISHED> <$ANY>PAPERS> <$THIS.YEAR>
HAS <$AUTHORS> <$WRITTEN/PUBLISHED> <$ANY>PAPERS> <$LATELY>
HAS <$AUTHORS> BEEN REFERENCED IN <$ANY>PAPERS>
HASN’T <$A>PAPER> <$ABOUT.TOPIC> BEEN RELEASED
HASN’T <$TOPICS> BEEN CONSIDERED IN <$JOURNAL>
HAVE <$ANY>PAPERS> <$BY.AUTHOR> APPEARED
HAVE <$ANY>PAPERS> APPEARED <$ABOUT.TOPIC>
HAVE <$AUTHORS> <$WRITTEN/PUBLISHED> <$THIS.YEAR>
HAVEN’T YOU FINISHED
HELP
HOW BIG IS THE DATA BASE
HOW CAN I USE THE SYSTEM EFFICIENTLY
HOW LONG <$DOES.IT.TAKE>
I’D LIKE TO KNOW THE <$AUTHORS> <$DATE/TITLE> OF <$THE.PAPER>
LIST <$QUANTITY> HUNDRED
LIST BETWEEN <$QUANTITY> AND <$QUANTITY> OF THEM
LIST <$PAPERS> <$BY.AUTHOR>

90
NO MORE PLEASE
NO THANKS
OK
PLEASE HELP ME
PLEASE LIST <$THE AUTHORS>
PLEASE MAKE ME A FILE OF THOSE
PRINT <$QUANTITY>
PRODUCE A COPY OF <$QUANTITY> <$PAPERS>
SELECT FROM <$PAPERS> <$ABOUT TOPIC>
SHOW ME <$QUANTITY>
SHOW ME ITS <$AND/OR!AUTHOR!DATE!TITLE>
SUBSELECT FROM <$TOPICS>
SURE THANKS
TELL ME <$WHAT TO DO>
TELL ME THE <$AND/OR!AUTHOR!DATE!TITLE> OF <$QUANTITY>
THANK YOU <$EVERYONE> DONE
TRANSMIT <$QUANTITY>
WHAT <$ABOUT TOPIC>
WHAT <$CAN I DO> TO SPEED UP YOU UP
WHAT <$DO I HAVE TO DO>
WHAT <$IS> ITS <$AND/OR!AUTHOR!DATE!TITLE>
WHAT <$JOURNALS> DURING <$DATE> AND <$DATE> <$MENTION TOPIC>
WHAT <$PAPERS> <$MENTION TOPIC>
WHAT <$SORT OF SUMMARY> IS AVAILABLE
WHAT ADDRESS IS GIVEN FOR <$THE AUTHORS>
WHAT ADDRESSES ARE GIVEN FOR <$THE AUTHORS>
WHAT CAN <$THE SYSTEM> DO
WHAT CONFERENCE WAS AT <$WORKPLACE> OR AT <$GEOPLACE>
WHAT CONFERENCE WAS AT <$WORKPLACE> OR AT <$WORKPLACE>
WHAT FACTS <$ARE> STORED
WHAT KEY WORD RELATES TO <$TOPICS>
WHAT KINDS <$ARE> STORED
WHAT KINDS OF SUBJECTS <$ARE> STORED
WHAT MUST I ASK
WHAT SHOULD I ASK
WHAT SHOULD I SAY
WHAT SORTS OF <$TOPICS> <$ARE> <$MENTIONED>
WHAT SUBJECT CAN I REQUEST
WHAT TOPIC MENU CAN I CHOOSE
WHAT TOPICS <$ARE> RELATED TO <$TOPICS>
WHAT TYPES OF <$RETRIEVAL CAN I HEAR SAY DO>
WHEN WILL YOU HAVE THE ANSWER
WHERE <$ARE> <$QUIT>
WHERE <$DO THEY WORK>
WHERE DID <$THE PAPERS> APPEAR
WHICH <$HAS TEXT> <$CONTAINED> <$TOPICS>
WHICH <$AUTHORS> <$MENTION TOPIC>
WHICH CONFERENCE WAS AT <$GEOPLACE> OR AT <$GEOPLACE>
WHICH <$IS> <$QUANTITY>
WHICH NOTES <$ABOUT TOPIC> <$ALSO MENTION TOPIC>
WHICH ONES
WHICH SORT OF <$RETRIEVAL KEYS> CAN I SEEK
WHICH TITLES <$MENTION TOPIC>
WHICH WAS THE LAST ARTICLE <$BY AUTHOR>
WHO
WHO HAS <$WRITTEN!PUBLISHED> <$ABOUT TOPIC>
WHO WAS QUOTED IN <$THE PAPER>
WHO WAS THE AUTHOR
WHO WERE <$THE AUTHORS> OF <$THE PAPER>
WHO WROTE <$PAPERS> <$ABOUT TOPIC> <$THIS YEAR>
WHO WROTE IT
WHY IS THE SYSTEM <$THIS SLOW>
WOULD YOU LIST <$QUANTITY>
WRITE A FILE OF THOSE
YES PLEASE
<$RETRIEVAL CAN HEARSAY DO>- <$RETRIEVAL CAN HEARSAY DO
<$DOES IT TAKE>- DOES IT TAKE
<$DATES OF THE CONFERENCE>- DATES OF <$THE CONFERENCE /S>
<$WHAT TO DO>- WHAT TO DO
<$CAN I DO>- CAN I DO
<$THE SYSTEM DO>- THE SYSTEM DO
<$DO I HAVE TO DO>- DO I HAVE TO DO
<$THE AUTHORS>- THE AUTHORS
ANY AUTHORS

<$KILL>- <CEASE PRINTING>
PLEASE <CEASE PRINTING>
<CEASE PRINTING> PLEASE
<CEASE PRINTING>- <CEASE> <PRINTING>
<CEASE>- <CEASE>
STOP
TERMINATE
FINISH
QUIT
KILL THE
<PRINTING>- <PRINTING
LISTING
TRANSMITTING
<$SHOW.MANY>- HOW MANY
<$SHOW.MANY.PAPERS>- HOW MANY <$PAPERS>
HOW MANY OF THESE
<$ARE.THERE>- ARE WERE THERE
<$DOIDIO>- YOU HAVE
<$DOIDIO>- YOU HAPPEN TO HAVE
<$AREWERE>- <$ARE>
<$WERE>
<$WERE>- WERE
WEREN'T
WERE NOT
<$DOIDIO>- DO
DON'T
DID
DIDN'T
<$DOESDIDN'T>- DOES
 DOES

<$GETME>- <$GETGIVE>- ME
<$GETGIVE>
TRY TO GET
TRY TO GET ME
COULD YOU RETRIEVE
<$HAVE>- <$DEMANDWANT>
<$HAVE>- <$DEMANDWANT> TO <$SEEGET>
<$IWED>- LIKE TO <$SEEGET>

<$GETGIVE>- GET
GIVE
<$HAVE>- I
WE
<$DEMANDWANT>- DEMAND
DEMAND
 WISH
WANT
<$IWED>- I'D
WE'D
<$SEEGET>- SEE
GET

<$DON'TGETME>- DON'T <$GETME>

<$WHERE.IS>- WHERE IS
<$IS>- IS
ISN'T
WAS
WASN'T
<$ARE>- ARE
ARE NOT
AREN'T
<$IS.THERE>- <$IS> THERE
<$HEHIS>- HER
HIS
<$WHAT.HAVE>- WHAT <HASHAVE>
<$HASHAVE>- HAS
HAVE
<$WHEN.WERE>- WHEN WAS
WHEN WERE
DO THEY WORK? DO THEY WORK
DO THEY? DO DO THEY DO THEY
 DOES HE SHE?
HE SHE

WE'RE INTERESTED IN THE AREA WE'RE INTERESTED IN
ONLY WE'RE INTERESTED IN
THE ONLY AREA WE'RE INTERESTED IN

LET'S RESTRICT OURSELVES TO
WE'RE INTERESTED IN
WE'VE BEEN WE HAVE BEEN
I'M I AM
INTERESTED IN INTERESTED IN
LET'S LET US
RESTRICT RESTRICT
CONFINE LIMIT
OURSELVES OURSELVES
OUR ATTENTION MYSELF

WHAT ABOUT WHAT ABOUT
WHAT ARE WHAT ARE
WHAT IS WHAT IS
WHICH OF THESE WHICH OF THESE
WHICH WHICH!
PAPERS PAPERS
JOURNAL JOURNAL
THEM THEM

THESE THEM

95
<$QUANTITY> » THE <LAST> <$NUMBER1-99>
   THE <LATEST> <$NUMBER1-99>
   THE <LATEST>
   <$NUMBER1-99>
   UP TO <$NUMBER1-99>

<LAST> » LAST
   NEXT
   FIRST
< LATEST> » EARLIEST
   LATEST
   NEWEST
   OLDEST
   MOST RECENT

< NUMBER> » <$NUMBER1-99>
  < HUNDREDS> <$NUMBER1-99>
  < HUNDREDS> » < DIGITS> HUNDRED
  A HUNDRED
<$NUMBER1-99> » < NUMBER2>
  < TEENS>
  < DIGITS>
< NUMBER2> » < TENS>
  < TENS> < DIGITS>
< TEENS> » NINETEEN
  TEN
  ELEVEN
  TWELVE
  THIRTEEN
  FOURTEEN
  FIFTEEN
  SIXTEEN
  SEVENTEEN
  EIGHTEEN
< TENS> » TWENTY
  THIRTY
  FORTY
  FIFTY
  SIXTY
  SEVENTY
  EIGHTY
  NINETY
< DIGITS> » ONE
  TWO
  THREE
  FOUR
  FIVE
  SIX
  SEVEN
  EIGHT
  NINE
<table>
<thead>
<tr>
<th>Authors</th>
<th>Dates</th>
<th>Publishers</th>
<th>Titles</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;AUTHOR&gt;</td>
<td>DATE</td>
<td>PUBLISHER</td>
<td>TITLE</td>
</tr>
<tr>
<td>&lt;AUTHOR&gt;</td>
<td>DATE</td>
<td>PUBLISHER</td>
<td>TITLE</td>
</tr>
<tr>
<td>&lt;AUTHOR&gt;</td>
<td>DATE</td>
<td>PUBLISHER</td>
<td>TITLE</td>
</tr>
</tbody>
</table>

<AND> <OR> 3> AND
<OR>
THAT
THE
<THESE|THOSE> = THESE
THOSE
<THESE|THOSE|THE> = THE
THESE
THOSE
<ANY|MANY|SOME> = ANY
MANY
SOME
TRANSACTIONS
VOLUMES
<ADJ> ISSUES
<OF.THE ADJ. CONFERENCE> OF THE <ADJ. JOURNAL
OF THE <ADJ. JOURNAL> <CONFERENCE.>
<$A\text{CONFERENCE}> \cdot \text{AN} <\text{ADJ}\text{CONFERENCE}\text{V}> <\text{CONFERENCE}>

<\text{RECENT}\text{ADJ}\text{CONFERENCE}> \cdot <\text{RECENT}\text{CURRENT}> <\text{ADJ}\text{CONFERENCE}> <\text{CONFERENCE}>

<\text{RECENT}\text{CURRENT}> <\text{CONFERENCE}>

<\text{ADJ}\text{CONFERENCE}> \cdot <\text{ADJ}\text{CONFERENCE}\text{V}>

<\text{ADJ}\text{CONFERENCE}\text{V}*> \cdot <\text{ADJ}\text{CONFERENCE}> <\text{CONFERENCE}>

<\text{IJCAI}>

<\text{ACM}>

<\text{IEEE}>

<\text{IFIP}>

<\text{CONFERENCE}> \cdot <\text{CONFERENCE}>

<\text{CONVENTION}>

<\text{MEETING}>

<\text{SESSION}>

<\text{CONFERENCES}> \cdot <\text{CONFERENCES}>

<\text{CONVENTIONS}>

<\text{MEETINGS}>

<\text{SESSIONS}>

<\text{RECENT}\text{CURRENT}>

<\text{RECENT}>

<\text{CURRENT}>

<\text{NEW}>

<\$\text{ANY}\text{CONFERENCE}\text{S}*> \cdot \text{ANY} <\text{RECENT}\text{ADJ}\text{CONFERENCE}\text{S}>

<\text{RECENT}\text{ADJ}\text{CONFERENCE}\text{S}*> \cdot <\text{RECENT}\text{CURRENT}> <\text{ADJ}\text{CONFERENCE}>

<\text{CONFERENCE}\text{S}*> \cdot <\text{CONFERENCE}>

<\text{CONFERENCES}>

<$\text{THE}\text{CONFERENCE}\text{S}*> \cdot \text{THE} <\text{RECENT}\text{ADJ}\text{CONFERENCE}\text{S}>

<\text{IJCAI}>

<\text{IFIP}>

<\text{IEEE}>

<\text{ACM}>

<\text{CONVEN}>
<AUTHORS> = <AUTHOR>
  <AUTHOR> <AND/OR> <AUTHOR>
  <AND/OR> = AND
  OR

<AUTHOR> = ALLEN COLLINS
  ALLEN NEWELL
  ANN RUBIN
  ANTHONY MARTELLI
  AZRIEL ROSENFELD
  BERNARD MELTZER
  BERT RAPHAEL
  BILL WOODS
  BONNIE NASH-WEBBER
  BRUCE BUCHANAN
  CARI HEWITT
  CHRISTOPHER RIESRECK
  CHUCK RIEGER
  DANNY BOHRNOW
  DAVE RUMELHART
  DAVID MARR
  DAVID MICHIE
  DICK SELTZER
  DONALD NORMAN
  DOUG LENAT
  DREW NICERMOFF
  DREYFUS
  EARL HUNT
  EARL SACHEROTI
  ED FEIGENBAUM
  ED RISEMAN
  ELLIOT SOLOWAY
  ERIK SANDEWALL
  EUGENE CHARNIAK
  FEIGENBAUM
  FEIGENBAUM
  FELDMAN
  GARY HENDRIX
  GEORGE ENNST
  GIPS
  HANS BERLINER
  HARRY BARKOW
  HERB SIMON
  HERBERT BROCK
  HILARY PUTNAM
  HOLLAND
  HUGH NAGEL
  JRY SOBEL
  ISSAC ASIMOV
  JACK MINER
  JACK MOSTOW
  JAMES SLagle
  JEAN SAKAKI
  JEFFREY BILLMAN
  JERRY FELDMAN
  JOHN GASCHNING
  JOHN HOLLAND
  JOHN MCCARTHY
  JOHN NEWCOMER
AUTOMATION
AXIOMATIC SEMANTICS
AXIOMS FOR GO
BACKGAMMON
BELIEF SYSTEMS
BINDINGS
BIOMEDICINE
BRAIN THEORY
BUSINESS PROBLEM SOLVING
CARTOGRAPHY
CASE SYSTEMS
CAUSAL REASONING
CELL ASSEMBLY THEORY
CHECKING PROOFS
CHESS
CHESS PLAYING PROGRAMS
CIRCUIT ANALYSIS
COGNITION
COGNITIVE ROBOTIC SYSTEMS
COGNITIVE SCIENCE
COMMON SENSE
COMMON SENSE THEORY FORMATION
COMPLEX WAVEFORMS
COMPUTATIONAL LINGUISTICS
COMPUTER ART
COMPUTER BASED CONSULTANT
COMPUTER BASED CONSULTATIONS
COMPUTER CONTROLLED MANIPULATORS
COMPUTER GRAPHICS
COMPUTER MUSIC
COMPUTER NETWORKS
COMPUTER VISION
CONCEPTUAL DESCRIPTIONS
CONCEPTUAL INFERENCE
CONCEPTUAL OVERLAYS
CONSTRAINT SATISFACTION
CONSTRUCTING PROGRAMS FROM EXAMPLES
CONSTRUCTION OF PROGRAMS
CONTEXT
CONTINUOUS PROCESSES
CONTROL
COORDINATING SOURCES OF KNOWLEDGE
COPYING LIST STRUCTURES
CURVED OBJECTS
CYBERNETICS
CYCLIC
DATABASES
DATABASES FOR INTERACTIVE DESIGN
DATA STRUCTURES
DECISION THEORY
DEDUCTION
DEDUCTIVE RETRIEVAL
DENOTATIONAL SEMANTICS
DEPTH PERCEPTION
DERIVATION PLANS
DESIGN
DESIGN AUTOMATION
DESIGN IN THE ARTS
DETECTION OF LIGHT SOURCES
DISPLAY TERMINALS
DRAGON
DRIVING A CAR
DYNAMIC BINDING
DYNAMIC CLUSTERING
DYNAMIC PROGRAMMING
ELECTRONIC CIRCUITS
ELECTRONICS
ENGLISH
EVALUATION FUNCTIONS
EXPERT SYSTEMS
EXPLANATION CAPABILITIES
FAIRIES OR FAIRY TALES
FEATURE-DRIVEN SYSTEMS
FIRST ORDER LOGIC
FORMAL SEMANTICS
FRAME THEORY
FRAMES
FRAMES AND THE ENVIRONMENT
FUZZY KNOWLEDGE
FUZZY PROBLEM SOLVING
GAME OF POKER
GAME PLAYING
GENERAL PURPOSE MODELS
GENERATION OF NATURAL LANGUAGE
GEOMETRIC MODELING
GO OR GO-MOKU
GOAL SEEKING COMPONENTS
GRAIN OF COMPUTATION
GRAMMATICAL INference
GRAPH INTERPRETABLE GAMES
GRAPH MATCHING
HEARSAY
HETEROSTATIC THEORY
HEURISTIC PROGRAMMING
HEURISTIC TECHNIQUES
HILL CLIMBING
HUMAN BEHAVIOR
HUMAN MEMORY
HUMAN VISION
HYPOTHESIS FORMATION
IMAGE INTENSITY UNDERSTANDING
IMPROVING PROGRAMS
INDUCTIVE ASSERTIONS
INDUSTRIAL APPLICATION
INEXACT REPRESENTATION
INFERENCe
INFERENCES
INFERENTIAL QUESTION ANSWERING
INFORMATION PROCESSING UNIVERSALS
INHERITANCE OF PROPERTIES
INTELLIGENT MACHINES
INTENTIONS
INTERACTIVE DESIGN
INTERACTIVE KNOWLEDGE SYSTEMS
INTERACTIVE PROGRAM SYNTHESIS
INTERPRETIVE SEMANTICS
INTONATION
INVARINACE FOR PROBLEM SOLVING
INVARINANCES IN THE PERCEPTION OF FACES
INVESTMENT ANALYSIS
ITERATION
KNOWLEDGE BASED SYSTEMS
KNOWLEDGE SYSTEMS
LAMBDA CALCULUS
LANGUAGE COMPREHENSION
LANGUAGE DESIGN
LANGUAGE PARAPHRASE
LANGUAGE PASCAL
LANGUAGE PRIMITIVES
LANGUAGE UNDERSTANDING
LARGE DATA BASES
LEARNING
LINEAR LEXICOMETRY
LOW ORDERS OF RECOGNITION PERFORMANCE
MACHINE INTELLIGENCE IN MEDICAL DIAGNOSIS
MACRO PROCESSING FOR AN ON-LINE NEWSLETTER
MANAGEMENT INFORMATION SYSTEMS
MEANS FOR COMPUTER MOVIES
MEDICAL CONSULTATION
MINIMAL SPANNING FORESTS OR TREES
MOTION IN SCENE DESCRIPTION
NEURAL NETWORKS
NEWSLETTER REPORTERS
NONDETERMINISTIC PROGRAMMING
OBJECT LOCATIONS AND MOVEMENTS IN NATURAL IMAGES
OBJECT MANIPULATING ROBOTS
OPERATIONAL REASONING
OPTIMAL PROBLEM SOLVING SEARCH
OPTIMIZED CODE FOR THE TRANSFER OF COMMENTS
PAPERS BY BILL WOODS
PARALLELISM IN PROBLEM SOLVING
PARTIAL EVALUATOR
PATTERN DIRECTED FUNCTION INVOCATION
PATTERN MATCHING
PATTERN RECOGNITION
PERCEPTRONS
PHOTOGRAMMETRY
PICTURE RECOGNITION
PLANNER-LIKE LANGUAGES
PREDICATE CALCULUS
PREFERENTIAL SEMANTICS
PRICE'S TUTORIAL
PROBLEM SOLVING
PROCEDURAL EVENTS
PRODUCTION SYSTEMS
PRODUCTIVITY TECHNOLOGY
PROGRAM VERIFICATION
PSYCHOLOGY
RECOGNITION DEVICES
REPRESENTING REAL-WORLD KNOWLEDGE IN RELATIONAL PRODUCTION SYSTEMS
RESOLUTION THEOREM PROVING
RESOURCE LIMITED PROCESSES
RETRIEVAL
ROBOTICS COOPERATION
UNDERSTANDING
UNIFORM PROOF PROCEDURES
USING S-L-GRAPHS
VISUAL COMMUNICATION
VISUAL PLANES IN THE RECOGNITION OF POLYHEDRA
Appendix III-C-3. AI Retrieval Language Grammar: AIX05

<SENT> = [ <SS> ]
<SS> = ANY ABSTRACTS REFERRING TO <$TOPICS>
    ARE <$AUTHOR/S> CITED BY ANY OF THOSE
    ARE <$AUTHOR/S> CITED IN ANY RECENT PAPERS
    ARE <$TOPICS> DISCUSSED IN RECENT JOURNALS
    ARE <$TOPICS> MENTIONED ANYWHERE
    ARE <$TOPICS> MENTIONED IN AN ABSTRACT
    ARE ANY ARTICLES ABOUT <$TOPICS>
    ARE ANY ARTICLES BY <$AUTHOR/S>
    ARE ANY BY <$AUTHOR/S>
    ARE ANY NEW BOOKS BY <$AUTHOR/S>
    ARE ANY OF THE PAPERS ON <$TOPICS> ALSO ABOUT <$TOPICS>
    ARE ANY OF THESE BY <$AUTHOR/S>
    ARE ANY OF THESE FROM AN ACM SESSION
    ARE ANY OF THESE FROM THE IFIP SESSIONS IN THE MONTH OF JUNE
    ARE ANY PAPERS ABOUT <$TOPICS>
    ARE ANY RECENT ISSUES ABOUT <$TOPICS> BUT NOT <$TOPICS>
    ARE NOT SOME OF THESE FROM COMPUTING SURVEYS
    ARE THERE ANY ABSTRACTS WHICH REFER TO <$TOPICS>
    ARE THERE ANY ABSTRACTS WHICH REFER TO PAPERS BY <$AUTHOR/S>
    ARE THERE ANY ARTICLES ABOUT <$TOPICS>
    ARE THERE ANY ARTICLES ABOUT <$TOPICS>
    ARE THERE ANY NEW ISSUES CONCERNING <$TOPICS>
    ARE THERE ANY NEW PAPERS ON <$TOPICS>
    ARE THERE ANY PAPERS THAT MENTION <$TOPICS>
    ARE THERE ANY RECENT ARTICLES IN CACM
    ARE THERE ANY RECENT BOOKS ABOUT <$TOPICS>
    ARE THERE SOME PAPERS ON <$TOPICS>
    ARE YOU ALWAYS THIS SLOW
    ARE YOU REGULARLY THIS SLOW
    ARE YOU USUALLY SO SLOW
    AREN'T THERE ANY ABSTRACTS SINCE NINETEEN SEVENTY-FIVE
    CAN I HAVE THESE ABSTRACTS LISTED
    CAN YOU HELP ME
    CLOSE PRINTING
    CHOOSE AMONG VOLUMES BEFORE NINETEEN SIXTY
    COULD YOU RETRIEVE SOMETHING FROM <$INFORMATION AND CONTROL> DISCUSSING <$TOPICS>
    DID <$AUTHOR/S> PRESENT A PAPER AT IJCAI
    DID <$AUTHOR/S> PRESENT A PAPER AT THE IFIP MEETINGS IN SEPTEMBER
    DID <$AUTHOR/S> PRESENT PAPERS AT IFIP
    DID <$AUTHOR/S> PRESENT PAPERS AT IJCAI
    DID <$AUTHOR/S> PUBLISH A PAPER
    DID <$AUTHOR/S> WRITE A BOOK
    DID <$AUTHOR/S> WRITE A BOOK RECENTLY
    DID <$AUTHOR/S> WRITE A PAPER THIS YEAR
    DID ANY <$AI JOURNAL> PAPERS CITE <$AUTHOR/S>
    DID ANY ACL PAPERS CITE <$AUTHOR/S>
    DID ANY NEIL CONVENTIONS PUBLISH PROCEEDINGS
    DID ANY OF THOSE PAPERS CITE <$AUTHOR/S>
    DID ANYONE PUBLISH ABOUT <$TOPICS> IN COMMUNICATIONS OF THE ACM
    DID THE SIGART NEWSLETTER PUBLISH ANYTHING IN OCTOBER OR NOVEMBER
    DIDN'T THAT PAPER QUOTE <$AUTHOR/S>
    DO ALL QUERIES TAKE THIS LONG
    DO ANY ARTICLES ON <$TOPICS> IN ADDITION CONSIDER <$TOPICS>
    DO ANY ARTICLES ON <$TOPICS> MENTION <$TOPICS>
    DO ANY ARTICLES REFER TO <$TOPICS>
HELP
HOW BIG IS THE DATABASE
HOW CAN I USE THE SYSTEM EFFICIENTLY
HOW LONG DOES IT TAKE
HOW MANY ABSTRACTS ARE THERE ON <$TOPICS>
HOW MANY ABSTRACTS REFER TO <$TOPICS>
HOW MANY ARTICLES DISCUSS <$TOPICS>
HOW MANY ARTICLES ON <$TOPICS> ARE THERE
HOW MANY ARTICLES WERE WRITTEN BY <$AUTHOR/S> AND NOT <$AUTHOR/S>
HOW MANY BOOKS DISCUSS <$TOPICS>
HOW MANY BOOKS WERE PRODUCED FROM MARCH TO DECEMBER
HOW MANY BOOKS WERE WRITTEN BY <$AUTHOR/S>
HOW MANY OF THESE ALSO DISCUSS <$TOPICS>
HOW MANY PAPERS ARE ABOUT <$TOPICS>
HOW MANY PAPERS CONSIDER <$TOPICS> SIMULTANEOUSLY
HOW MANY PAPERS DISCUSS <$TOPICS>
HOW MANY PAPERS FROM APRIL THROUGH AUGUST CONCERNED <$TOPICS>
HOW MANY PAPERS HAVE <$AUTHOR/S> WRITTEN SINCE JANUARY
HOW MANY PAPERS REFER TO <$TOPICS>
HOW MANY PAPERS THIS YEAR DISCUSS <$TOPICS>
HOW MANY PAPERS WERE WRITTEN BY <$AUTHOR/S>
HOW MANY RECENT ISSUES CONCERN <$TOPICS>
HOW MANY REFERENCES ARE GIVEN
HOW MANY SUMMARIES DISCUSS <$TOPICS>
I AM INTERESTED IN <$TOPICS>
I AM ONLY INTERESTED IN PAPERS ON <$TOPICS>
I DEMAND ANOTHER ARTICLE AFTER AUGUST NINETEEN THIRTEEN
I'D LIKE TO KNOW THE PUBLISHERS OF THAT STORY
I'D LIKE TO SEE THE MENUS
IS <$AUTHOR/S> BUT NOT <$AUTHOR/S> CITED IN SOME OF THOSE ARTICLES
IS <$AUTHOR/S> CITED BY THOSE ABSTRACTS
IS <$AUTHOR/S> CITED IN ANY OF THESE
IS <$TOPICS> DISCUSSED ANYWHERE
IS <$TOPICS> DISCUSSED IN A RECENT SUMMARY
IS <$TOPICS> MENTIONED
IS <$TOPICS> MENTIONED ANYWHERE
IS <$TOPICS> MENTIONED IN AN ABSTRACT
IS <$TOPICS> REFERRED TO
IS <$TOPICS> REFERRED TO ANYWHERE
IS THAT ABOUT <$TOPICS>
IS THERE A RECENT ARTICLE ABOUT <$TOPICS>
IS THERE A RECENT PAPER ABOUT <$TOPICS>
IS THERE A RECENT PAPER MENTIONING <$TOPICS>
IS THERE AN ARTICLE ABOUT <$TOPICS>
IS THERE AN IFIP CONVENTION ISSUE FROM MAY OR JUNE
IS THERE ANYTHING NEW REGARDING <$TOPICS>
ISN'T <$TOPICS> MENTIONED IN AN ABSTRACT
ISN'T THERE AN ARTICLE ABOUT <$TOPICS>
KILL THE LISTING
LET ME LIMIT MYSELF TO REPORTS ISSUED SINCE NINETEEN FIFTEEN
LET US CONFINE OURSELVES TO JOURNALS AFTER FEBRUARY NINETEEN FIFTY
LET'S RESTRICT OUR ATTENTION TO PAPERS SINCE NINETEEN SEVENTY FOUR
LIST BETWEEN TWELVE AND TWENTY OF THEM
LIST THE ABSTRACTS BY <$AUTHOR/S>
LIST THE NEXT FOURTEEN HUNDRED
NO MORE PLEASE
NO THANKS
OK
PLEASE HELP ME
PLEASE LIST THE AUTHORS
PLEASE MAKE ME A FILE OF THOSE
PLEASE TERMINATE TRANSMITTING
PRINT THE NEXT ONE
PRODUCE A COPY OF THE NEWEST EIGHTY ARTICLES
QUIT LISTING PLEASE
SELECT FROM ARTICLES ON <$TOPICS>
SHOW ME ITS PUBLISHER
SHOW ME THE LATEST ELEVEN
STOP TRANSMITTING PLEASE
SUBSELECT FROM <$TOPICS>
SURE THANKS
TELL ME THE TITLES OF THE EARLIEST TEN
TELL ME WHAT TO DO
THANK YOU I'M DONE
THE AREA I AM INTERESTED IN IS <$TOPICS>
THE AREA I'M INTERESTED IN IS <$TOPICS>
THE FIRST TWO
THE LATEST SIXTEEN PLEASE
TRANSMIT THE NEXT EIGHTEEN
TRY TO GET SURVEYS PRINTED IN THE LAST EIGHTY MONTHS
WAS <$AUTHOR/S> CITED BY ANY REPORTS ISSUED IN THE LAST NINETY YEARS
WAS <$AUTHOR/S> CITED IN THAT SUMMARY
WAS <$TOPICS> MENTIONED SOMEWHERE IN RECENT TIMES
WAS <$TOPICS> WRITTEN UP RECENTLY
WAS IT PUBLISHED BY <$THE-ASSOCIATION-FOR-COMPUTATIONAL-LINGUISTICS>
WAS IT PUBLISHED BY THE JOURNAL OF THE ACM
WAS THERE A CONFERENCE IN THE USSR
WASN'T <$TOPICS> MENTIONED RECENTLY
WASN'T <$TOPICS> REFERRED TO SOMEWHERE
WE DESIRE A PROCEEDING OF THE ACM MEETING REFERENCED BY <$AUTHOR/S>
WE WANT SOME REVIEWS CONCERNING <$TOPICS>
WE WISH TO GET THE LATEST FORTY ARTICLES ON <$TOPICS>
WE'D LIKE TO SEE THE TITLES FROM PROCEEDINGS OF THE ACM CONFERENCE
WE'RE INTERESTED IN <$TOPICS>
WE'RE INTERESTED IN ARTICLES PUBLISHED IN THE LAST THIRTY YEARS
WE'VE BEEN INTERESTED IN <$TOPICS>
WERE ANY OF THESE ARTICLES WRITTEN BY <$AUTHOR/S>
WERE ANY OF THESE PUBLISHED IN THE SUNSHINE STATE OR IN THE U.S.
WERE ANY OF THESE WRITTEN BY <$AUTHOR/S>
WERE ANY PUBLISHED AFTER JUNE NINETEEN SIXTY FIVE
WERE THERE ANY ARTICLES ABOUT <$TOPICS>
WEREN'T SOME ARTICLES PUBLISHED ON <$TOPICS>
WHAT ABOUT <$AUTHOR/S>
WHAT ABOUT <$TOPICS>
WHAT ADDRESS IS GIVEN FOR THE AUTHOR/S
WHAT ADDRESSES ARE GIVEN FOR THE AUTHOR/S
WHAT ARE SOME OF THE AREAS OF <$TOPICS>
WHAT ARE THE KEY PHRASES
WHAT ARE THE TITLES OF THE RECENT ARPA SURNOTES
WHAT ARE THEIR AFFILIATIONS
WHAT BOOKS MENTION <$TOPICS>
WHAT CAN I DO TO SPEED YOU UP
WHAT CAN THE SYSTEM DO
WHAT CONFERENCE WAS AT RUTGERS OR AT SRI
WHAT CONFERENCE WAS AT WATSON RESEARCH OR AT ILLINOIS
WHAT DO I HAVE TO DO
WHAT FACTS ARE STORED
WHAT HAS <$AUTHOR/S> WRITTEN LATELY
WHAT HAS <$AUTHOR/S> WRITTEN RECENTLY
WHAT HAVE <$AUTHOR/S> WRITTEN LATELY
WHAT IS HER AFFILIATION
WHAT IS HIS AFFILIATION
WHAT IS KNOWN ABOUT EVERY ARTICLE
WHAT IS THE SIZE OF THE DATA BANK
WHAT IS THE TITLE OF THAT PAPER
WHAT IS THE TITLE OF THE EARLIEST ONE
WHAT IS THE TITLE OF THE MOST RECENT ONE
WHAT ISSUES DURING JANUARY AND JULY CONCERN <$TOPICS>
WHAT KEY WORD RELATES TO <$TOPICS>
WHAT KEY WORDS SHOULD I USE FOR <$TOPICS>
WHAT KIND OF MENUS ARE THERE
WHAT KINDS OF SUBJECTS ARE STORED
WHAT MUST I ASK
WHAT PAPERS ON <$TOPICS> ARE THERE
WHAT SHOULD I ASK
WHAT SHOULD I SAY
WHAT SORT OF SUMMARY IS AVAILABLE
WHAT SORTS OF <$TOPICS> ARE WRITTEN UP
WHAT SUBJECT CAN I REQUEST
WHAT TOPIC MENU CAN I CHOOSE
WHAT TOPICS ARE RELATED TO <$TOPICS>
WHAT TYPES OF <$RETRIEVAL«CAN HEARSAY> DO
WHAT WAS ITS TITLE
WHAT'S THE PUBLISHER OF THAT PIECE
WHEN WAS <$HUMAN-PROBLEM-SOLVING> WRITTEN
WHEN WAS <$TOPICS> LAST MENTIONED
WHEN WAS <$TOPICS> LAST REFERRED TO
WHEN WAS IT PUBLISHED
WHEN WAS THAT BOOK WRITTEN
WHEN WAS THAT PAPER PUBLISHED
WHEN WAS THE LAST PAPER BY <$AUTHOR/S> PUBLISHED
WHEN WERE <$TOPICS> LAST REFERRED TO
WHEN WILL YOU HAVE THE ANSWER
WHERE ARE <$TOPICS> REFERRED TO
WHERE DID THAT ARTICLE APPEAR
WHERE DO THEY WORK
WHERE DOES HE WORK
WHERE IS <$TOPICS> MENTIONED
WHICH <$AIL-TEXT> CONTAINED <$TOPICS>
WHICH <$COGNITIVE-PsYCHOLOGY> CONTAINED <$TOPICS>
WHICH <$COGNITIVE-PsYCHOLOGY> CONTAINS <$TOPICS>
WHICH ABSTRACTS CONCERN <$TOPICS>
WHICH ABSTRACTS REFER TO <$TOPICS>
WHICH ARTICLES CONCERN <$TOPICS>
WHICH ARTICLES HAVE CONCERNED <$TOPICS>
WHICH ARTICLES ON <$TOPICS> ALSO CONCERN <$TOPICS>
WHICH ARTICLES REFER TO THESE
WHICH AUTHORS WORK AT HAMBURG OR AT EDINBURGH
WHICH AUTHORS WORK AT NIH OR AT STANFORD
WHICH AUTHORS WORK WITH SUMEX OR AT SUSSEX
WHICH BOOKS ON <$TOPICS> WERE PUBLISHED RECENTLY
WHICH BOOKS WERE WRITTEN BY <$AUTHOR/S> SINCE LAST YEAR
WHICH COMPUTING SURVEY ARTICLES RELATE TO <$TOPICS>
WHICH COMPUTING SURVEYS CONTAINED THE ARTICLE BY <$AUTHOR/S>
WHICH CONFERENCES WERE AT MASSACHUSETTS OR AT ROCHESTER
WHICH IS THE OLDEST
WHICH NOTES ON <$TOPICS> ALSO DISCUSS <$TOPICS>
WHICH OF THEM DISCUSSES <$TOPICS>
WHICH OF THESE APPEARED RECENTLY IN THE IEEE TRANSACTIONS
WHICH OF THESE ARE BY <$AUTHOR/S>
WHICH OF THESE CITES <$AUTHOR/S>
WHICH OF THESE WAS WRITTEN BY <$AUTHOR/S>
WHICH OF THESE WERE WRITTEN BY <$AUTHOR/S>
WHICH ONES
WHICH PAPER MENTIONS <$TOPICS>
WHICH PAPERS ARE ON <$TOPICS>
WHICH PAPERS BY <$AUTHOR/S> ARE REFERENCED
WHICH PAPERS CITE <$AUTHOR/S>
WHICH PAPERS DISCUSS <$TOPICS>
WHICH PAPERS HAVE MENTIONED <$TOPICS>
WHICH PAPERS ON <$TOPICS> ALSO CONCERN <$TOPICS>
WHICH PAPERS ON <$TOPICS> ALSO DISCUSS <$TOPICS>
WHICH PAPERS ON <$TOPICS> ARE ABOUT <$TOPICS>
WHICH PAPERS WERE WRITTEN AT NRL OR AT SMC
WHICH PAPERS WERE WRITTEN BY <$AUTHOR/S>
WHICH RECENT JOURNALS REFER TO <$TOPICS>
WHICH SORT OF <$RETRIEVAL KEYS> CAN I SEEK
WHICH STORIES IN THE SIGART NEWSLETTER HAVE BEEN DISCUSSING <$TOPICS>
WHICH SUMMARIES ON <$TOPICS> CONSIDER <$TOPICS> IN ADDITION
WHICH TECHNICAL PAPERS WERE WRITTEN BY <$AUTHOR/S>
WHICH TITLES CONTAIN THE PHRASE <$TOPICS>
WHICH WAS THE LAST ARTICLE BY <$AUTHOR/S>
WHO
WHO HAS WRITTEN ABOUT <$TOPICS>
WHO WAS QUOTED IN THAT ARTICLE
WHO WAS THE AUTHOR
WHO WERE THE AUTHORS OF THAT BOOK
WHO WROTE IT
WHO WROTE PAPERS ON <$TOPICS> THIS YEAR
WHY IS THE SYSTEM SO SLOW
WOULD YOU LIST UP TO SEVENTEEN
WRITE A FILE OF THOSE
YES PLEASE
<AUTHORS>:: ALLEN COLLINS
  ALLEN NEWELL
  ANN RUBIN
  ANTHONY MARTELLI
  AZRIEL ROSENFIELD
  BERNARD MELTZER
  BERT RAPHAEL
  BILL WOODS
  BONNIE NASH-WEBBER
  BRUCE BUCHANAN
  CARL HEWITT
  CHRISTOPHER RIESBECK
  CHUCK RIEGER
  DANNY BOBROW
  DAVE RUMELHART
  DAVID MARR
  DAVID MICHE
  DICK SALTZER
  DONALD NORMAN
  DOUG LENAT
  DREW MCDERMOTT
  DREYFUS
  EARL HUNT
  EARL SACKRIO
  ED FEIGENBAUM
  ED RISEMAN
  ELLIOT SOLOWAY
  ERIK SANDEWALL
  EUGENE CHARNIAK
  FEIGENBAUM
  FELDMAN
  GARY HENDRIX
  GEORGE ERNST
  GIPS
  HANS BERLINER
  HARRY BARROW
  HERB SIMON
  HERBERT BLOCK
  HILARY PUTNAM
  HOLLAND
  HUGH NAGEL
  IRY SOBOL
  ISSAC ASIMOV
  JACK MINKER
  JACK MOSTOW
  JAMES SLAGLE
  JEAN SAMUEL
  JEFFREY ULLMAN
  JERIY FELDMAN
  JOHN DASCHNIG
  JOHN HOLLAND
  JOHN MCCARTHY
  JOHN NEWCOMER
WINograd's Article

A Common Sense Algorithm
A Game Model
A Losing Move
A Multilevel Organization
A Packet Based Approach to Network Communication
A Partial Evaluator
A Program Synthesizer for Network Protocols
A Programming Apprentice
A Proof Checker for Protocol Termination Expressions
A Radio Interview on Science Fiction
A Region Analysis Subsystem
A Stereo Pair of Views
A Task Oriented Dialogue
A Thaumaturgist
A Theorem Prover Planning for Progress
A Time Domain Analyzer
A Tutor or Tutoring on TV
A TV Reporter
Abstraction
Acquisition of Knowledge
Active Knowledge
Acyclic Isomers
Adaptation
Adaptive Production Systems
Advising Physicians
AI
AI Lectures
Algebraic Reduction
Algol
Algorithmic Aesthetics
All-or-None Solutions
An Adaptive Natural Language System
An Assembly Robot
An Axiomatic System
Analogy in Problem Solving
Analysis of Context
Analysis of Sentences
Artificial Intelligence
Asimilation of New Information
Associative Memories
Associative Memory
Augmented Transition Networks
Automated Deduction
Automatic Coding
Automatic Computation
Automatic Mantra Generation
Automatic Program Synthesis from Example Problems
Automatic Program Writing
Automatic Programming
Automatic Proof of Correctness
Automatic Theorem Proving
DETECTION OF LIGHT SOURCES
DISPLAY TERMINALS
DRAGON
DRIVING A CAR
DYNAMIC BINDING
DYNAMIC CLUSTERING
DYNAMIC PROGRAMMING
ELECTRONIC CIRCUITS
ELECTRONICS
ENGLISH
EVALUATION FUNCTIONS
EXPERT SYSTEMS
EXPLANATION CAPABILITIES
FABLES OR FAIRY TALES
FEATURE-DRIVEN SYSTEMS
FIRST ORDER LOGIC
FORMAL SEMANTICS
FRAME THEORY
FRAMES
FRAMES AND THE ENVIRONMENT
FUZZY KNOWLEDGE
FUZZY PROBLEM SOLVING
GAME OF POKER
GAME PLAYING
GENERAL PURPOSE MODELS
GENERATION OF NATURAL LANGUAGE
GEOMETRIC MODELING
GO OR GO-MOKU
GOAL SEEKING COMPONENTS
GRAIN OF COMPUTATION
GRAMMATICAL INFERENCES
GRAPH INTERPRETABLE GAMES
GRAPH MATCHING
HEARSAY
HETEROSTATIC THEORY
HEURISTIC PROGRAMMING
HEURISTIC TECHNIQUES
HILL CLIMBING
HUMAN BEHAVIOR
HUMAN MEMORY
HUMAN VISION
HYPOTHESIS FORMATION
IMAGE INTENSITY UNDERSTANDING
IMPROVING PROGRAMS
INDUCTIVE ASSERTIONS
INDUSTRIAL APPLICATION
INEXACT REPRESENTATION
INFERENCES
INFERENTIAL QUESTION ANSWERING
INFORMATION PROCESSING UNIVERSALS
INHERITANCE OF PROPERTIES
INTELLIGENT MACHINES
INTENTIONS
INTERACTIVE DESIGN
INTERACTIVE KNOWLEDGE SYSTEMS
INTERACTIVE PROGRAM SYNTHESIS
INTERPRETIVE SEMANTICS
INTONATION
INVARJANCL FOR PROBLEM SOLVING
INVARIANCES IN THE PERCEPTION OF FACES
INVESTMENT ANALYSIS
ITERATION
KNOWLEDGE BASED SYSTEMS
KNOWLEDGE SYSTEMS
LAMBDA CALCULUS
LANGUAGE COMPREHENSION
LANGUAGE DESIGN
LANGUAGE PARAPHRASE
LANGUAGE PASCAL
LANGUAGE PRIMITIVES
LANGUAGE UNDERSTANDING
LARGE DATA BASES
LEARNING
LINEAR LEXICOMETRY
LOW ORDERS OF RECOGNITION PERFORMANCE
MACHINE INTELLIGENCE IN MEDICAL DIAGNOSIS
MACRO PROCESSING FOR AN ON-LINE NEWSLETTER
MANAGEMENT INFORMATION SYSTEMS
MEANS FOR COMPUTER MOVIES
MEDICAL CONSULTATION
MINIMAL SPANNING FORESTS OR TREES
MOTION IN SCENE DESCRIPTION
NEURAL NETWORKS
NEWSLETTER REPORTERS
NONDETERMINISTIC PROGRAMMING
OBJECT LOCATIONS AND MOVEMENTS IN NATURAL IMAGES
OBJECT MANIPULATING ROBOTS
OPERATIONAL REASONING
OPTIMAL PROBLEM SOLVING SEARCH
OPTIMIZED CODE FOR THE TRANSFER OF COMMENTS
PAPERS BY BILL WOODS
PARALLELISM IN PROBLEM SOLVING
PARTIAL EVALUATOR
PATTERN DIRECTED FUNCTION INVOCATION
PATTERN MATCHING
PATTERN RECOGNITION
PERCEPTRONS
PHOTOGRAMMETRY
PICTURE RECOGNITION
PLANNER-LIKE LANGUAGES
PREDICATE CALCULUS
PREFERENTIAL SEMANTICS
PRICE'S TUTORIAL
PROBLEM SOLVING
PROCEDURAL EVENTS
PRODUCTION SYSTEMS
PRODUCTIVITY TECHNOLOGY
PROGRAM VERIFICATION
PSYCHOLOGY
RECOGNITION DEVICES
REPRESENTING REAL-WORLD KNOWLEDGE IN RELATIONAL PRODUCTION SYSTEMS
RESOLUTION THEOREM PROVING
RESOURCE LIMITED PROCESSES
RETRIEVAL
ROBOTICS COOPERATION
RULE ACQUISITION CAPABILITIES
SCENE SEGMENTATION
SEMANTIC NETS
A SEMANTIC NETWORK
SEMANTIC NETWORKS
SENTENCE MEANING IN CONTEXT
SENTENCE MEANING IN CONTEXT
SERIAL PATTERN ACQUISITION
SEVERAL GOALS
SHAPE GRAMMARS
SHAPE TOPOLOGY
SIMULTANEOUS ACTIONS
SNAPPING DRAGONS
SOFTWARE INTERRUPTS
SPEECH UNDERSTANDING
STATE DESCRIPTION MODELS
STOCHASTIC MODELING
STORAGE REDUCTION
STRUCTURED PATTERN RECOGNITION
SYMBOL MAPPING IN BASEBALL
SYNCHRONIZATION OF CONCURRENT PROCESSES
SYNTACTIC METHODS
SYNTAX
SYNTHESIS OF LINE DRAWINGS
TELLELOGICAL REASONING
TEMPORAL SCENE ANALYSIS
TEXTURE ANALYSIS
THE ARTICLE BY ALLEN NEWELL
THE BAY AREA CIRCLE
THE BERKELEY DEBATE
THE COMPUTERS AND THOUGHT AWARD
THE DATES OF THE WORLD COMPUTER CHESS CONFERENCE
THE DEDUCTIVE PATHFINDER
THE DREYFUS DEBATE
THE ENVIRONMENT
THE FEDERAL JUDICIAL SYSTEM
THE GAME OF POKER
THE HISTORY OF AI
THE HUNGRY MONKEY
THE INSANE HEURISTIC
THE LANGUAGE PASCAL
THE LOCATION OF OBJECTS IN MAGAZINES
THE LOGICAL REDUCTION OF LISP DATA BASES
THE META-SYMBOLIC SIMULATION OF MULTIPROCESS SOFTWARE
THE META-MATHEMATICS OF LISP1 OR LISP2
THE NOMINATION OF NOMINEES BY A NATIONAL NOMINATING COMMITTEE
THE ONTOGENY OF NON-INDEPENDENT SUBPROBLEMS
THE PARRY SIMULATION OF PARANOIA
THE PERFORMANCE OF PATTERN MATCHING RULES
THE SIX SEVEN EIGHT NINE GAME
THE STOCK MARKET
THE STRUCTURE OF ANY VARIETY OF COMPUTER TERMINAL
THE TECH-II CHESS PROGRAM
THE WEAK LOGIC OF PROGRAMS
THREE DIMENSIONAL MODELS
TIME COMPLEXITY
TIME OR SPACE BOUNDS
TROUBLESHOOTING
UNDERSTANDING
UNIFORM PROOF PROCEDURES
USING S-L-GRAPHS
VISUAL COMMUNICATION
VISUAL PLANES IN THE RECOGNITION OF POLYHEDRA
A FUNCTIONAL DESCRIPTION OF THE
HEARSAY-II SPEECH UNDERSTANDING SYSTEM

Lee D. Erman

Carnegie-Mellon University, Pittsburgh, Pa. 15213

ABSTRACT

A description of the September, 1976, version of the
Hearsay-II system is given at the knowledge-source level,
indicating the actions of each knowledge-source and their
interactions.

INTRODUCTION

The Hearsay-II system has been described elsewhere in
terms of its system organization, including the model which has
driven that design [LeH75, ErMu75, FePa76]. Also, the
individual knowledge sources (KSs) have been separately
reported on in detail. In this paper, a description of the
September, 1976, version of the system is given in terms of
the functions and interactions of the KSs. This does not
include a description of how this configuration is realized
within the general Hearsay model and Hearsay-II system, nor
does it include many details of the inner workings of the KSs,
or comparisons of Hearsay-II with any other systems.

The task for the system is to answer questions about
and retrieve documents from a collection of computer science
abstracts (in the area of artificial intelligence). Example
sentences are:

1. "Which abstracts refer to theory of computation?"
2. "List those articles."
3. "What has Minsky written since nineteen seventy-four?"

The vocabulary contains 1011 words (in which each extended
form of a root, e.g., the plural of a noun, is counted separately,
if it appears). The grammar which defines the legal sentences
is context free and includes recursion. The style of the
grammar is such that there are many more non-terminals than
in conventional syntactic grammars; the information contained
in the greater number of nodes provides semantic and
pragmatic constraint within the grammatical structure. For
example, in place of "Noun* in a conventional grammar, this
grammar includes such non-terminals as 'Topic', 'Author', 'Year',
'Publisher', etc.

The grammar allows each word, on the average, to be
followed by seventeen other words of the vocabulary. The
standard deviation of this measure is very high (about 51),
since some words can be followed by many others (up to 300
in several cases). For the sentences used for performance
testing, the average length is seven words and the average
number of words that can follow any initial portion of the
sentence is thirty-four.

The September, 1976, configuration of the system
recognizes about 80% of its test utterances (run blind) word-
for-word correctly, with about 90% of the utterances being
interpreted semantically correct.

SIGNAL ACQUISITION, PARAMETER EXTRACTION,
SEGMENTATION, and LABELLING

An input utterance is spoken into a medium-quality
Electro-Voice RE-51 close-speaking headset microphone in a
fairly noisy environment (105 db). The audio signal is low-
passed filtered and 9-bit sampled at 10 KHz. All subsequent
processing, as well as controlling the A/D converter, is digital
and is done on a time-shared PDP-10 computer. Four
parameters (called "ZAPDASH") are derived by simple
algorithms operating directly on the sampled signal [GoZa77].
These parameters are extracted in real-time and are initially
used to detect the beginning and end of the utterance.

The ZAPDASH parameters are next used by the SEG
knowledge-source as the basis for an acoustic segmentation
and classification of the utterance. This segmentation is
accomplished by an iterative refinement technique: First,
silence is separated from non-silence; then, the non-silence is
broken down into the sonorant and non-sonorant regions, etc.
Eventually, five classes of segments are produced: silence,
sonorant peak, sonorant non-peak, fricative, and flap.
Associated with each classified segment is its duration,
absolute amplitude, and amplitude relative to its neighboring
segments (i.e., local peak, local value, or plateau). The
segments are contiguous and non-overlapping, with one class
designation for each.

Finally, the SEG KS does a finer labelling of each
segment. The labels are allophonic-like; there are currently 98
of them. Each of the 98 labels is defined by a vector of auto-
correlation coefficients [ItMi75]. These templates are
generated from speaker-dependent training data that have
been hand-labeled. The result of the labelling process, which
matches the central portion of each segment against each of
the templates using the Itakura metric, is a vector of 98
numbers; the i'th number is an estimate of the (negative log)
probability that the segment represents an occurrence of the
i'th allophone in the label set.
WORD SPOTTING

The initial generation of words, bottom-up, is accomplished by a three-step process.

First, using the labelled segments as input, the POM knowledge source [SmWa76] generates hypotheses for likely syllable classes. This is done by first identifying syllable nuclei, then "parsing" outward from each nucleus. The syllable-class parsing is driven by a probabilistic "grammar" of "syllable-class -> segment" productions; the rules and their probabilities are learned by an offline program which is trained on hand-labelled utterances. The current trained set covers about 360 word tokens. For each nucleus position, several competing syllable-class hypotheses are generated -- typically three to eight.

Next, the syllable classes are used to hypothesize words. Each of the 1011 words in the vocabulary is specified by a pronunciation description. In order to hypothesize words, a pronunciation is generated from among those words containing that syllable-class. The MOW KS [SmWo76] looks up each hypothesized syllable class and generates word candidates from among those words containing that syllable-class. For each word that is multi-syllabic, all of the syllables in one of the pronunciations must match above a threshold. Typically, 50 words of the 1011-word vocabulary are generated at each syllable nucleus position.

Finally, the generated word candidates are rated and their begin- and end-times adjusted by the WIZARD knowledge source [LoHa76]. For each word in the vocabulary, WIZARD has used a network which can parse the possible pronunciations. This rating is calculated by finding the path through the network which best matches the label segments, using the distances associated with each label for each segment; the rating is then based on the difference between this best path and the segment labels. The result of the word processing so far is a set of words. Each word includes a begin-time, an end-time, and a confidence rating. A policy KS, called WORD-CTL ('word control'), selects a subset of these words, based on their times and ratings, to be hypothesized: it is these selected word hypotheses that form much of the base for the "top-end" processing that now begins. Typically, these selected hypotheses include about 75% of the words actually spoken (i.e., "correct" word hypotheses) and with each correct hypothesis having a rating which ranks .1 on the average about three, as compared to the five to twenty-five or so hypotheses which compete with it (i.e., which significantly overlap it in time). The non-selected words are retained internally by WORD-CTL for possible later hypothesis generation.

TOP-END PROCESSING

Word-Island Generation

The WOSEQ knowledge source [LeSe77] has the job of generating, from the word hypotheses generated bottom-up, a small set (about three to ten) of word sequence hypotheses. Each of these sequences, or islands, can be used as the basis for expansion into larger islands, hopefully culminating in an hypothesis that spans the entire utterance. Multi-word islands are used rather than single-word islands because of the relatively poor reliability of ratings of single words as well as the limited syntactic constraint supplied by single words.

WOSEQ uses two kinds of knowledge to generate multi-word islands:

A table derived from the grammar indicates for every ordered pair of words in the vocabulary (1011 x 1011) whether that pair can occur in that order in some sentence of the defined language. This binary table (which contains about 172,115 "1"s) thus defines "language-adjacency".

Acoustic-phonetic knowledge, embodied in the JUNCT KS, is applied to pairs of word hypotheses and is used to decide if that pair might be considered to be language-adjacent in the utterance. JUNCT uses the dictionary pronunciations and examines the segments at their juncture (gap or overlap) in making its decision.

WOSEQ takes the highest-rated single words and generates multi-word sequences by expanding them with other hypothesized words that are both time- and language-adjacent. This expansion is controlled by heuristics based on the number and ratings of competing word hypotheses. The best of these word sequences (which occasionally includes single words) are hypothesized.

The top-end processing is started by the creation of these word-sequence hypotheses. Subsequently, WOSEQ may generate additional hypotheses if the recognition process seems not to be making those already hypothesized. These additional hypotheses may include shorter, decomposed versions of some of the original ones.

Word-Sequence Parsing

Because the syntactic constraints used in the generation of the word sequences are only pair-wise, a sequence longer than two words may not be syntactically acceptable. A component of the SASS [HaSy77, HaLa77] knowledge source can parse a word sequence of arbitrary length, using the full constraints given by the language. This parsing does not require that the word sequence form a complete non-terminal in the grammar nor that the sequence be sentence-initial or sentence-final, only that the words occur contiguously in some sentence of the language. If a sequence hypothesis does not parse, the hypothesis is marked as "rejected". Otherwise, a phrase hypothesis is created. Associated with the phrase hypothesis is the word sequence of which it is composed, as well as information about the way (or ways) the words parsed.
Word Predictions from Phrases

Another component of the SASS knowledge source that, for any phrase hypothesis, generates predictions of all words which can immediately precede and all which can immediately follow the phrase in the language. In doing the computation to generate these predictions, this KS uses the parsing information attached to the phrase hypothesis by the parsing component.

Word Verification

An attempt is made to verify the existence of or reject each such predicted word, in the context of its predicting phrase. If verified, a confidence rating for the word is also generated. First, if the word has been hypothesized previously and passes the test for time-adjacency (by the JUNCT KS), it is marked as verified and the word hypothesis is associated with the prediction. (Note that a single word may thus become associated with several different phrases.) Second, a search is made of the internal store of WORD-CTL to see if the candidate can be matched to a previously generated candidate which has not been hypothesized. Again, JUNCT makes a judgment about time-adjacency. Finally, WIZARD compares its word-pronunciation network to the segments in an attempt to verify the prediction.

For each of these different kinds of verification, the approximate begin-time (end-time) of the word being predicted to the right (left) of the phrase is taken to be the end-time (begin-time) of the phrase. The end-time (begin-time) of the predicted word is not known and, in fact, one requirement of the verification step is to generate an approximate end-time (begin-time) for the verified word. In general, several different “versions” of the word may be generated which differ primarily in their end-times; since no context to the right (left) of the predicted word is given, several different estimates of the end (beginning) of the word may be plausible based solely on the segmental information.

Word-Phrase Concatenation

For each verified word and its predicting phrase, a new and longer phrase may be generated. This process, accomplished by a component of SASS similar to the Word-Sequence recognition component, involves parsing the words of the original phrase augmented by the newly verified word. The extended phrase is then hypothesized and includes a rating based on the ratings of the words that compose it.

Complete Sentences and Halting Criteria

Two unique “word” hypotheses are generated before the first and after the last segment of the utterance to denote begin and end of utterance, respectively. These same “words” are included in the syntactic specification of the language and appear as the first and last terminals of every complete sentence. Thus, any verified phrase that includes these as its extreme constituents is a complete sentence and spans the entire utterance. Such a sentence becomes a candidate for selection as the system’s recognition result.

In general, the control and rating strategies do not guarantee that the first such complete spanning hypothesis found will have the highest rating of all possible spanning sentence hypotheses that might be found if the search were allowed to continue, so the system does not just stop with the first one generated. However, the characteristics of such an hypothesis are used to prune from further consideration other partial hypotheses which, because of their low ratings, are unlikely to be extendable into spanning hypotheses with ratings higher than the best already-discovered spanning sentence. This heuristic pruning procedure is based on the form of the ratings function (i.e., how the rating of the phrase is derived from its constituent words). The pruning procedure considers each partial phrase and uses the ratings of other word hypotheses in the time areas not covered by the phrase to determine if the phrase might be extendable to a phrase rated higher than the spanning hypothesis; if not, the partial phrase is pruned. This pruning process and the rating and halting policies are discussed in [HaPo77].

The recognition process finally halts in one of two ways: First, there may be no more partial hypotheses left to consider for predicting and extending. Because of the combinatorics of the grammar and the likelihood of finding some prediction that is rated at least above the absolute rejection threshold, this form of termination happens when the pruning procedure has been effective and has eliminated all competitors. Second, the expenditure of a preselected amount of computing resources (time or space) also halts the recognition process; the actual thresholds used are set according to the past performance of the system on similar sentences (i.e., of the given length and over the same vocabulary and grammar).

Once the recognition process is halted, a selection of one or more phrase hypotheses is made to represent the result. If at least one spanning sentence hypothesis was found, the highest-rated such hypothesis is chosen; otherwise, a selection of several of the highest-rated of the partial phrase hypotheses is made, basing the selection on the longest ones which tend to overlap (in time) the least.

Attention Focusion

The top-end processing operations include (a) word-island generation, (b) word sequence parsing, (c) word prediction from phrases, (d) partial phrase generation, and (e) word-phrase concatenation. Of these, (b), (c), and (d) are the most frequently performed. In general, there are a number of these actions waiting to be performed at various places in the utterance. The selection at each point in the processing of which of these actions to perform is a problem of combinatorial control, since the execution of each action will, in general, generate more such actions to be done.

To handle this problem, the Hearsay-II system has a statistically-based scheduler [HaFo77] which calculates a priority for each action and selects, at each time, the waiting action with the highest priority. The priority calculation attempts to estimate the usefulness of the action in fulfilling the overall system goal of recognizing the utterance. The calculation is based on information specified when the action is triggered. For example, the word verifier is triggered whenever words are predicted from a phrase hypothesis; the information passed to the scheduler in order to help calculate the priority of this instantiation of the verifier includes such
things as the time and rating of the predicting phrase and the number of words predicted. In addition to the action-specific information, the scheduler keeps track of the overall state of the system in terms of the kinds and quality of hypotheses in each time area.

INTERPRETATION and RESPONSE

The SEMANT knowledge-source [HaDi77] accepts the word sequence(s) result of the recognition process and generates an interpretation in an unambiguous format for interaction with the data base that the speaker is querying. For example, the spoken sentence

"What has Minsky written since 1974?"

is represented in this format as

Type: SREQUEST
Subtype: SQUERY!AUTHOR!DATE
[Date: >1974; Author: "MINSKY"]

The interpretation is constructed by actions associated with "semantically interesting" non-terminals in the parse tree(s) of the recognized sequence(s). If recognition results in two or more partial sequences, SEMANT constructs a consistent interpretation based on all of the partial sentences, taking into account for each partial sentence its rating, temporal position, and consistency (or competitiveness) as compared to the other partial sentences.

The DISCO knowledge-source [HaDi77] accepts the formatted interpretation of SEMANT and produces a response to the speaker. This response is often the display of a selected portion of the queried data base. In order to retain a coherent interpretation across sentences, DISCO has a finite-state model of the discourse which is updated with each interaction.

REFERENCES


[HaSy77] F. Hayes-Roth, L. D. Erman, M. Fox, and D. J. Mostow, "Syntactic Processing in Hearsay-II", in [CmSp77].

[HaSe77] F. Hayes-Roth, M. Fox, G. Gill, and D. J. Mostow, "Semantics and Pragmatics in the Hearsay-II Speech Understanding System", in [CmSp77].
ABSTRACT

In Hearsay-II, a word recognizer hypothesizes words bottom-up from acoustic data. Usually many competing words are hypothesized for each time interval of speech, with the correct word rarely too-ranked. Due to the unreliable ratings of words and the limited syntactic constraint supplied by single words, the use of single-word islands would cause the recognition system to explore many blind alleys before abandoning an incorrect island. In addition, the multiplicity of words makes the parsing of all possible word sequences extremely time-consuming. The Hearsay-II island selection strategy uses (1) knowledge of what word adjacencies are allowed by the grammar, (2) analysis of acoustic data at the junctures between word hypotheses, and (3) heuristics based on the number of competing word hypotheses, to form multi-word islands which the syntax-level knowledge source first checks for grammatically and then attempts to extend to form a complete recognition.

INTRODUCTION

Conventional strategies for controlling the search in a continuous speech understanding system fall into two major categories: left-to-right (HARPY [Lowerra, 1976], Hearsay-I [Reddy, 1973]) and island-driven (SRI [Paxton, 1975], SPCHILS [Woods, 1975], Hearsay-II [Lesser, 1975]) strategies. In the left-to-right strategy, as the name implies, the search always begins at the start of the utterance and continues to extend in a left-to-right manner each partially hypothesized phrase that appears plausible. In contrast, the island-driven strategy, before beginning the process of phrase hypothesization and extension, first performs a scan of the entire utterance in an attempt to spot likely words [Smith 1976, Klovstad 1976]. The best found in this phase are chosen as the initial phrasal hypotheses for the second phase of the search. In this second phase, a partial phrase chosen for further extensions can be extended by prediction of grammatically legal word extensions on either the left or right or in both directions, depending, for instance, on the constraints given by the grammar about which direction has fewer extensions [see Hayes-Roth and Lesser 1976, Paxton and Robinson 1975, and Woods 1975 for a discussion of techniques for choosing the next hypothesis to extend]. This strategy allows the phrasal hypothesis to be concatenated with existing partial phrases to construct new, enlarged hypotheses.

The advantages of the left-to-right strategy over the more sophisticated island-driven strategy are mainly in the area of efficiency: (1) the computationally expensive word-spotting phase is bypassed and (2) the application of grammatical knowledge and the overhead for controlling the search is much less expensive. The major disadvantage of the left-to-right scheme is that the beginning of the utterance may not contain very good acoustic data and thus lead to initial word predictions that are very poor; in this case, it may be very difficult or impossible (if the correct word was not hypothesized) to recognize the utterance. The major advantage of island-driven strategy is its robustness; there may be hypothesized more than one correct initial island, and thus there exists more than one sequence of steps to achieve the correct recognition. In addition, the island-driven strategy would seem to have a higher probability of starting the search with an initial island that is valid because of its word-spotting phase. However, this word-spotting search may not in practice produce results as valid as would be expected because words are predicted based only on acoustic constraints, neither grammatical nor co-articulation constraints are used except at the beginning and end of the utterance. Another advantage of the island-driven strategy is that it can use variations in the branching factor of the grammar at different points in the utterance to reduce the space needed to be searched.

The major disadvantage of both of these search strategies is that they are particularly sensitive to major rating 'errors' on single words--cases where a valid word is rated lower than an invalid word in the same time area. If the correct word in the starting area is very poorly rated, a best-first search from the higher-rated alternatives will develop a very large search space, and backtracking all the way to the initial incorrect decision will be very expensive and unlikely.

Two means of overcoming this shortcoming exist. First, in the left-to-right--first search, the N top rated words in an area are used to begin searches, and as long as one of these is correct, recognition is not precluded. The second alternative is to identify multi-word sequences of word hypotheses that are most probably correct as the
starting islands in an island-driven strategy. In comparison with single-word islands or left-to-right single-word starting hypotheses, multi-word sequences are more reliable for two reasons: under certain generally applicable conditions, the credibility of a sequence hypothesis exceeds that of a single word hypothesis and, secondly, the reliability of a validity rating for a sequence is greater than that of a single word hypothesis.

To substantiate this conjecture, consider the following average rank order statistics for initial islands based on the three different approaches. These data were collected over 34 training utterances, with each island generation strategy applied to all utterances. The average sentence length was 5.5 words. The left-to-right and the single-word island-driven strategies have the same rank order statistic which is 2.6 (i.e., there are on the average 1.6 islands with ratings better than the correct one). It is interesting to note that in none of the 34 utterances did the left-to-right strategy not hypothesize the correct word in the initial utterance position; the average number of words hypothesized for the initial position was eleven words. The average rank order statistic for the multi-word island strategy, if one utterance is eliminated in which the rank order is 30, is 2.0; the average length of the best correct multi-word island is 2.3 words, whereas the average number of correct words hypothesized bottom-up is 3.0.

A MULTI-LEVEL ISLAND-DRIVEN STRATEGY

The strategy found to be most effective in the Nearest-II system (as applied to a 1000-word vocabulary with an average word length of 3.3) is to select multi-word sequences of word hypotheses as starting islands for syntax-level processing. This strategy introduces a new level of hypothesis, the word sequence, between the conventional lexical and phrase levels. A word-sequence hypothesis is a concatenation of one or more word hypotheses. In contrast with a phrasal hypothesis, a word-sequence hypothesis is created before the syntax-level knowledge source begins its work, and may not be grammatical (i.e., it may represent a sequence of words which does not appear in any sentence in the language defined by the grammar).

The decision to create word-sequence hypotheses arose from the realization that the combinational space of all possible sequences of word hypotheses, generated as a result of the word-spotting scan, can be reduced sharply by applying a computationally inexpensive filter to the data. This filter is based on simple kinds of grammatical and co-articulation knowledge about which word pairs are possible. The grammatical constraints are specified through a square bit matrix, whose order is the size of the vocabulary, and each entry (ij) in the matrix indicates whether word j can follow word i in the grammar. If two words can follow each other, they are called "language-adjacent." The co-articulation constraints are specified through another square matrix, whose order is the size of the number of phonemes. Each entry (ij) in the matrix indicates what type of acoustic segments are allowed in the juncture between two words, the first word ending with the phoneme i and the second word beginning with phoneme j. The appendix contains a more detailed description of how the co-articulation constraints are implemented. If two words pass these co-articulation constraints, they are said to be "time-adjacent." A word-sequence hypothesis always consists of word hypotheses which are pair-wise language-adjacent and time-adjacent.

Consider a pair of word hypotheses that are language- and time-adjacent. If there is a third hypothesis that is language- and time-adjacent, either to the left of the first word of the pair or to the right of the second, it can be concatenated onto the pair to form a three word hypothesis. This action of extending could be repeated (leftward and rightward) until there were no more possible extensions. If there were many alternative extensions at each point, this process would result in the creation of a larger number of partially similar word sequences. However, it is clear that a sequence of more than two words may not be grammatical, since language-adjacency is defined only between successive two word pairs. The determination of the grammaticality of a sequence by the syntax-level knowledge source is a relatively expensive operation (between 1 and 1 seconds on a PDP-10 KA10); thus, there is a bias against creating word sequences which have a high probability of being incorrect.

The factors which are of interest in deciding whether a word sequence is good are: the length of the sequence, the ratings of its individual word hypotheses, and the number of other word hypotheses competing (overlapping in time) with each of them. The best starting island is the longest one which has a very high probability of being correct, with correctness taking precedence over length; correctness is a function of both individual word validity rating and the lack of similar alternative sequences. These considerations led to the following algorithm for sequence creation:

(1) The 30 highest-rated word hypotheses anywhere in the utterance are chosen as initial one word sequences. Those with ratings less than some cutoff are discarded unless doing so would leave less than five, in which case the five top words are kept.

(2) Each initial sequence is assigned a competing sequence count (CSC) of 1.

(3) For each current sequence, the sets of all word hypotheses left-(right) language- and time-adjacent to the beginning (ending) words of the sequence are found. If the current sequence has CSC=N, and R right-adjacent words are found, then a right extension would have CSC=N*R.

(4) Only those extensions whose average word ratings exceed a cutoff proportional to the square root of RAR are formed. The direction chosen for extension is a function of CSC count for the direction and the validity of the highest word that remains to be extended in the specific direction.

(5) Steps 3 and 4 are repeated in a recursive manner until no more extensions can be formed.

All sequences that are generated as a result of this process which are subsequences of another sequence are discarded.

This algorithm produces a large number of potential word sequences, usually between 10 and 100. The cost of validating them all for grammaticality is expensive. Thus,
another level of filtering is performed, based on a rating attached to each word sequence. The rating of a sequence is an increasing function of these quantities: (1) the duration-weighted average word rating, $AVGRATE$, computed by summing the product (word's rating)$\times$(number of syllables it contains) over all words in the sequence and then dividing by the number of syllables in the sequence; (2) the duration, $DUR$, computed as the percent of the utterance's syllables contained in the sequence; (3) the number of words in the sequence, $NWORDS$. The rating function is

$$RATE = AVGRATE + 0.1 \times NWORDS + AVGRATE + 0.5 \times DUR$$

The highest rated word sequence plus word sequences whose rating is some epsilon away from the highest are chosen as candidates for further evaluation. In addition, another criterion is employed to choose sequences for further evaluation: if at all possible, there should be at least one word sequence for each area of utterance; the time areas not covered by the highest rated word-sequences are the areas that are attempted to be covered by lesser rated word-sequences. Word sequences not chosen by this filtering are not discarded but rather are held in abeyance until either processing later on stagnates, or an existing word sequence candidate has been found to be ungrammatical or cannot be successfully extended; in these cases, these lower-rated sequences are hypothesized for consideration by the rest of the system. This process of word sequence generation for the 34 utterances results in an average of 8.1 initial candidates, with an average of 6.6 more candidates being generated during the run.

The basic reason that this algorithm is the identification of sequences of time-adjacent and language-adjacent words whose credibility is high. Although a large proportion of these sequences may not be grammatical, very few highest-rated sequences are ever incorrect (unless no successive correct word pairs have been hypothesized). Furthermore, the computation of CSC biases against forming long sequences except when the chance occurrence of a language-adjacent pair is small; thus, in only ten percent of utterances does a highly-rated incorrect sequence contain a correct subsequence of length greater than one which does not occur in a longer correct sequence. In such a case, if the grammaticality of the incorrect long sequence is rejected by the syntax knowledge source, a decomposition of the sequence into two maximal subsequences occurs; these decompositions will be hypothesized subsequently if rated sufficiently high. This is a form of backtracking and, therefore, is subject to the same weaknesses as other backtracking search algorithms. In this case, however, the probability of a false initial island has been greatly reduced. As a result, the chance of a totally fruitless search is correspondingly reduced.

The effectiveness, in terms of both total system error rate and amount of search performed, of this multi-word island approach over both the left-to-right and single-word island-driven strategies is indicated by the following statistics; the overall error rate for the three strategies is 67%, 47% and 54%, respectively. In the ten sentences that were recognized correctly by all three strategies, the average number of phrases hypothesized are 47, 68 and 68, respectively.

**CONCLUSION**

The multi-word sequence generation procedure is a key knowledge sources in Hearsay-II. By exploiting the redundancy of the language to identify plausible word sequences and, incidentally, increasing the probability that a valid starting island hypothesis will be more highly rated than an incorrect one, this source of knowledge provides very reliable and useful knowledge to direct the overall search. In our opinion, this knowledge source is a paradigmatic example of the effective use of redundancy and statistical sampling to achieve a reduction of uncertainty in problems characterized by fuzzy and partial information.

**REFERENCES**


APPENDIX

This appendix describes the word pair adjacency acceptance procedure (JUNCT) developed for Hearsay-II, the knowledge it uses, and the current results. Such a procedure must be computationally inexpensive, making decisions on hundreds of pairs of hypothesized words. It must rely upon knowledge of word junctures and upon the information contained in the segmental transcription of the spoken utterance. And it must reject as many incorrect pairs (word pairs not actually spoken) as possible, without rejecting any of the correct pairs.

As input, JUNCT receives a pair of word hypotheses. If it determines, based upon the times associated with the hypotheses, the juncture rules contained in the procedure, and the segmental description of the spoken utterance, that the words are adjacent, the pair is accepted as a valid sequence; otherwise it is rejected.

The word junctures upon which JUNCT must make its decisions fall within three distinct cases: (1) Time-contiguous hypotheses: Words which are time contiguous are immediately accepted by JUNCT as a possible sequence; no further tests for adjacency are performed. (2) Overlapping hypotheses: When two words overlap in time, juncture rules are applied in the context of the segmental interpretation of the utterance to determine if such a juncture is allowable for the word pair. (3) Separated hypotheses: When the words are separated by some interval of time, rules are applied, as in the overlap case, to determine whether the pair can be accepted as a valid sequence in the utterance.

The juncture rules used by JUNCT are of two types: (1) allowable overlaps of word end- and begin-phonemes, and (2) tests for disallowed segments within the word juncture. A bit matrix of allowable overlaps is precompiled into the procedure, and is indexed by the end- and begin-phonemes of the word pair. Any overlap juncture involving phonemes which are not allowed to share segments is rejected by JUNCT. In the separation case, as in allowed overlaps, the segmental description of the spoken utterance is examined in the context of the end- and begin-phonemes of the word pair to determine if any disallowed segments are present in the juncture. If such segments are found, the word pair is rejected. Only when a word pair passes all rule tests which apply in the segmental context of its juncture is it accepted as a valid sequence.

Examples of allowable phoneme overlaps are the following: (1) Allow words to share a flap-like segment if one of the juncture phonemes is a stop. (2) Allow nasal-like segment overlaps in nasal-stop phoneme junctures. (3) In a fricative-stop phoneme juncture, allow sharing of aspirates, fricatives, silences, and flap-like segments.

Examples of non-allowed segments in a juncture are the following: (1) Do not allow a vowel segment in any juncture (overlap or separation), unless it is a vowel-vowel phoneme juncture. (2) Do not allow a fricative segment in any non-fricative juncture.

Current Results

Stand-alone performance evaluation runs were made over 60 utterances using words generated from files produced by the Hearsay-II word hypothesizer. Syntactically adjacent pairs of words whose ratings were 40 and above (on a scale from 0 to 100) and whose times (left-word end time and right-word begin time) were within a 200 millisecond interval were considered. All of the words used for testing the procedure were hypothesized “bottom-up” in Hearsay-II; no grammatically based predictions were used in the evaluation runs. Table 1 summarizes the performance of the JUNCT procedure.

It is expected that, as lower-level sources of knowledge provide more accurate times for word hypotheses, the rules for acceptance of valid word pairs may be tightened, further increasing the speed and performance of Hearsay-II.

<table>
<thead>
<tr>
<th>CORRECT WORD PAIRS</th>
<th>INCORRECT WORD PAIRS</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCEPTED</td>
<td>188 (95.4%)</td>
<td>2891 (41%)</td>
</tr>
<tr>
<td>REJECTED</td>
<td>5 (2.5%)</td>
<td>4224 (59%)</td>
</tr>
<tr>
<td>TOTAL</td>
<td>197</td>
<td>7115</td>
</tr>
</tbody>
</table>

Table 1. JUNCT performance (60 utterances)
ABSTRACT
A key problem for speech understanding systems is the verification of word hypotheses generated by various knowledge sources in the system. In this paper we will discuss the general problem of word verification in speech understanding systems. A description of our matching algorithm for word verification which is based on that used in the HARPY system, a general connected speech recognition system (Auerre, 1976), is given. An example of the verification of a word hypothesis using this algorithm is presented. Problems which arose in applying this technique to verification of individual words in a connected speech understanding system and their solutions are discussed. A performance analysis of the verifier in terms of accuracy and speed is given and directions for future work are indicated.

INTRODUCTION
Word verification is the evaluation of word hypotheses in speech understanding or recognition systems. The aim of this evaluation is to decide which hypotheses are worthy of further processing by other parts of the system. This evaluation is generally performed by measuring how closely a given word matches its premised representation. The representation and the match of the acoustic signal may be performed at various representational levels such as the parametric, phonetic and syllabic. Since errors are introduced and propagated as information is encoded from the parametric to the syllabic level, accurate matching becomes increasingly difficult at each successive level of abstraction. However the computation time for matching decreases since there are fewer match elements each containing more information.

Words may be hypothesized from many diverse sources of knowledge not solely based upon acoustic evidence. If 50 to 80% of the vocabulary is hypothesized for each word position in the utterance (the current HEARSAY bottom-up performance), the verifier must distinguish between 50 to 80 competing word candidates in a 1000 word vocabulary. Even with significant improvements in word hypothesization (i.e. decreasing the effective vocabulary hypothesized to 50% per word position), as we move to systems with large vocabularies (~100,000 words see Smith 1977) the number of potential verifiable words remains quite large.

The verifier must assign a likelihood score which is commensurate with the match between the underlying acoustic data and the phonetic description of the word. The goodness of a score may be only temporarily significant so the scores should rank order the competitive words in any time area such that the correct word is high in the ordering.

Besides this acceptance criteria, it is also necessary for the verifier to reject absolutely a large percentage of the hypothesized words, without rejecting significant numbers of correct words, in order to constrain the combinatoric explosion of hypotheses at higher levels.

THE HEARSAY ENVIRONMENT
Word verification is performed within HEARSAY II in the following environment. Word candidates may be supplied from a bottom-up word hypothesizer (POKOW) based on acoustic information or from a top-down syntax and semantics knowledge source (SASS) based on syntactic information and constraints provided by the grammar. POKOW (Smith 1976) provides word hypotheses which have reasonable underlying acoustic support over a definite portion of the utterance. The times supplied are used to guide verification but do not preclude change. SASS (Hayes-Roth 1977) provokes words which can be characterized as being syntactically plausible in a particular time area of the utterance. No pruning is performed according to the credibility of the underlying acoustic information. Since these words are always hypothesized based on a previously verified word or from the boundaries of the utterance, only one time is known. This requires that the verifier must not only rate the hypothesis, but must also predict the missing time. In addition, since words may be predicted to the left or right of a verified word, the verifier must have the ability to match words in both directions.

HEARSAY operates under the hypothesize-and-test paradigm to produce many competing hypotheses which overlap in time. Each word hypothesis must be verified and assigned a rating before it can be used by other sources of knowledge. Each of these verified hypotheses can in turn be used as seeds to generate new sets of syntactically plausible words. A measure of the fan-out from each word is the effective branching factor of the HEARSAY II grammar (Goodman 1976) which is between 5 and 15. Thus regardless of the scoring performance, a verifier must be computationally efficient in order to be useful in this type of system.

VERIFICATION MODEL
WIZARD can be decomposed into three major parts: word networks, a segmentation of the utterance, and a control structure which implements the matching algorithm. First, each word in the lexicon is represented by a statically defined network which embodies alternate pronunciations of the word. Each node in the word network represents a phone and arcs indicate successor/predecessor relationships between phones.

This work was supported by the Defense Advance Research Projects Agency under contract F44620-73-C-0074 which is monitored by the Air Force Office of Scientific Research.

The acoustic information is a segmentation of the utterance where each segment is represented as a vector of the same templates and segmentation as the HARPY system (Lowerrre, 1976). As in HARPY the phone probabilities are distance measures between each segment and acoustic-phonetic templates in the phonetic dictionary. This value is a scaled log likelihood measure (since the probabilities do not sum to 1) and is used directly in computing the word match score over the given segments. WIZARD uses approximately 90 templates to cover all phonetic variations in its 1024 word vocabulary.

The last component is the dynamic matching algorithm. Although there is no speech dependent knowledge embodied in this module, several heuristics are employed to find optimal starting points and to choose the best final segment. These heuristics are discussed in the following section on implementation issues.

Figure 2 gives the phone probabilities for each phone in the network in each of the segments over which the match is performed. Those scores in Figure 3 marked with * indicate the best path through the mapping. The begin time of each segment is given, along with the segment number, on the top of the figure. The left side is labeled with each phone in the network. Entries in the table of oo indicate that a phone was hypothesized at many places in the utterance, with good function words such as on, the, of, to, in, tend to be remembered at some later time, proved intractable. Problems in the generation of end times are mapped directly into their respective segments and verification is performed across these segments. It takes approximately 60 milliseconds of CPU time on a PDP-KA10 to perform matching in this mode.

Non-pad mode was added to handle the problem that bottom-up times may be incorrect. This mode is currently used to verify all bottom-up hypothesis. In this mode the begin/end times are mapped into segments as in non-pad mode. Then a one segment uncertainty is allowed during the matching. Thus if the begin segment, \( E \) is the end segment, segments \( B-I/B/E \) are allowable starting points for the match and \( E/I/E/E+1 \) are the allowable ending points. The nine paths between the boundary segments are evaluated in parallel by modifying the boundary conditions in the matching algorithm. As a result WIZARD must backtrack from each of the final end segments in order to find the correct segment and verification is performed across these segments. It takes approximately 100 milliseconds of CPU time on the PDP-KA10 and is about 5 times faster than exploring each of the nine paths in non-pad mode.

As we have mentioned before it is necessary to perform verification where only one of the word times is known. Two prediction modes are implemented in WIZARD, one where the end time is unknown (right) and the other predicts a missing begin time (left). As in pad mode a one segment window is evaluated around the given starting point. Then each successive segment is matched and the match score is computed as if the match were ending in that segment. The scores are ordered and the score for the best path is returned. Several experiments were performed to determine the best way to normalize the match scores. The technique employed was to verify approximately 10000 bottom-up word hypotheses from 60 utterances, normalize the scores and calculate the average rank order of the correct words. The rank order gives the number of incorrect words that
were rated as high as, or higher than, the correct word. This ordering is a measure of how many words per word position must be considered by the top level knowledge sources in order to have confidence that the correct word is present, assuming it has been hypothesized. Normalizing the scores by the time duration of the word amplified the problem of function words receiving unusually good scores. Handling compensations to word scores based on assignment scaling were also rejected. Segmental normalizations employing penalties for mapping the same phone into many successive segments proved to be too time consuming in light of the benefit derived. Currently, predict mode scores are normalized by the number of phone segments in pairs of phone segments. The other modes are normalized by N-I. These normalizations are computationally simple and embellishments tried to date have not performed significantly better.

The conversion of internal WIZARD scores to HEARSAY II hypothesis ratings was accomplished by conducting a statistical analysis of correct/incorrect word ratings over approximately 50000 verifications. By knowing the distribution of correct and incorrect word scores over each of the internal score values (dynamic range of 64), a corresponding distribution of HEARSAY II scores was calculated. The HEARSAY II score distribution allows for the absolute rejection of verified words. This threshold was set so as to reject no correct words. Scores above that threshold were distributed as to capitalize on the distributions of correct words. Several tradeoffs are possible here. If one requires that no potential correct words be rejected then WIZARD was able to reject 122 to 19% of the incorrect words hypothesized. On the other hand if it were possible for the system to perform with a small number of the correct words being rejected, a higher percentage of incorrect words could be rejected. Allowing a 5% rejection rate of correct words approximately 50% of the incorrect words can be eliminated from consideration by the higher level knowledge sources.

To aid in compensating for the apparent temporal difference in word scores, the acoustic match probabilities generated by the segmenter were normalized such that the score of the best phone in a segment had the absolute best match score. This alleviated the problem and improved the reliability of the normalized match score while leaving the rank order statistics unchanged.

RESULTS
The results summarized in Figure 4 are for five data sets, containing 100 utterances, in which 332 correct words were hypothesized bottom-up by POMOW. In addition, 13053 incorrect words were generated. The vocabulary size for POMOW and WIZARD was approximately 550 words. WIZARD rated each of the words using pad mode normalization. For each rating threshold (15, 10) the number of correct and incorrect words that were accepted or rejected is tabulated. From this data the number of words hypothesized per word position, and the percent of the vocabulary per word position, can be calculated. These numbers give a vocabulary independent measure of performance, allowing comparisons between various system configurations. An average rank order of the correct word is provided which measures, at each threshold, the number of words in each word position that must be examined in order to include the correct word. The range of rank orders between the data sets (20 utterances/set) is also noted.

DISCUSSION
The major direction of this work is the application of the HARPY network representations to the verification of single words in a connected speech understanding system. This includes the modifications to allow the various verification modes dictated by the HEARSAY II system strategies. We feel that WIZARD makes an important contribution to the overall performance of HEARSAY II and forms a groundwork upon which more sophisticated verifiers can be developed.

Several problems with the current word verification system can not be solved within the existing framework. Future work is required in the following areas. New schemes for normalization of scores have been proposed to improve the performance in segmentations having many very short transition segments. These segments in general have poor ratings and often degrade the composite word score.

Although we feel that the matching algorithm was computationally efficient when first implemented, as system strategies evolved it was found that a significant portion of recognition time was being spent in verification. A sizable increase in speed can be obtained by coding certain of the inner loops in assembly language. Other implementation oriented optimizations may be needed.

A most useful addition to WIZARD would be the ability to verify sequences of words by dynamic generation of multiple word networks. These networks would embody the appropriate word juncture rules and would allow WIZARD to rate phrasal hypotheses directly rather than having other knowledge sources calculate a composite score from the individual word scores. Along these lines, perhaps as a first step, it is necessary to handle word juncture problems which cannot be stastically encoded in the single word networks themselves. These juncture problems are a major cause of incorrect times on word hypotheses.

It will be necessary to augment the word verification system with a component to perform direct signal matching. The purpose of this addition is to disambiguate competing words which have good WIZARD scores in the same time area. We propose to extract word templates at the parametric level and perform matching using Itakura's method (Itakura, 1975). The philosophy here is to store templates for a small number of potentially different words rather than synthesize the templates by a rule-based system. This time consuming matching would be performed when indicated by higher sources of knowledge.

ACKNOWLEDGEMENTS
The original idea to implement a word verifier using a network representation was that of Raj Reddy who made continuing suggestions for refinements to the basic algorithm and for improvements to its performance. Bruce Lowerre cheerfully shared his dictionary and network generation expertise. Lee Erman and Richard Smith aided in integrating WIZARD into HEARSAY and provided the impetus for many of its interesting features.

REFERENCES
The $d'$ Model of Signal Detection Applied to Speech Segmentation

Henry G. Goldberg

Department of Computer Science
Carnegie-Mellon University, Pittsburgh, Pa. 15213

Abstract: The statistical measure, $d'$, from Signal Detection Theory, [Swe64] has been shown to parametrize the "detectability" of signal over noise in a wide variety of perceptual situations. Its usefulness is extended to the problem of quantifying error rates for segmentation of continuous speech. It has often been impossible to accurately compare different machine techniques for segmentation since errors occur as either missing or extra segment boundaries whose rates are related by internal decision thresholds. The basic $d'$ model is shown to accurately (95% confidence) describe the missing versus extra segment trade-off found in at least one, non-trivial, speech segmentation program. [Gol75]

Introduction: The last few years of computer speech recognition research have produced, among other things, a number of techniques for machine segmentation of the speech signal into phonetic (or acoustic) units, [e.g. Dix75, Bak75, Gol75]. The difficulties involved in evaluating and comparing the performance of segmenters seem to occur in two areas. First, one must acquire a definition of the "correct" segments for some large set of data. This is usually done by hand, although some automatic techniques are available.* Since the production of "correct" segmentations and their comparison with machine segmentations (e.g. What amount of mis-alignment, etc. should one allow?) involve a number of linguistic as well as recognition system-specific issues, we will not deal further with these problems here.

However, a second problem is that segmentation errors occur in two types: missed boundaries (segments) and extra boundaries (segments). There is clearly a trade-off between these two types of error, but we have not understood it well enough in a quantitative sense to compare different segmenters (or even the same segmenter "tuned" to a different point of the M/E trade-off). What was needed was a model of this trade-off which yielded a single, comparable measure of segmentation efficacy for any set of data with errors marked missing or extra. Such a model is provided by Signal Detection Theory. We will show that the theory agrees quite well with the results of a set of segmentation trials run to explore the M/E trade-off.

*The Harpy speech recognition system [Low76] can be forced to the correct words. This produces a "best" fit of the system's acoustic and phonological knowledge to the signal. When a very fine grained fit is made (average acoustic segment duration, 30 ms.), the resulting phonetic segments are very close to those produced by humans.

Signal Detection Theory: The theory of Signal Detection, as formulated by Tanner, Swets, and Green, [Tan64, Lic64] is primarily applied to detection trials which may be considered similar to the segmentation process. A detection trial presents a stimulus, which may be composed of noise or of noise and some known signal, and requires a decision to be made about the presence of the signal. This is not unlike the decision process resulting in the placement of a segment boundary based upon local information only. It is assumed that the prior likelihoods and costs of various errors are known to a decision process which senses and possibly transforms the stimulus into some internal signal space before it yields a decision on the presence of the signal. The detector's sensory data is considered, in this model, to be reduced to a single decision parameter. An optimal one, according to decision theory, is the ratio of the probabilities of two hypotheses -- that the input stimulus was signal plus noise or that it was noise alone. A simple threshold on this single parameter may be placed to optimize the expected costs given a priori likelihoods, costs of misses, false alarms, etc. Figure 1 represents such a hypothetical internal decision parameter, $L$.

Figure 1: Signal Detection Model

Very simply stated, the model assumes a single decision parameter, $L$, which may be any sensory measurement one wishes. The distribution of $L$ values for the two types of stimuli, signal-plus-noise and noise-alone, are assumed to be normal (with equal variance in the simplest version of the model). Their means differ by $d'$ times the standard deviation. Rates of "hit" and "false

alarm" — $Pr(accept|signal)$ and $Pr(accept|noise)$ respectively — are sufficient to determine the least $d'$ for which an optimal decision process can display the observed rates. When the hit and false alarm rates are plotted against one another for a number of sets of trials where the detector’s acceptance threshold has been altered, a response operator characteristic (ROC) curve is obtained (see figure 2).

When the hit and false alarm rates are plotted against one another for a number of sets of trials where the detector’s acceptance threshold has been altered, a response operator characteristic (ROC) curve is obtained (see figure 2).

The theory states that the curve is totally determined by $d'$. When the axes of the ROC curve are transformed by the inverse function of the Normal distribution function, the curve is approximately a straight line with slope=$\sigma(\text{noise})/\sigma(\text{signal})$ and $x$-intercept=$d'$.[Eg64]

This theory has been most often applied to detection trials to provide estimates of the detectability of the signal as it appears in a human perceiver’s internal sensory signal space. The estimate of $d'$ provided by the signal detection model may then be compared with well known properties of visual or auditory signals to provide a bound on the efficacy of the perceiver’s transduction process — the sensory channel. While the main thrust of its application is not relevant here, the signal detection model and the dimensionless measure $d'$ can be used as a normalized measure of segment boundary detection that is relatively unaffected by adjustments in the proportion of missing versus extra segment errors. Furthermore, the $d'$ value, once estimated, may be used to predict the entire response-operator characteristic.

**Segmentation** The results reported here are, for the most part, obtained from a segmentation program written for a comparison study of parametric representations [Gol73] and used for a while as the initial signal-to-symbol stage of the Hearsay II speech understanding system. [Erm74] A short description of the segmenter is therefore called for.

The signal amplitude, and measures of signal and of amplitude change,** (each measured over both 10 and 30 ms. intervals), are input. Speech is separated from silence and from near-silence, and flaps are detected by their amplitude contours. Then the measures of change are inspected for significant peaks (possible boundaries). The union of all such detections is processed by a correction routine to merge multiple boundaries caused by the same underlying phonetic change. The program has two advantages for this study. First, the input parametric representation is easily changed, and second, the internal, segment detection process is easily tuned along the M/E trade-off.

Results from this program were compared with a "corrected" hand segmentation. That is, the machine segmentation was compared to a phonemic-level human segmentation for discovering missing segments, and to a finer-grained phonetic-level segmentation for discovering extra segment errors.

Results The first experiment validates the Signal Detection model assumption of two (nearly) normal distributions in a signal, hypothetical decision variable. A set of 40 sentences with 1083 phonemes and 1541 phonetic segments was segmented seven times. Internal thresholds were varied to produce segmentations performing over a wide range of the M/E trade-off. Internal threshold were varied to produce segmentations performing over a wide range of the M/E trade-off. The resultant error rates are plotted on a normal-normal grid in Figure 3. A least-squares regression fit a line with slope=1.00 (Noise standard deviation / Signal standard deviation), and x-intercept=2.25 (the separation of the means of the two distributions).

![Figure 2: Typical ROC Plot](image)

The line is the ROC of the segmenter with this particular parametric representation, "correct" segment definitions, etc. for all M/E trade-off tuning.

A second experiment, run with different input parameters, gives a measure of confidence in the $d'$ estimates. When the 40 sentence were divided into 10 groups, and estimates of $d'$ made for each group, the 95% confidence interval in $d'$ was found to be $\pm 0.14$ (i.e. the estimate from 4 sentences fits the $d'$ computed from all 40 within the confidence interval). Since this interval is considerably smaller than the differences found between segmentation programs, or between input parametric...
representations, we feel such comparisons are meaningful using $d'$. For example, four representation of the signal were tested [Gol75] yielding $d'$ values from 1.29 to 2.38. Furthermore, published results of two other segmenters [Bak75, Dix75] allowed estimates of $d'$ to be made of 2.26 and 2.73. The ordering of all these segmentation runs agrees very well with our intuitions about the programs, as well as with the (somewhat sparse) results of speech recognition use of them.

**Conclusions**

We believe that the model provided by Signal Detection Theory, and particularly the $d'$ parameter of that model, offers a highly suitable and attractive measure of segmentation efficacy, and a means of better understanding the M/E trade-off. Different segmenters, conforming to needs of different speech recognition systems, can be quantitatively compared, and their performance under different "tuning" of the M/E trade-off can be predicted.

**References**


AN APPLICATION OF CONNECTED SPEECH TO THE CARTOGRAPHY TASK

Gary Goodman, Don Scleva, and Bruno Beek

Department of Computer Science
Carnegie-Mellon University, Pittsburgh, Pa. 15213

ABSTRACT

This paper summarizes initial development of a system for visual and verbal data acquisition in the cartography task. Visual input and output is provided by a graphics tablet in conjunction with a graphic display terminal. Verbal input consists of sequences of commands and map feature descriptors which are recognized by the Harpy speech recognition system. An important and interesting aspect of this research involves the design and analysis of vocabularies and grammars for tasks of this nature.

INTRODUCTION

The cartography task is an interesting application in man-machine communication combining several forms of input. It is a practical task, used daily by map makers, and has a well defined protocol. In this task features are selected and traced from a map and further described by a sequence of descriptor phrases. The graphical input is obtained using an x-y coordinate input device, such as a graphics tablet. In currently used cartography systems, the textual descriptions are entered via keyboard. This paper describes the VICS system, a cartography system in which connected speech input replaces keyboard input. VICS stands for Voice Input Cartography System.

This project was undertaken because it represented a practical and useful application for speech input of sufficient size to be interesting, but small enough to be feasible. An important aspect of the research is the pursuit of a methodology for language design for man-machine voice communication. Interaction with the user is sufficiently flexible to allow the investigation of several different methods of language structure, from little or no constraint to highly constrained sequences. Further, since a smoothly interacting system with adequate response would have immediate application, there is great potential for study of the many problems associated with man-machine systems.

In order to combine voice and graphical input in a practical system, one needs 1) a speech recognition system capable of recognizing utterances from a language as complex as required by the task, 2) a graphics system sufficiently flexible to allow graphical input and visual feedback as necessary for the task, and 3) some method of interfacing them so the system behaves in a way which appears as natural as possible to the user. Two systems designed at Carnegie-Mellon University provide the necessary tools. The Harpy speech recognition system [Lowerre, 1976 and 1977] recognizes live voice input with the ability to apply grammatical constraints. The SPACS graphic system [Greer, 1976], originally built as a stand alone interactive graphics editor, uses a tablet input device in conjunction with a graphics display terminal. Its capabilities include free-hand line drawing and the ability to create tables, flow charts, logic diagrams, and other schematic diagrams. The interfacing problem is solved by the use of a task module in the Harpy system.

Other systems for speech input are available. The isolated word recognition system developed by Threshold Technology [Martin, 1975] and the Bell Labs connected speech system [Sambur and Rabiner, 1976] are accurate systems, but at present lack the desired flexibility in structuring the grammar. Other successful systems, such as Hearsay-II [Ermans et al., 1976 and Lesser et al., 1975], HWIM [Woods, 1976], and the IBM system [Johnson et al., 1975 and Bahl, et al., 1976], have much more elaborate control structures and were designed for larger tasks. The overhead involved in those systems is considered unacceptable for tasks such as this one.

THE HARPY CONNECTED SPEECH RECOGNITION SYSTEM

In the Harpy system the recognition process consists of searching for the best path through a precompiled network, given the acoustic evidence present in the utterance. The search scheme uses heuristics to reduce the number of paths considered, resulting in only a few "best" paths being searched in parallel. The recognized utterance

The authors wish to acknowledge Raj Reddy for assisting in the overall design of the system, Bruce Lowerre for creating dictionaries and designing the task module interface, and Ken Greer for making the necessary changes to the graphics editor.

This work was supported in part by the Defense Advanced Research Projects Agency (F44620-73-C-0074) and is monitored by the Air Force Office of Scientific Research.

is then turned over to a task module, a program whose purpose is to respond to the user in a way appropriate to the task. The simplest task module would simply type the recognized utterance on some output device such as a CRT. In more complicated cases, such as the AI abstract retrieval task, the task module would extract the intent (meaning) of the utterance, consult its data base, and supply an appropriate response, eg. "There are 17 articles on that topic".

The recognition process in Harpy uses a precompiled network which integrates syntactic, lexical, and word juncture knowledge. Syntactic knowledge is specified by a context-free grammar defining the input language. Lexical information is embodied in a symbolic phonetic dictionary containing pronunciations and alternate pronunciations for each word in the task language. Word juncture phenomena are characterized by a set of juncture rules giving alternate pronunciations of word beginnings and endings based on the context of adjacent words. All these sources of knowledge serve as inputs to a program which compiles a network representing all possible pronunciations of all possible input utterances.

The acoustic evidence used to determine the best path in the network is obtained by segmenting the input and extracting LPC parameters for each segment. These LPC parameters are matched with phone templates to produce a metric between the segments and the symbols (phones) associated with network states. This metric is in the form of the probability that the segment is an instance of the symbol. Probabilities are learned from exemplars taken as training data.

Creating a new task for Harpy consists of defining the language, training the phone templates, and specifying the task module. To define the language one first specifies the grammar for the input language and then obtains from it a list of all the words used in the language. For each of these words a description of its allowed pronunciations is entered into the dictionary. These descriptions are in terms of a standard set of phones.

THE VICS SYSTEM

The task module coordinates verbal and graphical input and controls discourse with the user. Figure 1 shows a user at the graphics display interacting with the VICS system. Verbal input is a sequence of words or phrases which may be commands for the task module or descriptions of the map feature. Graphic input is via a graphics tablet x-y sensor. There are two graphic input modes: point mode and trace mode. The user enters point mode by saying "point" or "point mode". In this mode the user defines one position on the map corresponding to the location of an feature such as a well, pond, or water tank. For more complicated and larger features, such as lakes, islands, shorelines, and harbors, trace mode is entered. In this mode the x-y sensor position is continuously monitored giving a graphical description consisting of a set of lines. In both modes the graphical description is displayed on a CRT for visual verification. Figure 2 shows how the graphics display appears after the user has traced an intermittent stream. At this point the user describes the feature verbally according to the vocabulary and grammatical structure. The display after verbally describing the stream is shown in figure 3. Figure 4 shows the display after another trace-describe cycle describing an adjacent pond. After the description is complete the user may reject or accept it using voice commands. If accepted, the description is stored for future use.
The vocabulary for the VICS system consists of task module commands and words or phrases for describing the map feature. These phrases are familiar content phrases used by map makers and are contained in a document produced jointly by the Department of Commerce and the Department of Defense [U.S. Dept. of Commerce, 1975]. Some examples from this document are shown in figure 5. We have chosen, in cooperation with RADC, 591 phrases from this document. A 77 phrase subset, used in the description of features in the class drainage, has been chosen for test purposes. The first few lines of the task dictionary are shown in figure 6.

The choice of grammar is dictated both by the nature of the task, eg, the description of map features, and by the desired user interactions, eg, user commands. A factor relating to user satisfaction is grammatical constraint. A grammar with high constraint implies, in general, fewer recognition errors and therefore greater satisfaction. Care must be taken, however, to not constrain the grammar so much that interaction becomes unnatural for the user.

There are several ways of imposing grammatical structure on the phrases which make up the verbal description. We are currently experimenting with two methods, which represent the extremes of constraint. The first method is unstructured where any phrase may be followed by any phrase, i.e. not constraint. This gives the user complete freedom to describe the map feature in the most natural way. Since there are other methods which allow the naturalness but also have some constraint, this mode is used for the investigation of what accuracies are attainable in the worst case. If accuracy is adequate in this case, then it will be more than adequate in situations with greater constraint. The second method is complete constraint, or tree-like, where each description is represented by a path from the root of a tree to the one of its leaves. In this method menus representing all possible choices at a node of the tree are shown to the user. After one of these possible utterances is spoken and recognized, the system uses the recognized phrase to move to the appropriate new node and presents the next menu according to the choices at the new node. The first menu (top or root node) presented to the user is shown in figure 7. This menu describes the major classification of the feature being described. Each menu contains "restart" and "backup" as possible verbal commands. Restart means go back to the root node of the grammar tree and start the current description again. Backup means move back to the previous node of the tree. This command be used when an error was encountered. As the description is entered verbally, the recognized phrases are placed on the display, near the graphical description, for verification. The final menu contains "ok", "accept", "backup", and "restart" as possible inputs.
Neither of these methods for grammatical structure is viewed as being entirely appropriate to the task. Another method which we intend to investigate is an unordered tree-like scheme where each description is a path thru a tree structure, but phrases can be entered in any order and the user need supply only enough of the path to make it unique. A variation allows features to have certain default attributes, eg. "river" implies "natural". The default would be used to construct the unique description unless some other countering choice, such as "man-made" were mentioned.

The VICS system was first demonstrated in September 1976 after less than a man-month of effort. Recent emphasis has been on investigation of various language studies. While no extensive accuracy studies have been made, it appears that 98% accuracies are attainable with moderate grammatical constraint.

DISCUSSION

The research reported represents initial progress toward the development of a system combining visual and verbal data acquisition for the cartography task. We have shown that a new task can be constructed in a relatively short time. The system is still in its infancy and many interesting research problems remain in vocabulary analysis and design, language analysis and design (Goodman, 1976), effects of language structure and user discourse, interactive techniques, and the investigation of recognition characteristics under various vocabulary and grammatical complexities. We look forward to pursuing these areas of research.

REFERENCES


Dynamic Speaker Adaptation in the Harpy Speech Recognition System

Bruce T. Lowerre

Department of Computer Science
Carnegie-Mellon University, Pittsburgh, Pa. 15213

ABSTRACT

The Harpy speech recognition system works optimally when it "knows" the speaker, i.e. when it has learned the speaker dependent characteristics (speaker dependent parameters) of the speaker. There are three methods of learning these parameters. One way is to generate them from a set of training data which covers all the allophones that occur in the task language. A second method is to use "speaker independent" parameters with a resulting reduction in accuracy performance. Since it is inconvenient for a "new" speaker to say a set of training data before using the system and the low accuracy with speaker independent parameters is unacceptable, a third method has been devised to allow the system to dynamically learn the speaker dependent parameters while using the system. The new speaker starts with a set of speaker independent parameters. These parameters are then altered after correct recognition (which can be forced if necessary) to match the spoken utterance.

INTRODUCTION

This paper presents a method by which the Harpy is able to adapt to non-familiar speakers. The first section gives a short description of the Harpy system, its data structures, and its current performance. The following sections discuss the speaker variability issue and several approaches that have been taken towards its solution. These approaches include speaker specific tuning, speaker independent tuning, and dynamic speaker adaptation. The last section discusses how these averaging techniques can also be used in isolated word recognition systems.

THE HARPY SYSTEM

The Harpy system is the first system to be demonstrated with a vocabulary of over 1000 words. The system was demonstrated at the completion of the five year Advanced Research Projects Agency (ARPA) speech research project in September, 1976. It had a sentence accuracy, across five speakers (both male and female), of 91% and ran in about 30 MIPS (a MIPS is millions of machine instructions executed per second of speech). Since that time, improvements have been made in the speed of the system. The current system runs in less than 7 MIPS. The system is a recognition system rather than an understanding system since it uses no semantic knowledge about the task in decoding the utterance. However, there are several other sources of knowledge in the system such as syntactic, lexical, word juncture phenomena, speaker characteristics, and intrinsic phoneme durations (see Lowerre, 1976 for complete details).

In the Harpy system, the syntactic, lexical, and word juncture knowledge are combined together into one integral network representation similar to that of the Dragon system (Baker, 1975). The syntactic knowledge is specified by a context-free set of production rules for the task language. A dictionary is used to represent the lexical knowledge. The dictionary contains symbolic phone spellings and specifies alternate pronunciations of the words in the task language. Word juncture rules are also included in the network to account for inter-word phonetic phenomena. The network consists of a set of states and inter-state pointers. Each state has associated with it phonetic, lexical, and duration information. The pointers indicate what states may follow any given state. Two special states in the network, the initial state and the final state, indicate the starting point and ending point for all utterances respectively. The network is, therefore, a complete representation of all possible pronunciations of all possible utterances in the task language. This network is used to guide the recognition process.

The recognition process of the Harpy system is based on the Locus model of search. The Locus model rejects all but a narrow beam of paths around the most likely path through the network. These "best" paths are searched in parallel with one pass through the speech data and therefore does not require backtracking.

The following is a short description of the recognition process: The utterance is digitized at 10 KHz. This continuous signal is segmented into consecutive acoustically similar sound units (based on distance measures of the data) and autocorrelation values and linear predictor coding (LPC) coefficients are extracted for each segment. The segments are then mapped to the network states based on the probability of match (distance match) of the LPC data and the expected phones of each state. The matching of the LPC's and the network states is accomplished by use of phone templates. The templates contain the idealized parameters for each phone that occurs in the network states and they may be either speaker specific or speaker independent. The metric used for this matching is Itakura's minimum prediction residual error (see Itakura, 1975).

The current system achieves a sentence accuracy of 90.02% and a word accuracy of 94.37% on a 1011 word task and runs in 6.8 MIPSS.

SPEAKER ADAPTATION IN THE HARPY SYSTEM

Speaker variability Speaker variability generally occurs in three forms, dialectic, contextual, and acoustic. Dialectic variability involves changes in the pronunciation of words among speakers. Contextual variability involves changes in word pronunciation due to the context of the words. Acoustic variability results from vocal tract changes among speakers. Either or all types of variability can occur when changing speakers. The Harpy system attempts to recognize these different variabilities and to separate the effects made by each. Dialectic variability is an effect across a broad group of speakers and the variability is encoded into the lexicon. Many dialects can be encoded into the lexicon or different lexicons can be used for different dialects. The current Harpy system uses the "mid-western American" dialect of English. The contextual variability is handled in the word juncture phenomena rules and, to a lesser extent, in the lexicon itself. The acoustic variability is a speaker dependent phenomenon and can be separated from the other types of variability.

Approach to speaker variability Many proposals and attempts have been made, from such groups as SDC, BBN, Lincoln Labs, etc., as to how to handle the speaker variability problem. These proposals include such ideas as vowel formant normalizations, attempts to determine speaker independent characteristics of the speech signal. The Harpy system handles speaker variability by the use of phone templates to capture the vocal tract characteristics. We achieve this by identifying all the unique sounds that occur in the task language (called phones). It is important to realize that these phones may or may not bear a resemblance to what may be usually thought of as a phonetic sound in the English language. For example, there are usually several occurrences of one vowel (allophones) in our set of phones each of which has a unique name. Also, there could be a single phone which represents what is usually thought of as a combination of phones (e.g. the phone "WH" represents the characteristics of the aspiration sound when pair "K W" that occurs together as in the word 'whatch'. Each of the phones used in the Harpy system represents one unique phonetic sound.

Phonetic knowledge in the Harpy system The Harpy system uses a phonetic dictionary (along with word juncture rules) to represent the lexicon of the task language. The spellings in the dictionary are strings of phones (along with a special syntax) which are used to represent primary and alternate pronunciations of the words in the lexicon. The phonetic dictionary is a representation of the actual realizations of the task language words rather than a pronunciation dictionary. A set of speaker dependent phone templates (one per phone) is used to match the symbolic lexicon to the actual acoustic signal. The phones of the lexicon represent all the unique phonetic sounds that occur in the task language. Since the lexicon contains symbolic spellings which are speaker independent mappings of the templates to the phones, the acoustic speaker variability can be handled effectively by using a unique set of templates for each speaker. The templates model speaker dependent vocal characteristics. For example, the dictionary spelling for "CONCERN" is "('K-,'C-,'N,,'E-,'R,,'N,S,I,,'O,,'R-I,)' (K0) (I7,I3) N S ER (N,D,X)". Optional paths are enclosed within parenthesis and are separated by commas (the "0" represents the null option). The spelling is interpreted as either a voice bar ("-"), followed by an optional silence ("-") or just a silence, followed by an optional "K", followed by either a "IH7" or "IH3", followed by an "N", followed by a "S", followed by an "ER", followed by either an "N" or "DX". See McKeown, 1977, for an example network.

Averaging of template exemplars The success of the speaker dependent phonetic templates depends on the ability to average many exemplars of each phone together to generate each template. This averaging enables the automatic cancelation of errors (provided they are small). Since the template is an average, there is no need to find the single "ideal" exemplar that best fits all occurrences of the phone. The averaged template will usually match all exemplars of the phone in the training data to a high degree of accuracy. If a match of an exemplar in the training data is too far from the average template, then this indicates a missing phone.

The metric used by the Harpy system is Itakura's minimum prediction residual error of the LPC data. A method was needed to average samples together that could be used for generating the templates for this metric. The method we use is to sum the autocorrelation data of the samples that are used in generating the template. The justification of this is that the LPCs are independent of the number of autocorrelation samples that are used to generate them. The obvious danger is that non-similar sounds may be averaged resulting in a poor spectrum. This is a real problem and is handled by a semi-automatic procedure for generation of the phones, templates, lexicon, and word juncture rules described below.

Speaker specific tuning The phones, templates, lexicon, and word juncture rules are generated from a set of training data that contains occurrences (and hopefully all contexts) of all the words in the lexicon. A semi-automatic iterative procedure is used to generate (or more precisely, update) these knowledge sources. There is a "chicken-egg" problem with this iterative procedure in that the data sources must already exist in order to update them. The generation of the initial knowledge sources is a tedious manual bootstrapping procedure. The training data must be carefully hand labeled (both at the word level and the phone level) and initial guesses are made about what phones, word spellings, juncture rules, etc. are needed. This manual effort is the main bottle-neck for developing larger vocabulary systems. Automatic methods must be developed before larger systems can be attempted.

The following is the semi-automatic procedure used to update the data sources: The Harpy system is run in a forced recognition mode with a previously generated set of templates (which can be from some other speaker) to produce a parsing of the phones to the acoustic data. This
forced recognition can be done either by using a unique network for each utterance (which represents only one utterance) or by considering only paths in a large network that represent each single utterance. The parsings generated from the forced recognition runs are used to locate the autocorrelation data for the averaging of the templates. After the averaging is completed, a new set of templates is generated and used to again run the training cycle. This cycle is run several times until the templates converge. If the templates do not converge, then this indicates an error in either the lexicon or word juncture rules or a missing phone which must be manually analyzed and corrected.

**Speaker independent tuning** The speaker dependent templates are an averaging of many phone exemplars for each template. Since there is a unique set of templates for each speaker, they capture the individual vocal tract characteristics. This idea of capturing vocal tract characteristics by the use of templates can be extended to multiple speakers. When a number of these speaker dependent sets of templates are generated, another set of templates can be generated from all of them by a similar averaging technique. This set of templates, since they are an averaging of several speakers, will be speaker independent. The performance with speaker independent templates will of course be lower than with the speaker dependent templates. For example, one experiment done with connected digits gave the following result: Ten speakers (including males and females) were used to produce ten speaker dependent sets of templates. The average word accuracy for all ten speakers (when tested on the speaker dependent templates with a total of 1000 three words per speaker) was 98%. These ten template sets were then used to generate a set of speaker independent templates. These same ten speakers plus ten new speakers were then tested with the system. The word accuracy for all 20 speakers (on 1200 utterances) was 93%. An interesting observation is that there was no significant difference between the accuracies of the ten speakers whose templates were used to generate the speaker independent set and the ten new speakers.

**Dynamic speaker adaptation** The high error rate (72%) with the speaker independent templates makes this alternative to the handling of acoustic variability unacceptable. Further, the training cycle mentioned earlier to generate the speaker dependent templates is inconvenient do to the large amount of training data needed and is computationally expensive. A third scheme was devised which allows a new user immediate use of the system but also allows for the speaker dependent vocal characteristics. This idea of capturing vocal tract characteristics. This idea of capturing vocal tract characteristics by the use of templates can be extended to multiple speakers. When a number of these speaker dependent sets of templates are generated, another set of templates can be generated from all of them by a similar averaging technique. This set of templates, since they are an averaging of several speakers, will be speaker independent. The performance with speaker independent templates will of course be lower than with the speaker dependent templates. For example, one experiment done with connected digits gave the following result: Ten speakers (including males and females) were used to produce ten speaker dependent sets of templates. The average word accuracy for all ten speakers (when tested on the speaker dependent templates with a total of 1000 three words per speaker) was 98%. These ten template sets were then used to generate a set of speaker independent templates. These same ten speakers plus ten new speakers were then tested with the system. The word accuracy for all 20 speakers (on 1200 utterances) was 93%. An interesting observation is that there was no significant difference between the accuracies of the ten speakers whose templates were used to generate the speaker independent set and the ten new speakers.

**Dynamic speaker adaptation** The high error rate (72%) with the speaker independent templates makes this alternative to the handling of acoustic variability unacceptable. Further, the training cycle mentioned earlier to generate the speaker dependent templates is inconvenient do to the large amount of training data needed and is computationally expensive. A third scheme was devised which allows a new user immediate use of the system but also allows for the speaker dependent vocal characteristics. This idea of capturing vocal tract characteristics by the use of templates can be extended to multiple speakers. When a number of these speaker dependent sets of templates are generated, another set of templates can be generated from all of them by a similar averaging technique. This set of templates, since they are an averaging of several speakers, will be speaker independent. The performance with speaker independent templates will of course be lower than with the speaker dependent templates. For example, one experiment done with connected digits gave the following result: Ten speakers (including males and females) were used to produce ten speaker dependent sets of templates. The average word accuracy for all ten speakers (when tested on the speaker dependent templates with a total of 1000 three words per speaker) was 98%. These ten template sets were then used to generate a set of speaker independent templates. These same ten speakers plus ten new speakers were then tested with the system. The word accuracy for all 20 speakers (on 1200 utterances) was 93%. An interesting observation is that there was no significant difference between the accuracies of the ten speakers whose templates were used to generate the speaker independent set and the ten new speakers.

**DISCUSSION**

**Summary** In this paper we have considered several sources of variability in the connected speech signal, i.e. dialectic, contextual, and speaker dependent variability, and described how the Harpy system attempts to cope with all these sources of variability. The dialectic and contextual variability are encoded into the lexicon and word juncture rules. The speaker dependent sources of variability are handled by averaging phone parameters (i.e., the autocorrelation coefficients, not the LPC's) from among several exemplars of a given phone by the same speaker (for speaker specific templates) or from many speakers (for speaker independent templates). In the case of dynamic adaptation, a set of speaker independent templates are used initially and the system automatically alters the templates during use to adopt to the specific speaker.

It appears straightforward to adopt the above techniques for isolated word recognition systems also. Given several training samples of the same word, one can align the speech signal by dynamic programming techniques and average the autocorrelation coefficients as in the connected speech case. Since this averaging would be independent of word representation used, i.e. whether one uses segmentation and phone templates to represent words or the conventional brute force word templates, one can still use the above averaging technique to generate better templates.

**REFERENCES**


USE OF SEGMENTATION AND LABELING IN ANALYSIS-SYNTHESIS OF SPEECH

Raj Reddy and Robert Watkins

Department of Computer Science
Carnegie-Mellon University, Pittsburgh, Pa. 15213

ABSTRACT
We have been attempting to produce further bandwidth reduction in LPC based analysis-synthesis techniques by using the segmentation and labeling algorithms used in the Harpy and Hearsay-II systems. Preliminary results indicate that a factor of 3 to 5 further reduction in bandwidth might be possible using segmentation and labeling in conjunction with LPC vocoders.

INTRODUCTION
An important application of speech analysis-synthesis is digital voice transmission. Real-time transmission at low bandwidths can only be achieved through efficient analysis and encoding techniques. While present analysis methods, based on signal processing techniques, have been used successfully to obtain bandwidth reductions of over an order of magnitude, further improvement is possible if higher level properties of speech are also taken into account. In this paper, we demonstrate how segmentation and labeling, two techniques commonly used in connected speech recognition, can be applied to vocoder systems as a means of improving coding efficiency.

In the remaining sections, we describe segmentation and labeling techniques, and their use in vocoder systems. Results from three different vocoder simulations based on these techniques are presented and evaluated. We then consider some of the practical aspects of real-time speech transmission using these methods. Finally, the advantages and disadvantages of high level speech processing as applied to vocoders are discussed.

APPROACH
Our goal in this study was to evaluate the usefulness of segmentation and labeling as techniques for improving vocoder coding efficiency. To accomplish this, two vocoder simulations using each of these techniques separately, and a vocoder simulation which combined the techniques, were run. The results were compared with those obtained using conventional parameter encoding methods, and evaluated in terms of bandwidth reduction and quality of synthetic speech.

Of the several techniques for speech analysis that exist, this paper considers only those based on the autocorrelation method of linear prediction. A complete vocoder simulation based on this technique has already been developed by Markel and Gray [Markel and Gray, 1974]. Since a detailed discussion can be found in this reference, we consider only those aspects relevant to the bandwidth problem here.
this is not always the case. Near segment boundaries, the vocal tract is changing, and cannot be assumed to have constant resonances. To eliminate possible errors due to these changes, the parcor coefficients are computed at the segment midpoint. Once calculated, those coefficients, along with the segment duration, are transmitted. Pitch and amplitude are then extracted from each frame in the segment, encoded, and transmitted. Thus, with this scheme, pitch and amplitude are still transmitted at the constant rate of once per frame, but parcor coefficients are transmitted at the rate of once per segment, which is not necessarily constant.

Except at its boundaries, the same set of parcor coefficients is used to synthesize speech for each frame within a segment. Near boundaries variation in the parcor coefficients due to vocal tract changes must be taken into account. Good results have been obtained using simple linear interpolation. For most segments it is adequate to interpolate over 5 centiseconds, from 2 centiseconds before the segment boundary, to 2 centiseconds after. For shorter segments, indicating rapid changes in the spectral structure, interpolation is done from the segment midpoint.

The effects of parcor coefficient interpolation are illustrated in Figure 1. This figure shows the spectral envelopes for a transition from one segment to the next. The darker curve represents the conventional synthetic speech, the lighter represents the speech synthesized from interpolated parcor coefficients. Note that although the peak amplitude and shape differ slightly, the peak locations are nearly identical.

Figure 3 shows a digital spectrogram for the utterance “The area I’m interested in is understanding,” synthesized with the Segment-coder. For comparison purposes, a digital spectrogram of the utterance synthesized with conventional methods is shown in Figure 2. As can be seen, the spectrograms resemble each other closely. In informal listening tests, the synthetic speech generated with parcor parameters transmitted only once per segment was nearly indistinguishable from that generated with parcor parameters transmitted every frame.

The degree of improvement in coding efficiency will vary from system to system, depending on frame rate, and the precision to which each of the parameters are encoded. For the system described earlier, a total of \((6+5+64) \times 100 = 7500\) bits/sec are required to encode the analysis parameters. Using segmentation, pitch period and amplitude information are still transmitted for each frame, but parcor coefficients are transmitted only once per segment. Another parameter, the segment duration, must also be transmitted with each segment. Allocation of 4 bits for this parameter allows for segment lengths up to 16 centiseconds. Segments exceeding this length are rarely encountered, and can easily be split into multiple segments.

On average, the segmentation algorithm produces 15 segments per second of speech. Thus, the total bit rate needed for this scheme is \((6+5) \times 100 \times (4+64) \times 15 = 2120\) bits/sec. This represents improvement by a factor of about 3.5 over the conventional method.

Reductions of this order have been obtained in conventional vocoders by using reduced frame rate. Rather than transmitting one frame per centisecond, these vocoders might transmit one frame every 3 centiseconds, indiscriminately ignoring data between frames. This has a smoothing effect which results in the loss of short events that may be perceptually significant. Thus, the overall quality of the synthetic speech should be lower than that obtained with the segmentation scheme.

A second technique makes use of an assumption that all speech, regardless of its complexity, can be formed by combinations of a small number of basic sounds. The VORTRAX speech synthesizer is an example of one such system based on this assumption. Associated with each sound is unique formation of the vocal tract, and associated with each vocal tract formation is a set of parcor coefficients. If speech at each analysis frame can be identified and classified as one of these sounds, then it might only be necessary to transmit a label identifying the sound, rather than the entire set of parcor parameters. Since the number of sounds is small, significantly fewer that 64 bits are needed to encode the label, and an improvement in coding efficiency would result.

Prior to the development of a vocoder simulation, the properties of each sound must be determined and represented in a format usable by the system. A procedure to accomplish this was developed for use with the Harpy system(Lowerre, 1976). Segments from several utterances, spoken by a particular speaker, are identified and grouped according to their sound class. Autocorrelation coefficients for each segment are computed and averaged over all segments in the same class. For each averaged autocorrelation sequence, hereafter referred to as a template, linear prediction coefficients, parcor coefficients,
Figure 2. Digital spectrogram of synthetic speech for the utterance "The area I'm interested in is understanding," generated using conventional signal encoding techniques.

Figure 3. Spectrogram of the synthetic speech generated by the Segment-coder.

Figure 4. Spectrogram of the synthetic speech generated by the Label-coder.

Figure 5. Spectrogram of the synthetic speech generated by the Segment-label-coder.
and b-coefficients [Itakura, 1975] are computed. This information is made available to both the transmitter and receiver portions of the vocoder.

The task of the vocoder, then, is to determine, for each analysis frame, which template best matches the speech signal. The LPC matching technique developed by Itakura [Itakura, 1975] has been used for this purpose. A distance metric is applied between each frame and all templates. The best template, in terms of minimum distance, is selected. A label identifying this template, along with pitch and amplitude information, is transmitted. At the receiver, a simple table lookup, using the label as an index, is performed to determine the parcor parameters of each frame. From this point on, synthesis proceeds normally.

Figure 6 shows the spectral mismatch between original spectra and the labels assigned to them. The darker curve corresponds to the original speech, the lighter to speech synthesized with the labeling method. The curves illustrate typical spectral errors that occur with the labeling method.

Displayed in Figure 4 is a digital spectrogram of the test utterance, synthesized with the Label-coder. This may be compared with the spectrogram of the conventional synthetic speech in Figure 2. Although the synthetic speech was intelligible, there was considerable distortion. We believe that this can be eliminated by changes in the template generation and matching algorithms.

Again, the bandwidth reduction afforded by this technique depends on how accurately the parameters are quantized, but in this case it is independent of frame rate. As before, we base our comparison on the system described earlier. For this system, a total of 6+5+64+75 bits/frame are needed to encode the speech. For the system with labeling, a label, along with the encoded pitch and amplitude, is transmitted for each frame. To uniquely identify each of the 96 templates used in this simulation, 7 bits were allocated for the label. Thus, with labeling, only 6+5+7+18 bits are needed to encode each frame. This represents a bandwidth reduction by a factor of 4.

SEGMENT-LABEL-CODER

Clearly, if only one set of parcor coefficients is necessary to encode the spectral structure of each segment, and if each spectral structure can be identified by a label, then it should be possible to transmit only one label per segment. Examination of the analysis parameters from the labeling system reveals that this is indeed the case. Most frames within a segment were found to be labeled with the same label. Those that were not, were labeled with an acoustically similar label. Once again, a vocoder simulation to test the hypothesis was developed.

The separate use of segmentation and labeling has already been discussed. This system is merely a combination of the two previous ones. After segmentation, the labeling algorithm is applied at the midpoint of each segment. The label which best characterizes the spectral properties of that segment, and the segment duration are encoded for transmission. Of course, pitch and amplitude information are still transmitted for every frame. Received labels are first used to determine the parcor parameters associated with each segment, which in turn are used to synthesize speech for all frames within that segment. Interpolation at segment boundaries is carried out as previously described.

The spectrogram for speech synthesized by this system is shown in Figure 5. Note its similarity to the spectrogram for speech synthesized by the labeling system. This is to be expected, since it was already determined that segmentation causes no significant degradation. The differences between this and the other spectrograms are due to degradation introduced by labeling.

Again, we calculate coding efficiency by comparison with the conventional system. With this encoding scheme, a total of 6 bits for pitch, and 5 bits for amplitude are transmitted every frame. An additional 4 bits for segment duration, and 7 bits to identify each template are transmitted for each segment. Using a frame rate of 100 frames/sec, and an average of 15 segments per second of speech, a data rate of (6+5)x100+(4+7)x15=1265 bits/second is obtained. This is approximately 5.9 times smaller than the 7500 bits/sec of the conventional system.

DISCUSSION

We have shown that segmentation and labeling can be used as a means of reducing bandwidth in speech analysis-synthesis systems. Since the primary application of such systems is secure voice communications, it is appropriate to mention some of the practical aspects of a vocoder based on these techniques.

A problem arises when the vocoder is converted to real-time operation. Since analysis parameters for each segment are not transmitted until the entire segment has been spoken, it is possible for the synthesizer to complete synthesis of one segment before it receives parameters for
the next. If this happens, a pause in the synthesizer output will occur. To avoid these pauses it is necessary to define a maximum segment duration, and delay the synthesis by this amount. We have already indicated that 16 centiseconds is a reasonable choice for maximum segment duration. If the synthesizer lags the transmitter by this amount, plus an additional 2 centiseconds to allow for interpolation, continuous synthetic speech can be guaranteed. In practice, this is not a serious drawback. Delays of this magnitude are secondary in nature to those normally encountered in satellite-based transmission systems.

From the discussion of labeling it should be clear that both transmitter and receiver must access to the same set of templates. Since the templates vary from speaker to speaker, it is impractical to make them a permanent part of the system. Rather, at the beginning of a conversation, templates for each speaker could be loaded into the corresponding transmitter and transmitted to the connecting receiver. Another possibility would be to use a single set of templates which has been averaged over many speakers. However, lower quality synthesis can be expected with this method.

In addition to the obvious reduction in bit rate, there are other advantages to the use of these techniques. At first, the additional processing needed to segment and classify speech would seem to result in slower vocoder operation, however this is not the case. Once the segments are known, the time consuming autocorrelation analysis need be performed only once per segment. Thus, overall vocoder operation is actually faster. Furthermore, since gross segment classifications are obtained during the segmentation process, specialized processing, depending on the segment class can be performed. For example, silences can be dismissed with no processing, and low coefficient LPC analysis can be performed for fricatives. This should result in a more accurate synthesis.

The main point should be clear: through the use of specialized knowledge of the nature of speech, and higher level signal-to-symbol transformation techniques, incrementally better vocoders can be obtained. We have demonstrated two steps in this progression. The first was the transition from systems based solely on spectral analysis, to a system that combined knowledge of segments with spectral analysis. The next step was the use of labeling in addition to segmentation to give even further bandwidth reduction. As speech recognition systems evolve, better and better encodings will become practical. Eventually, it should be possible to transmit syllable sized units.

Finally, improvement in coding efficiency is obtained at the expense of generality. As more specialized knowledge of speech and language is used, the variety of sounds that can be transmitted is reduced. At the lowest level is the system that transmits sampled speech directly. With this system, arbitrary sounds can be represented accurately. The step to conventional vocoders limits those sounds which can be transmitted to speech. Greater restrictions occur as the vocoder becomes more and more language-oriented.

CONCLUSIONS

We have presented two techniques, based on algorithms developed for the Hearsay and Harpy speech recognition systems, which use knowledge about speech phenomena, to yield reductions in vocoder bandwidth. While the degree of improvement varies from system to system, typical reduction factors ranging from 3 to 4 can be expected from each method. Furthermore, improvements by a factor of 5 or more can be realized if the techniques are combined.

Use of segmentation caused no noticeable degradation in the synthetic speech quality. With labeling, considerable degradation occurred, however it is felt that this can be eliminated with better templates.

Some of the practical aspects of vocoder implementation based on these techniques, along with the advantages and disadvantages to the use of specialized knowledge, were discussed. On the basis of arguments presented then, we believe that speech analysis-synthesis using segmentation and labeling is worthy of further research.

REFERENCES

H. Q. Goldberg and R. Reddy (1976), "Feature extraction, segmentation, and labeling in the Harpy and Hearsay-II systems," JASA, 60, 511.


A HALTING CONDITION AND RELATED PRUNING HEURISTIC
FOR COMBINATORIAL SEARCH

David Jack Mostow

Computer Science Department
Carnegie-Mellon University

May 1977

ABSTRACT

Many combinatorial search problems can be viewed within the "Chinese restaurant menu selection paradigm" of "choose one from Column A, one from Column B, ..." A solution to such a problem consists of a set of selections which are mutually consistent according to some set of constraints. The overall value of a solution is a composite function of the value of each individual selection. The goal of the search is to find the best (highest-rated) solution. Examples of such search problems occur in the domains of speech understanding, vision, and medical diagnosis.

This paper describes a search-pruning heuristic and a halting condition which are conservative in that they will not miss the best solution by pruning it out of the search or by terminating the search before it is found. The method exploits information about already-found solutions in order to prune the search and decide when to terminate it. An implementation of the halting condition and pruning heuristic within the Hearsay-II speech understanding system is described and evaluated, and the conditions governing its applicability and performance are discussed.

INTRODUCTION: SOME EXAMPLES

A frequently-occurring problem in AI involves finding the best combination of choices for a set of interdependent multiple-choice decisions. The possible combinations form a combinatorial search space. Each decision corresponds to a data element which can be labelled (explained, interpreted) in several alternative ways, some of which may be preferable to (more appropriate than) others. Legal solutions (combinations of labels) must satisfy certain domain-specific consistency constraints governing the interdependencies between the various elements to be labelled.

One example of combinatorial search occurs in the domain of speech understanding. A spoken utterance can be viewed as a set of contiguous points in time. The combinatorial search task of a speech understanding system is to label each time interval with the word apparently spoken during that interval. Several labels may appear plausible due to the uncertainty of the speech signal and the word recognition process [7]. A solution consists of a transcription of the utterance, i.e., a sequence of word labels, which is syntactically and semantically consistent. The credibility (probability of correctness) of such a solution depends on the overall goodness of fit between the labels and their time intervals.

Another example comes from the domain of vision. The contour detection
problem can be described as follows: given a scene represented by an array of pixel gray levels, label each pixel with a vector corresponding to the apparent intensity gradient at that point in the image [9]. A consistent interpretation of the scene assigns parallel gradients to contiguous pixels on a contour and null gradients to pixels in the interior of a region. The accuracy of an interpretation depends on the overall degree to which the labels match the visual data they attempt to describe.

A third example can be found in the domain of medical diagnosis. Here the data elements to be explained are the patient's symptoms. A diagnosis provides consistent explanations for all the symptoms. The plausibility of a diagnosis depends on the overall plausibility with which the individual symptoms are accounted for [1].

**PROPERTIES OF COMBINATORIAL SEARCH**

Let us now examine these search problems in order to discover common properties which can be exploited in designing halting conditions and pruning heuristics. In each example, the set of data elements (points in time, pixels, symptoms) to be explained or labelled is known at the beginning of the search. (Actually, this assumption does not hold for systems like MYCIN which collect data during the course of the search. However, as we shall see, it is sufficient for the set of elements to be determined anytime before the first solution is found.)

A partial solution consists of consistent explanations for a subset of the elements. Combinatorial search algorithms typically extend and combine such partial solutions. In fact, each step in the search can be characterized as examining a collection of partial solutions $I_1, \ldots, I_k$, and then possibly creating a new partial solution $I'$. We can use rating information about partial solutions in order to decide when to halt the search once some solution has been found. For example, suppose we examine the ratings of all existing partial solutions and conclude that none of them can be extended into a complete solution rated higher than the best one found so far. Under this condition, it is safe to halt the search; the best solution found is the best one possible. This condition is the desired conservative halting condition.

A similar technique can be used to prune the search. If a partial solution cannot possibly be extrapolated into a complete solution superior to the best existing one, it can be rejected -- i.e., all efforts to extend it or combine it with other partial solutions can safely be abandoned. This pruning heuristic is conservative but also rather weak. A more powerful heuristic depends on certain properties of the function used for rating solutions. Let us consider this function in more detail.

**THE RATING FUNCTION**

A complete solution consistently explains all the elements and is rated according to how well each element is explained. I.e., if the rating function $R(I, S)$ measures how well the interpretation $I$ explains the elements of the set $S$, then $R(I, S) = \sum_{e \in S} R(I, e)$, where $R(I, e)$ measures how well $I$ explains the element $e$. $R(I, S)$ is assumed to be an increasing function of the terms $R(I, e)$. The interpretation $I$ is a set of labels for the elements of $S$, i.e., for all $e$ in $S$ $e \rightarrow I(e)$. The rating $R(I, e)$ may be context-sensitive, i.e., depend on how other elements besides $e$ are labelled (e.g., its neighbors, if $e$ is a pixel). A considerable

---

1 This condition could be relaxed by allowing complete solutions to label some elements "IGNORED." The rating function would then have to reflect the relative significance of explaining or ignoring a given element, so as to allow meaningful comparison between solutions accounting for different subsets of the element set.
simplification is possible if $R(I,e)$ is context-free, i.e., $R(I,e) = R(l_j(e),e)$, where $l_j(e)$ is the label assigned by $I$ to $e$, and $R(I,e)$ measures the goodness of fit between the label $l$ and the element $e$. In this case, $R(I,S) = \sum f(R(l_j(e),e) \mid e \in S)$. If $f$ is a simple averaging function, then $R(I,S) = \text{Average} \{ R(l_j(e),e) \mid e \in S \}$.

The best solution $I$ maximizes $R(I,S)$ subject to the consistency constraints. Note that the function $R$ may produce higher values if applied to inconsistent interpretations (non-solutions). For example, the interpretation $l_{max}\rightarrow e$ is the highest-rated label for $e$, will in general maximize $R(I,S)$ but is not in general consistent.

**A HALTING CONDITION AND PRUNING HEURISTIC**

We can now precisely define our halting condition and pruning heuristic in terms of the rating function $R$. Let $S'$ be a subset of the element set $S$, and let $I'$ be a partial solution which explains $S'$. Let $I$ be the highest-rated solution found so far during the search.

$I'$ can be extended into a complete (not necessarily consistent!) interpretation $I''$ by assigning $l_{max}(e)$ to every $e$ in $S-S'$. $I''$ is the highest-rated possible complete extrapolation of $I'$. Thus if $R(I'',S) \leq R(I,S)$, $I'$ cannot be extended into a solution better than $I$, and it is safe to reject $I'$ and all its potential extensions. Unfortunately, this condition is too strong and is not often satisfied. A more powerful (but still conservative) pruning heuristic is made possible by assuming that $R$ is context-free in the sense defined earlier.

**A MORE POWERFUL PRUNING HEURISTIC**

Suppose that $R$ is context-free and that a solution $I$ has been found. If a better solution is possible, there must exist a partial solution $I'$ which is locally superior to $I$. $I'$ is locally superior to $I$ over domain $S'$ if $R(I',S') > R(I,S')$. Intuitively, $I'$ explains some subset $S'$ better than $I$ does. If no such $I'$ exists, then $I$ is the best solution, and it is safe to halt the search.

This reasoning requires some justification. We consider all individual element labels to be one-element partial solutions, and assume that they are available to the search algorithm as such. If some potential complete solution $I''$ is better than $I$, then there must exist at least one element $e$ in $S$ such that $R(I'',e) = R(l_{max}(e),e) > R(l_j(e),e) = R(I,e)$. (Otherwise $R(I'',S) \leq R(I,S)$.) This one-element partial solution can be extended step by step into $I''$ so that the partial solution $I'$ at each step is locally superior to $I$. We assume that such a sequence of partial solutions can be found by the search algorithm. This is a strong assumption. Many sequences of partial solutions may lead by stepwise extension and combination to the same solution, but not all will maintain local superiority at each step, and not all may be realizable by the search algorithm being used.

With this caveat, we now observe a happy property of context-free rating functions: once a solution has been found, only partial solutions which are locally superior to it need be considered. All others may be deactivated, i.e., ignored except for combination with active partial solutions.

We can now express a powerful conservative pruning condition: A proposed search operation based on partial solutions $I_1, \ldots, I_k$ may safely be cancelled if

1. Any of the $I_j$ has been rejected, or
2. All of the $I_j$ have been deactivated.
The halting condition is trivial: halt when all pending search operations have been cancelled.

UNDERLYING ASSUMPTIONS

Let us now re-examine some of the assumptions on which this method is based, and the motivations for making them.

(1) The rating function is context-free. Otherwise the local superiority criterion is not valid.

(2) The labels lmax(e) are known at the beginning of the search, and exist as one-point partial solutions. Otherwise correct but low-rated partial solutions might be erroneously rejected. Actually, in order to avoid erroneous rejection, it is only necessary to know an upper bound function Rmax(e) ≥ R(e) for all e in S. The tighter this upper bound, the more partial solutions can be rejected. The Rmax function used by the HWIM speech understanding system is defined by the score of the best phonetic label for each segment [8]. Since this score is based on the best possible word match for each segment rather than on the best actual word match, it provides a poor (over-optimistic) upper bound on the actual word ratings, and produces mediocre results. The Rmax function used in Hearsay-II is defined by the score of the highest-rated hypothesized word at each point in the utterance, and produces good results.

(3) If a potential solution I" is better than an existing solution I, the search algorithm must be capable of building I" in such a way that each partial solution I in the derivation sequence is locally superior to I. Otherwise the derivation of I" might require operating on a set of deactivated partial solutions and be blocked by the deactivation pruning heuristic.

EXAMPLE FROM HEARSAY-II

The Hearsay-II speech understanding system [2] segments a spoken utterance into syllable-length time intervals. These are the elements. The labels for each element are taken from a 1,000-word vocabulary. A complete solution is a grammatical transcription spanning the utterance. A partial solution is a grammatical phrase spanning part of the utterance. The rating function is a simple average of label fit goodness. A (partial) solution I covers a time interval [firstsyl:lastsyl]. Its rating is its average word rating weighted by the number of syllables in each word. I.e., R(I,[firstsyl:lastsyl]) = Average (R(Wj(syl) | A(syl))) where firstsyl ≤ syl ≤ lastsyl, A(syl) represents the acoustic data in the interval syl, Wj(syl) is the word label assigned by I to syl, and R(W | A) measures how closely the word W matches the acoustic data A. R(W | A) is computed by the word verifier [6].

In Hearsay-II, partial solutions are explicitly represented as hypotheses on a global data structure called a blackboard. Search operations are proposed by various knowledge sources which monitor the data on the blackboard. The operations relevant to the discussion at hand are [5]

(1) Recognition: given a sequence of words, parse it and record it as a partial solution if it is grammatical.

(2) Prediction: given a recognized phrase, propose words which can grammatically precede or follow it. Predictions which are rated above a specified threshold by the word-verifier are recorded on the blackboard as one-word hypotheses. Thus prediction
dynamically assigns extra labels to elements, and could potentially violate our earlier assumption that \( R_{\text{max}}(e) \) is known before the rejection pruning heuristic is applied. This is not a problem in practice, however, since most label assignment (word recognition) is done at the beginning of the search or before the first complete solution is found, and predicted words are seldom rated higher than the best previously-recognized words.

(3) **Concatenation:** given two temporally adjacent phrases (or a phrase and a word predicted next to it and subsequently verified), concatenate them and record the result as a partial solution if it is grammatical.

These search operations are performed in order of their priorities, which are assigned by a central focus-of-attention module [3]. The focus module tries to order the search in a best-first manner, and succeeds about 50% of the time on the corpus tested for this paper. This figure seems to increase as the constraints on grammatical consistency are increased, i.e., as the branching factor of the language is reduced. For a best-first search, the best halting policy is to terminate the search as soon as a solution is found. Note that the rejection and deactivation pruning heuristics are inapplicable if this policy is used, since these heuristics do not become applicable until some solution is found.

**EVALUATION**

The deactivation and rejection heuristics were evaluated on a corpus of 34 utterances drawn from a 262-word vocabulary. Utterance length ranges from 3 to 9 words, with an average of 6. The fanout (number of grammatical word successors in each word position) averages 27 for the corpus.

Each utterance was processed in 5 modes. Mode \( N \) uses neither heuristic; mode \( R \) uses rejection; mode \( D \) uses deactivation; and mode \( B \) uses both. In mode \( F \), the system accepts the first solution it finds and immediately halts. The results of the experiment are shown in Table 1.

The simple accept-the-first-solution policy used in mode \( F \) is fastest, but at a considerable cost in accuracy, since it fails for those runs (about 50%) in which the highest-rated solution is not the first one found. A more conservative policy finds these solutions at the cost of extra search, in those runs where the best solution is found first. The correct choice of policy (simple versus conservative) depends on a tradeoff between efficiency and accuracy. Since accuracy is very important in speech understanding, the conservative policy is preferred despite its extra cost.

The heuristics can be evaluated according to two criteria. First, how fast is the best solution found once the first solution is found? As Table 1 shows, deactivation is about twice as powerful as rejection in speeding up this phase of the search. The combination of heuristics is only slightly more effective than using deactivation alone.
<table>
<thead>
<tr>
<th>Mode</th>
<th>N</th>
<th>R</th>
<th>D</th>
<th>B</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>153</td>
<td>157</td>
<td>145</td>
<td>152</td>
<td>152</td>
</tr>
<tr>
<td></td>
<td>71</td>
<td>58</td>
<td>30</td>
<td>26</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>48%</td>
<td>37%</td>
<td>21%</td>
<td>17%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1.2</td>
<td>2.4</td>
<td>2.7</td>
<td>Infinity</td>
</tr>
<tr>
<td></td>
<td>223</td>
<td>215</td>
<td>175</td>
<td>178</td>
<td>152</td>
</tr>
<tr>
<td></td>
<td>241</td>
<td>153</td>
<td>89</td>
<td>52</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>108%</td>
<td>71%</td>
<td>51%</td>
<td>29%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>286</td>
<td>282</td>
<td>253</td>
<td>226</td>
<td>152</td>
</tr>
</tbody>
</table>

### Table 1. Results of experimental evaluation of pruning heuristics.

1 Ideally these numbers should be equal, since the heuristics are not applied until the first solution is found. The variation in these figures is caused by some randomness in the Hearsay-II scheduler in choosing between equally promising search operations.

2 The halting condition is satisfied when no more search operations are pending, or when all the pending operations are considered unpromising by the system.

3 Speedup ratios between different modes are not meaningful here since the set of excluded utterances varies from mode to mode.
Second, how fast is the halting condition satisfied once the best solution is found? An ideal policy would halt as soon as the best solution was found. The deviation of an actual policy from this ideal can be measured by its “halting overhead,” i.e., the amount of extra search performed after the best solution is found. When neither heuristic is used, the halting condition is satisfied in only 12% of the runs (time or space bounds are exceeded in the others) and the halting overhead in those runs is 108%. The rejection heuristic succeeds in satisfying the halting condition in 50% of the runs, with an overhead of 71%. Deactivation leads to halting in 94% of the runs, with 51% overhead. The combination of both heuristics also causes halting in 94% of the runs, but reduces overhead to only 29%.

These results can be summarized as follows:

(1) Deactivation is about twice as powerful as rejection in accelerating the search for the best solution once the first solution has been found. This difference in empirical performance substantiates the intuitive notion that the conditions for deactivating a partial solution are substantially easier to satisfy than the conditions for rejecting it. The combined heuristics speed up this phase of the search by a significant factor (2.7).

(2) The combined heuristics succeed most (94%) of the time in satisfying the halting condition, at a reasonable cost (29%) compared to the time it takes to find the best solution. The large variance in this cost and the failure to satisfy the halting condition in the other 6% of the runs suggest that other techniques are needed to further streamline the search without eliminating the best solution.

DISCUSSION OF APPLICABILITY

What properties of Hearsay-II make this method applicable?

(1) Most of the word labelling is performed before the first solution is found and the heuristics are applied. Seldom is a new word subsequently hypothesized with a rating higher than all the other words in its time interval. Thus the necessary information (the Rmax function) is determined before the heuristics are applied. Exceptions do not automatically cause erroneous rejection, since the Rmax function generally provides a safety margin by overestimating the rating of the best possible solution.

(2) A solution must account for the whole time interval of the utterance, i.e., for every element (syllable). This facilitates the comparison of extrapolated potential solutions with already-found solutions.

(3) The rating function for evaluating solutions is context-free. This facilitates the local comparison of partial solutions with complete solutions.

The context-free property is somewhat counter-intuitive since the consistency criteria are in general context-sensitive, i.e., the admissibility of a label depends on the labels assigned to other elements. The rating function might be expected to rate solutions (consistent interpretations) higher than inconsistent explanations, but a context-free rating function does not have this intuitively satisfying trait. Our approach separates two properties of a solution:

(1) satisfaction of consistency constraints.

(2) goodness of fit between labels and data.
Consistency is considered to be an all-or-none property and is guaranteed by the form of the search. Relative goodness of fit is assumed to be local, rather than context-sensitive. When this assumption approximates the truth, it becomes possible to apply the powerful deactivation heuristic.

**CONCLUSIONS**

Conservative pruning heuristics for combinatorial search have been presented. They operate by eliminating branches of the search which cannot lead to solutions better than those found already. In this respect, they can be thought of as alpha-beta pruning heuristics in a one-player game. The pruning heuristics and associated halting condition have been implemented in Hearsay-II and shown to be effective in the real-world problem domain of speech understanding.

When the object of a search is to find the best solution (not just any solution), there is an important tradeoff between speed and accuracy. The simplest halting policy accepts the first solution found. This policy is correct if the search is always best-first; the closer the search is to best-first, the more attractive such a simple policy becomes. More sophisticated policies increase accuracy at the expense of prolonging the search so as to guarantee that the best solution is not missed.

In a nearly-best-first search, the discovery of a solution changes the purpose of the search from one of finding the best possible solution to one of verifying that there is no better solution than the one found. This change of purpose should be reflected in the search-guiding policies.

The approach described exploits certain assumptions about the search.

1. The search space can be represented by a set of elements (data) each of which can be labelled in several ways. A solution labels all the elements and satisfies specified consistency constraints.

2. A rating function evaluates how well a given label fits a given element. An upper bound on the best label rating for each element should be determined by the time the first solution is found. The tighter the bound, the better the performance of the pruning heuristics.

3. The rating of a solution should be a function of the ratings of its individual labels. It should be possible to compute an upper bound on the rating of the best possible extrapolation of a given partial solution. The tighter the bound, the better the performance.

4. The better the found solution relative to the best (generally inconsistent) interpretation Imax (which assigns each element its highest-ranked label), the more pruning can be done. The stronger the consistency constraints, the lower a solution will tend to be rated compared to Imax, and the worse the performance.

Many search problems (e.g., speech and image understanding, medical diagnosis) appear to fit the paradigm of "choose one from Column A, one from Column B," i.e., given alternative choices for a set of decision points, find the best-rated consistent set of choices. When efficient best-first search algorithms are infeasible, some mechanism is needed for deciding when to halt the search and accept the best solution found so far. Such a mechanism should terminate the search as soon as possible without ignoring better solutions. This paper has shown how such a mechanism can exploit information about already-found solutions to accelerate the search conservatively, i.e., without ignoring better solutions.
ACKNOWLEDGEMENTS

The author wishes to acknowledge the intellectual contributions of Rick Hayes-Roth and Victor Lesser, and to call attention to their related work [3].

REFERENCES


V. CMU Computer Science Department Speech Publications

-- 1977 --

CM77Su CMU Computer Science Speech Group. Summary of the CMU Five-year ARPA effort in speech understanding research. Tech. Report, CMUCSD, 1977. Includes Cr77Wo, Er77Fu, GI77Mo, GI77Za, Gd77Ap, Gd77Ef, Ha77Di, Ha77Fc, Ha77Po, Ha77Se, Ha77Sy, Le77Pa, Le77Se, Lo77Dy, Mc77Pa, MK77Wo, Mo77Ha, Re77Mu and Re77Us. Harpy and Hearsay-II -- "Final" report following the close of the five-year ARPA effort in September, 1976.

Cr77Wo Cronk, R. Word adjacency acceptance procedure in Hearsay-II. In CM77Su.


GI77Za Goldberg, K, Reddy, D. R. and Gill, G. The ZAPDASH parameters, feature extraction, segmentation, and labeling for speech understanding systems. In CM77Su. Bottom-end processing for Hearsay-II and later Harpy systems.


Ha77Di Hayes-Roth, F., Gill, G. and Mostow, D. J. Discourse analysis and task performance in Hearsay-II. In CM77Su.


Ha77Fc Hayes-Roth, F. and Lesser, V. R. Focus and control in Hearsay-II. In CM77Su.

Ha77Po Hayes-Roth, F., Lesser, V. R., Mostow, D. J. and Erman, L. D. Policies for rating hypotheses, halting, and selecting a solution in Hearsay-II. In CM77Su.

Ha77Se Hayes-Roth, F., Fox, M. and Gill, G. Mostow, D. J. Semantics and pragmatics in Hearsay-II. In CM77Su.

Ha77Sy Hayes-Roth, F., Erman, L. D., Fox, M. and Mostow, D. J. Syntactic processing in Hearsay-II. In CM77Su.


Mc77Pa McCracken, D. A parallel production system for speech understanding. In CM77Su. A production-system version of Hearsay-II.


Mo77Ha Mostow, D. J. A halting condition and related pruning heuristic for combinatorial search. In CM77Su. Used in Hearsay-II.


Re77Mu Reddy, D. R. The multi-system approach to speech understanding. In CM77Su.


-- 1976 --


CM76W4CMU Computer Science Speech Group. Working papers in speech recognition IV: The Hearsay-II system. Tech. Report, CMUCSD, Feb., 1976. Includes Cr76Wo, Er750v, Er75Mu, Fe75Pa, Gi76Se, Ha76Fo, Ha76Hy, Ha75Au, Ha76Sy, Ha76Di, Le750r, Sh76Ph and Sm76Wo. This is a fairly complete description of Hearsay-II as of February, 1976. Descriptions here of many of the knowledge sources were made obsolete by the September, 1976, system (see CM77Su).


Sh76Ph Shockey, L. and Adam, C. The phonetic component of the Hearsay-II speech understanding system. In CM76W4.


--- 1975 ---


--- 1974 ---


CM74W3 CMU Computer Science Speech Group. Working papers in speech recognition III. Tech. Report, CMUCSD, 1974. Includes Bk75Dr, Bm74Ne, Gl74Pa, Kn74Re, Kr74Ad, Le750r, Ri74In and Sh74Su.

Er74Co Erman, L. D. (Ed.) *Contributed Papers of the IEEE Symp. on Speech Recognition*. IEEE, NY, NY, 1974. Includes Bk75Dr, Bm74Ne, Gl74Pa, Kn74Re, Kr74Ad, Le750r, Ri74In and Sh74Su.


-- 1973 --


CM73W2CMU Computer Science Speech Group. Working papers in speech recognition II. Tech. Report, CMUCSD, 1973. Includes Bk73Ma, Bm73Ne, Er73Sy, Re73Mo and Re73Hx.


Re73Hf Reddy, D. R. *The Hearsay system*. 20 minute 16mm sound film. Describes the speech understanding problem and demonstrates the Hearsay-I system. Prints may be borrowed.


--- 1972 ---

CM72W1 CMU Computer Science Speech Group. Working papers in speech recognition. I. Tech. Report, CMUCSD, 1972. Includes Er71Im, Ne71Sp, Re70Sp, Re70Cm, Re71Sp, Re71Sm and Re72Me.


--- 1971 ---


**1970**


(The following earlier papers were done at Stanford and are included here for reference.)

**1969**

Ne69Ex Neely, R. B. Experimental conversational computer system. *J. Acous. Soc. Amer.* 46 (July 1969) 89. (Abstract only)


**1968**


**Abbreviations:**
ASSP -- Acoustics, Speech and Signal Processing.
CMUCSD -- Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA, 15213. (412) 621-2600 x. 141.
IJCAI -- International Joint Conferences on Artificial Intelligence.