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

DEPARTMENT
of
COMPUTER SCIENCE

Carnegie-Mellon University
Best Available Copy
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.
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

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFOSR)
NOTICE OF TRANSMISSION TO FDA
This technical report is for Internal use and is approved for public release under AR 15-12 (b-3).
Distribution is unlimited.
A. D. 85035
Technical Information Officer

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

Bill Brodley, 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

   Introduction ................................................................. 1
   Systems ........................................................................ 4
   Hearsay-I
   Dragon
   Harpy
   Hearsay-II
   Locust
   Parallel Systems
   Discussion ..................................................................... 8

II. Knowledge Sources and Techniques .............................................. 10

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

III. Performance Measurement ............................................................ 32

   Performance of the Harpy and Hearsay-II Systems ......................................................... 32
   Connected Digit Recognition using Symbolic Representation of Pronunciation Variability ............ 37
   Effects of Branching Factor and Vocabulary Size on Performance ........................................ 39
   Appendix III-A: Test sentences ................................................................................ 41
   Appendix III-B: AI Retrieval Language Dictionary ............................................................. 46
   Appendix III-C: AI Retrieval Language Grammars ............................................................. 63
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 133
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.
I. MULTI-SYSTEM APPROACH TO SPEECH UNDERSTANDING*

Raj Re'idy

INTRODUCTION

In 1971, a group of scientists recommended the initiation of a five-year research program toward 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 lead 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 tinkered or ignored so as to gain deeper understanding of some aspect of the overall system. The purpose of this paper is to


* 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></td>
</tr>
<tr>
<td>vocabulary and syntax</td>
<td></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></td>
</tr>
<tr>
<td>transducer characteristics</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Signal transformations</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>digitization speed and accuracy</td>
<td></td>
</tr>
<tr>
<td>special-purpose hardware required</td>
<td></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></td>
</tr>
<tr>
<td>level transformation occurs</td>
<td></td>
</tr>
<tr>
<td>label selection technique</td>
<td></td>
</tr>
<tr>
<td>amount of training required</td>
<td></td>
</tr>
</tbody>
</table>

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

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

<table>
<thead>
<tr>
<th>System organization</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>levels of representation</td>
<td></td>
</tr>
<tr>
<td>single processor / multi-processor</td>
<td></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 69% with a word error rate of 45%. 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 other 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 51%. 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
be-, but 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 combing 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 Peddy)

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 (Wishboard). 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 II-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, Lowe, 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 minicomputers 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 (KAI0). 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 system 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 Crmmp and Cms.
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
Ph.D. Dissertation, Dept. of Computer Science, Carnegie-Mellon University,
Pittsburgh, PA 15213.
J. K. Baker (1975). "Stochastic modeling for automatic speech understanding", in
Computer Science Speech Group (1976). "Working Papers in Speech Recognition IV -
The Hearsay-II System", Tech. Report, Dept. of Computer Science, Carnegie-
Mellon Univ., Pittsburgh, PA 15213.
connected speech", Ph.D. dissertation, Computer Science Dept., Stanford
University, Technical Report, Computer Science Dept., Carnegie-Mellon University,
Pittsburgh, PA.
system", J. Acoust. Soc. Amer., Vol. 56, 177(A)
V. R. Lesser, R. D. Fennell, L. D. Erman, and D. P. Reddy (1975). "Organization of the
Science Dept., Carnegie-Mellon University, Pittsburgh, PA.
A. Newell, J. Barnett, J. Forgie, C. Green, D. Katl, J. C. R. Licklider, J Munson, R. Reddy,
machine recognition of speech", IEEE Trans. Audio and Electroacoustics Vol. AU-
21, (3), 229-238.
understanding system: an example of the recognition process", Proc. 3rd Int.
Joint Conf. on Artificial Intelligence, Stanford, CA, 185-193.
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 hypothesization 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 (POKOW) 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/beg+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 $4 \times \text{mintran}$, 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 POKOW 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 POKOW. In addition, 13053 incorrect words were generated. The vocabulary size for POKOW 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, 100 ms 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 15</th>
<th># HYPED BY POMOW</th>
<th>ACCEPTED</th>
<th>REJECTED</th>
<th>5.6 RANK ORDER</th>
</tr>
</thead>
<tbody>
<tr>
<td>CORRECT</td>
<td>332</td>
<td>326 (98%)</td>
<td>6 (2%)</td>
<td>(3.6 - 7.1)</td>
</tr>
<tr>
<td>INCORRECT</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>48 (8%)</td>
<td>32 (6%)</td>
<td>8 (2%)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>THR 18</th>
<th># HYPED BY POMOW</th>
<th>ACCEPTED</th>
<th>REJECTED</th>
<th>4.5 RANK ORDER</th>
</tr>
</thead>
<tbody>
<tr>
<td>CORRECT</td>
<td>332</td>
<td>312 (94%)</td>
<td>20 (6%)</td>
<td>(3.4 - 5.6)</td>
</tr>
<tr>
<td>INCORRECT</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>BEST RANKED PREDICTED WORD BOUNDARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIST</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</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 results it produces.

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 (957)</td>
<td>2891 (417)</td>
<td>3079 (422)</td>
</tr>
<tr>
<td>REJECTED</td>
<td>5 (52%)</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 \( S \rightarrow A T, T \rightarrow w_1 V, V \rightarrow UX, U \rightarrow \ldots w_n, A \rightarrow w_0 \), 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 \( w_1 \) 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 \( <a_1 w_1 \ldots w_n b_j> \) as well as all sequences \( <x_1 x_2 \ldots x_p a_1 \ldots w_1 \ldots w_n b_j y_1 y_2 \ldots y_q> \) where \( <x_1 x_2 \ldots x_p a_1 \ldots w_1 \ldots w_n b_j y_1 y_2 \ldots y_q> \) is a previously parsed sequence of words on the blackboard that is time-adjacent to and precedes (succeeds) \( <w_1 \ldots w_n>a \). 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/labeler 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 $H_1, ..., H_n$ via respective links $L_1, ..., L_n$ is given by

$$\max VLD(H_i) \times \text{IMPLICATION}(L_i)/100, \quad (1 \leq n).$$

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

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

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) = \left( \text{Sum } VLD(W_i) \times \text{length}(Wi) \right) / \text{Sum length}(Wi), (1 \leq 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 parsing 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 \[ VLD(H) \cdot length(H) \]. 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 speaker 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

[ WE'RE INTERESTED IN LEARNING ]

is recognized by the semantics module as an instance of the $SELECTION template, and the semantic feature $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 $SELECTION node in the finite state model. DISCO then predicts that the next utterance will be an instance of the $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 $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

[ 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;
O indicates empty retrieval set.
$\text{SELECTION [ WE'RE INTERESTED IN LEARNING ]}
\quad \text{Specifies a menu. DISCO responds by printing keywords and phrases from the menu.}

$\text{QUEST [ WERE ANY ARTICLES ON LEARNING WRITTEN IN MAY 1974 ]}
\quad \text{Specifies a set of articles. DISCO retrieves the articles and asks for further directions.}

$\text{SPRINGLIST [ WHICH OF THESE MENTION ROBOTS ]}
\quad \text{Further specifies a set of articles. DISCO removes articles from the currently retrieved set which don't satisfy the new restrictions.}

$\text{GETINFO [ WHO WROTE THESE ]}
\quad \text{Requests information about the retrieved articles. DISCO prints the requested information.}

$\text{LISTTHEM [ PLEASE LIST THEM ]}
\quad \text{Requests output of a set of articles. DISCO prints all the articles in the currently retrieved set.}

$\text{LISTIT [ PRINT IT ]}
\quad \text{Requests output of a single article. DISCO prints the article most recently referred to.}

$\text{STOPLISTING [ STOP LISTING PLEASE ]}
\quad \text{Requests cessation of output. DISCO aborts the current output operation.}

$\text{MAKEFILE [ MAKE A FILE OF THESE PIECES ]}
\quad \text{Requests file output. DISCO creates a file containing the retrieved articles.}

$\text{CONTENTSMENU [ WHAT ARE THE KEYWORDS FOR LEARNING ]}
\quad \text{Requests the contents of a menu. DISCO lists the keywords and phrases of the menu.}

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

$\text{WHY [ WHY ARE YOU SO SLOW ]}
\quad \text{Complaint. DISCO responds with a pacifying message.}

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

$\text{SWANTSEE}
\quad \text{DISCO has printed the title of an article and asked, "Do you want to see this article?"}

$\text{SWANTMORE}
\quad \text{DISCO has printed some of the articles in the currently retrieved set and asked, "Do you want to see another of these articles?"}

$\text{SMOREONMENU}
\quad \text{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
PERCEPTIONS
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 A. V. FINDLER

PROCEDURAL LEARNING IN WORLDS OF ROBOTS
LAURENT SIKLOPSY

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, prosodics) 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 data components available within that organization. Knowledge sources are intended to be largely

References


independent and capable of adynchronous 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 1002).
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-I1 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-I1 system on the HSII/C.mmp system.

References
A Parallel Production System for Speech Understanding (McCracken)

The question addressed by his 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 experiment—tion, three specific ones—AIX05, AIX15, and AIXF—are 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 template 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 91.7% 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 91.7% 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 Al 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 OS 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 Al retrieval task test data.
### RESULTS OF LIVE SENTENCES HARPY VERSION

<table>
<thead>
<tr>
<th>UTT</th>
<th>TIME IN</th>
<th>WORDS IN</th>
<th>WORDS 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>70</td>
</tr>
<tr>
<td>3</td>
<td>3.8</td>
<td>9</td>
<td>11</td>
<td>5</td>
<td>55.6</td>
<td>301.1</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>3</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>53.0</td>
<td>36.6</td>
<td>352</td>
<td>53</td>
</tr>
</tbody>
</table>

Correct utts=4/5 = 80.0%

### RESULTS OF LIVE SENTENCES HARPY VERSION

**UTT 1**
- **UTT**: "ARE ANY PAPERS ABOUT SEMANTIC NETWORKS"  
  - **REC**: "ARE ANY PAPERS ABOUT SEMANTIC NETWORKS"  
  - **CORRECT=6/6 AVE. PRB.=-.4954980

**UTT 2**
- **UTT**: "DOES SEMANTIC NETS GET MENTIONED ANYWHERE"  
  - **REC**: "DOES SEMANTIC NETS GET MENTIONED ANYWHERE"  
  - **CORRECT=6/6 AVE. PRB.=-.5610780

**UTT 3**
- **UTT**: "WHICH PAPERS ON REGION ANALYSIS ALSO DISCUSS LANGUAGE UNDERSTANDING"  
  - **REC**: "WHICH PAPERS ON REGION ANALYSIS SUBSYSTEM AND DESIGN MENTION UNDERSTANDING"  
  - **CORRECT=5/9 AVE. PRB.=-.663689

**UTT 4**
- **UTT**: "HOW MANY ARTICLES ON CHESS AND LEARNING ARE THERE"  
  - **REC**: "HOW MANY ARTICLES ON CHESS AND LEARNING ARE THERE"  
  - **CORRECT=5/9 AVE. PRB.=-.552164

**UTT 5**
- **UTT**: "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: 1811
Branching factor: 3.53
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
4849 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
85.8 Mips

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 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 12%.

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>18</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>1</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

| Training Set | | | |
| No. of Sentences | 300 | 30 | 120 | 300 |
| No. of Words | 980 | 210 | 360 | 900 |
| Mins. of Speech | 7.5 | 1.4 | 3.1 | 7.6 |
| Words/minute | 120 | 150 | 116 | 118 |

| Test Set | | | |
| No. of Sentences | 1000 | 100 | 400 | 1200 |
| No. of Words | 3880 | 700 | 1200 | 3680 |
| Mins. of Speech | 25.1 | 4.8 | 10.3 | 33.8 |
| Words/minute | 120 | 146 | 117 | 109 |

| Performance on Test Data | | | |
| %Word Accuracy | 98 | 99 | 93 | 93 |
| %Sent. Accuracy | 96 | 96 | 82 | 83 |
| Mips/s | 3.5 | 3.5 | 3.5 | 3.5 |
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 measures.

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>MIPESS</th>
<th>BRANCHING ERROR</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1506</td>
<td>6.63</td>
<td>4.6</td>
</tr>
<tr>
<td>A1510</td>
<td>9.36</td>
<td>8.2</td>
</tr>
<tr>
<td>A1515</td>
<td>13.65</td>
<td>11.9</td>
</tr>
<tr>
<td>A1530</td>
<td>44.72</td>
<td>33.3</td>
</tr>
<tr>
<td>A1540</td>
<td>59.15</td>
<td>39.5</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>MIPESS</th>
<th>BRANCHING</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1510</td>
<td>9.36</td>
<td>8.2</td>
</tr>
<tr>
<td>A1512</td>
<td>16.77</td>
<td>10.5</td>
</tr>
<tr>
<td>A1503</td>
<td>26.00</td>
<td>3.5</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></th>
<th>AF</th>
<th></th>
<th></th>
<th>100</th>
<th>6.83 (average over all)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIX05</td>
<td>33.4</td>
<td>AIX15</td>
<td>46.5</td>
<td>68.0</td>
<td></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>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>39.7</td>
<td>41.5</td>
<td>68.3</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>78.3</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=36.598
{ 66 DOES 196 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 196 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 DISCUSS 192 EVALUATION 1 FUNCTIONS 3 words=6 AF=40.857
{ 65 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
{ 65 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=12.250
{ 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 1 HAVE 1 THESE 1 ABSTRACTS 1 LISTED 1 words=6 AF=18.422
{ 66 ARE 292 ANY 5 ARTICLES 2 ABOUT 192 STRUCTURED 1 PATTERN 1 RECOGNITION 3 words=7 AF=70.375
{ 66 DO 6 ANY 6 OF 3 THE 1 ABSTRACTS 1 MENTION 192 LEARNING 3 words=7 AF=34.750
{ 66 HOW 4 MANY 8 OF 1 THESE 1 ALSO 1 DISCUSS 192 ABSTRACTION 3 words=7 AF=34.500
{ 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 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
Let's restrict our attention to papers since nineteen seventy-four.

...d...
 WHICH 2 PAPERS 7 CITE 36 FELDMAN 3 words=4 AF=38.600
 WHO 5 HAS 1 WRITTEN 1 ABOUT 192 AUTOMATIC 7 PROGRAMMING 3
 words=6 AF=39.286
 WHO 5 WAS 2 QUOTED 1 IN 1 THAT 1 ARTICLE 1 words=6 AF=11.000
 WHICH 21 IS 1 THE 1 OLDEST 1 words=4 AF=18.000
 ARE 292 ANY 6 NEW 1 BOOKS 1 BY 76 TERRY 1 WINGRAD 3 words=7
 AF=58.250
 CAN 2 I 1 HAVE 1 THESE 1 ABSTRACTS 1 LISTED 1 words=6 AF=10.429
 DID 99 CARL 1 HEWITT 5 PRESENT 2 A 1 PAPER 1 AT 2 THE 1 IFIP 1
 MEETINGS 1 IN 1 SEPTEMBER 1 words=12 AF=14.000
 DID 99 ANY 4 ACL 1 PAPERS 1 CITE 96 RICK 1 HAYES-ROTH 3 words=7
 AF=33.875
 DO 6 ANY 6 OF 3 THOSE 1 PAPERS 1 MENTION 192 AXIOMATIC 1
 SEMANTICS 3 words=8 AF=31.000
 DURING 1 WHAT 1 MONTHS 1 WERE 1 THEY 1 PUBLISHED 1 words=6
 AF=10.286
 HOW 4 MANY 8 RECENT 1 ISSUES 1 CONCERN 192 INVARIANCE 1 FOR 1
 PROBLEM 1 SOLVING 3 words=9 AF=27.800
 HOW 4 MANY 8 SUMMARIES 1 DISCUSS 192 KNOWLEDGE 2 BASED 1 SYSTEMS
 3 words=7 AF=34.625
 HAVE 97 ANY 2 NEW 1 PAPERS 1 BY 96 LEE 1 ERMAN 3 APPEARED 1
 words=8 AF=29.778
 I'D 1 LIKE 1 TO 2 KNOW 1 THE 1 PUBLISHERS 1 OF 1 THAT 1 STORY 1
 words=9 AF=7.600
 IS 290 HUMAN 3 BEHAVIOR 5 OR 191 HUMAN 3 MEMORY 3 DISCUSSED 2 IN
 1 A 1 RECENT 1 SUMMARY 1 words=11 AF=47.250
 LIST 2 THE 2 ABSTRACTS 1 BY 96 HERB 1 SIMON 3 words=6 AF=24.429
 WAS 290 ALLEN 2 NEWELL 3 CITED 2 IN 1 THAT 1 SUMMARY 1 words=7
 AF=45.75d
 WHAT 26 ABOUT 288 ALLEN 2 COLLINS 3 words=4 AF=77.000
 WHERE 5 DID 1 THAT 1 ARTICLE 1 APPEAR 1 words=5 AF=12.500
 WHO 5 HAS 1 WRITTEN 1 ABOUT 192 LANGUAGE 6 COMPREHENSION 3 AND
 LANGUAGE 6 DESIGN 3 words=9 AF=47.200
 QUIT 1 LISTING 1 PLEASE 1 words=3 AF=17.250
 WEREN'T 1 SOME 1 ARTICLES 1 PUBLISHED 1 ON 192 GOAL 1 SEEKING 1
 COMPONENTS 3 words=8 AF=23.667
 WHAT 26 SORTS 1 OF 192 LANGUAGE 6 PRIMITIVES 3 ARE 1 WRITTEN 1
 UP 1 words=8 AF=33.000
 HASN'T 192 A 21 CURRENT 1 REPORT 1 ON 192 PRODUCTION 1 SYSTEMS 3
 BEEN 1 RELEASED 1 words=9 AF=47.900
 ARE 292 THERE 2 ANY 5 ISSUES 1 ABOUT 192 COOPERATING 1 SOURCES 1
 OF 1 KNOWLEDGE 3 words=9 AF=56.400
 DID 99 VIC 1 LESSER 5 PRESENT 2 PAPERS 1 AT 2 IFIP 1 words=7
 AF=22.125
 DID 99 ANYONE 1 PUBLISH 1 ABOUT 192 LARGE 1 DATA 1 BASES 3 IN 1
 COMMUNICATIONS 1 OF 1 THE 1 ACM 1 words=12 AF=28.385
 DOES 195 HE 1 WORK 1 AT 1 CMU 1 words=5 AF=44.333
 DO 6 ANY 6 RECENT 4 ACM 1 CONFERENCES 1 CONSIDER 192 SEMANTIC 2
 NETS 3 OR 191 SEMANTIC 2 NETWORKS 1 words=11 AF=39.583
DO 6 RESPONSES I EVER 1 COME 1 FASTER 1 words=5 AF=12.667
HAS 96 LEE 1 ERMAN 4 BEEN 1 REFERENCED 1 IN 1 ANY 1 OF 1 THOSE 1 words=9 AF=17.300
HAS 96 ALLEN 2 NEWELL 4 PUBLISHED 2 ANYTHING 1 RECENTLY 1 words=6 AF=24.571
HAVE 97 ANY 2 NEW 1 PAPERS 1 BY 56 TERRY 1 WINograd 3 APPEARED 1 words=8 AF=23.778
HOW 4 BIG 1 IS 1 THE 1 DATABASE 1 words=6 AF=10.714
HOW 4 MANY 8 OF 1 THESE 1 ALSO 1 DISCUSS 192 DYNAMIC 3 BINDING 3 words=8 AF=31.000
HOW 4 MANY 8 RECENT 1 ISSUES 1 CONCERN 192 DISPLAY 1 TERMINALS 3 words=7 AF=34.500
KILL 1 THE 1 LISTING 1 words=3 AF=17.250
PLEASE 4 MAKE 1 ME 1 A 1 FILE 1 OF 1 THOSE 1 words=7 AF=9.500
WHAT 26 IS 4 HIS 1 AFFILIATION 1 words=4 AF=19.600
WHICH 21 OF 2 THESE 5 CITED 96 PERRY 1 THORNDYKE 3 words=6 AF=27.714
WHICH 21 PAPERS 7 ON 192 DESIGN 6 IN 1 THE 1 ARTS 4 ALSO 2 words=9 AF=41.667
WHO 5 WAS 2 QUOTED 1 IN 1 THAT 1 ARTICLE 1 words=6 AF=11.800
WHICH 21 PAPERS 7 WERE 1 WRITTEN 2 AT 1 NRL 1 OR 1 AT 1 SMC 1 words=9 AF=18.200
CONCEPTUAL

CONCLUSION

CONSCIENTIOUS

CONCURRENT

CONFERENCE

CONTRIBUTES

CONFINE

CONSISTENT

CONSIDER

CONSIDERED

CONSTRAINT

CONSTRUCTING

CONSTRUCTION

CONSULTANT

CONSULTATIONS

CONTAIN

CONTAINED

CONTAINS

CONTEXT

CONTINUOUS

CONTROL

CONTROLLED

CONVEXION

CORREPTING

CURRENT

CURVED

CYCLES

DANKY

DATA

DATE

DATE'S

DAVID

DEBATE

DECEMBER

DECISION

DEDUCTION

DEDULATIVE

DEMAND

DEKOTATIONAL

DEPTH

DERIVATION

DESCRIBE

DESCRIPTION

DESCRIPTIONS

DESIRED

DESIRED'S

DEVICES

DEPARTMENT
MEANING (-0) M IV V (Z;4;2,0,0) S
MEDICAL (-0) M EN (<= (-,0),-) (U,0) DX IV2 (- (-,0),|-4,0) (K,0) EL
MEETING (-0) M IV (<= (-,0),-) T DX I H2 N X
MEETINGS (-0) M IV (<= (-,0),-) T DX IH2 NH Z, IH3
MEET/Z (-0) M AA2 EL3 (- (-,0),-) S ER
MEMORIES (-0) M EN2 (O,0) (K,H2) IV2
MEMORY (-0) M EN2 M (O,0) (K,H2) IV2
MENTION (-0) M (E2,H2,0) N (<= (-,0),-) S H1,10 IH5 N
MENTIONED (-0) M (E2,H2,0) N (<= (-,0),-) S H1,10 IH5 N ((-,0),-4,0) (D,0) DX
MENTIONING (-0) M (E2,H2,0) N (<= (-,0),-) SH1,10 N (IH3,14,0) NH
MENTIONS (-0) M (E2,H2,0) N (<= (-,0),-) S H1,10 N (Z;4;2,0,0) S
MENTH (-0) M EN2 V (A,H2,0)
MENTS (-0) M EN2 N V (A,14;2,0,0) S
META-SYMBOLIC (-0) M EN (<= (-,0),-) T DX IH3 S IH5 N (<= (-,0),-4,0) (B,0) B AG6 EL IH3 (- (-,0),-4,0) (K,0)
MATHEMATICS (-0) M EN (<= (-,0),-) T DX IH3 M AF TH AH2 MI AE5 (<= (-,0),-) T DX IH3 (- (-,0),-) S (H;4,0)
METHODS (-0) M EN2 TH IH3 (- (-,0),-) S (H;4,0)
MICHAEL (-0) M (A,H2,0) AC3 (AHR,0) (<= (-,0),-) (K,0) EL
MICHAILSKY (-0) M IH (- (-,0),-) (K,0) AO EL S (- (-,0),-) IV
MICHEL (-0) M IH2 (- (-,0),-) SH1,10 IH4 IV
MICHIE (-0) M (A,H2,0) AC3 (AHR,0) (<= (-,0),-4,0) (K,0)
MINIMAL (-0) M IH3 N SH1,10 IH4 EL2
MINIMAX (-0) M IH2 NH (- (-,0),-) (K,0) ER2
MINISKY (-0) M IH3 N S (- (-,0),-) IV
MITCHELL (-0) M IH3 (- (-,0),-) S H1,10 IH3 EL3
MISP (-0) M EN2 M (-,0) L IH6 S - T IH2 UW (U;2,0)
MODEL (-0) M AA (- (-,0),-) D DX EL2
MODELING (-0) M AA (- (-,0),-) D DX EL (H3,14) NH
MODELS (-0) M AA (A,H2,0) (- (-,0),-) D DX EL3 (Z;4;2,0,0) S
MONITOR (-0) M AA N IH1 (<= (-,0),-) T DX (H2,0) ER
MORSE (-0) M (A,H2,0) IH2 NH (- (-,0),-) (K,0) IV
MOUTH (-0) M AA11 H29 N (- (-,0),-) IH (H;4,0)
MONTHS (-0) M (A,H2,0) N (- (-,0),-) (T,0) T IH3 S (H;4,0)
MOTION (-0) M AA21 ER2
MOTION (-0) M OW SH IH3 N
MOVE (-0) M I W2 V (I,0)
MOVEMENTS (-0) M UW2 V M E H2 N (-,0,) - S (H;4,0)
MOVING (-0) M UW2 V (I,0) IV (Z;4;2,0,0) S
MULTILEVEL (-0) M EL2 (<= (-,0),-4,0) (T,0) DX IV L IH V EL
MULTIPROCESS (-0) M OW4 EL2 (<= (-,0),-4,0) (T,0) DX IV (- (-,0),-) (P,0) PR (R,0) AO S IH7 S (H;4,0)
MUSIC (-0) M (Y,0) IV IH4 (Z;4;2,0,0) S IH4 (- (-,0),-) (K,0)
MUST (-0) M IH2 (-,0) S =T; (H;4,0)
MYSELF (-0) M (A,H2,0) AC3 (AHR,0) S A H A EL F (H;4,0)
NAME (-0) M (E,Y2,0) EYC (EYR,0) (- (-,0),-) (G,0) EL
NAME/WEATHER (-0) M AE5 SH W E H3 (- (-,0),-4,0) (B,0) ER
NATIONAL (-0) M AE5 SH W IH3 N EL2
NATURAL (-0) M AE5 SH W IH3 N EL2
NATURE (-0) M AE5 SH W IH3 N EL2
NEW (-0) M AA1 (R,E2) EL2
NEW (-0) M IH2 (0,0) UW1 (UW2,0)
NEWBORN (-0) M (IH3,0) UW1 (- (-,0),-4,0) (B,0) AA4 ER2 (N,DX)
NEWCOMER (-0) M (IH3,0) UW1 (- (-,0),-) (K,0) AA M ER
NEWELL (-0) M (IH5,14,0) UW2;10,30) EL
NEWEST (-0) M LW IH3 S (-,0) (T,0)
Appendix III-C-1. AI Retrieval Language Grammar: AIXF

<SENTENCE> = [ <SENTENCE1> ]
<SA> = THE
   A
   AN
<ACQUIRE> = HAVE
   SEE
   KNOW
   GUESS
<AFFILIATION> = <ADDRESS/S>
   <AFFILIATION/S>
<ADDRESS/S> = ADDRESSES
   ADDRESS
<AFFILIATION/S> = AFFILIATIONS
   AFFILIATION
<AI> = AI
   ARTIFICIAL INTELLIGENCE
<ALSO> = ALSO
   IN ADDITION
<ALSO/ATION/STOPICS> = <SMSTRICATION> <STOPICS>
   <SMSTRICATION> <STOPICS> <ALSO>
   <ALSO> <SMSTRICATION> <STOPICS>
<SMENTION> = CITE
   REFER TO
   <REF.> <REF1>
   <DISCUSS/S>
   CONCERN
   CONTAIN THE PHRASE
   DESCRIBE
   RELATE TO
   <HAVE> <SMENTION/HAVE>
   CONSIDER
   <MENTION/S>
<STOPICS> = <TOPIC>
   <TOPIC> <CONJUNCTION> <TOPIC>
<ALWAYS> = ALWAYS
   USUALLY
   REGULARLY
<ANY/NOD/A/Pieces> = <PIECE/S>
   <PIECE/S>
   <SOME> <PIECE1>
<PIECE/S> = <STORY/S>
   <ARTICLE/S>
   <BOOK/S>
   <PAPER/S>
   <ABSTRACT/S>
   <PROCEEDINGS>
   <REPORT/S>
   <ISSUE/S>
   <JOURNAL/S>
   NOTES
   <REVIEW/S>
   <VOLUME/S>
   PIECE
   <SURVEY/S>
   <SUMMARY/S>
   TECHNICAL PAPERS


A]XF
S<$OuRCV>
.P;ECE1
..<PRLCKI
I)ING/S>
$FROM>,S<
SCONFERINCE>
SCI
CI:1I Ci>,S<
PIECc12>
SARTICL-
HUMAN
PROOLLfel
THOUGHT
AND
LANGUAGE
ASK:

REQUEST
DEMAND
SAY

$AUTHORS1:

$AUTHORS1:

$AUTHORS1:

$AUTHORS1:

$AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

$AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:

AUTHORS1:
$\textit{CONJUNCTION}$ = AND
    NOT
    OR
    BUT NOT
    AND NOT
    OR NOT
$\textit{AUTHOR/DATE}$ = $\textit{AUTHOR/S}$ AND $\textit{DATE/S}$
    $\textit{DATE/S}$ AND $\textit{AUTHOR/S}$
$\textit{AUTHOR/S}$ = AUTHORS
$\textit{DATE/S}$ = DATES
$\textit{BE}$ = $\textit{BE}$
    $\textit{HAVE}$ BEEN
$\textit{BE}$ = $\textit{BE/PAST}$
$\textit{HAVE}$ = HAVE
$\textit{HAPPEN/S}$ = HAPPEN
$\textit{BE}$ = $\textit{BE}$
    $\textit{HAVE}$ NOT
    ISN'T
    ARN'T
    WASN'T
    WEREN'T
$\textit{BE/TOPICS/MENTIONED}$ = $\textit{HAVE}$ $\textit{TOPICS}$ BEEN $\textit{MENTIONED}$ [PP]
    $\textit{TOPICS}$ $\textit{MENTIONED}$[PP]
$\textit{HAVE}$ $\textit{TOPICS}$ BEEN $\textit{MENTIONED}$ [PP]
$\textit{DO}$ $\textit{TOPICS}$ GET $\textit{MENTIONED}$[PP]
$\textit{DO}$ $\textit{TOPICS}$ GET $\textit{MENTIONED}$[PP]
$\textit{HAVE}$ = $\textit{HAVE}$
    $\textit{HAVE}$ NOT
    $\textit{HAVE}$ HADN'T
$\textit{MENTIONED}$[PP] = $\textit{CITED}$
    DISCUSSED
    MENTIONED
    CONSIDERED
    $\textit{WRITTEN}$ ABOUT
$\textit{SOMEWHERE}$ = IN $\textit{ANYPIECE}$
    ANYWHERE
    AT ALL
$\textit{SOMEWHERE}$ = IN $\textit{ANYPIECE}$
    SOMEWHERE
    ANYWHERE
    AT ALL
$\textit{DO}$ = $\textit{DO}$
    $\textit{DON'T}$
$\textit{BE/PAST}$ = 15
$\textit{ARE}$
<$RE[PAST]> + WAS
WERE
<$RE[HERE]> + <$RE>
<$DO> <$HEARSAY> HAVE
<$RE[HERE]>
<$HEARSAY> - YOU
THE DATA BANK
THE DATA BASE
HEARSAY
THE SYSTEM
<$RE[HERE] + <$RE> THERE
<$HAVE> THERE BEEN
<$RE[HERE]ANYPIECES> + <$RE[HERE] <$ANYPIECES>
DO YOU HAPPEN TO HAVE <$ANYPIECES>
<$SHOW/MANYPIECES2> ARE THERE
<$SHOW/MANYPIECES2> + <$SHOW/MANY>
<$SHOW/MANY> <$PIECES>
<$CHESS> + 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>
<$CONFERENCE3> + <$A> <$CONFERENCE>
<$CONFERENCE1>
<$CONFERENCE> + <$CONFERENCE1>
<$CONFERENCE1> + <$CONFERENCE2>
<$CONFERENCE1> + IJCAI
ACM
IEEFT
IFIP
<$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
GAME PLAYING
LEARNING
INFLUENCE
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
$1$ - I
WE
$WHAT:BE$ - $WHAT:WHICH$ $BE$
WHAT'S
$DATE$ - $DATE$
THE LAST $NUMBER$ $TIMES$
$DATE$ - $CONJUNCTION$ $DATE$
$DATE$ - $THROUGH$ $DATE$
$DATE1$ - $YEAR$
$MONTH$
THE $MONTH/S$ OF $YEAR$
$MONTH$ - $YEAR$
$NUMBER$ - $HUNDREDS$
$NUMBER$
$HUNDREDS$ $NUMBER$
$TIMES$ - $MONTHS$
ISSUES
VOLUMES
YEARS
TIMES
$THROUGH$ - TO
THROUGH
TILL
$YEAR$ - NINETEEN $NUMBER$
$MONTH$ - MAY
JANUARY
FEBRUARY
MARCH
APRIL
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER
$MONTH/S$ - $MONTHS$
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
ONES
$WANNA = <$1D> LIKE
$1 = $DESIRED
$LEMMI = IF $ME
LET'S
$MAY = <$1>
$WOULD = WOULD
CAN
$LIST = LIST
PRINT
TRANSMIT
WRITE
$GIVE = <$GIVE>
GET FOR
TELL
$ME = MI.
US
$GIVE1 = GET
GIVE
SHOW
$GRIPE = <$ME - <$Hearsay> $ALWAYS $SLOW
HAVEN'T YOU FINISHED
WHY <$ME - <$Hearsay> SO SLOW
DO RESPONSES EVEN COME FASTER
HOW <$MAY = <$1> $IMPROVEHIS
DO ALL QUERIES TAKE THIS LONG
HOW LONG DOES IT TAKE
WHEN WILL <$Hearsay> HAVE THE ANSWER
DOES IT ALWAYS TAKE THIS LONG TO ANSWER <$ME
WHAT <$MAY = <$1> DO TO <$IMPROVEHIS

$SLOW = SO SLOW
SLOW
THIS SLOW
$IMPROVEHIS = HELP
SPEED <$Hearsay> UP
HELP <$Hearsay>
USE <$Hearsay> EFFICIENTLY

$HELP = HELP
HOW BIG IS THE DATA BASE
$WHATSORTOF = RETRIEVAL/S CAN <$Hearsay> DO
TELL <$ME = WHAT TO DO
$WHATWHICH = <SMENU> <$MAY = <$1> $SEEK
$WHATSORTOF = RETRIEVAL <KEY/S> <$MAY = <$1> $SEEK
$WHATSORTOF = <PIECE/S> <$DEPREQ> AVAILABLE
$WHATSORTOF = <$SMENU> <$DE> STORED
WHAT IS KNOWN <$KEYS EVERY <$PIECE/S>
WHAT DO <$1> HAVE TO DO
CAN YOU HELP
$WHATSORTOF = <$SMENU> <$BEHERE>
CAN YOU HELP <$ME
HELP <$ME
$PROVIDE <$A> <$SMENU>
$WHATIS = <$SOME <SMENU> <$FROM <$1>
$WHATWHICH = FACTS ARE STORED
$WHATIS = THE SIZE OF <$Hearsay>
WHAT <$MAY = <$1> $ASK

73
WE'VE
WE HAVE
$1D$ $<<$1$ WOULD
I'D
WE'D
$1TS$ $THE$ 
THEIR
ITS
$1OF$ $THAT$PIECE$ OF $THAT$PIECE
FROM $THAT$PIECE
$PUBLISHER$ $PUBLISHER$
$REFERENCE$ $REFERENCE$
REFERENCES
$TITLE$ $TITLES$
TITLE
$WORD$ $WORDS$
WORD
$PHRASE$ $PHRASES$
PHRASE
$LAST$ $LAST$
MOST RECENTLY
$LEARNING$ $LEARNING$
GRAMMATICAL INFERENC
NEURAL NETWORKS
ABSTRACTION
DYNAMIC CLUSTERING
CELL ASSEMBLY THEORY
$LIST$ $THE$ $NEWEST$
$THENEXT$
$NAME$ $THENEXT$
$MAY$ $HAVE $THAT$PIECE $LISTED$
$LIST$ $THE$ $NEWEST$
$LIST$ $THAT$PIECE
$THENEXT$ $THE NEXT$
The NEXT $NUMBER$
THE FIRST
UP TO $NUMBER$
BETWEEN $NUMBER$ AND $NUMBER$ OF THEM
$NUMBER$ MORE
THE FIRST $NUMBER$
$LISTED$ $LISTED$
PRINTED
WRITTEN
$LISTING$ $LISTING$
PRINTING
TRANSMITTING
WRITING
$MAKE$ $CONV$
WRITE
MAIL
PRODUCE
GEN
$MAKE$ $A$ $TITLE$
$MAKE$ $A$ $TITLE$ OR THE $NEWEST$
$MAKE$ $A$ $TITLE$
$MAKE$ $A$ $TITLE$
$MAKE$ $A$ $TITLE$
$MAKE$ $A$ $TITLE$ $FROM$ $THAT$PIECE
<$REF1> - RELATED TO
ABOUT
REGARDING
ON
RELATING TO
DISCUSSING
CONCERNING
MENTIONING
<DISCUSS/S> - DISCURES
<MENTIONS/D> - <BE> <$RE>
<MENTIONS/HAVE>
<MENTIONS/HAVE> - <$MENTIONED[PP]>
RELATED TO
CONCERNED
<MENTIONS/S> - MENTIONS
MENTION
<WRITTEN - <$WRITTEN>
<$RECENTLY> - <$WRITTEN>
<$WRITTEN> - <$RECENTLY>
<$WRITTEN 2> - <$WRITTEN 2>
<$RECENTLY 2> - <$WRITTEN 2>
<$RECENTLY 2>
<TOPIC/S> - TOPICS
TOPIC
<SCULPS> - MENC
MENU
<SUBJECTS> - SUBJECTS
SUBJECT
<AREA'S> - AREAS
AREA
<MY - CUR
<STATEMENTS> - <$S> <$WISH> TO <$ACQUIRE> <$WHAT>
<$DON'T> <$GIVE> <$WISH> <$WHAT>
<$WISH> - WOULD NOT LIKE
<$DON'T> <$WISH>
<NEWEST> - LAST
NEWEST
<$LAST> <$WRITTEN>
LATEST
MOST RECENT
<NO> - NO
<STEPS> - <$STEPS>
<$STEPS> <$NUMBERS>
<TENS> - NINETY
TEN
ELEVEN
TWELVE
THIRTEEN
FOURTEEN
FIFTEEN
SIXTEEN
SEVENTEEN
EIGHTEEN
$TEN> - TWENTY
THIRTY
FOURTY
FIFTY
SIXTY
SEVENTY
EIGHTY
NINETY
<OLDEST1> • OLDEST
FIRST
EARLIEST
FIRST <$WRITTEN>
<OURSELVES> • <RETRIEVAL/S>
<SME• ATTENTION
MYSELF
OURSELVES
<WHEN/DATE> • THIS YEAR
LAST YEAR
SINCE LAST YEAR
<WHEN> • <DATE>
STORY/S • STORIES
STORY
ARTICLE/S • ARTICLES
ARTICLE
BOOK/S • BOOKS
BOOK
PAPER/S • PAPERS
PAPER
ABSTRACT/S • ABSTRACTS
ABSTRACT
PROCEEDINGS/S • PROCEEDINGS
PROCEEDING
REPORT/S • REPORTS
REPORT
ISSUE/S • ISSUES
ISSUE
JOURNAL/S • JOURNALS
JOURNAL
REVIEW/S • REVIEWS
VOLUME/S • VOLUME
SURVEY/S • SURVEYS
SURVEY
SUMMARY/S • SUMMARIES
SUMMARY
STORY/S 2 • STORIES
STORY
ARTICLE/S 2 • ARTICLES
ARTICLE
BOOK/S 2 • BOOKS
BOOK
PAPER/S 2 • PAPERS
PAPER
ABSTRACT/S 2 • ABSTRACTS
ABSTRACT
PROCEEDINGS/S 2 • PROCEEDINGS
PROCEEDING
REPORT/S 2 • REPORTS
REPORT
ISSUE/S 2 • ISSUES
ISSUE
JOURNAL/S 2 • JOURNALS
WHAT'S OPTS OF TOPICS?
WHERE SE TOPICS MENTIONED
SHOW MANY PIECES SE TOPICS ALSO MENTIONED TOPICS?
WHERE SE TOPICS MENTIONED
TOPICS MENTIONED SOMEWHERE 2
TOPICS MENTIONED SOMEWHERE 2 RECENTLY
SHOW MANY PIECES RE TOPICS BEEN WRITTEN
SHOW MANY PIECES RE TOPICS
TOPICS MENTIONED INS ANY PIECES
SHOW MANY PIECES APPEARED WHICH MENTION TOPICS
SHOW MANY PIECES RE TOPICS THAT
SHOW MANY PIECES RE TOPICS SE WRITTEN
SHOW MANY PIECES RE TOPICS EXIST
QUERY CITATION SHOW MANY PIECES CITE AUTHORS
SE SEARCH AUTHOR MENTIONED PP SOMEWHERE
SHOW WHICH PIECES WRITTEN BY OFFICIALS
SHOW MANY PIECES MENTIONED ANY PIECES SWHAT WHICH
SHOW MANY PIECES MENTIONED ANY PIECES EXIST
QUERY WHEN CONFERENCE + SHOW MANY CONFERENCE'S SE PAST WHERE
SHOW MANY CONFERENCE'S WHERE
QUERY DATE SHOW MANY PIECES SE PAST WHERE WRITTEN WHEN DATE
SHOW WHICH PIECE THE NEWEST WRITTEN BY OFFICIALS
SHOW MANY PIECES ANY DATED PIECES WHEN DATE
SHOW MANY PIECES ANY PIECES WRITTEN WHEN DATE
SHOW MANY PIECES ANY PIECES WRITTEN BY OFFICIALS
SHOW MANY PIECES ANY PIECES CITATION WHEN PIECES
SHOW MANY PIECES ANY PIECES WRITTEN WHEN PIECES
SHOW MANY SOURCE PIECES CONTAINED ANY PIECES WRITTEN BY OFFICIALS
SHOW MANY SOURCE PIECES CONTAIN WISCONSIN'S ARTICLE
DOI OFFICIALS PRESENT ANY DATED PIECES AT CONFERENCE'S
DOI OFFICIALS PRESENT ANY DATED PIECES AT CONFERENCE'S WHEN DATE
QUERY SOURCE CITATION DNA SHOW MANY SOURCE PIECES CITE OFFICIALS
QUERY WHEN PUBLISH OFFICIALS SOMETHING WHEN DATE
SHOW MANY SOURCE PIECES WHEN DATE
QUERY CONFERENCE'S SHOW MANY SOURCE PIECES CITED BY OFFICIALS
QUERY SOURCE CONFERENCE'S SHOW MANY ANY PIECES FROM SOURCE2 RE TOPICS
DOI ANY PUBLISH IN RE TOPICS IN SOURCE2
SHOW MANY SOURCE PIECES MENTIONATION TOPICS
SOURCE終於 CONTAINED IN SOURCE2
QUERY INDEX SHOW MANY PIECES WHEN DATE MENTIONED TOPICS
RELATE WHERE RELATES
QUERY - CONPAND SEARCH AUTHOR
QUERY STATE search WHAT FILL SE SOURCE PIECES
\$SELECTION$ - \$WHAT$ - \$SOME$ - \$MENU$ - \$FROM$ - \$SUPERMENU$ - \$SUB$ - \$INTERESTED$ - \$SUPERMENU$ - \$WHAT$ - \$WHICH$ - \$MENU$ - \$ONLY$ - \$INTERESTED$ - \$PRES$ - \$RELATION$ - \$TILL$ - \$MENU$ - \$INTERESTED$ - \$MENU$ - \$CHOOSE$ - \$FROM$ - \$WHAT$ - \$MENU$ - \$ANY$ - \$PIECES$ - \$REL$ - \$SUPERMENU$ - \$WHAT$ - \$MENU$ - \$CONJUNCTION$ - \$WHAT$ - \$MENU$ - \$SEMANTIC$ - \$NETWORKS$ - \$SEMANTIC$ - \$NETWORK$ - \$SEMANTIC$ - \$NET$ - \$UNDERSTANDING$ - \$LANGUAGE$ - \$UNDERSTANDING$ - \$NATURAL$ - \$LANGUAGE$ - \$ENGLISH$ - \$NATURAL$ - \$LANGUAGE$ - \$UNDERSTANDING$ - \$SYNTAX$ - \$SENTENCE$ - \$CONTENTS$ - \$MENU$ - \$INFO$ - \$SCRIPT$ - \$SHEEP$ - \$SISTEM$ - \$MAKE$ - \$ICE$ - \$SNO$ - \$REQUEST$ - \$REQUEST$ - \$SELECTION$ - \$YES$ - \$STOP$ - \$LISTING$ - \$YES$ - \$SURE$ - \$STOP$ - \$MORE$ - \$STOP$ - \$FINISH$ - \$STOP$ - \$TILL$ - \$STOP$ - \$LISTING$ - \$SENTENCE$ - \$SENTENCE$ - \$SPOIL$ - \$TESS$ - \$SENTENCE$ - \$SPOIL$ - \$TESS$ - \$THAT$ - \$PIECES$ - \$THE$ - \$SORT$ - \$SORT$ - \$KIND$ - \$KINDS$ - \$SORT$ - \$TYPES$ - \$SORTS$ - \$KIND$ - \$KINDS$ - \$TYPES$ - \$VARIE$ - \$VARIE$ - \$TRANSA$ - \$TRANSA$
<STOP> - STOP
CEASE
TERMINATE
KILL
FINISH
QUIT
<THAT> - THIS
THAT
THOSE
<THESE> - IT
<THAT>
THEY
EACH
<PIECES> - <PIECES>
<ONE/S>
<THEM>
<THEM>
<THAT>
THEM
<TIME> - TIME
SPACE
TIME <CONJUNCTION> SPACE
SPACE <CONJUNCTION> TIME
<TOPIC> - <TOPIC>
PROBLEM SOLVING
GIPS
SCHLSS
STARNING
INFEERENCE
SYMBINONET
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
TEXTURAL ANALYSIS
A THALAMIC GEST
SHAPE TOPOLOGY
THREE DIMENSIONAL MODELS
A TUTOR OR TUTORING ON TV
THE WEAK LOGIC OF PROGRAMS
AIXF

THE DATES OF THE WORLD COMPUTER CHESS CONFERENCE
NEWSLETTER REPORTERS
OBJECT LOCATIONS AND MOVEMENTS IN NATURAL IMAGES
PARALLELISM IN PROBLEM SOLVING
THE PERFORMANCE OF PATTERN MATCHING RULES
A PROGRAM SYNTHESIZER FOR NETWORK PROTOCOLS
A PROGRAMMING APPRENTICE
A PROTOCOL CHECKER FOR PROTOCOL TERMINATION EXPRESSIONS
A RADIO INTERVIEW ON SCIENCE FICTION
A TIME-DOMAIN ANALYZER
INVARIA NCE S IN THE PERCEPTION OF FACES
THE LOCATION OF OBJECTS IN MAGAZINES
THE LOGICAL REDUCTION OF LISP DATA BASES
DATA BASES
A LOSING MOVIE
MACHINE INTELLIGENCE IN MEDICAL DIAGNOSIS
MANAGEMENT INFORMATION SYSTEMS
OBJECT MANIPULATING ROBOTS
AUTOMATIC MANTRA GENERATION
SYMBOL MAPPING IN BASEBALL
THE STOCK MARKET
THE METAMETHANOLIC SIMULATION OF MULTIPROCESS SOFTWARE
THE METAMATHMATICS OF MLISP OR MLISP2
MINIMAL SPANNING FORESTS OR TREES
MOTION IN SCENE DESCRIPTION
A MUL TICLEVEL ORGANIZATION
THE NOMINATION OF NOMINEES BY A NATIONAL NOMINATING COMMITTEE
NONDETERMINISTIC PROGRAMMING
MACRO PROCESSING FOR AN ON-LINE NEWSLETTER
THE ONTOLOGY OF NON-INDEPENDENT SUBPROBLEMS
OPERATIONAL REASONING
LANGUAGE PARADIGM
OPTIMAL PROBLEM SOLVING SEARCH
OPTIMIZED CODE FOR THE TRANSFER OF COMMENTS
A PACKET BASED APPROACH TO NETWORK COMMUNICATION
THE PARODY SIMULATION OF PARANOID
LINEAR CODE COMING
MEANS FOR COMPUTER MOVIES
LOW ORDER OF RECOGNITION PERFORMANCE
A TWO-PARTER
A THEOREM PROVER PLANNING FOR PROGRESS
THE STRUCTURE OF ANY VARIETY OF COMPUTER TERMINAL
A TAX MONITOR
A COMMON SENSE ALGORITHM
ACQUISITION OF KNOWLEDGE
ACTIVE KNOWLEDGE
CYCLIC AND ACYCLIC ISOMERS
ADAPTIVE PRODUCTION SYSTEMS
PRODUCTION SYSTEMS
ADVISING PHYSICIANS
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
CONTEXT
ANALYSIS OF SENTENCES
ASSIMILATION OF NEW INFORMATION
AUGMENTED TRANSITION NETWORKS
AUTOMATED DEDUCTION
DEDUCTION
AUTOMATIC CODING
AUTOMATIC COMPUTATION
AUTOMATIC PROGRAM SYNTHESIS FROM EXAMPLE PROBLEMS
AUTOMATIC PROGRAM WRITING
AUTOMATIC PROOF OF CORRECTNESS
AUTOMATIC THEOREM PROVING
AXIOMATIC SEMANTICS
BACKGAMMON
BELIEF SYSTEMS
BINDINGS
BIOMIMICRY
BRAIN THEORY
BUSINESS PROBLEM SOLVING
CARTOGRAPHY
CASE SYSTEMS
CAUSAL REASONING
CHECKING PROOFS
CHESS PLAYING PROGRAMS
CIRCUIT ANALYSIS
COGNITIVE ROBOTIC SYSTEMS
COMMON SENSE
COMMON SENSE THEOREY FORMATION
COMPLEX WAVEFORMS
COMPUTER ART
COMPUTER BASED CONSULTATIONS
COMPUTER CONTROLLED MANIPULATORS
COMPUTER GRAPHICS
COMPUTER MUSIC
COMPUTER VISION
CONCEPTUAL DESCRIPTIONS
CONCEPTUAL INHIBICE
CONCEPTUAL OVERLAYS
CONSTRAINT SATISFACTION
CONSTRUCTING PROGRAMS FROM EXAMPLES
CONSTRUCTION OF PROGRAMS
CONTINUOUS PROCESSES
COORDINATING SOURCES OF KNOWLEDGE
COPYING LIST STRUCTURES
CURVED OBJECTS
DATA BASES FOR INTERACTIVE DESIGN
DECISION THEORY
THE DEDUCTIVE PARADIGM
DENOTATIONAL SEMANTICS
DEPTH PERCEPTION
DERIVATION PLANS
DESIGN AUTOMATION
DESIGN IN THE ARTS
DETECTION OF LIGHT SOURCES
DISPLAY TERMINALS
DRAGON
DRIVING A CAR
DYNAMIC BINDING
DYNAMIC PROGRAMMING
ELECTRONIC CIRCUITS
ELECTRONICS
THE ENVIRONMENT
EXPERT SYSTEMS
EXPLANATION CAPABILITIES
FAIRY TALES OR FABLES
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
HEURISTIC 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
THE BERKELEY DEBATE
THE DRESDEN DEBATE
THE HISTORY OF AI
THE HUNGRY MUSHROOM
THE INNATE HEURISTIC
AXIOMS FOR GO
COMPUTER-BASED CONSULTANT
IMAGE INTENSITY UNDERSTANDING
TROUBLESHOOTING
LANGUAGE COMPREHENSION
- STIMULUS - DOUNDS
PERCEPTIONS
COMPUTER NETWORKS
GRAPH MATCHING
ASSOCIATIVE MEMORY
UNIFORM PROOF PROCEDURES
PLANNER-LIKE LANGUAGES
HILL CLIMBING
- STIMULUS - 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
PREFERENTIAL SEMANTICS
THE GAME OF POKER
PROCEDURAL EVENTS
PRICE'S TUTORIAL
PRODUCTIVITY TECHNOLOGY
A RELATION ANALYSIS SUBSYSTEM
REPRESENTING REAL-WORLD KNOWLEDGE IN RELATIONAL PRODUCTION SYSTEMS
ROBOTICS COOPERATION AND RESOURCE LIMITED PROCESSES
USING S-L-GRAPHS
RULE ACQUISITION CAPABILITIES
SCENE SEGMENTATION
SLOW 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
AT LECTURES
THE COMPUTERS AND THOUGHT AWARD
<MEMORY/S> - MEMORY
<MEMORIES>
<SIGHTS> - <EASY PIECES> - <RETROMES>
WHAT

WHAT

WHERE

WHAT

WHERE

WHAT

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

WHERE

WHAT

W
Appendix III-C-2  AI Retrieval Language Grammar: AI15

$SENT = [}$TOPIC$]

$SS = <$ANY PAPERS$> <$ABOUT TOPIC$>

$TOPIC = <$ANY JOURNALS$> <$ABOUT TOPIC$>

$TOPIC = <$ANY PAPERS$> <$ABOUT TOPIC$>

$TOPIC = <$ANY PAPERS$> IN <$JOURNAL$>

$TOPIC = <$ANY PAPERS$> SINCE <$DATE$>

$TOPIC = <$ANY PAPERS$> THAT MENTION THE <$DATES OF THE CONFERENCE$>

$TOPIC = <$ANY PAPERS$> WHICH <$CITE AUTHOR$>

$TOPIC = <$ANY PAPERS$> <$ABOUT TOPIC$>

$TOPIC = <$ANY JOURNALS$> <$ABOUT TOPIC$> BUT NOT <$TOPICS$>

$TOPIC = <$ANY PAPERS$> <$ABOUT TOPIC$>

$TOPIC = <$ANY PAPERS$> <$ABOUT TOPIC$> <$ALSO ABOUT TOPIC$>

$TOPIC = <$ANY PAPERS$> <$ABOUT TOPIC$> <$BY AUTHOR$>

$TOPIC = <$ANY PAPERS$> FROM <$CONFERENCE$>

$TOPIC = <$ANY PAPERS$> FROM <$JOURNAL$>

$TOPIC = <$ANY PAPERS$> FROM <$THE CONFERENCES$> IN THE MONTH OF <$DATE$>

$TOPIC = <$AUTHORS$> CITED BY <$ANY PAPERS$>

$TOPIC = <$AUTHORS$> CITED IN <$ANY PAPERS$>

$TOPIC = <$TOPIC$> <$MENTIONED$> ANYWHERE

$TOPIC = <$TOPIC$> <$MENTIONED$> IN <$ANY PAPER$>

$TOPIC = <$TOPIC$> <$MENTIONED$> IN <$ANY JOURNALS$>

$TOPIC = <$ANY$> <$BY AUTHOR$>

$TOPIC = <$ANY$> <$THIS SLOW$>

$TOPIC = <$ANY CONFERENCE$> <$MENTION TOPIC$>

$TOPIC = <$ANY CONFERENCE$> PUBLISH <$JOURNALS$>

$TOPIC = <$ANY JOURNALS$> <$MENTION TOPIC$>

$TOPIC = <$ANY PAPERS$> <$ABOUT TOPIC$> <$ALSO MENTION TOPIC$>

$TOPIC = <$ANY PAPERS$> <$ABOUT TOPIC$> <$MENTION TOPIC$>

$TOPIC = <$ANY PAPERS$> <$ABOUT TOPIC$> EXIST

$TOPIC = <$ANY PAPERS$> <$MENTION TOPIC$>

$TOPIC = <$ANY PAPERS$> <$CITE AUTHOR$>

$TOPIC = <$ANY PAPERS$> <$MENTION TOPIC$>

$TOPIC = <$ANY PAPERS$> <$MENTION TOPIC$> BUT NOT <$TOPICS$>

$TOPIC = <$ANY PAPERS$> <$THIS YEAR$> <$CITE AUTHOR$>

$TOPIC = <$AUTHORS$> PRESENT <$ANY PAPER$> AT <$THE CONFERENCES$>

$TOPIC = <$AUTHORS$> PRESENT <$ANY PAPER$> AT <$THE CONFERENCES$> IN <$DATE$>

$TOPIC = <$AUTHORS$> PRESENT <$ANY PAPER$> AT <$THE CONFERENCES$>

$TOPIC = <$AUTHORS$> PUBLISH <$ANY PAPER$>

$TOPIC = <$AUTHORS$> WRITE <$ANY PAPER$>

$TOPIC = <$AUTHORS$> WRITE <$ANY PAPER$> <$ANY$>

$TOPIC = <$AUTHORS$> WRITE <$ANY PAPER$> <$THIS YEAR$>

$TOPIC = <$THE AUTHORS$> <$MENTION TOPIC$>

$TOPIC = <$THE JOURNAL$> PUBLISH <$ANYTHING$> IN <$DATE$> OR <$DATE$>

$TOPIC = <$THE PAPERS$> <$CITE AUTHOR$>

$TOPIC = <$ANY$> QUILLES TAK THIS LONG

$TOPIC = <$ANY$> PUBLISH <$ABOUT TOPIC$> IN <$THE JOURNAL$>

$TOPIC = <$ANY$> RESPONSE EVER COME FASTER

$TOPIC = <$ANY$> WORK AT <$WORKPLACE$>

$TOPIC = <$ANY$> PUBLISH <$MENTION TOPIC$>

$TOPIC = <$ANY$> PUBLISH REFERENCE <$ANY JOURNAL$>

$TOPIC = <$TOPIC$> <$MENTION TOPIC$> <$MENTION TOPIC$> ANYWHERE

$TOPIC = <$ANY$> TAK THIS LONG TO ANSWER ME

$TOPIC = <$ANY$> <$ABOUT TOPIC$>

$TOPIC = <$ANY$> <$ABOUT TOPIC$>

$TOPIC = <$ANY$> <$ABOUT TOPIC$> IN <$DATE$>

$TOPIC = <$ANY$> <$ABOUT TOPIC$>

$TOPIC = <$ANY$> <$ABOUT TOPIC$> IN <$DATE$>
WHAT ARE THE KEY PHRASES
WHAT ARE THEIR AFFILIATIONS
WHAT HAS $AUTHOR$'S $PAPERS$ BEEN PUBLISHED LATELY
WHAT IS $IN UNITS$ AFFILIATION
WHAT IS KNOWN ABOUT EVERY ARTICLE
WHAT IS THE $ANDORAUTHORDATETITLE$ OF $QUANTITY$
WHAT IS THE $ANDORAUTHORDATETITLE$ OF $THE PAPER$
WHAT IS THE $ANDORAUTHORDATETITLE$ OF THAT PIECE
WHAT IS THE SIZE OF THE DATA BANK
WHEN WAS $HUMAN PROBLEM SOLVING$ PUBLISHED
WHEN WAS $THE PAPER$ PUBLISHED
WHEN WAS $TOPICS$ MENTIONED
WHEN WAS $PAPERS$ PUBLISHED LATELY
WHEN WAS THE LAST PAPER $AUTHOR$ PUBLISHED
WHICH $TOPICS$ ARE MENTIONED
WHICH AUTHORS WORK AT $PLACE$ OR AT $GEOPLACE$
WHICH AUTHORS WORK AT $WORKPLACE$ OR AT $PLACE$
WHICH AUTHORS WORK WITH $WORKPLACE$ OR AT $GEOPLACE$
WHICH OF THESE ARE ABOUT TOPICS ALSO MENTION TOPIC
WHICH OF THESE ARE ABOUT TOPICS MENTION TOPIC
WHICH OF THESE ARE $TOPICS$ $were$ WITTEN PUBLISHED LATELY
WHICH OF THESE ARE $TOPICS$ $were$ $AUTHOR$
WHICH OF THESE ARE $TOPICS$ $are$ $AUTHOR$
WHICH OF THESE ARE $TOPICS$ MENTIONED $AUTHOR$
WHICH OF THESE ARE $TOPICS$ $contained$ $PAPERS$ $AUTHOR$
WHICH OF THESE ARE $TOPICS$ $content$ $TOPICS$
WHICH OF THESE ARE $TOPICS$ REFER TO THESE
WHICH OF THESE WERE $AUTHOR$
CAN I HAVE $PAPERS$ LISTED
CAN YOU HELP ME
CHOOSE AMONG $JOURNALS$ BEFORE $DATE$
DURING WHAT MONTHS $EVERY$ THEY PUBLISHED GENERATE A COPY OF $DOE$ HAS $AUTHOR$'S $PAPERS$ $ANY$ THIS YEAR
HAS $AUTHOR$'S $PAPERS$ ANYTHING LATELY
HAS $AUTHOR$ BEEN REFERENCED IN $PAPERS$
HASN'T $AUTHOR$ PAPER BEEN RELEASED
HASN'T $TOPICS$ BEEN CONSIDERED IN $JOURNAL$
HAVE $PAPERS$ $AUTHOR$ APPEARED
HAVE $PAPERS$ $AUTHOR$'S PAPERS APPEARED ABOUT TOPIC
HAVE $AUTHOR$ $PAPERS$ LATESTLY PUBLISHED $THIS YEAR$
HASN'T YOU FINISHED
HELP
HOW BIG IS THE DATABASE
HOW CAN I USE THIS SYSTEM EFFICIENTLY
HOW LONG $SEQUENCE$ IS
I'D LIKE TO KNOW THE $ANDORAUTHORDATETITLE$ OF $THE PAPER$
LIST $QUANTITY$ UNPUBLISHED
LIST BETWEEN $QUANTITY$ $QUANTITY$ OF THEM
LIST THE $PAPERS$ $AUTHOR$
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 <$AUTHOR$ <$DATE$ <$TITLE$>
SUBSELECT FROM <$TOPICS$>
SURE THANKS
TELL ME <$WHAT TO DO$>
TELL ME THE <$AUTHOR$ <$DATE$ <$TITLE$> OF <$QUANTITY$>
THANK YOU <$WE'RE DONE$>
TRANSMIT <$QUANTITY$>
WHAT <$ABOUT TOPIC$>
WHAT <$SCAN$ TO SPEED YOU UP
WHAT <$DO I HAVE TO DO$>
WHAT <$IS IT <$AUTHOR$ <$DATE$ <$TITLE$>
WHAT <$JOURNAL$ <$DATE$ <$DATE$>$ <$DATE$ <$Mention TOPIC$>
WHAT <$PAPERS$ <$Mention TOPIC$>
WHAT <$SORT$ OF <$SUMMARY$> IS AVAILABLE
WHAT <$ADDRESS$ IS GIVEN FOR <$THE AUTHORS$>
WHAT <$ADDRESS$ ARE GIVEN FOR <$THE AUTHORS$>
WHAT CAN <$THE SYSTEM$>
WHAT CONFERENCE WAS AT <$WORKPLACE$> OR AT <$PLACE$>
WHAT CONFERENCE WAS AT <$WORKPLACE$> OR AT <$PLACE$>
WHAT <$FACT$ <$ARE$ <$STORY$I$>
WHAT <$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 <$SHOULD I ASK$>
WHAT <$SHOULD I SAY$>
WHAT <$SORTS$ OF <$TOPICS$ <$ARE$ <$MENTIONED$>
WHAT <$SUBJECT$ CAN I REQUIRE$>
WHAT <$TOPIC$ MENU$ CAN I CHOOSE$>
WHAT <$TOPICS$ <$ARE$ <$RELATI$ ON$ <$TOPICS$>
WHAT <$TYPES$ OF <$RETRIEVAL$ CAN I ASK DO$>
WHEN WILL YOU HAVE THE MENU$>
WHEN <$ARE$ <$TOPICS$ <$MENTIONED$>
WHEN <$DO$ <$THEY$ WORK$>
WHEN <$ID$ <$SAME$ PAPER$ AFTER$>
WHICH <$STAGE$ <$CONTAINS$ <$TOPICS$>
WHICH <$AUTHORS$ <$MENTION TOPIC$>
WHICH CONFERENCE WAS AT <$PLACE$> OR AT <$PLACE$>
WHICH <$IS$ <$QUANTITY$>
WHICH <$NOTES$ <$ABOUT TOPIC$ <$ALSO$ <$MENTION TOPIC$
WHICH <$ONE$>
WHICH <$SORT$ OF <$RETRIEVAL$ CAN I ASK$>
WHICH <$TITLES$ <$Mention TOPIC$>
WHICH <$WAS$ THE LAST ARTICLE <$BY$ <$AUTHOR$>
WHICH <$HAS$ <$WRITTEN$ <$PUBLISHED$ <$ABOUT TOPIC$>
WHO <$WAS$ QUOTED IN <$THE PAPER$>
WHO WAS THE AUTHOR
WHO WERE THE AUTHORS OF THE PAPER
WHO WROTE THE 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
AIX15

<s>RETRIEVAL CAN HEARSAY DO</s> \ <s>RETRIEVAL CAN HEARSAY DO
<s>DOES IT TAKE</s> \ DOES IT TAKE
<s>DATES OF THE CONFERENCE</s> \ DATES OF <s>THE CONFERENCE/S</s>
<s>WHAT TO DO</s> \ WHAT TO DO
<s>CAN I DO</s> \ CAN I DO
<s>THE SYSTEM DO</s> \ THE SYSTEM DO
<s>DO I HAVE TO DO</s> \ DO I HAVE TO DO
<s>THE AUTHORS</s> \ THE AUTHORS
ANY AUTHORS

<s>CEASE</s> \ <s>CEASE PRINTING</s>
PLEASE <s>CEASE PRINTING</s>
<s>CEASE PRINTING</s> PLEASE
<s>CEASE PRINTING</s> \ <s>CEASE</s> <s>PRINTING</s>
<s>CEASE</s> \ CEASE
STOP
TERMINATE
FINISH
QUIT
KILL THE
<s>PRINTING</s> \ PRINTING
LISTING
TRANSMITTING
SHOW MANY • HOW MANY
SHOW MANY PAPERS • HOW MANY PAPERS
HOW MANY OF THESE
ARE THERE • ARE THERE
DO YOU HAVE
DO YOU HAPPEN TO HAVE
ARE THERE • ARE THERE
WERE
WERE NOT
DO
DON'T
DID
DIDN'T
DOES
DOESN'T
GET ME • GET GIVE ME
GET GIVE
TRY TO GET
TRY TO GET ME
COULD YOU RETRIEVE
I WE
I'D
ID
WE'D
SEE
GET
DON'T GET ME • DON'T GET ME
WHO IS
WHO IS
IS
IS
IS
IS
IS
IS
ARE
ARE
NOT
NOT
HERE
HERE
HER
HER
WHAT HAS
WHAT HAS
HAS
HAS
WHEN WAS
WHEN WAS
WHEN WERE
<DO THEY WORK>  +  <DO THEY WORK
<DO THEY>  +  <$DO:DO> THEY
<$DO:DOESNT>  +  <$HESH:HE
<HESH:HE>  +  HE
SHE

<$WERE:INTERESTED IN>  +  <$WERE:INTERESTED IN
<$WERE:ONLY>  +  <$INTERESTED IN
THE AREA: <$WERE>  +  <$INTERESTED IN> IS
THE ONLY AREA: <$WERE>  +  <$INTERESTED IN> IS
<LET'S:RESTRICT>  +  <$OURSelves:OURSelves> TO

<$WERE:WERE>  +  <$WERE:WERE
WE'VE BEEN
WE HAVE BEEN
<<I'M>

<<I'M:IM>  +  <I'M
I AM

<$INTERESTED IN>  +  <$INTERESTED IN
<LET'S:LETS>  +  <LET'S
LET <$US:US>

<US:US>  +  US

<RESTRICT:RESTRICT>  +  <$OURSelves:OURSelves
CONFINE
LIMIT

<$OURSelves:OURSelves>  +  <$OURSelves:OURSelves
OUR ATTENTION
OURSELVES
MYSELF

<$WHAT ABOUT>  +  <$WHAT ABOUT
<$WHAT ARE>  +  <$WHAT ARE
<$WHAT IS>  +  <$WHAT IS
WHAT'S

<$WHICH AUTHORS>  +  <$WHICH AUTHORS
<$WHICH OF THESE>  +  <$WHICH OF THESE
WHICH PAPER

WHICH <$PAPERS:
WHICH <$JOURNAL:
WHICH <$JOURNALS:

<$THESE:THEM>  +  <$THESE:THEM
THEM
$\text{QUANTITY} = \text{THE LATEST} < \text{NUMBER1-99} \\
\text{THE LATEST} < \text{NUMBER1-99} \\
\text{THE LATEST} < \text{NUMBER1-99} \\
\text{UP TO} < \text{NUMBER1-99} \\
< \text{LAST} > = \text{LAST} \\
\text{NEXT} \\
\text{FIRST} \\
< \text{LATEST} > = \text{EARLIEST} \\
\text{LATEST} \\
\text{NEWEST} \\
\text{OLDEST} \\
\text{MOST RECENT} \\
< \text{NUMBER} > = < \text{NUMBER1-99} > \\
< \text{HUNDREDS} > < \text{NUMBER1-99} > \\
< \text{HUNDREDS} > < \text{DIGITS} > \text{HUNDRED} \\
A \text{HUNDRED} \\
< \text{NUMBER1-99} > < < \text{NUMBER2} > \\
< < \text{TEENS} > \\
< < \text{DIGITS} > \\
< \text{NUMBER2} > < < \text{TENS} > \\
< < \text{TENS} > < < \text{DIGITS} > \\
< < \text{TEENS} > < \text{NINETEEN} \\
\text{TEN} \\
\text{ELEVEN} \\
\text{TWELVE} \\
\text{THIRTEEN} \\
\text{FOURTEEN} \\
\text{FIFTEEN} \\
\text{SIXTEEN} \\
\text{SEVENTEEN} \\
\text{EIGHTEEN} \\
< < \text{TENS} > < \text{TWENTY} \\
\text{THIRTY} \\
\text{FOURTY} \\
\text{FIFTY} \\
\text{SIXTY} \\
\text{SEVENTY} \\
\text{EIGHTY} \\
\text{NINETY} \\
< < \text{DIGITS} > < \text{ONE} \\
\text{TWO} \\
\text{THREE} \\
\text{FOUR} \\
\text{FIVE} \\
\text{SIX} \\
\text{SEVEN} \\
\text{EIGHT} \\
\text{NINE}
TRANSACTIONS
VOLUMES
<JOURNAL>• V • ISSUES
<OF THE ADJ. CONFERENCE> • OF THE <ADJ.:JOURNAL>
OF THE <ADJ.:JOURNAL> <CONFERENCE>
A Conference is an ADJ:Conference. Another ADJ:Conference is a Recent ADJ:Conference. A Recent ADJ:Conference is a Recent Current ADJ:Conference. The Next Line should be an ADJ:Conference. An ADJ:Conference is an ADJ:Conference? An ADJ:Conference? is A1

\[ \text{A Conference} \]
\[ \text{ACM} \]
\[ \text{IEE} \]
\[ \text{IFIP} \]
\[ \text{A Conference} = \text{Conference} \]
\[ \text{Convention} \]
\[ \text{Meeting} \]
\[ \text{Session} \]
\[ \text{Conferences} = \text{Conferences} \]
\[ \text{Conventions} \]
\[ \text{Meetings} \]
\[ \text{Sessions} \]
\[ \text{Recent Current} = \text{Recent Current} \]
\[ \text{New} \]
\[ \text{Any Conferences?} = \text{Any Recent ADJ:Conferences?} \]
\[ \text{Recent ADJ:Conferences?} = \text{Recent Current ADJ:Conference} \]
\[ \text{Recent Current ADJ:Conference?} = \text{Recent Conference?} \]
\[ \text{ADJ:Conference?} = \text{Conference?} \]
\[ \text{Conferences?} = \text{Conferences?} \]
\[ \text{The Conferences?} = \text{The Recent ADJ:Conferences?} \]
AXIS

AXIOMATIC SEMANTICS
AXIOMS FOR GO
BACKWARD CHAINING
CASE SYSTEMS
CATEGORIZATION
BIOMETRIC
MIND-THEORY
BUSINESS-PROBLEM SOLVING
GEOGRAPHY
CASE SYSTEMS
CAUSAL-REASONING
CELL-ASSEMBLY THEORY
CHECKING PROOFS
CHESS
CHESS-PLAYING PROGRAMS
CIRCUIT ANALYSIS
COGNITION
COGNITIVE ROBOTICS SYSTEMS
COGNITIVE SCIENCE
COMMON SENSE
COMMON SENSE THEORY FORMATION
COMPUTER WAVEFORMS
COMPUTATIONAL LINGUISTICS
COMPUTER ART
COMPUTER BASED CONSULTATION
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
COOPERATING SOURCES OF KNOWLEDGE
COPYING LIST STRUCTURES
COPYING OBJECTS
COPYING TECHNICS
CYCLES
DATABASES
DATABASES FOR INTERACTIVE DESIGN
DATA STRUCTURES
DECISION THEORY
DEDUCTION
DEDUCTIVE RETRIEVAL
DENOTATIONAL SEMANTICS
DEPTH PERCEPTION
DESIGNATION PLANS
DESIGN
DESIGN AUTOMATION
DESIGN IN THE ARTS
INTRODUCTION
INVARINACE FOR PROBLEM SOLVING
INVARIANCES IN THE PERCEPTION OF FACES
INVESTIGATION
ITERATION
KNOWLEDGE BASED SYSTEMS
KNOWLEDGE SYSTEMS
LAMBDACALCULUS
LANGUAGE COMPREHENSION
LANGUAGE DESIGN
LANGUAGE PARAPHRASE
LANGUAGE PASCAL
LANGUAGE PRIMITIVES
LANGUAGE UNDERSTANDING
LARGE DATA BASE
LEARNING
LINEAR LEXICOMATARY
LOW ORDERS OF RECOGNITION PERFORMANCE
MACHINE INTELLIGENCE IN MEDICAL DIAGNOSIS
MACRO PROCESSING FOR AN ONLINE NEWSLETTER
MANAGEMENT INFORMATION SYSTEMS
METHODS FOR COMPUTER MOVIES
MEDICAL CONSULTATION
MINIMAL SPANNING FORESTS ON 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
PATH AN MATCHING
PATH RECOGNITION
PERCEPTIONS
PHOTOGRAMMETRY
PICTURE RECOGNITION
PLANNER LIKE LANGUAGES
PREDICATE CALCULUS
PREFERENTIAL SEMANTICS
PRIMITIVES TUTORIAL
PROBLEM SOLVING
PROCEDURAL EVENTS
PRODUCT SYSTEMS
PRODUCTIVITY TECHNOLOGY
PROGRAM VERIFICATION
PSYCHOLOGY
RECOGNITION DEVICES
REPRESENTING REAL-WORLD KNOWLEDGE IN RELATIONAL PRODUCTION SYSTEMS
RESEARCH ON REASONING
RESOURCE LIMITED PROCESSES
RETRIEVAL
ROBOTICS COOPERATION
UNDERSTANDING
UNIFORM PROOF PROCEDURES
USING S.L-GRAPHS
VISUAL COMMUNICATION
VISUAL PLANES I: 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
AR. <$TOPICS$ 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 ISSUES 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

CLEAR PRINTING

CHOOSE AMONG VOLUMES BEFORE NINETEEN SIXTY
COULD YOU RETRIEVE SOMETHING FROM <$INFORMATION-AND-CONTROL> DISCUSING <$TOPICS$>
DID <$AUTHOR$'S> PRESENT A PAPER AT IFIP?
DID <$AUTHOR$'S> PRESENT A PAPER AT THE IFIP MEETINGS IN SEPTEMBER
DID <$AUTHOR$'S> PRESENT PAPERS AT IFIP?
DID <$AUTHOR$'S> PRESENT PAPERS AT IFIP?
DID <$AUTHOR$'S> PUBLISH A PAPER
DID <$AUTHOR$'S> WRITE A BOOK
DID <$AUTHOR$'S> WRITE A BOOK RECENTLY
DID <$AUTHOR$'S> WRITE A PAPER THIS YEAR
DID ANY <$JOURNAL$> PAPERS CITE <$AUTHOR$'S>
DID ANY ACM PAPERS CITE <$AUTHOR$'S>
DID ANY IFIP CONVENTIONS PUBLISH PROCEEDINGS
DID ANY OF THOSE PAPERS CITE <$AUTHOR$'S>
DID ANYONE PUBLISH ABOUT <$TOPICS$> IN COMMUNICATIONS OF THE ACM
DID THE SIGART NEWSLETTER PUBLISH ANYTHING IN OCTOBER OR NOVEMBER
DIDN'T THAT PAPER CITE <$AUTHOR$'S>
DO ANY QUESTIONS TAKE THIS LONG
DO ANY ARTICLES ON <$TOPICS$> IN ADDITION CONSIDER <$TOPICS$>
DO ANY ARTICLES ON <$TOPICS$> MENTION <$TOPICS$>
DO ANY ARTICLES REFER TO <$TOPICS$>
DO ANY AUTHORS DISCUSS <TOPICS>?
DO ANY NEW ARTICLES MENTION <TOPICS>?
DO ANY OF THE ABSTRACTS MENTION <TOPICS>?
DO ANY OF THESE ALSO DISCUSS <TOPICS>?
DO ANY OF THESE ALSO MENTION <TOPICS>?
DO ANY OF THE CITATION <AUTHORS>?
DO ANY OF THESE MENTION <TOPICS>?
DO ANY OF THOSE PAPERS MENTION <TOPICS>?
DO ANY PAPERS ABOUT <TOPICS> ALSO CONSIDER <TOPICS>?
DO ANY PAPERS CITE <AUTHORS>?
DO ANY PAPERS DISCUSS <TOPICS>?
DO ANY PAPERS DISCUSS <TOPICS> BUT NOT <TOPICS>?
DO ANY PAPERS ON <TOPICS> EXIST?
DO ANY PAPERS THIS YEAR CITE <AUTHORS>?
DO ANY RECENT ACM CONFERENCE CONSIDER <TOPICS>?
DO ANY RECENT BOOKS CITE <AUTHORS>?
DO ANY RECENT BOOKS MENTION <TOPICS>?
DO ANY RECENT JOURNALS DISCUSS <TOPICS>?
DO ANY RECENT SUMMARIES DISCUSS <TOPICS>?
DO MANY ABSTRACTS DISCUSS <AUTHORS>?
DO MANY ABSTRACTS DISCUSS <TOPICS>?
DO THEY WORK AT THE GM RESEARCH LABS?
DO YOU HAPPEN TO HAVE ANY RECENT PAPERS ON <TOPICS>?
DO YOU HAVE ANY ARTICLES ON <TOPICS>?
DO YOU HAVE ANY NEW PAPERS ON <TOPICS>?
DO YOU HAVE ANY RECENT PAPERS ON <TOPICS>?
DO YOU HAVE ANY SUMMARIES ABOUT <TOPICS>?
DO YOU HAVE NEW PAPERS ON <TOPICS>?
DOES <TOPICS> GET DISCUSSED ANYWHERE?
DOES <TOPICS> GET MENTIONED ANYWHERE?
DOES HE WORK AT CMU?
DOES IT ALWAYS TAKE THIS LONG TO ANSWER ME?
DOES SHE WORK AT THE INSTITUTE FOR SEMANTIC AND COGNITIVE STUDIES?
DOES THAT ARTICLE MENTION <TOPICS>?
DOESN'T THIS PAPER REFERENCE AN IFF TRANSACTION?
DON'T GET ME ANY ARTICLES WHICH MENTION <TOPICS>?
DURING WHAT MONTHS WERE THEY PUBLISHED?
FINISH PRINTING?
GENERATE A COPY OF THOSE?
GET ME ANY BOOKS WRITTEN BY <AUTHORS>?
GET ME EVERYTHING ON <TOPICS>.
GIVE ME ANY ABSTRACTS MENTIONING <TOPICS> BUT NOT <TOPICS>?
GIVE ME ANY ARTICLES ABOUT <TOPICS>.
GIVE ME ANY PAPERS ON <TOPICS> FROM JUNE TILL AUGUST.
GIVE ME ONE MORE PLEASE.
GIVE ME SOMETHING MENTIONING <TOPICS>.
GIVE ME THE DATE OF THAT ABSTRACT.
GIVE THE AUTHOR AND DATE OF EACH.
HAS <AUTHORS> BEEN REFERENCES IN ANY OF THOSE?
HAS <AUTHORS> PUBLISHED ANY PAPERS THIS YEAR?
HAS <AUTHORS> PUBLISHED ANYTHING RECENTLY?
HASN'T <TOPICS> BEEN CONSIDERED IN COMPUTER REVIEWS?
HASN'T A CURRENT REPORT ON <TOPICS> BEEN RELEAS?
HAVE <AUTHORS> PUBLISHED THIS YEAR?
HAVE ANY ARTICLES APPEAR WHICH MENTION <TOPICS>?
HAVE ANY NEW PAPERS BY <AUTHORS> APPEARED?
HAVEN'T YOU FINISHED?
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$> AND NOT <$AUTHOR$>
HOW MANY BOOKS DISCUSS <$TOPICS$>
HOW MANY BOOKS WERE PRODUCED FROM MARCH TO DECEMBER
HOW MANY BOOKS WERE WRITTEN BY <$AUTHOR$>
HOW MANY OF THESE ALSO DISCUSS <$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$> WRITTEN SINCE JANUARY
HOW MANY PAPERS REFER TO <$TOPICS$>
HOW MANY PAPERS THIS YEAR DISCUSS <$TOPICS$>
HOW MANY PAPERS WERE WRITTEN BY <$AUTHOR$>
HOW MANY RECENT ISSUES CONCERN <$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$> BUT NOT <$AUTHOR$> CITED IN SOME OF THOSE ARTICLES
IS <$AUTHOR$> CITED BY THOSE ABSTRACTS
IS <$AUTHOR$> CITED IN ANY OF THESE
IS <$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$>
IS THERE AN AIP CONVENTION ISSUE FROM MAY OR JUNE
IS THERE ANYTHING NEW REGARDING <$TOPICS$>
ISN'T <$TOPICS$> MENTIONED IN AN ABSTRACT
ISN'T THERE AN ARTICLE ABOUT <$TOPICS$>
KILL THE LISTING
LET ME LIMIT MYSELF TO REPORTS ISSUED SINCE NINETEEN FIFTEEN
LET US CONFINCE OURSELVES TO JOURNALS AFTER FEBRUARY NINETEEN FIFTY
LET'S INSTRUCT OUR ATTENTION TO PAPERS SINCE NINETEEN SEVENTY FOUR
LIST BETWEEN TWELVE AND TWENTY OF THEM
LIST THE ABSTRACTS BY <$AUTHOR$>
LIST THE NEXT FOUR HUNDRED
NO MORE PLEASE
NO THANKS
OK
PLEASE HELP ME
PLEASE LIST THE AUTHORS
PLEASE MAKE ME A FILE OF THOSE
PLEASE TERMINATE TRANSMITTING
PRINT THE NEXT ONE
PRODUCE A COPY OF THE NEWEST EIGHTY ARTICLES
QUIT LISTING PLEASE
SELECT FROM ARTICLES ON <$TOPICS>
SHOW ME ITS PUBLISHER
SHOW ME THE LATEST EVEN
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 <$TOPICS>
WE'RE ANY OF THESE ARTICLES WRITTEN BY <$AUTHOR/S>
WE'RE ANY OF THESE PUBLISHED IN THE SUNSHINE STATE OR IN THE US
WE'RE ANY OF THESE WRITTEN BY <$AUTHOR/S>
WE'RE ANY PUBLISHED AFTER JANUARY NINETEEN SIXTY-FIVE
WE'RE THEIR ANY ARTICLES ABOUT <$TOPICS>
WE'RE NOT SOME ARTICLES PUBLISHED ON <$TOPICS>
WHAT ABOUT <$AUTHOR/S>?
WHAT ABOUT <$TOPICS>?
WHAT ADDRESS IS GIVEN FOR THE AUTHORS
WHAT ADDRESSES ARE GIVEN FOR THE AUTHORS
WHAT ARE SOME OF THE AREAS OF <$TOPICS>?
WHAT ARE THE KEY PHRASES
WHAT ARE THE TITLES OF THE RECENT ARPA SURVEYS
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

119
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 ONLY
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 <$AUTHOR/S>
WHICH RECENT JOURNALS REFER TO <TOPICS>
WHICH SORT OF <RETRIEVAL-KEYS> CAN I SEEK
WHICH STORIES IN THE SIGART NEWSLETTER HAVE BEEN DISCUSSING <TOPICS>
WHICH SUMMARY ON <TOPICS> CONSIDER <TOPICS> IN ADDITION
WHICH TECHNICAL PAPERS WERE WRITTEN BY <$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
AUTHOR/S: ALLEN COLLINS

ALLEN NEWELL
ANN RUBIN
ANTHONY MARTELLI
AZRIEL ROSENFELD
BERNARD MILLIZER
BERT RAFAEL
BILL WOODS
BONNIE NAGIH-WEBBER
BRUCE BUCHANAN
CARL HEWITT
CHRISTOPHER RIESBECK
CHUCK RIEGER
DANNY ROBOROW
DAVE RUMFIHART
DAVID MARR
DAVID MICHE
DICK SELTZER
DONALD NOHMAN
DOUG LENAT
DREW MCIDEMOTT
DREYFUS
EARL HUNT
EARL SACKRIO
ED FEIGENBAUM
ED RISEMAN
ELLIOT SOLOWAY
ERIK SANDVEWALL
EUGENE CHARNI
FEIGENBAUM
FELDMAN
GARY HENDRIX
GEORGE ERNST
GIPS
HANS BERLINER
HARRY HARKOW
HERB SIMON
HERBERT BLOCK
HILARY PUTNAM
HOLLAND
HUGH NAGEL
IRV SOREL
ISSAC ASIMOV
JACK MINNER
JACK MOSTOW
JAMES SLAGE
JEAN SAMMET
JEFFREY UELMAN
JERRY FELDMAN
JOHN GASCHNIG
JOHN HOLLAND
JOHN MCCARTHY
JOHN NEWCOMER
JOSEPH WILFENBAUM
JULIE PEARL
KARL PINGS LI
KLITH PRICE
KEND Colby
KON RALSTON
KING SONG FU
LAUREN SIKLOSVY
LEE ERMAN
LEONARD LIHR
LES EARNEST
LINDA MASINTER
MADELINE BATES
MARVIN MINSKY
MARY NEWHORN
MARY SHAW
MICHAEL ARITTH
MIKE RYCHNER
MINSKY
MITCHELL KEWEY
NEWELL
NILS NILSSON
NILSSON
NORI SUZUKI
PAMFIA MCCORDUCK
PAT WINSTON
PENNY THORNOYKE
PETER KIGEL
RAJ REDDY
RAVAN Banerji
RAYMOND SPROULL
REDDY
RICH FIKE
RICH SMITH
RICHARD MICHALSKI
RICHARD WALDINGER
RICK HAYES ROTH
ROBERT FELLER
ROCK SCHAUK
RON OHLANDER
ROSENBERG
SCOTT FAHMAN
SEYMOUR PAPERT
SIMON
STEVE COLES
STEVE REED
STEVE ZUCKER
TED SHORTLIFFE
TERRY WINGRAO
THOMAS MARGAND
THOMAS SYKES
LIHR
VIC LESSER
WALLY RHOMBERG
WOODS
WOODY BLEINODE
WOGICK WILKS
ZOHAR MANNA
INFORMATION
INVARIANCE FOR PROBLEM SOLVING
INVARIANCES IN THE PERCEPTION OF FACES
INVESTMENT ANALYSIS
ITURATION
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 RILL WOODS
PARALLELISM IN PROBLEM SOLVING
PARTIAL EVALUATOR
PATTERN DIRECTED FUNCTION INVOCATION
PATTERN MATCHING
PATTERN RECOGNITION
PERCEPTIONS
PHOTOGRAMMETRY
PICTURE RECOGNITION
PLANNER-LIKE LANGUAGES
PREDICATE CALCULUS
PRETERIENIAL 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
RISK ACQUISITION CAPABILITIES
SECRET SIGNIFICATION
SEMANTIC HITS
A SEMANTIC NETWORK
SEMANTIC NETWORKS
SENTENCE MEANING IN CONTEXT
SENTENCE MEANING IN CONTEXT
SERIAL PATTERN ACQUISITION
SEVERAL GOALS
SHAPE GRAMMARS
SHAPE TOPOLOGY
SIMILARITY OPERATIONS
SHARING 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
TELEOLOGICAL REASONING
TEMPORAL SCENE ANALYSIS
TEXTURE ANALYSIS
The ARTICLE BY ALLAN NEWELL
The BAY AREA CIRCLE
The BERKELEY DEBATE
The COMPUTERS AND THOUGHT AWARDS
The DATES OF THE WORLD COMPUTER CHESS CONFERENCE
The DEDUCTIVE PATHFINDER
The DRESDEN 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 LIFE DATA Bases
The META SYMBOLIC SIMULATION OF MULTIPROCESS SOFTWARE
The META-SYMBOLICS OF MISP OR MISP2
The NOMINATION OF NOMINEES BY A NATIONAL NOMINATING COMMITTEE
The ONTOLOGY OF NON-INDEPENDENT SUBPROBLEMS
The PARODY SIMULATION OF PARAIDA
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 NITCH PROGRAM
The WEAK LOGIC OF PROGRAMS
TEXTUAL SEMANTIC MODELS
TEXT COMPLEXITY
TIME OR SPACE ROUNDS
TROUBLE SHOOTING
UNDERSTANDING
UNITOPM FOOD PROCEDURES
USING 3-D GRAPHS
VISUAL COMMUNICATION
VISUAL PLANES IN THE RECOGNITION OF POLYHEDRA
IV. COLLECTED PAPERS


A FUNCTIONAL DESCRIPTION OF THE HEARSAY-II SPEECH UNDERSTANDING SYSTEM

Lee D. Erman
Department of Computer Science
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 (Erman75, Erman76, 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:

"What abstracts refer to theory of computation?"
"List those articles."
"What hasinsky written since nineteen seventy-four?"

The vocabulary contains 101, 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 57), 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 LABELING

An input utterance is spoken into a medium-quality Electro-Voice RE-51 close-speaking headphone microphone in a fairly noisy environment (65 dB). The audio signal is low-pass 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 ZAPCASH) 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 ZAPCASH 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 class segment is its duration, absolute amplitude, and amplitude relative to its neighboring segment (i.e., peak, valley, or plateau). The segments are contiguous and non-overlapping, with one class designation for each.

Finally, the SEG KS does a finer labeling of each segment. The labels are allopheonic, there are currently 98 of them. Each of the 98 labels is defined by a vector of autocorrelation coefficients (FM76). These templates are generated from speaker-dependent training data that have been non-novel. The result of the labeling process, which matches the central portion of each segment against each of the templates using the Itakura metric, is a vector of 98 numbers, i.e., the number is an estimate of the negative log probability that the segment represents an occurrence of the ith allophone in the set.

The next generation of words, bottom-up, is accompanied by a three-step process.

First, using the labeled segments as input, the PCM knowledge source (5MW90) generates hypotheses for likely syllable classes. This is done by first identifying syllable nuclei and 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 off-line program which is trained on manually annotated utterances. The current training, which covers over 600,000 segments, results in about 300 word tokens. For each nucleus position, several competing syllable-class hypotheses are generated — typically three to eight.

The syllable classes are used to hypothesize words. Each of the 10,137 words in the vocabulary is specified by a pronunciation description. For word hypothesization purposes, an inverted form of the dictionary is used, in which there is an entry for each syllable-class and the words which have some pronunciation containing that syllable-class. The VCO 95 (5MW95) uses 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 10,137-word vocabulary are generated in each syllable classification.

Finally, the generated word candidates are rated and the best and end-state adjusted by the WIZARD knowledge source (5MW77). For each word in the vocabulary, WIZARD has a network which describes 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 edge. For each segment, the rating is then based on the difference between the best path and the segment edge costs.  

The result of the word processing so far is a set of words. Each word includes a base form, an ending, and a confidence rating. A policy KD, and WOCOT, limit control, selects a subset of these words, according to their bases and qualities, to be hypothesized. It is these selected word hypotheses that form much of the base for the “top-down” processing that now begins. In particular, these selected word hypotheses include about 95% of the words actually spoken in these correct word hypotheses. With each correct hypotheses making a rating which ranges from the average of three, compared to the five to twenty-five or so hypotheses with complete confidence, with significantly overlapping times. The non-selected words are retained internally by WOCOT to provide possible later hypothesization.

**TOP-DOWN PROCESSING**

Word Generation

WOSEQ knowledge source (5MW77) 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 is then ranked by a statistical grammar indicating the probability of a word sequence being correct. This ranking is then used to select which sequence to use as the basis for further processing. Each sequence begins with a single-word hypothesis and is built up to a multi-word sequence. It is used rather than whole-word hypotheses, because of the relative 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 sequences:

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 1.77 * 10^12) thus defines “language adjacency.”

Acoustic phonetic knowledge, embodied in the WOCOT (5MW76) subroutines, which generates word hypotheses and is used to decide if that pair might be considered to be time-adjacent in the utterance. WOCOT uses the dictionary, in conjunction with the sentences that have been processed in making its decision about which pair to accept.

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 next of these word sequences (which occasionally includes single words) are hypothesized.

The top-down processing is started by the creation of these word sequence hypotheses. Subsequently, WOCOT may generate additional hypotheses in the recognition process itself, not to be making progress based on those already hypothesized. These additional hypotheses may include shorter, decomposed versions of some of the original ones.

**WOCOT Processing**

Because the syntactic constraints used in the generation of word sequences are only part-wise, a sequence longer than two words may not be syntactically acceptable. A component of the 5MW77, 5MW94 knowledge source (5MW94) can parse a word sequence of arbitrary length, using the fixed constraint known as the language. In a sequence hypotheses that are not “correct,” the hypothesis is marked as “flawed.” Otherwise, a phrase hypothesis is created associated with the phrase hypotheses of the word sequence at which it is combined, as well as information about the way the words are parsed.
Word-Phrases from Parses

Another component of the SASS knowledge source can, for any phrase hypothesis, generate predictions of all words which can immediately precede and which can immediately follow the phrase in the language. In doing the computation to generate these predictions, the 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 must also be generated. First, if the word has been hypothesized previously and passes the test for time-synchrony (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 state of WIZARD to see if the candidate can be matched by a previously generated candidate which may not have been hypothesized. Again, JUNCT makes a judgment about time-synchrony. Finally, WIZARD compares its word-phonotactic network to the segments in an attempt to verify the prediction.

For each of the different pairs of verification, the approximate time and/or time of the predicted prediction to the right of the phrase is taken to be the end-time of the phrase. The end-time of the right of the predicted word is known and in fact, one requirement of the verification step is to generate an approximate end-time for the verified word. In general, several different "versions" of the word may be generated which differ primarily in the end-time, since no context to the right of the predicted word is given, several different estimates of the end-beginning of the word may be plausible based solely on the segments-information.

Word-Phrase Completion

For each verified word and its predicting phrase, a new and longer phrase hypothesis is generated. This process is carried out by a component of SASS similar to the Word-Sequence recognition component, matching the words of the original phrase against the new verified word. The extended phrase is then hypothesized and includes a rating based on the ratings of the words that compose it.

Complete Sentences via Matching Constraints

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 cover as the first and last "words" 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 as described above ensure that the first, such complete spanning hypotheses found will have the highest ratings of all possible spanning sentences 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 in future further consideration of other word hypotheses in the vicinity of the spanning ratings, and are unique to be extended into spanning hypotheses with ratings higher than the best already-discovered spanning sentence. This heuristic pruning procedure is based on the form of the rating function, and the way the phrase is derived from its constituent words. The pruning procedure considers each partial case and uses the ratings of other word hypotheses in the tree areas, not covered by the phrase to determine if the phrase might be extendable to a phrases rated higher than the spanning hypothesis, if not, the partial phrase is pruned. This pruning process and the rating and ranking processes are discuss in [4080?].

The recognition processing 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 combinations of the grammar and the likelihood of finding some partial product that is rated above the absolute reject or limit, this form of termination happens when the pruning procedure has been effective and has eliminated all competitors. Second, the exhaustion of a preassigned amount of computing resources (time or space) also halts the recognition process. The value thresholds used are set relative to the tree-criteria that reference hypotheses in similar sentences i.e., at 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 part of the recognized sentence. This selection is made by the highest-rated such hypothesis is chosen, otherwise, a selection of several of the highest-rated of the partial phrase hypotheses is made, using the selection to the lowest ones which tend to be spanned by the best.

Attested Rules

The topmost processing operations include (a) word- and generation, (b) word sequence parsing, (c) word sequence generation, (d) word segmentation, (e) word verification, and (f) word completion. In addition, there are also the repeat parsing and repeat verifying of each word, as well as the generation of each word. In general, the selection 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.

In many cases, the hear-say-lll system has a statistical, fixed scheduler [4080?], 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 filling the overall system gap of recognizing the utterance. The calculation is based on information specified when the action is triggered. For example, the word verifier is triggered when a word is generated from a word 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 preceding 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 [HaGo77] 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:** REQUEST
- **Subtype:** QUERY/AUTHORDATE
- **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 (for competitiveness) as compared to the other partial sentences.

The DISCO knowledge-source [HaGo77] accepts the generated 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**


[GaRo77] H. Goldberg, R. Reedy, and J. Gil, "The ZARFAS Parsers, Feature Extraction, Segmentation, and Labeling for Speech Understanding Systems", in [CmSp77].

[HaGo77] F. Hayes-Roth, G. Gil, and D. J. Moslow, "Discourse Analysis and Task Performance 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 (HARRY: Lowe, 1975), Hearsay-I (Reddy, 1973) and island-driven (ER: Paxton, 1975), SPCILIS (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, Kozierad, 1976). The best words found in this phase are chosen as the initial phonetic 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, 1975, Paxton and Robinson, 1975, and Woods' 1975 for a discussion of techniques for choosing the next hypothesis to extend; this strategy allows the prnasal 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: (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 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 there exists more than one sequence of steps to arrive at 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 might not in practice produce results as good 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 limited-breadth-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 hypotheses 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
two different approaches. These data were collected over
34 training utterances, with each island generation strategy
applied to all utterances. The average sentence length was
55 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 2.6 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 island in the initial utterance
position; the average number of islands hypothesized for the
initial position was twelve. 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, where the average number of correct words
hypothesized bottom-up is 30.

A MULTI-LEVEL ISLAND-DRIVEN STRATEGY

The strategy found to be most effective in the
Hearsay-II system as applied to a 1000-word vocabulary
with an average word (a total of 33) is to select multi-word
sequences of word hypotheses as starting islands for
syntactic-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 syntactic-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).

A word-sequence hypothesis is generated as a result of the word-splitting scan, can be reduced sharply by
applying a computationally inexpensive filter to the data.
This filter is based on simple kinds of grammatical and
correction 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; each
entry (i,j) in the matrix indicates whether word i can follow
word j in the grammar. If two words can follow each other,
they are called "language-adjacent." The correction
constraints are specified through another square matrix,
whose order is the size of the number of phones. Each
entry (i,j) in the matrix indicates whether two acoustic
segments are allowed in the juncture between two words,
the first word ending with the phone i and the second
word beginning with phone j. The appendix contains a
more detailed description of how the correction
constraints are implemented. If two words pass these
correction 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 word, 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 3
seconds on a PDP-10 KAIO); 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 time) with
each of them. The best starting island is the longest one
which has a very high probability of being correct, with
correctness taken precedence over length; correctness is a
function of both individual word validity ratings 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.
These with ratings less than some cutoff are discarded
unless doing so would leave less than five, in which case the
five lowest 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 CSCA, and R right-adjacent words are
found, then a right extension would have CSCA+R.

4. Only those extensions whose average word ratings
exceed a cutoff proportional to the square root of CSC are
formed. The direction chosen for extension is a function of
the 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 excessive. Thus,
Another level of filtering is performed, based on a rating attached to each word sequence. The rating of a word sequence is an increasing function of three quantities: (1) the duration-weighted average word rating, AVGRATE, computed by summing the product (word's rating x 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

\[ \text{RATE} = \text{AVGRATE} + 0.1 \times \text{NWORDS} \times \text{AVGRATE} + 0.5 \times \text{DUR} \]

The highest rated word sequence plus word sequences whose rating is some epsilon away from the highest rated sequence are used for further evaluation. In addition, another criteria 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 stabilizes, or an existing word sequence candidate has been found to have been ungrammatical or cannot be successfully extended; in those cases, these poorer-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 result of 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 chosen as evidence for further evaluation; in addition, another criteria 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 stabilizes, or an existing word sequence candidate has been found to have been ungrammatical or cannot be successfully extended; in those cases, these poorer-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 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 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 both strategies, the average number of phrases hypothesized are 37, 68 and 68, respectively.

**Conclusion**

The multi-word sequence generation procedure is a key knowledge source in Hearst's system. By exploiting the redundancy of the language to identify plausible word sequences and, incidently, 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, aksines, 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 tags produced by the Hearsay-II word hypotherizer. 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 times for acceptance of valid word pairs may be tightened, further increasing the speed and performance of Hearsay-II.

<table>
<thead>
<tr>
<th>CORRECT</th>
<th>INCORRECT</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCEPTED</td>
<td>156 (95.4%)</td>
<td>260 (41.5%)</td>
</tr>
<tr>
<td>REJECTED</td>
<td>5 (2.5%)</td>
<td>422 (58.4%)</td>
</tr>
<tr>
<td>TOTAL</td>
<td>161</td>
<td>712</td>
</tr>
</tbody>
</table>

Table 1. JUNCT performance 60 utterances)
Word Verification in the HEARSAY II Speech Understanding System

David M. McKeown, Jr.

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

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 RITOS speech recognition system (Lowratt, 1976), is given. An example of the verification of a word hypothesis using this algorithm is presented. Problems which arise 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 a predefined representation. The representation and the match of the acoustic signal may be evaluated at various representation levels such as the parametric, phonetic, and syntactic. Since errors are introduced and propagated as information is encoded from the parametric to the symbolic 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 on acoustic evidence. If 57 to 87 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 57 per word position), as we move to systems with larger vocabularies (10,000 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 to the scores should 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 hypothesizes at higher levels.

THE HEARSAY II ENVIRONMENT
Word verification is performed within HEARSAY II in the following environment. Word candidates may be supplied from a bottom-up word hypothesizer (PCOMAW) based on acoustic information or from a top-down syntactic and semantics knowledge source (SASS) based on syntactic information and constraints provided by the grammar POMAW (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) supplies 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 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 II operates under the hypothesis-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 1974) 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 encodes alternate pronunciations of the word. Each node in the word network represents a phone and arcs indicate successor/predecessor relationships between phones.
Figure 1 gives an example of the network for the word ABSTRACT. These networks are stored as a static data structure in a packed format. The uniform representation of words by a single network which embodies all speech dependent knowledge gives several advantages over other approaches. First, the generation of proper network structures can be handled on a case-by-case basis without the need for a general theory for all. This also eliminates the need for special case solutions when the general theory fails or is found incomplete. Tools are also available to generate word descriptions and tune the acoustic-phonetic templates (Lowrette, 1976).

The acoustic information is a segmentation of the utterance where each segment is represented as a vector of phone probabilities. WIZARD benefits from the use of the same templates and segmentation as the HAPPY system (Lowrette, 1976). As in HAPPY, the phone probabilities are distance measures between each segment and acoustic-phonetic templates in the phonetic dictionary. This value is a scalar 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 embedded 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 illustrates the matching of the word ABSTRACT to ten segments of an utterance. The performance score for any phone in the 5th segment can be calculated from the following:

\[ S_j = M_{j-1} + P_j \]

Where \( M_{j-1} \) is the best match score in the previous segment for phone \( j \) Where

\[ J = 1 \text{ or } J \text{ precedes } I \text{ in the network and } P_j \text{ is the acoustic match score of phone } j \text{ in segment } I \]

Figure 2 gives the phone probabilities for each phone in the network in each of the segments over which the match is performed. These scores are from Figure 3 marked with a mpgm. The best path through the mapping. The beginning 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 M indicate that a phone mapping to that segment is not allowed. The final mapping is given at the bottom of the figure. The final match score of 40.2 is the best phone which transitions into the final state J plus the acoustic match probability of J which is defined to be zero. This represents the score of the best path through the network. This score would be normalized by the number of segments mapped. If this wouldn't receive a HIGSAY score of 90 out of a possible 100. Other paths can be found by tracing back from the other possible ending phones: (48), (49), (48), (49), and (46).

IMPLEMENTATION ISSUES

Initially several problems arose while integrating this knowledge source into HIGSAY. The following is a discussion of the problems addressed during the implementation of the system. First, since we were dealing with single words and attempting to find them as they existed in isolation, many of the constraints provided by word juncture rules and syntactic knowledge were unavailable for use. In light of the issue that these constraints give to similar systems (Lowrette, 1976) would verification be acceptable?

Words could be hypothesized bottom-up with incorrect times. This meant that procedures had to be employed to search the segmentation for the local best starting and ending point around the given points. Words predicted top-down always had a missing time, and procedures for predicting these times accurately had to be developed. Problems in the generation of end and begin times of words which can be used for word pairs to be rejected by higher level knowledge sources.

The conversion of internal match scores to HIGSAY II ratings, while maintaining consistency of the ratings proved to be a major concern when it was noticed that the average internal ratings for words varied considerably depending on where in the utterance the words occurred. A solution to this problem directly rather than pass it on to the higher level knowledge sources.

SOLUTIONS

Several methods of verification are supported within WIZARD. Each represents a partial solution to one or more of the problems outlined. Non-pad mode uses no heuristics to determine the boundaries of the match. The predicted begin/end times are mapped directly into their respective segments and verification is performed on those segments. It takes approximately 30 milliseconds on CPU time on a PDP-11 to perform matching in this mode.

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 hypotheses. In this mode the begin/end times are mapped into segments as in non-pad mode. The given hypotheses are then used in the matching. Thus if B is the begin segment E the end segment, segments B to E are allowed starting point for the match and C(E) is 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 begin segment associated with the path. This is necessary as the begin time of the segment can be determined at the begin time of the word and to determine the path length number of phones in the path for scoring the node takes about 100 milliseconds of CPU time on the PDP-11 and is about 35 times faster than exploring each of the nine paths in non-pad mode.

As we have mentioned before it is necessary to perform verification on all of the words in the given times is known. Two basic modes are employed in WIZARD, one where the end time is unknown (in the other the other node a missing time issue) as in pad mode one segment window is centered around the given starting segment. Then each successive segment is matched and the match score computed as if the match were ending in that segment. The scores are ordered and the score for the best path is returned along with the matching time. Several heuristics are used to prune the number of end segments actually looked at as possible end states. This is the most computationally expensive of the verification modes taking about 160 milliseconds per verification on a PDP-11 processor.

Several experiments were performed to determine the best way to normalize the match scores. The technique employed was to try approximately 1000 bottom-up word hypotheses from 50 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 high scores. More complex normalizations based on non-linear time scaling were also attempted. 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 segments in the match path N while the other modes are normalized by N-1. These normalizations are computationally simple and embody the idea that data 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 500000 verifications. By knowing the distribution of correct and incorrect words over each of the internal score values (dynamic range of 64), a corresponding distribution of HEARSAY scores was calculated. The HEARSAY score distribution rows for the absolute rejection of verified words. This threshold was set so as to reject no correct words. Scores above this threshold were divided into scores that were not as important. One threshold is possible here. If one requires that no potential correct words be rejected then WIZARD was able to reject 121 to 137 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. A 62 rejection rate of correct words approximately 51% 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 has 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 102 utterances, in which 332 correct words were hypothesized. A total of 11053 incorrect words were generated. The vocabulary size for POA012 and WIZARD was approximately 550 words. WIZARD rated each of the words using ppa mode voting. For each rating threshold (15,103) 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 (12 utterances/120) is also noted.

DISCUSSION

The major direction of this work is the application of the HARNP model to word representations to the verification of single words in a connected speech understanding system. This includes the modifications to show the various verification modes dictated by the HEARSAY II system strategies. We feel that WI2277 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 of segmentations having many very short transition segments. The segment in general have poor ratings and often degrade the composite word score. Although we felt 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 stable increase in speed can be obtained by coding certain of the flight loops in assembly language. Other implementation oriented optimizations may be needed.

A most useful addition to WIZARD would be the ability to verify segmentic strategies. This would include words to dynamic generation of multiple word networks. These networks would embody the appropriate word juncture rules and would allow WIZARD to rate prusal 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 statically 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 this word verification system with a word segment hypothesis to perform more direct signal matching. The purpose of this addition is to distribute competing words which have good WIZARD scores in the same time area. We propose to extract word templates at the single word level and perform matching using Itakura's method (Itakura, 1975). The philosophy here is to store templates for a small number of potentially difficult 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 RO Eddy who made continuing suggestions for refinements to the basic algorithm and for testing works for performance. Bruce Lowenwe 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

**ABSTRACT**

(-0) AE3 ((-0),-) S (-0) (DR (R,0), T) R) AE2! ((-(-0),-) (T,0), DX)

**Figure 1**

**TABLE 1**

<table>
<thead>
<tr>
<th>S+E+</th>
<th>20</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
<th>25</th>
<th>26</th>
<th>27</th>
<th>28</th>
<th>29</th>
<th>30</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE3</td>
<td>28</td>
<td>18</td>
<td>22</td>
<td>20</td>
<td>24</td>
<td>20</td>
<td>26</td>
<td>22</td>
<td>20</td>
<td>18</td>
<td>28</td>
</tr>
<tr>
<td>RE2</td>
<td>22</td>
<td>24</td>
<td>20</td>
<td>18</td>
<td>22</td>
<td>20</td>
<td>26</td>
<td>22</td>
<td>20</td>
<td>18</td>
<td>28</td>
</tr>
</tbody>
</table>

**TABLE 2**

<table>
<thead>
<tr>
<th>S+E+</th>
<th>20</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
<th>25</th>
<th>26</th>
<th>27</th>
<th>28</th>
<th>29</th>
<th>30</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE3</td>
<td>28</td>
<td>18</td>
<td>22</td>
<td>20</td>
<td>24</td>
<td>20</td>
<td>26</td>
<td>22</td>
<td>20</td>
<td>18</td>
<td>28</td>
</tr>
<tr>
<td>RE2</td>
<td>22</td>
<td>24</td>
<td>20</td>
<td>18</td>
<td>22</td>
<td>20</td>
<td>26</td>
<td>22</td>
<td>20</td>
<td>18</td>
<td>28</td>
</tr>
</tbody>
</table>

**TABLE 3**

<table>
<thead>
<tr>
<th>TIME</th>
<th>92</th>
<th>93</th>
<th>100</th>
<th>112</th>
<th>116</th>
<th>123</th>
<th>128</th>
<th>130</th>
<th>145</th>
<th>148</th>
<th>180</th>
</tr>
</thead>
<tbody>
<tr>
<td>S+E+</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td>AE3</td>
<td>28</td>
<td>18</td>
<td>22</td>
<td>20</td>
<td>24</td>
<td>20</td>
<td>26</td>
<td>22</td>
<td>20</td>
<td>18</td>
<td>28</td>
</tr>
</tbody>
</table>

**FIGURE 2**

**FIGURE 3**

**FIGURE 4**
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 (Swets 64) has been shown to parameterize 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 (Go75).

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 units (e.g. G675, Bax75, Go75). 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, using some automatic techniques available*. Since the production of "correct" segmentations and their comparison with machine segmentations (e.g. amount of misalignment, etc. should one allow?) involve a number of linguistic as well as recognition systems-specific issues, we will not deal further with these problems here.

However, 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 errors, but we have not understood it well enough in a quantitative sense to compare different segmenters for even the same segmenter "tuned" to a different point on 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 missed, marked, or extra. Such a model is provided by Signal Detection Theory. We will show that the trade-off curves fall with the results of a set of segmentation trials run to explore the M/E trade-off.

*The happy speech recognition system (Law69) can be forced to the correct words. This provides a "best-fit" of the system's acoustic and phonological knowledge to the signal. When a very fine grained it is made coarse 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, (T64, Sw64) C is primarily applied to detection trials with a single, or a 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 a prior knowledge 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 convolved, in this model, to be reduced to a single decision parameter. An optimum 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 the models, costs of misses, false alarm, etc.

Figure 1 represents such a hypothetical internal decision parameter, d'.

![Figure 1: Signal Detection Model](image)

Figure 1: Signal Detection Model

Very simply stated, the model assumes a single decision parameter, $d'$, which may be any sensory measurement one wishes. The distribution of $d'$ values for the two types of signal, signal-plus-noise and noise alone, are assumed to be normal with equal variance in the signal condition of the model. The decision criterion is then set by $d'$ times the "standard error" of the $d'$ test.
alarm" -- Pr(accept|signal) and Pr(accept|noise) -- are sufficient to determine the least δ 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 trials where the detector's acceptance threshold has been altered, a response operator characteristic (ROC) curve is obtained (see figure 2).

![ROC Curve](image)

Figure 2: Typical ROC Plot

The theory states that the curve is totally determined by δ. 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 (signal/noise)/(signal + noise) and y-intercept = 1 (Eq. 6.14).

This theory has been most often applied to detection trials to provide estimates of the detectability of the signal as δ appears in a human perceiver's internal sensory signal space. The estimate of δ 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 of δ 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 δ 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 [Go761] and used for a while as the initial signal-to-symptom stage of the Hearsay II speech understanding system [En74]. 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 50 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 detect 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 of this program were compared with a "corrected" hand segmentation. That is, the machine segmentation was compared to a phoneme-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 1093 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. The resultant error rates are plotted on a normal-normal grid in figure 3. A least-squares regression fits a line with slope = 0.60 (noise standard deviation / signal standard deviation), and y-intercept = 2.25 (δ' -- the separation of the means of the two distributions).

![Signal to Noise](image)

Figure 3: δ' Trade-off

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, gave a measure of confidence in the δ' estimates. When the 40 sentence were divided into 10 groups, and estimates of δ' made for each group, the 95% confidence intervals in δ' was found to be ± 0.14 (i.e., the estimate of δ' for 10 sentences is ± 0.14 away from the estimate computed from all 40 within the confidence interval). Since this interval is considerably smaller than the differences found between segmentation program or between most parametric
representations, we feel such comparisons are meaningful using d'. For example, our representation of the signal were tested (Gold75), yielding d' values from 1.29 to 2.38. Furthermore, published results of two other segmenters (Bak75, Dot75) 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, offer 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, Dan Seidov, and Bruno Bask

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 graphics display terminal. Verbal input consists of sequences of commands and map feature descriptors which are recognized by the Harpy speech recognition system. An important 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 xy-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 user-favorable 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 the rigid constraint to highly constrained sentences. 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 graphic 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 were the Harpy speech recognition system [Lowerrre, 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 and 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 [Santhan and Rabiner, 1976] are accurate systems, but at present lack the desired flexibility in structuring the grammar. Other successful systems, such as Norlayee-I [Ehman et al., 1975 and Lesser et al., 1975] HWM [Woods, 1976], and the IBM system [Jelinek et al., 1975 and Bani et al., 1976], have much more elaborate control structures and were designed for larger tasks. The overhead involved in these 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 precomputed 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
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 database, 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 and pronunciation 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 a few juncture rules giving alternate pronunciations of word beginnings and endings based on the context of adjacent words. All of 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 in instance of the symbol. Probabilities are learned from examples 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 or cursor. 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 object or 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 graphic 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-descrbe 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, 651 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, e.g., the description of map features, and by the desired user interactions, e.g., 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, not to constrain the grammar too much that interaction becomes unnatural for the user.

There are several ways of imposing grammatical structure on the phrases when we 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., no constraint. This gives the user complete freedom to describe the map feature in any natural way. Since there are other methods which allow the naturalness but also have some constraint, this mode is used for what accuracy is attainable in the worst case. If accuracy is adequate in this case, then it will be more 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 stop or root node presented to the user is shown in figure 7.

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 a 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 "rver" implies "natural". The default would be used to construct the unique description unless some other counteracting 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 varying 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. 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 91.2 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

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. 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.

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 (and pre-compiled) 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 mapping scheme used is a modified graph search in which heuristics are used to reduce the number of paths that are checked. The result is that only a few "best" paths are searched in parallel during the recognition processes thus greatly reducing the computational overhead.

Results. The current system achieves a sentence accuracy of 90.02% and a word accuracy of 94.12% on a 1011 word task and runs in 6.8 MIPS.

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 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 decision variables 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 "midwestern 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 as an attempt 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 phoneticsound in the English language. For example, there are usually several occurrences of one vowel (all the phones in one set or phone, 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 "KW" that occurs together as in the word "queen"). 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 spelling 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 the unique phoneme sounds that occur in the task language. Since the lexicon contains symbolic spellings which are speaker independent and there is a one to one mapping 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 "(c) (v) (n) (e) (r) (n) (c)" and its pronunciation is "(c) (v) (n) (e) (r) (n) (c)". Optional paths are enclosed within parentheses and are separated by commas (the "O" represents the null option). The spelling is interpreted as either a word bar ("-"), followed by an optional word ("-"), or a single word ("-"), followed by another word ("-"), followed by either a "-" or a "-", followed by an "-", followed by an "-" or a "-".

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 "best" 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 Takura'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 (which can be from some other sources). Since the template is an average, there is no need to find the single "best" 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 Takura'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 (which can be from some other sources). Since the template is an average, there is no need to find the single "best" 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 Takura'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 (which can be from some other sources). Since the template is an average, there is no need to find the single "best" 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.
forced recognition can be done either by using a unique network for each utterance (which represents only the one utterance) or by considering only paths in a large network that represent each single utterance. The parses 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 the speaker dependent templates with a total of 1000 three word utterances) 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 (75%) 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 due to the large amount of training data needed and is computationally expensive. A third scheme was devised which allows a new user the immediate use of the system but also allows for the speaker dependent vocal characteristics. This is the dynamic tuning of the speaker templates. A new speaker to the system starts with the set of speaker independent templates. The system will, upon all correct recognitions, automatically average the autocorrelation data with the corresponding templates and update the template parameters. The first occurrence of a phone spoken by the speaker will replace the speaker independent template. Further occurrences of the same phone will add to the average of the template. This will result in the phone template being altered quickly for the first occurrences of a phone and a gradual tuning of the template by additional occurrences of the phone. In this method, the system quickly adapts itself to the speaker’s acoustic characteristics. If the system makes an error in recognition, one can either speak the same utterance again with the hope that it will be recognized correctly the second time or the system can be rerun on the same utterance and forced to recognize the utterance. To force a recognition, the appropriate switch is set and the correct utterance is typed to the system. The system will then only consider paths in its network which represent the spoken utterance.

The error rate when first starting is, of course, 75% but quickly drops off towards the 25% error rate of the speaker dependent templates. The time needed for the updating of the templates is zero during the actual recognition but requires up to one times real time after recognition depending on the number of templates that are updated. Therefore, the overhead of doing the dynamic speaker adaptation is minimal.

DISCUSSION

Summary. In this paper we have considered several sources of variability in the connected speech signal, i.e. dialect, 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 adapt to the specific speaker.

It appears straight forward to adopt the above techniques to 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

Analysis parameters in autocorrelation based linear prediction systems consist of pitch period, a voiced/unvoiced decision, amplitude information, and parcor coefficients. These parameters are generally encoded into a minimal bit representation and transmitted at a constant frame rate. The system on which our comparisons are based uses a frame rate of 100 frames/sec, where each frame consists of 200 speech samples. A total of 64 bits are allocated to the 14 parcor parameters, which are quantized as described in [Markel and Gray, 1974]. Pitch period and the voiced/unvoiced decision are encoded together in 6 bits, and the amplitude is coded into 5 bits.

SEGMENT-CODER

Classically, information concerning the vocal tract shape is transmitted in the form of parcor parameters once per analysis frame. Speech, however, can be segmented into events, or the duration of which, vocal tract shape may be considered approximately constant. Cases where this is not true, such as glides and ophonms, may be approximated by a series of shorter segments. Therefore, it should be possible, without significant degradation in synthetic speech quality, to transmit parcor parameters once per segment, rather than once per frame. Since segment duration is relatively long compared with analysis frame length, a savings in the number of bits needed to encode the analysis parameters should result. A vocoder simulation based on this hypothesis was developed.

Segmentation is performed using algorithms developed for the Hearsay speech recognition system [Goldberg and Reddy, 1976]. Three stages are involved in the overall process: parametrization, segmentation, and classification. The first step in parametrization is to generate smoothed and differentiated waveforms from the sampled speech. Next, peak to peak amplitudes and zero crossing counts are extracted from each waveform once per centisecond of speech. Segmentation is based on these parameters.

Segment boundaries are determined by successive subdivision of the waveform. First, silences and unvoiced fricatives are detected by a thresholding technique. Next, the remaining segments are divided where significant dips in the smoothed peak to peak parameter occur. A region growing technique is then applied to further subdivide the segments. Finally, the resulting segments may optionally be classified in terms of manner of articulation. Decision rules based on the averaged parameter values for each segment are used for this purpose.

Operation of the vocoder is relatively straightforward. Speech is segmented as it enters the system. When a segment boundary occurs, parcor parameters for that segment are calculated. By definition, a frame within a segment should have similar spectral properties, however...

For very nearly indistinguishable "informal" with purposes, a digital spectrogram of synthesized with the 7 coder. For utterance interpolated parcor coefficients. Note that although the sound speech, the darker curve represents the vocal tract, 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+5n)100+(4+64)n=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 less than that obtained with the segmentation scheme.

**LABEL-CODER**

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 would 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 than 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 (Lawrence, 1978). 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 speech 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-CODE**

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 is then 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 the 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)+100×(64+7)+15×1263 bits/sec 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 continue synthesis of one segment before it receives parameters for...
the next. If the receiver 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. Indeed, this is 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 the same set of templates. Once 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 in 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 Hearsay-II 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,

**REFERENCES**

H. G. 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 HEATSAY-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

---

1 This work was supported in part by the Defense Advanced Research Projects Agency under contract no. F44620-73-C-0074 and monitored by the Air Force Office of Scientific Research. In addition, the author was partially supported by a National Science Foundation Graduate Fellowship.
A 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\(^1\) 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 \(R(I,S) = \{R(I,e) \mid e \in S\}\), 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 \(e \in S\), \(I(e)\). 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., \( R(I, e) = R(l_1(e), e) \), where \( l_1(e) \) is the label assigned by \( I \) to \( e \), and \( R(I, e) \) measures the goodness of fit between the label \( I \) and the element \( e \). In this case, \( R(I, S) = \{ R(l_1(e), e) \mid e \in S \} \). If \( I \) is a simple averaging function, then \( R(I, S) = \text{Average} \{ R(l_1(e), e) \mid e \in S \} \).

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 \( I_{\text{MAX}}: e \rightarrow I_{\text{MAX}}(e) \), where \( I_{\text{MAX}}(e) \) 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 \( I_{\text{MAX}}(e) \) 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 \( R(I'', e) = R(l_1(e), e) > R(l_1(e), e) = R(l, e) \). (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 \( l_1, \ldots, l_k \) may safely be cancelled if

1. Any of the \( l_i \) has been rejected, or
2. All of the \( l_i \) 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 $l_{\text{max}}(e)$ 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 $R_{\text{max}}(e) \geq R(e)$ for all $e$ in $S$. The tighter this upper bound, the more partial solutions can be rejected. The $R_{\text{max}}$ 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 $R_{\text{max}}$ 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 $[\text{first'syl}, \text{last'syl}]$. Its rating is its average word rating weighted by the number of syllables in each word. I.e., $R(I, [\text{first'syl}, \text{last'syl}]) = \text{Average} \left( R(W_l(syl) \mid A(syl)) \right)$, where $\text{first'syl} \leq syl \leq \text{last'syl}$, $A(syl)$ represents the acoustic data in the interval $syl$, $W_l(syl)$ is the word label assigned by $I$ to $syl$, and $R(W \mid A)$ measures how closely the word $W$ matches the acoustic data $A$. $R(W \mid A)$ 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.
Mode: N R D B F

Average number of search operations (Hearsay-II knowledge source and precondition executions) to find first solution.¹

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>153</td>
<td>157</td>
<td>145</td>
<td>152</td>
<td>152</td>
</tr>
</tbody>
</table>

Average number of (percent) extra search operations to find the best solution once the first solution has been found:

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>71</td>
<td>58</td>
<td>30</td>
<td>26</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>46%</td>
<td>37%</td>
<td>21%</td>
<td>17%</td>
<td>0%</td>
</tr>
</tbody>
</table>

Speedup in this phase of the search relative to mode N:

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1.2</td>
<td>2.4</td>
<td>2.7</td>
<td>Infinity</td>
</tr>
</tbody>
</table>

Average total number of search operations to find best solution:

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>223</td>
<td>215</td>
<td>175</td>
<td>178</td>
<td>152</td>
</tr>
</tbody>
</table>

Average number of (percent) extra search operations to satisfy halting condition² once the best solution has been found (excluding runs in which time or space is exhausted):³

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<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>
</tbody>
</table>

Average total number of search operations until halting condition is satisfied:

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>286</td>
<td>282</td>
<td>253</td>
<td>226</td>
<td>152</td>
</tr>
</tbody>
</table>

Number (percent) of utterances in which halting condition is satisfied before system exceeds predefined limits on time (800 search operations) or space (193K):

|   |   |   |   |   |
|---|---|---|---|
|   | 4 | 17 | 32 | 34 |
|   | 12% | 50% | 94% | 94% |
|   | 100% |   |   |   |

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 labeling 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 \( I_{\text{max}} \) (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 \( I_{\text{max}} \), 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


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, 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, R, 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 --


CM76W4 CMU 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, Le75Or, 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).


Ha76Ch Hayes-Roth, F. and Burge, J. Characterizing syllables as sequences of phonelabels derived automatically from live continuous speech: a study in symbolic pattern learning using a conjunctive feature learning and classification system. Proc. 3rd Inter. Joint Conf. on Pattern Recognition, Coronado, CA, 1976.

Ha76Di Hayes-Roth, F., Gill, G. and Mostow, D. J. Discourse analysis and task performance in the Hearsay-II speech understanding system. In CM76W4.


-- 1975 --


-- 1974 --


CM74W3CMU Computer Science Speech Group. Working papers in speech recognition III. Tech. Report, CMUCSD, 1974. Includes Bk75Dr, Bm74Ne, Gi74Pa, Kn74Re, Kr74Ad, Le75Or, R74In and Sh74Su.

Er74Co Erman, L. D. (Ed.) Contributed Papers of the IEEE Symp. on Speech Recognition. IEEE, NY, NY, 1974. Includes Bk75Dr, Bm74Ne, Gi74Pa, Kn74Re, Kr74Ad, Le75Or, R74In and Sh74Su.


Reddy, D. R. and Newell, A. Knowledge and its representation in a speech
understanding system. In Knowledge and Cognition. (Gregg, L. W., Ed.)

Symp. Speech Recognition, Pittsburgh, PA, Apr., 1974, 242. Also appeared in
Er74Co and CM74W3.

Shockey, L. and Reddy, D. R. Quantitative analyses of speech perception:
results from transcription of connected speech from unfamiliar

Shockey, L. and Erman, L. D. Sub-lexical levels in the Hearsay-II speech
understanding system. Proc. 1974 IEEE Symp. Speech Recognition, Pittsburgh,

Shockey, L. and Erman, L. D. Sub-lexical levels in the Hearsay-I speech
understanding system. Proc. 1974 IEEE Symp. Speech Recognition, Pittsburgh,


-- 1973 --


Brooks, R., Erman, L. D. and Neely, R. Jabberwocky: a semi-automated system
for the transcription of verbal protocols. Behavioral Research Methods and
Instrumentation (May 1973).

CMU. 3W2 CMU Computer Science Speech Group. Working papers in speech recognition
II. Tech. Report, CMUCS-73, 1973. Includes Bk73Ma, Bk73Ne, Er73Re, Re73Mo
and Re73Hy.

Erman, L. D., Lifter, B. T. and Reddy, D. R. Representation and use of
acoustic-phonetic knowledge in the Hearsay-I system. 8th Meeting Acous.
Soc. Amer., Los Angeles, CA, Nov., 1973, 49. (Abstract only)

and labeling of connected speech. 8th Meeting Acous. Soc. Amer., Los
Angeles, CA, Nov., 1973, 51. (Abstract only)

Erman, L. D., Fennell, R. D., Lesser, V. R. and Reddy, D. R. System organizations
for speech understanding: Implications of network and multiprocessor
Also appeared in CM73W2 and IEEE Trans. Computers, C-25, No. 4, April,

Neely, R. B. On the use of syntax and semantics in a speech understanding
University). Description of top-end processing in Hearsay-I.

Newell, J., Barnett, J., Forgie, J., Green, C., Klatt, D., Licklider, J. C. R., Munson,
work set the goals and orientation for the DARPA SIR effort.

Reddy, D. R. Eves and ears for computers. In I. G. Fachthakung Cognitive
Verfahren und Systeme. (Beckmann, M., Goos, C. and Knecht, H. G., Ed.)

Re73Hi 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)


Re69Se Reddy, D. R. Segment-synchronization problem in speech recognition. J. Acous. Soc. Amer. 46 (July 1969) 89. (Abstract only)


1968 --


-- 1966 --


-- 1964 --


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.