The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty

LEE D. ERMAN

USC/Information Sciences Institute, Marina del Rey, California 90291

FREDERICK HAYES-ROTH

The Rand Corporation, Santa Monica, California 90406

VICTOR R. LESSER

University of Massachusetts, Amherst, Massachusetts 01003

D. RAJ REDDY

Carnegie-Mellon University, Pittsburgh, Pennsylvania 15213

The Hearsay-II system, developed during the DARPA-sponsored five-year speech-understanding research program, represents both a specific solution to the speech-understanding problem and a general framework for coordinating independent processes to achieve cooperative problem-solving behavior. As a computational problem, speech understanding reflects a large number of intrinsically interesting issues. Spoken sounds are achieved by a long chain of successive transformations, from intentions, through semantic and syntactic structuring, to the eventually resulting audible acoustic waves. As a consequence, interpreting speech means effectively inverting these transformations to recover the speaker's intention from the sound. At each step in the interpretive process, ambiguity and uncertainty arise.

The Hearsay-II problem-solving framework reconstructs an intention from hypothetical interpretations formulated at various levels of abstraction. In addition, it allocates limited processing resources first to the most promising incremental actions. The final configuration of the Hearsay-II system comprises problem-solving components to generate and evaluate speech hypotheses, and a focus-of-control mechanism to identify potential actions of greatest value. Many of these specific procedures reveal novel approaches to speech problems. Most important, the system successfully integrates and coordinates all of these independent activities to resolve uncertainty and control combinatorics. Several adaptations of the Hearsay-II framework have already been undertaken in other problem domains, and it is anticipated that this trend will continue; many future systems necessarily will integrate diverse sources of knowledge to solve complex problems cooperatively.

Discussed in this paper are the characteristics of the speech problem in particular, the special kinds of problem-solving uncertainty in that domain, the structure of the Hearsay-II system developed to cope with that uncertainty, and the relationship between Hearsay-II's structure and those of other speech-understanding systems. The paper is intended for the general computer science audience and presupposes no speech or artificial intelligence background.

Keywords and Phrases: artificial intelligence, blackboard, focus of control, knowledge-based system, multiple diverse knowledge sources, multiple levels of abstraction, problem-solving system, speech-understanding systems, uncertainty resolving

CONTENTS

INTRODUCTION

The Hearsay-II speech-understanding system (SUS) developed at Carnegie-Mellon University recognizes connected speech in a 1000-word vocabulary with correct interpretations for 90 percent of test sentences. Its basic methodology involves the application of symbolic reasoning as an aid to signal processing. A marriage of general artificial intelligence techniques with specific acoustic and linguistic knowledge was needed to accomplish satisfactory speech-understanding performance. Because the various techniques and heuristics employed were embedded within a general problem-solving framework, the Hearsay-II system embodies several design characteristics that are adaptable to other domains as well. Its structure has been applied to such tasks as multisensor interpretation [NII78], protein-crystallographic analysis [ENGE77], image understanding [HANS76], a model of human reading [RUME76], and dialogue comprehension [MANN79]. This paper discusses the characteristics of the speech problem in particular, the special kinds of problem-solving uncertainty in that domain, the structure of the Hearsay-II system developed to cope with that uncertainty, and the relationship between Hearsay-II's structure and the structures of other SUSs.

Uncertainty arises in a problem-solving system if the system's knowledge is inadequate to produce a solution directly. The fundamental method for handling uncertainty is to create a space of candidate solutions and search that space for a solution. "Almost all the basic methods used by intelligent systems can be seen as some variation of search, responsive to the particular knowledge available" [NEWE77, p. 13]. In a difficult problem, i.e., one with a large search space, a problem solver can be effective only if it can search efficiently. To do so, it must apply knowledge to guide the search so that relatively few points in the space need be examined before a solution is found. A key way of accomplishing this is by augmenting the space of candidate solutions with candidate *partial* solutions and then constructing a complete solution by extending and combining partial candidates. A candidate partial solution represents all complete candidates that contain it. By considering partial solution candidates, we can often eliminate whole subspaces from further consideration and simultaneously focus the search on more promising subspaces.

To solve a problem as difficult as speech understanding, a problem solver requires several kinds of capabilities in order to search effectively: It must collect and analyze data, set goals to guide the inferential search processes, produce and retain appro-

This research was supported chiefly by Defense Advanced Research Projects Agency contract F44620-73-C-0074 to Carnegie-Mellon University. In addition, support for the preparation of this paper was provided by USC/ISI, Rand, and the University of Massachusetts. We gratefully acknowledge their support. Views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official opinion or policy of DARPA, the U.S. government, or any other person or agency connected with them.

priate inferences, and decide when to stop working for a possibly better solution. Years ago, when AI problem solvers first emerged, they attempted to provide these capabilities through quite general domain-independent methods, the so-called *weak* methods [NEWE69]. A prime example of such a problem solver is GPS [ERNS69]. More recently, several major problem-solving accomplishments, such as Dendral [FEIG71] and Mycin [SHOR76], have reflected a different philosophy: Powerful problem solvers depend on extensive amounts of knowledge about both the problem domain and the problem-solving strategies effective in that domain [FEIG77]. Much of what we view as expertise consists of these two types of knowledge; without capturing and implementing this knowledge, we could not create effective computer problem solvers. Because knowledge plays a crucial role in these kinds of tasks, many people call the corresponding problem solvers *knowledge-based systems* [BARN77]. The design of Hearsay-II is responsive to both concerns. While formulated as a general system-building framework that would structure and control problem-solving behavior involving multiple, diverse, and error-full sources of knowledge, the current Hearsay-II system consists of a particular collection of programs embedding speech knowledge that are capable of solving the understanding problem.[1]

The difficulty of the speech-understanding problem, and hence the need for powerful problem-solving methods, derives from two inherent sources of uncertainty or error. The first includes ordinary variability and noise in the speech waveform, and the second includes the ambiguous and inaccurate judgments arising from an application of incomplete and imprecise theories of speech. Because we cannot resolve these uncertainties directly, we structure the speech-understanding problem as a space in which our problem solver searches for a solution. The space is the set of (partial and complete) *interpretations* of the input acoustic signal, i.e., the (partial and complete) mappings from the signal to the possible messages. The goal of our problem-solving system is to find a complete interpretation (i.e., a message and mapping) which maximizes some evaluation function based on knowledge about such things as acoustic-phonetics, vocabulary, grammar, semantics, and discourse. This resolution of the combined sources of uncertainty requires the generation, evaluation, and integration of numerous partial interpretations. The need to consider many alternative interpretations without spawning an explosive combinatorial search thus becomes a principal design objective. Each of these issues is discussed in more detail in the following section.

Dimensions of the Problem: Uncertainty and Hypothetical Interpretations

The first source of difficulty in the speech problem arises from the speaking process itself. In the translation from intention to sound, a speaker transforms concepts into speech through processes that introduce variability and noise (see Figure 1). If, for example, we consider the semantic, syntactic, lexical, and phonemic stages, the types of variance introduced from one level to the next would correspond to errors or peculiarities of conceptualization, grammar, word choice, and articulation. In addition

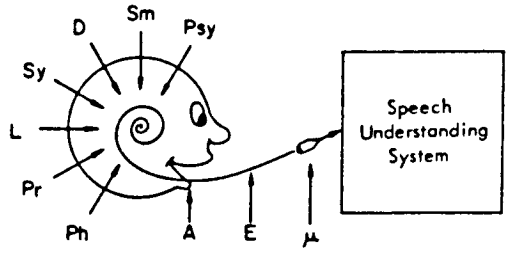


FIGURE 1. Some of the mechanisms that affect the message: psychology of the speaker, semantics, rules of discourse, syntax, lexicon, prosodic system, phonemic system, speaker's articulatory apparatus, ambient environmental noise, and microphone and system. [After NEWE75.]

[1] The problem of speech understanding has been actively pursued recently [REDD75, REDD76, CMU77, BERN76, WALK78, WOOD76, KLAT77, MEDR78, LEA80]. With the exception of HARPY [LOWE80], however, none of the other efforts has been presented as a structure for problem solving in other domains.

to these sources of variability, speech is often affected by pauses, extraneous sounds, or unnecessary phrase repetitions. The effect of these factors upon the physical sound signal is to distort it significantly from the *ideal message*, that is, from the message that would be produced if the production mechanisms did not introduce variability and noise. Accordingly, we speak of the disparity between the ideal and actual signals as *error*, and of the variety of factors that contribute to such distortion as *sources of error*. Thus the first source of error is inherent in the speaker and his environment.

The second source of error in the understanding process is intrinsic to the listener. Just as the speaker must transform his intention through successive intermediate levels of representation, so we presume the listener must accomplish the inverse of those transformations; from the physical signal the listener must detect acoustic-phonetic elements, syllables, words, and syntactic and conceptual structures corresponding to the speaker's intentions. At each step in this reconstruction the listener may introduce new errors corresponding to incorrect perceptual or interpretive judgments.[2] Because a machine speech-understanding system must also develop interpretations of what was spoken and what was intended, it is likely to commit similar mistakes in judgment. These judgmental errors can be viewed as the result of applying inadequate or inaccurate theoretical models to the speech-analysis task. If the first source of error is deviation between ideal and spoken messages due to inexact production, the second source of error is deviation between spoken and interpreted messages due to imprecise rules of comprehension.

To comprehend an utterance in the context of such errors, a speech-understanding system must formulate and evaluate numerous candidate interpretations of speech fragments. Understanding a message requires us to isolate and recognize its individual words and parse their syntactic and conceptual relationships. Each intermediate state of this process can be viewed as either the generation or evaluation of symbolic interpretations for portions of the spoken utterance. We use the term *hypothesis* to refer to a partial interpretation actually constructed. During the process of speech interpretation, hypotheses may vary from highly confident identification of particular words to great confusion concerning particular portions of the utterance. Between these two extremes, the listener may entertain simultaneously several competing hypotheses for what was said. Competing alternatives might occur at any of several levels of abstraction. For example, at the word level the listener may struggle to distinguish whether "till" or "tell" was spoken in one portion of the utterance while simultaneously attempting to differentiate the words "brings" and "rings" in another interval. These uncertainties derive from comparable uncertainties at lower levels of interpretation, such as syllabic and acoustic, where multiple competing hypotheses can also exist simultaneously. Similarly, uncertainty among word hypotheses at the lexical level engenders uncertainty at higher levels of interpretation. Thus the previously discussed inability to distinguish between alternative words may be the underlying cause of an inability to distinguish between the four hypothetical phrase interpretations:

till Bob rings
tell Bob rings
till Bob brings
tell Bob brings

Just as this example suggests, higher level interpretations incorporate lower level ones. A phrase-level hypothesis consists of a selection of word hypotheses from each interval of time spanned by the higher level hypothesis. Only one lower level hypothesis in any time interval can be incorporated into the higher level interpretation. Thus a phrase consists of a sequence of words, a word consists of a sequence of syllables, a syllable consists of a sequence of acoustic-phonetic segments, and so on. An overall interpretation of an entire utterance would consist of a syntactic or semantic analysis that recursively incorporated one hypoth-

[2] Though the levels of representation appear to be linearly ordered, the encoding and decoding processes do not necessarily operate sequentially through this ordering.

esis from each level of interpretation for each temporal interval of the utterance.

A fundamental assumption underlying the understanding problem is that a correct interpretation of an utterance should minimize the difference between those properties of the speech that the hypothetical interpretation would predict and those that are observed. This gives rise to the notion of the *consistency* between an interpretation and its supporting data. Thus certain parameter values derived from an acoustic waveform are more or less consistent with various phonetic classifications, particular sequences of phones are more or less consistent with various monosyllabic categorizations, and various syllable sequences are more or less consistent with particular lexical and phrase interpretations. The concept of consistency between two adjacent levels of interpretation can be generalized to permit consideration of the consistency between hypotheses at any two levels and, in particular, the consistency between an overall interpretation of the utterance and its supporting hypotheses at the lowest, acoustic-parametric level. A central assumption is that the greater the consistency between the overall interpretation and the acoustic data, the more likely the interpretation is to be correct.

We refer to the likelihood that some hypothesis is correct as its *credibility*. As the preceding suggests, the credibility of each hypothesis is a measure of consistency between the data generating the hypothesis and the expectations it engenders. A credibility calculation involves a judgment about the knowledge used in creating the hypothesis and therefore is itself subject to uncertainty.

To assess the credibility of a hypothesis, we need basically to evaluate two things: all plausible alternatives to this hypothesis and the degree of support each receives from data. Consider, for example, the evaluation of word hypotheses. Initially, nearly all words in the language are plausible candidates for occurring within any time interval. As a consequence, our uncertainty at the outset, as approximated by the number of equally plausible alternatives, is maximal. Over time we accrue evidence to eliminate some of these alternatives. Moreover, by eliminating one particular hypothesis, we may logically exclude others that are in temporally adjacent regions and that depend directly on that hypothesis. For example, if we have ruled out all possible adjectives and nouns in a particular location, we can also rule out adjectives in the preceding interval. Conversely, if we can identify a particular word as an adjective, we can increase our belief that the following word will be an adjective or noun. In general, each individual hypothesis is strengthened by its apparent combinability with others. Thus we say uncertainty is reduced by detecting mutually supporting hypotheses that are consistent with the acoustic data. Equivalently, the credibility of hypotheses increases as a function of their involvement in such mutually supportive clusters.

This technique for reducing uncertainty leads to the following incremental problem-solving method: The goal of the problem solver is to construct the most credible overall interpretation. The fundamental operations in the construction are hypothesis generation, hypothesis combination, and hypothesis evaluation. At each step in the construction, sources of knowledge use these operations to build larger partial interpretations, adding their constraints to the interpretation. The accrual of constraints reduces the uncertainty inherent in the data and in the knowledge sources themselves.

Three requirements must be met for such a problem solver to be effective:

(1) At least one possible sequence of knowledge-based operations must lead to a correct overall interpretation.
(2) The evaluation procedure should assess the correct overall interpretation as maximally credible among all overall interpretations generated.
(3) The cost of problem solving must satisfy some externally specified limit. Usually this limit restricts the time or space available for computing. As a consequence, it leads to restrictions on the number of alternative partial interpretations that can be considered. Alternative partial solutions must be considered in order to ensure that a correct

one is included. The greater the uncertainty in the knowledge used to generate and evaluate hypotheses, the greater the number of alternatives that must be considered, leading to a possible combinatorial explosion.

As we have seen, the speech-understanding problem is characterized by the need for highly diverse kinds of knowledge for its solution and by large amounts of uncertainty and variability in input data and knowledge. The diversity of knowledge leads to a search space of multilevel partial solutions. The uncertainty and variability mean that the operators used for searching the space are themselves error-prone; therefore many competing alternative hypotheses must be generated. To avoid a combinatorial explosion, a powerful control scheme is needed to exploit selectively the most promising combinations of alternatives. As systems tackle more such difficult real-world problems, such multilevel representations and powerful control schemes will become increasingly important [HAYE78a]. The next section discusses how the Hearsay-II system copes with these representation and control problems.

Hearsay-II Problem-Solving Model

The key functions of generating, combining, and evaluating hypothetical interpretations are performed by diverse and independent programs called *knowledge sources* (KSs). The necessity for diverse KSs derives from the diversity of transformations used by the speaker in creating the acoustic signal and the corresponding inverse transformations needed by the listener for interpreting it. Each KS can be schematized as a condition–action pair. The condition component prescribes the situations in which the KS may contribute to the problem-solving activity, and the action component specifies what that contribution is and how to integrate it into the current situation.[3] According to the original conception of the diverse stages and processes involved in speech understanding, KSs have been developed to perform a variety of functions. These include extracting acoustic parameters, classifying acoustic segments into phonetic classes, recognizing words, parsing phrases, and generating and evaluating predictions for undetected words or syllables. Figure 2 presents a schematic view of the KSs in the September 1976 configuration of the Hearsay-II speech-understanding system. Figure 3 gives a brief functional description of these KSs.

Because each KS is an *independent* condition–action module, KSs communicate through a global database called the *blackboard*. The blackboard records the hypotheses generated by KSs. Any KS can generate a hypothesis (record it on the blackboard) or modify an existing one. These actions in turn may produce structures that satisfy the applicability conditions of other KSs. In this framework the blackboard serves in two roles: It represents intermediate states of problem-solving activity, and it communicates messages (hypotheses) from one KS that activate other KSs.

The blackboard is subdivided into a set of information levels corresponding to the intermediate representation levels of the decoding processes (phrase, word, syllable, etc.). Each hypothesis resides on the blackboard at one of the levels and bears a defining label chosen from a set appropriate to that level (e.g., the word FLYING, the syllable ING, or the phone NG). The hypothesis contains additional information, including its time coordinates within the spoken utterance and a credibility rating. The sequence of levels on the blackboard forms a loose hierarchical structure: hypotheses at each level aggregate or abstract elements at the adjacent lower level. The possible hypotheses at a level form a search space for KSs operating at that level. A partial

[3] The condition and action components of a KS are realized as arbitrary programs. To minimize reevaluating the condition programs continuously, each condition program declares to the system the primitive kinds of situations in which it is interested. The condition program is triggered only when there occur changes that create such situations (and is then given pointers to all of them). This changes a polling action into an interrupt-driven one and is more efficient, especially for a large number of KSs. When executed, the condition program can search among the set of existing hypothetical interpretations for arbitrarily complex configurations of interest to its KS.

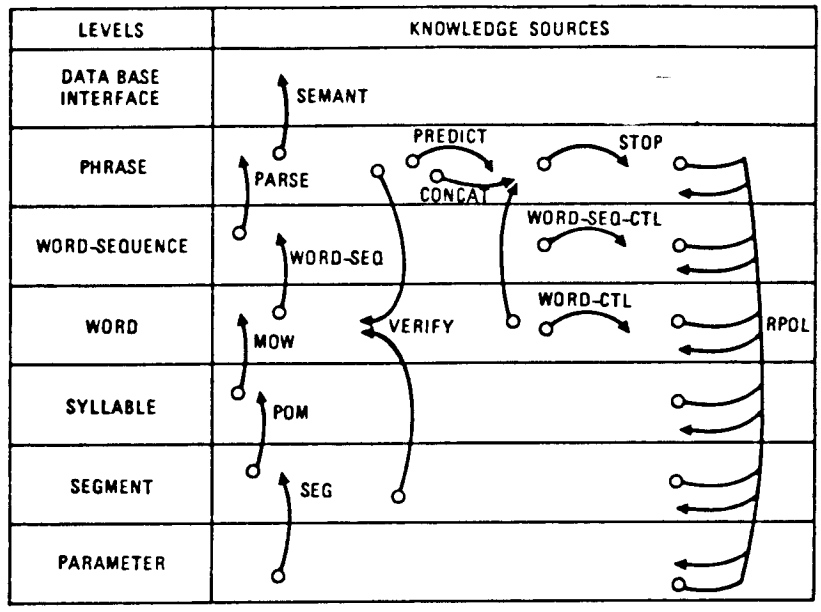


FIGURE 2. The levels and knowledge sources of September 1976. KSs are indicated by vertical arcs with the circled ends indicating the input level and the pointed ends indicating output level.

FIGURE 3. Functional description of the speech-understanding KSs.

Signal Acquisition, Parameter Extraction, Segmentation, and Labeling:
- SEG: Digitizes the signal, measures parameters, and produces a labeled segmentation.

Word Spotting:
- POM: Creates syllable-class hypotheses from segments.
- MOW: Creates word hypotheses from syllable classes.
- WORD-CTL: Controls the number of word hypotheses that MOW creates.

Phrase-Island Generation:
- WORD-SEQ: Creates word-sequence hypotheses that represent potential phrases from word hypotheses and weak grammatical knowledge.
- WORD-SEQ-CTL: Controls the number of hypotheses that WORD-SEQ creates.
- PARSE: Attempts to parse a word sequence and, if successful, creates a phrase hypothesis from it.

Phrase Extending:
- PREDICT: Predicts all possible words that might syntactically precede or follow a given phrase.
- VERIFY: Rates the consistency between segment hypotheses and a contiguous word-phrase pair.
- CONCAT: Creates a phrase hypothesis from a verified contiguous word-phrase pair.

Rating, Halting, and Interpretation:
- RPOL: Rates the credibility of each new or modified hypothesis, using information placed on the hypothesis by other KSs.
- STOP: Decides to halt processing (detects a complete sentence with a sufficiently high rating, or notes the system has exhausted its available resources) and selects the best phrase hypothesis or set of complementary phrase hypotheses as the output.
- SEMANT: Generates an unambiguous interpretation for the information-retrieval system which the user has queried.

interpretation at one level can constrain the search at another level.

Within this framework we consider two general types of problem-solving behaviors. The first type, associated with means-ends analysis and problem-reduction strategies [ERNS69, NILS71, SACE74], attempts to reach a goal by dividing it into a set of simpler subgoals and reducing these recursively until only primitive or immediately solvable subgoals remain. Such a strategy is called *top-down* or *analysis-by-synthesis*. In speech understanding, where the goal is to find the most credible high-level interpretation of the utterance, a top-down approach would reduce recursively the general sentential concept goal into alternative sentence forms, each sentence form into specific alternative word sequences, specific words into alternative phone sequences, and so forth, until the one alternative overall interpretation most consistent with the observed acoustic parameters is identified. The second, or *bottom-up*, method attempts to synthesize interpretations directly from characteristics of the data provided. One type of bottom-up method would employ procedures to classify acoustic segments within phonetic categories by comparing their observed parameters with the ideal parameter values of each phonetic category. Other bottom-up procedures might generate syllable or word hypotheses directly from sequences of phone hypotheses, or might combine temporally adjacent word hypotheses into syntactic or conceptual units. For a hypothesis generated in either the top-down or bottom-up mode, we would like to represent explicitly its relationship to the preexisting hypotheses that suggested it. *Links* are constructed between hypotheses for this purpose.

Both types of problem-solving behaviors can be accommodated simultaneously by the condition-action schema of a Hearsay-II KS. Top-down behaviors represent the reduction of the higher level goal as the condition to be satisfied and the generation of appropriate subgoals as the associated action. Bottom-up behaviors employ the condition component to represent the lower level hypothesis configurations justifying higher level interpretations, and employ the action component to represent and generate such hypotheses. In both cases the condition component performs a test to determine if there exists an appropriate configuration of hypotheses that would justify the generation of additional hypotheses prescribed by the corresponding action component. Whenever such conditions are satisfied, the action component of the KS is *invoked* to perform the appropriate hypothesis generation or modification operations. For example, the action of the POM KS (see Figures 2 and 3) is to create hypotheses at the syllable level. The condition for invoking the MOW KS is the creation of a syllable hypothesis. Thus the action of POM triggers MOW. The invocation condition of RPOL, the rating KS, is the creation or modification of a hypothesis at any level; thus POM's actions also trigger RPOL. In short, control of KS activation is determined by the blackboard actions of other KSs, rather than explicit calls from other KSs or some central sequencing mechanism. This *data-directed* control regime permits a more flexible scheduling of KS actions in response to changing conditions on the blackboard. We refer to such an ability of a system to exploit its best data and most promising methods as *opportunistic* problem solving [NII78, HAYE79a].

While it is true that each condition-action knowledge source is logically independent of the others, effective problem-solving activity depends ultimately on the capability of the individual KS actions to construct cooperatively an overall interpretation of the utterance. This high-level hypothesis and its recursive supports represent the *solution* to the understanding *problem*. Since each KS action simply generates or modifies hypotheses and links based on related information, a large number of individual KS invocations may be needed to construct an overall interpretation.

Any hypothesis that is included in the solution is *cooperative* with the others. Conversely, any hypothesis that is unincorporated into the solution is *competitive*. In a similar way, KS invocations can be considered cooperative or competitive depending on whether their potential actions

would or would not contribute to the same solution. Because of the inherent uncertainty in the speech-understanding task, there are inevitably large numbers of plausible alternative actions in each time interval of the utterance. Before the correct interpretation has been found, we cannot evaluate with certainty the prospective value of any potential action. Actions appear cooperative to the extent to which they contribute to the formation and support of increasingly comprehensive interpretations. Conversely, any hypothesis occupying the same time interval as another hypothesis but not part of its support set must be considered competitive. That is, two hypotheses compete if they represent incompatible interpretations of the same portion of the utterance. As a result, KS invocations can be viewed as competitive if their likely actions would generate inconsistent hypotheses, and they can be viewed as cooperative if their actions would combine to form more comprehensive or more strongly supported hypotheses.

The major impediment to discovery of the best overall interpretation in this scheme is the combinatorial explosion of KS invocations that can occur. From the outset, numerous alternative actions are warranted. A purely top-down approach would generate a vast number of possible actions, if unrestrained. Because certainty of recognition is practically never possible and substantial numbers of competing hypotheses must be entertained at each time interval of analysis, any bottom-up approach generates a similarly huge number of competing possible actions. Thus additional constraints on the problem-solving activity must be enforced. This is accomplished by selecting for execution only a limited subset of the invoked KSs.

The objective of *selective attention* is to allocate limited computing resources (processing cycles) to the most important and most promising actions. This selectivity involves three components. First, the probable effects of a potential KS action must be estimated before it is performed. Second, the global significance of an isolated action must be deduced from analysis of its cooperative and competitive relationships with existing hypotheses; *globally significant actions* are those that contribute to the detection, formation, or extension of combinations of redundant hypotheses. Third, the desirability of an action must be assessed in comparison with other potential actions. While the inherent uncertainty of the speech task precludes error-free performance of these component tasks, there have been devised some approximate methods that effectively control the combinatorics and make the speech-understanding problem tractable.

Selective attention is accomplished in the Hearsay-II system by a heuristic scheduler which calculates a priority for each action and executes, at each time, the waiting action with the highest priority [HAYE77a]. The priority calculation attempts to estimate the usefulness of the action in fulfilling the overall system goal of recognizing the utterance. The calculation is based on information provided when the condition part of a KS is satisfied. This information includes the *stimulus frame*, which is the set of hypotheses that satisfied the condition, and the *response frame*, a stylized description of the blackboard modifications that the KS action is likely to perform. For example, consider a syllable-based word hypothesizer KS (such as MOW); its stimulus frame would include the specific syllable hypothesis which matched its condition, and its response frame would specify the expected action of generating word hypotheses in a time interval spanning that of the stimulus frame. In addition to this action-specific information, the scheduler uses global state information in its calculations and considers especially the credibility and duration of the best hypotheses in each level and time region and the amount of processing required from the time the current best hypotheses were generated. The latter information allows the system to reappraise its confidence in its current best hypotheses if they are not quickly incorporated into more comprehensive hypotheses.

Hearsay-II Architecture

Figure 4 illustrates the primary architectural features of the Hearsay-II system. At the start of each cycle, the scheduler, in

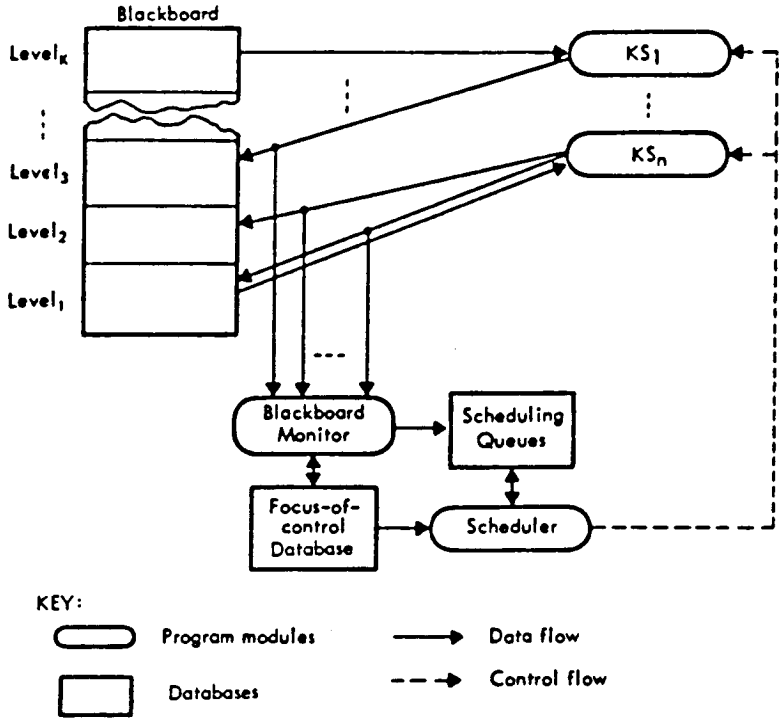


FIGURE 4. Schematic of the Hearsay-II architecture.

accordance with the global state information, calculates a priority for each activity (KS condition program or action program) in the scheduling queues. The highest priority activity is removed from the queues and executed. If the activity is a KS condition program, it may insert new instances of KS action programs into the scheduling queues. If the activity is a KS action program, the blackboard monitor notices the blackboard changes it makes. Whenever a change occurs that would be of interest to a KS condition program, the monitor creates an activity in the scheduling queues for that program. The monitor also updates the global state information to reflect the blackboard modifications.

1. AN EXAMPLE OF RECOGNITION

In this section we present a detailed description of the Hearsay-II speech system understanding one utterance. 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:

"Which abstracts refer to theory of computation?"

"List those articles."

"What has McCarthy written since nineteen seventy-four?"

The vocabulary contains 1011 words (in which each extended form of a root, e.g., the plural of a noun, is counted separately if it appears). The grammar defining the legal sentences is context-free and includes recursion. The style of the grammar is such that there are many more nonterminals than in conventional syntactic grammars; the information contained in the greater number of nodes imbeds semantic and pragmatic constraint directly within the grammatical structure. For example, in place of 'Noun' in a conventional grammar, this grammar includes such nonterminals as 'Topic', 'Author', 'Year', and 'Publisher'. Because of its emphasis on semantic categories, this type of grammar is called a *semantic template grammar* or simply a *semantic grammar* [HAYE75, BURT76,

HAYE80]. The grammar allows each word to be followed, on the average, by 17 other words of the vocabulary.[4] The standard deviation of this measure is very high (about 51), since some words (e.g., "about" or "on") can be followed by many others (up to 300 in several cases).

1.1 Introduction to the Example

We will describe how Hearsay-II understood the utterance "ARE ANY BY FEIGENBAUM AND FELDMAN?"[5] Each major *step* of the processing is shown; a step usually corresponds to the action of a knowledge source. Executions of the condition programs of the KSs are not shown explicitly, nor do we list those potential knowledge-source actions which are never chosen by the scheduler for execution. Executions of RPOL are also omitted; in order to calculate credibility ratings for hypotheses, RPOL runs in high priority immediately after any KS action that creates or modifies a hypothesis.

The waveform of the spoken utterance is shown in Figure 5a. The "correct" word boundaries (determined by human inspection) are shown in Figure 5b for reference. The remaining sections of Figure 5 contain all the hypotheses created by the KSs. Each hypothesis is represented by a box; the box's horizontal position indicates the location of the hypothesis within the utterance. The hypotheses are grouped by level: segment, syllable, word, word sequence, and phrase. Links between hypotheses are not shown. The processing will be described in terms of a sequence of *time steps*, where each step corresponds approximately to KS execution governed by one scheduling decision. Within each hypothesis, the number preceding the colon indicates the time step during which the hypothesis was created. The symbol following the colon names the hypothesis. At the word level and above, an asterisk (*) following the symbol indicates that the hypothesis is correct. The trailing number within each hypothesis is the credibility rating on an arbitrary scale ranging from 0 to 100.

In the step-by-step description, the name of the KS executed at each step follows the step number. An asterisk following the KS name indicates that the hypotheses in the stimulus frame of this KS instantiation are all correct. Single numbers in parentheses after hypotheses are their credibility ratings. All times given are in centisecond units; thus the duration of the whole utterance, which was 2.25 seconds, is marked as 225. When begin- and end-times of hypotheses are given, they appear as two numbers separated by a colon (e.g., 52:82). As in the figure, correct hypotheses are marked with an asterisk.

1.2 The Example

The utterance is recorded by a medium-quality Electro-Voice RE-51 close-speaking headset microphone in a moderately noisy environment (>65 dB). The audio signal is low-pass filtered and 9-bit sampled at 10 kHz. All subsequent processing, including the control of the A/D converter, is performed digitally on a time-shared PDP-10 computer. Four acoustic parameters (called ZAPDASH) are derived by simple algorithms operating directly on the sampled signal [GOLD77]. These parameters are extracted in real time and are used initially to detect the beginning and end of the utterance.

Step 1. KS: SEG.
Stimulus: Creation of ZAPDASH parameters for the utterance.
Action: Create segment hypotheses.

The ZAPDASH parameters are used by the *SEG* knowledge source as the basis for an acoustic segmentation and classification of the utterance [GILL78]. This segmentation is accomplished by an iterative refinement technique: First, silence is separated from nonsilence; then the nonsilence is broken down into the sonorant and nonsonorant regions, and so on. Eventually five

[4] Actually, a family of grammars, varying in the number of words (terminals) and in the number and complexity of sentences allowed, was generated. The grammar described here and used in most of the testing is called X05.

[5] To improve clarity, the description differs from the actual computer execution of Hearsay-II in a few minor details.

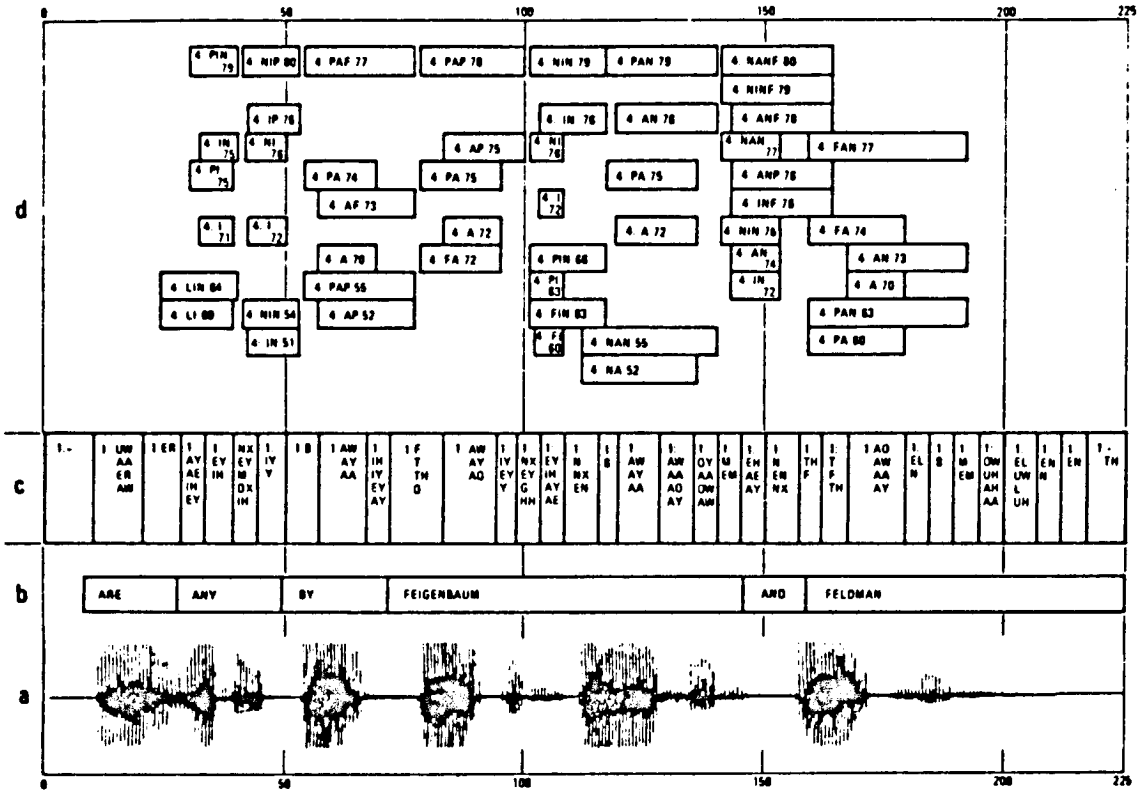


FIGURE 5. The example utterance: (a) the waveform of "Are any by Feigenbaum and Feldman?"; (b) the correct words (for reference); (c) segments; (d) syllable classes; (e) words (created by MOW); (f) words (created by VERIFY); (g) word sequences; (h) phrases. (See facing page for Figure 5e–h.)

classes of segments are produced: silence, sonorant peak, sonorant nonpeak, fricative, and flap. Associated with each classified segment is its duration, absolute amplitude, and amplitude relative to its neighboring segments (i.e., local peak, local valley, or plateau). The segments are contiguous and nonoverlapping, with one class designation for each.

SEG also does a finer labeling of each segment, using a repertory of 98 phonelike labels. Each of the labels is characterized by a vector of autocorrelation coefficients [ITAK75]. These template vectors were generalized from manually labeled speaker-specific training data. The labeling process matches the central portion of each segment against each of the templates using the Itakura metric and produces a vector of 98 numbers. The ith number is an estimate of the (negative log) probability that the segment represents an occurrence of the ith allophone in the label set. For each segment, SEG creates a hypothesis at the segment level and associates with it the vector of estimated allophone probabilities. The several highest rated labels of each segment are shown in Figure 5c.

Step 2. KS: WORD-CTL.
Stimulus: Start of processing.
Action: Create goal hypotheses at the word level. These will control the amount of hypothesization that MOW will do. (The goal hypotheses are not shown in Figure 5.)

Step 3. KS: WORD-SEQ-CTL.
Stimulus: Start of processing.
Action: Create goal hypotheses at the word-sequence le• These will control the amount of hypothesization that WORD-SEQ will do.

Step 4. KS: POM.
Stimulus: New segment hypotheses.
Action: Create syllable-class hypotheses.

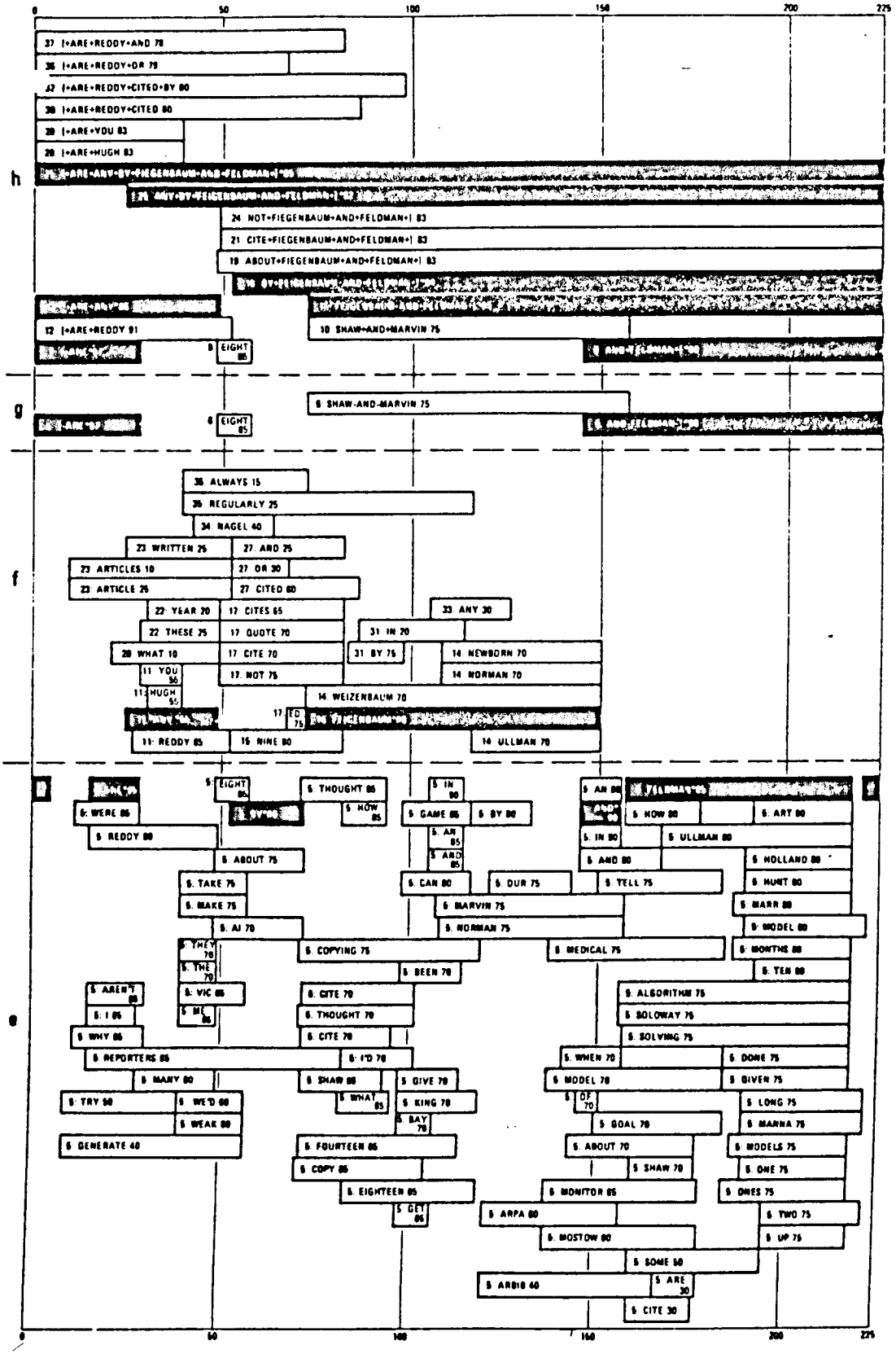

0
50
100
150
200
225
h
g
f
e
d
c
b
a
37 |+ARE+REDDY+AND 78
36 |+ARE+REDDY+OR 79
32 |+ARE+REDDY+CITED+BY 80
30 |+ARE+REDDY+CITED 80
28 |+ARE+YOU 83
28 |+ARE+HUGH 83
24 NOT+FIEGENBAUM+AND+FELDMAN+| 83
21 CITE+FIEGENBAUM+AND+FELDMAN+| 83
19 ABOUT+FIEGENBAUM+AND+FELDMAN+| 83
12 |+ARE+REDDY 91
10 SHAW+AND+MARVIN 75
8 EIGHT 85
6 SHAW-AND-MARVIN 75
6 EIGHT 85
36 ALWAYS 15
35 REGULARLY 25
34 NAGEL 40
23 WRITTEN 25
27 AND 25
23 ARTICLES 10
27 OR 30
23 ARTICLE 25
27 CITED 60
22 YEAR 20
17 CITES 65
33 ANY 30
22 THESE 25
17 QUOTE 70
31 IN 20
20 WHAT 10
17 CITE 70
31 BY 75
14 NEWBORN 70
11 YOU 55
17 NOT 75
14 NORMAN 70
11 HUGH 55
14 WEIZENBAUM 70
17 ED 75
11 REDDY 85
15 NINE 80
14 ULLMAN 70
5 EIGHT 85
5 THOUGHT 85
5 HOW 85
5 IN 80
5 WERE 85
5 GAME 85
5 BY 80
5 AN 85
5 AND 85
5 REDDY 80
5 ABOUT 75
5 TAKE 75
5 CAN 80
5 OUR 75
5 TELL 75
5 MAKE 75
5 MARVIN 75
5 AI 70
5 NORMAN 75
5 THEY 70
5 COPYING 75
5 MEDICAL 75
5 THE 70
5 BEEN 70
5 VIC 85
5 CITE 70
5 THOUGHT 70
5 I 85
5 CITE 70
5 WHY 85
5 REPORTERS 85
5 I'D 70
5 MANY 80
5 SHAW 80
5 GIVE 70
5 TRY 90
5 WE'D 80
5 WHAT 85
5 KING 70
5 WEAK 80
5 BAY 70
5 GENERATE 40
5 FOURTEEN 85
5 COPY 85
5 EIGHTEEN 85
5 GET 85
5 AN 80
5 HOW 80
5 ART 80
5 IN 80
5 ULLMAN 80
5 AND 80
5 HOLLAND 80
5 YELL 75
5 HUNT 80
5 MARR 80
5 MODEL 80
5 MONTHS 80
5 TEN 80
5 ALGORITHM 75
5 SOLOWAY 75
5 SOLVING 75
5 WHEN 70
5 DONE 75
5 MODEL 70
5 GIVEN 75
5 OF 70
5 LONG 75
5 GOAL 70
5 MANNA 75
5 ABOUT 70
5 MODELS 75
5 SHAW 70
5 ONE 75
5 MONITOR 85
5 ONES 75
5 ARPA 80
5 TWO 75
5 MOSTOW 80
5 UP 75
5 SOME 50
5 ARBIB 60
5 ARE 30
5 CITE 30

TABLE 1. PHONE CLASSES USED TO DEFINE THE SYLLABLE CLASSES

Code	Phone Class	Phones in Class
A	A-like	AE, AA, AH, AO, AX
I	I-like	IY, IH, EY, EH, IX, AY
U	U-like	OW, UH, U, UW, ER, AW, OY, EL, EM, EN
L	Liquid	Y, W, R, L
N	Nasal	M, N, NX
P	Stop	P, T, K, B, D, G, DX
F	Fricative	HH, F, TH, S, SH, V, DH, Z, ZH, CH, JH, WH

Using the labeled segments as input, the *POM* knowledge source [SMIT76] generates hypotheses for likely syllable classes. This is done by first identifying syllable nuclei and then parsing outward from each nucleus, using a probabilistic grammar with production rules of the form:

syllable-class → segment-sequence.

The rules and their probabilities are induced by an off-line program that trains on manually segmented and labeled utterances. For each nucleus position, several (typically three to eight) competing syllable-class hypotheses may be generated.

Figure 5d shows the syllable-class hypotheses created. Each class name is made up of single-letter codes representing classes of phones, as given in Table 1.

Step 5. KS: MOW.
Stimulus: New syllable hypotheses.[6]
Action: Create word hypotheses.

The syllable classes are used by MOW in step 5 to hypothesize words. Each of the 1011 words in the vocabulary is specified by a pronunciation description. For word hypothesization purposes, an inverted form of the dictionary is kept; this associates each syllable class with all words whose pronunciation contains it. The *MOW* KS [SMIT76] looks up each hypothesized syllable class in the dictionary and generates word candidates from among those words containing that syllable class. For each word that is multisyllabic, all of the syllables in one of the pronunciations must match with a rating above a specified threshold. Typically, about 50 words of the 1011-word vocabulary are generated at each syllable nucleus position.

Finally, the generated word candidates are rated and their begin- and end-times adjusted by the *WIZARD* procedure [MCKE77]. For each word in the vocabulary, WIZARD has a network description of its possible pronunciations. A word rating is calculated by finding the one path through the network which most closely matches the labeled segments, using the probabilities associated with the segment for each label; the resultant rating reflects the difference between this optimal path and the segment labels.[7]

Processing to this point has resulted in a set of bottom-up word candidates. Each word includes a begin-time, an end-time, and a credibility rating. MOW selects a subset of these words, based on their times and ratings, to be hypothesized; these selected word hypotheses form the base for the top-end processing. Words not immediately hypothesized are retained internally by MOW for possible later hypothesization.[8]

The amount of hypothesization that MOW does is controlled by the *WORD-CTL* (Word Control) KS. At step 2, WORD-CTL created initial goal hypotheses at the word level; these are interpreted by MOW as indicating how many word hypotheses to attempt to create in each time area. Subsequently, WORD-CTL may retrigger and modify the goal hypotheses (and thus retrigger MOW) if the overall search process stagnates; this condition is recognized when there are no waiting KS instantiations above a threshold priority or when the global measures of current state of the problem solution have

[6] MOW will also be reinvoked upon a modification to the word goal hypotheses by WORD-CTL.

[7] WIZARD is, in effect, a miniature version of the HARPY speech-recognition system (see Section 2.3), except that it has a network for each word, rather than one network containing all sentences.

[8] Since the September 1976 version, the POM and MOW KSs have been replaced by *NOAH* [SMIT77, SMIT81]. This KS outperforms, in both speed and accuracy, POM and MOW (with WIZARD) on the 1011-word vocabulary and is able to handle much larger vocabularies; its performance degradation is only logarithmic in vocabulary size, in the range of 500 to 19,000 words.

not improved in the last several KS executions.

WORD-CTL (and WORD-SEQ-CTL) are examples of KSs not directly involved in the hypothesizing and testing of partial solutions. Instead, these KSs control the search by influencing the activations of other KSs. These *policy* KSs impose global search strategies on the basic priority scheduling mechanism. For example, MOW is a generator of word hypotheses (from the candidates it creates internally) and WORD-CTL controls the number to be hypothesized. This clear separation of policy from mechanism has facilitated experimentation with various control schemes. A trivial change to WORD-CTL such that goal hypotheses are generated only at the start of the utterance (left-hand end) results in MOW creating word hypotheses only at the start, thus forcing all top-end processing to be left-to-right (see Section 3.2).

In this example four words (ARE, BY, AND, and FELDMAN) of the six in the utterance were correctly hypothesized; 86 incorrect hypotheses were generated (see Figure 5e). The 90 words that were hypothesized represent approximately 1.5 percent of the 1011-word vocabulary for each one of the six words in the utterance.

In addition, two unique word-level hypotheses are generated before the first and after the last segment of the utterance to denote the start and end of utterance, respectively. They are denoted by [and].

Step 6. KS: WORD-SEQ.
Stimulus: New words created bottom-up.
Action: Create four word-sequence hypotheses:
[–ARE*(97, 0:28),
AND–FELDMAN–]*(90, 145:225),
EIGHT(85, 48:57).
SHAW–AND–MARVIN(75, 72:157),

The *WORD-SEQ* knowledge source [LESS77a] has the task of generating, from the bottom-up word hypotheses, a small set (about three to ten) of word-sequence hypotheses. Each of these sequences, or *islands*, can be used as the basis for expansion into larger islands, which it is hoped will culminate in a hypothesis spanning the entire utterance. Multiword islands are used rather than single-word islands because of the relatively poor reliability of ratings of single words. With multiword islands, syntactic and coarticulation constraints can be used to increase the reliability of the ratings.

WORD-SEQ uses three kinds of knowledge to generate multiword islands efficiently:

(1) A table derived from the grammar indicates for every ordered pair of words in the vocabulary (1011 × 1011) whether that pair can occur in sequence within some sentence of the defined language. This binary table, whose density of ones for the X05 grammar is 1.7 percent, defines a *language-adjacent* relation.
(2) Acoustic-phonetic knowledge, embodied in the *JUNCT* (juncture) procedure [CRON77], is applied to pairs of word hypotheses and is used to decide if that pair might be considered to be *time-adjacent* in the utterance. JUNCT uses the dictionary pronunciations, and examines the segments at their juncture (gap or overlap) in making its decision.
(3) Statistical knowledge is used to assess the credibility of generated alternative word sequences and to terminate the search for additional candidates when the chance of finding improved hypotheses drops. The statistics are generated from previously observed behavior of WORD-SEQ and are based on the number of hypotheses generable from the given bottom-up word hypotheses and their ratings.

WORD-SEQ takes the highest rated single words and generates multiword sequences by expanding them with other hypothesized words that are both time- and language-adjacent. This expansion is guided by credibility ratings generated by using the statistical knowledge. The best of these word sequences (which occasionally includes single words) are hypothesized.

The *WORD-SEQ-CTL* (Word-Sequence-Control) KS controls the amount of hypothesization that WORD-SEQ does by creating "goal" hypotheses that WORD-SEQ interprets as indicating how many hypotheses to create. This provides the same kind of separation of policy and mechanism

achieved in the MOW/WORD-CTL pair of KSs. WORD-SEQ-CTL fired at the start of processing, at step 3, in order to create the goal hypotheses. Subsequently, WORD-SEQ-CTL may trigger if stagnation is recognized; it then modifies the word-sequence goal hypotheses, thus stimulating WORD-SEQ to generate new islands from which the search may prove more fruitful. WORD-SEQ may generate the additional hypotheses by decomposing word sequences already on the blackboard or by generating islands previously discarded because their ratings seemed too low.

Step 6 results in the generation of four multiword sequences (see Figure 5g). These are used as initial, alternative anchor points for additional searching. Note that two of these islands are correct, each representing an alternative search path that potentially can lead to a correct interpretation of the utterance. This ability to derive the correct interpretation in multiple ways makes the system more robust. For example, there have been cases in which a complete interpretation could not be constructed from one correct island because of KS errors but was derived from another island.

High-level processing on the multiword sequences is accomplished by the following KSs: PARSE, PREDICT, VERIFY, CONCAT, STOP, and WORD-SEQ-CTL. Since an execution of the VERIFY KS will often immediately follow the execution of the PREDICT KS (each on the same hypothesis), we have combined the descriptions of the two KS executions into one step for ease of understanding.

Because the syntactic constraint used in the generation of the word sequences is only pairwise, a sequence longer than two words might not be syntactically acceptable. The *PARSE* knowledge source [HAYE77b] can parse a word sequence of arbitrary length, using the full grammatical constraints. This parsing does not require that the word sequence form a complete nonterminal in the grammar or that the sequence be sentence-initial or sentence-final; the words need only occur contiguously in some sentence of the language. If a sequence hypothesis does not parse, it is marked as "rejected." Otherwise a phrase hypothesis is created. Associated with the phrase hypothesis is the word sequence that supports it, as well as information about the parse(s).

Steps 7 through 10 show the PARSE KS processing each of the multiword sequences. In this example all four multiword sequences were verified as valid language fragments. However, if a multiword sequence had been rejected, the WORD-SEQ KS might have been reinvoked to generate additional multiword sequences in the time area of the rejected one. WORD-SEQ would generate the additional hypotheses by decomposing (shortening) word-sequence islands already on the blackboard or by regenerating islands which may not have been hypothesized initially owing to low ratings. Additional word-sequence hypotheses might also be generated in response to the modification of "goal" hypotheses at the word-sequence level by the WORD-SEQ-CTL. Such a structuring of a KS as a *generator* is a primary mechanism in Hearsay-II for limiting the number of hypotheses created on the blackboard and thereby reducing the danger of a combinatorial explosion of KS activity in reaction to those hypotheses.

The scheduling strategy is parameterized to delay phrase-level processing until an adequate number of highly rated phrase hypothesis islands is generated. This strategy is not built directly into the scheduler, but rather is accomplished by (1) appropriately setting external scheduling parameters (i.e., the high setting of the priorities of WORD-SEQ and PARSE KS actions in contrast to those of PREDICT, VERIFY, and CONCAT),[9] and (2) taking into account the current state of hypotheses on the phrase level of the blackboard in evaluating the usefulness of potential KS actions as described by their response frames.

Step 7. KS: PARSE*.
Stimulus: [−ARE* (word sequence).
Action: Create phrase: [+ARE* (97, 0:28).

Step 8. KS: PARSE*.
Stimulus: AND−FELDMAN−]* (word sequence).

[9] These settings are determined empirically by observing a number of training runs. They are not adjusted during test runs of the system.

Action: Create phrase:
AND+FELDMAN+]* (90, 145:225).

Step 9. KS: PARSE.
Stimulus: EIGHT (word sequence).
Action: Create phrase EIGHT (85, 48:57).

Step 10. KS: PARSE.
Stimulus: SHAW–AND–MARVIN (word sequence).
Action: Create phrase: SHAW+AND+MARVIN (75, 72:157).

Each of the four executions of the PARSE KS (steps 7–10) results in the creation of a phrase hypothesis; these are shown in Figure 5h. Each of these hypotheses causes an invocation of the PREDICT KS.

The *PREDICT* knowledge source [HAYE 77b] can, for any phrase hypothesis, generate predictions of all words which can immediately precede and all which can immediately follow that phrase in the language. In generating these predictions this KS uses the parsing information attached to the phrase hypothesis by the parsing component. The action of PREDICT is to attach a "word-predictor" attribute to the hypothesis which specifies the predicted words. Not all of these PREDICT KS instantiations are necessarily executed (and thus indicated as a step in the execution history). For instance, further processing on the phrases [+ARE and AND+FELDMAN+] is sufficiently positive that the scheduler never executes the instantiation of PREDICT for the phrase SHAW+AND+MARVIN (created in step 10).

The *VERIFY* KS can attempt to verify the existence of or reject each such predicted word in the context of its predicting phrase. If verified, a confidence rating for the word is also generated. The verification proceeds as follows: First, if the word has been hypothesized previously and passes the test for time-adjacency (by the JUNCT procedure), it is marked as verified and the word hypothesis is associated with the prediction. (Note that some word hypotheses may thus become associated with several different phrases.) Second, a search is made of the internal store created by MOW to see if the prediction can be matched by a previously generated word candidate which had not yet been hypothesized. Again, JUNCT makes a judgment about the plausibility of the time-adjacency relationship between the predicting phrase and the predicted word. Finally, WIZARD compares its word-pronunciation network with the segments in an attempt to verify the prediction.

For each of these different kinds of verification, the approximate begin-time (end-time if verifying an antecedent prediction) of the word being predicted following (preceding) the phrase is taken to be the end-time (begin-time) of the phrase. The end-time (begin-time) of the predicted word is not known, and in fact one function of the verification step is to generate an approximate end-time (begin-time) for the verified word. In general, it is possible to generate several different "versions" of the word which differ primarily in their end-times (begin-times); since no context following (preceding) the predicted word is given, several different estimates of the end (beginning) of the word may be plausible solely on the basis of the segmental information. These alternatives give rise to the creation of competing hypotheses.

VERIFY is invoked when a KS (PREDICT) places a "word-predictor" attribute on a phrase hypothesis. For each word on the attribute list that it verifies, it creates a word hypothesis (if one does not already exist), and the word is placed on a "word-verification" attribute of the phrase hypothesis. Word hypotheses created by VERIFY are shown in Figure 5f.

The *CONCAT* KS [HAYE77b] is invoked on phrase hypotheses which have word-verification attributes attached. For each verified word, the phrase and new word are parsed together and a new, extended phrase hypothesis is created (as shown in Figure 5h). The new hypothesis receives a rating based on the ratings of its component words. If all word predictions preceding or following the phrase had been rejected, the phrase hypothesis itself would be marked as "rejected." Similarly, the underlying word-sequence hypothesis is rejected if all the phrase hypotheses it supports are rejected. (This action would retrigger WORD-SEQ to generate more word sequences.)

Step 11. KS: PREDICT & VERIFY*.
Stimulus: [+ARE* (phrase).

Action: Predict (from the grammar) 292 words following. Reject (using the acoustic information) 277 of them. The four highest rated of the 15 verified words are
REDDY(85, 26:52),
ANY*(65, 24:49),
HUGH(55, 30:39), and
YOU(55, 28:39).

Step 12. KS: CONCAT.
Stimulus: [+ARE* (phrase), REDDY (word).
Action: Create phrase: [+ARE+REDDY (91, 0:52).

Step 13. KS: CONCAT*.
Stimulus: [+ARE* (phrase), ANY* (word).
Action: Create phrase: [+ARE+ANY* (86, 0:49).

In steps 11 through 13 the highly rated phrase [+ARE is extended and results in the generation of the additional phrases [+ARE+REDDY and [+ARE+ANY. These phrases, however, are not immediately extended because the predicted words REDDY and ANY are not rated sufficiently high. Instead, the scheduler, pursuing a strategy more conservative than strict best-first, investigates phrases that look almost as good as the best one. This scheduling strategy results in the execution of the PREDICT and VERIFY KSs on two of the other initial phrase islands: AND+ FELDMAN+] and EIGHT.

Step 14. KS: PREDICT & VERIFY*.
Stimulus: AND+FELDMAN+]* (phrase).
Action: Predict 100 words preceding. Reject 76 of them. The best of the verified 24 (in descending rating order) are
FEIGENBAUM*(80, 72:150),
WEIZENBAUM(70, 72:150),
ULLMAN(70, 116:150),
NORMAN(70, 108:150), and
NEWBORN(70, 108:150).

Step 15. KS: PREDICT & VERIFY.
Stimulus: EIGHT (phrase).
Action: Predict the word NINE following and verify it (80, 52:82). Predict SEVEN preceding, but reject this because of mismatch with the acoustic segments.

The attempt to extend the phrase EIGHT at step 15 is not successful; none of the grammatically predicted words are acoustically verified, even using a lenient threshold. Thus this phrase is marked rejected and is dropped from further consideration.

Step 16. KS: CONCAT*.
Stimulus: FEIGENBAUM* (word), AND+ FELDMAN+]* (phrase).
Action: Create phrase: FEIGENBAUM+ AND+FELDMAN+]* (85, 72:225).

Beginning with step 16, extending the phrase AND+FELDMAN+] with the highly rated word FEIGENBAUM looks sufficiently promising for processing to continue now in a more depth-first manner along the path FEIGENBAUM+AND+ FELDMAN+] through step 25.[10] Processing on the path [+ARE+REDDY does not resume until step 26.

Step 17. KS: PREDICT & VERIFY*.
Stimulus: FEIGENBAUM+AND+FELDMAN+]* (phrase).
Action: Predict eight preceding words. Reject one (DISCUSS). Find two already on the blackboard:
BY*(80, 52:72) and
ABOUT(75, 48:72).
Verify five others:
NOT(75, 49:82),
ED(75, 67:72),
CITE(70, 49:82),
QUOTE(70, 49:82),
CITES(65, 49:82).

In steps 18 through 24, alternative word extensions of FEIGENBAUM+AND+ FELDMAN+] are explored. As a result of this exploration the phrase BY+FEIGENBAUM+AND+FELDMAN+] is considered the most credible.

Step 18. KS CONCAT*.
Stimulus: BY* (word), FEIGENBAUM+ AND+FELDMAN+]* (phrase).
Action: Create phrase: BY+FEIGENBAUM+AND+FELDMAN+]* (84, 52:225).

Step 19. KS: CONCAT.
Stimulus: ABOUT (word), FEIGENBAUM+ AND+FELDMAN+]* (phrase).
Action: Create phrase: ABOUT+FEIGENBAUM+AND+FELDMAN+] (83, 48:225).

Step 20. KS: PREDICT & VERIFY.
Stimulus:
ABOUT+FEIGENBAUM+AND+ FELDMAN+] (phrase).

[10] The rating on a hypothesis is only one parameter used by the scheduler to assign priorities to waiting KS instantiations. In particular, the length of a hypothesis is also important. Thus, FEIGENBAUM with a rating of 80 looks better than REDDY with a rating of 85 because it is much longer.

Action: Predict one preceding word: WHAT. Verify it (10, 20:49).

Step 21. KS: CONCAT
Stimulus: CITE (word), FEIGENBAUM+ AND+FELDMAN+] (phrase).
Action: Create phrase: CITE+FEIGEN-BAUM+AND+FELDMAN+] (83, 49:225).

Step 22. KS: PREDICT & VERIFY.
Stimulus: CITE+FEIGENBAUM+AND+ FELDMAN+] (phrase).
Action: Predict four preceding words. Reject two of them: BOOKS, PAPERS. Verify
THESE (25, 28:49),
YEAR (20, 30:49).

Step 23. KS: PREDICT & VERIFY*.
Stimulus: BY+FEIGENBAUM+AND+ FELDMAN+]* (phrase).
Action: Predict ten preceding words. Reject five: ABSTRACTS, ARE, BOOKS, PA-PERS, REFERENCED. Find two already on the blackboard:
ANY* (65, 24:49),
THESE (25, 28:49).
Verify three more:
ARTICLE (25, 9:52),
WRITTEN (25, 24:52),
ARTICLES (10, 9:52).

Step 24. KS: CONCAT.
Stimulus: NOT (word), FEIGENBAUM+ AND+FELDMAN+]*.
Action: Create phrase: NOT+FEIGEN-BAUM+AND+FELDMAN+] (83, 49:225).

Step 25. KS: CONCAT*.
Stimulus: ANY* (word), BY+FEIGEN-BAUM+AND+FELDMAN+]* (phrase).
Action: Create phrase: ANY+BY+FEIGEN-BAUM+AND+FELDMAN+]* (82, 24:225). [+ARE+ANY+BY+FEIGENBAUM+ AND+FELDMAN+]* (85, 0:225) is also created, from [+ARE+ANY and BY+FEIGEN-BAUM+ AND+FELDMAN+].

In step 25 the word ANY is concatenated onto the phrase BY+FEIGENBAUM+ AND+FELDMAN+]. However, instead of only creating this new combined phrase, the CONCAT KS also notices that the word ANY is the last word of the phrase [+ARE+ANY; this leads the CONCAT KS to merge the two adjacent phrases [+ARE+ANY and BY+FEIGENBAUM+ AND+FELDMAN+] into a single phrase, after first ascertaining that the resulting phrase is grammatical. This merging bypasses the several single-word PREDICT, VERIFY, and CONCAT actions needed to generate the enlarged hypothesis from either of the two original hypotheses in an incremental fashion. Thus the recognition process is sped up in two ways: (1) several single-word actions are eliminated, and (2) the scheduler postpones KS actions on competing (incorrect) hypotheses since these potential actions seem less attractive than actions on the new, enlarged hypothesis. Such mergings occur in approximately half of the runs on the 1011-word grammar with the small branching factor (X05); in grammars with higher branching factors, the merging of phrase hypotheses occurs with even higher frequency.

It has been our experience that just as a multiword island is more credible than the individual words that compose it, so a merged phrase hypothesis is more credible than its two constituent phrases. For example, about 80 percent of the mergings in X05 runs produce correct hypotheses. In more complex grammars this statistic drops to about 35 percent, but correspondingly more phrase mergings occur.

The newly created merged phrase also happens to be a complete sentence; i.e., it has begin- and end-of-utterance markers as its extreme constituents. Thus it is a candidate for the interpretation of the utterance.

Step 26. KS: STOP.
Stimulus: [+ARE+ANY+BY+FEIGEN-BAUM+AND+FELDMAN+]* (complete sentence).
Action: Deactivation of several dozen competing hypotheses.

These start- and end-of-utterance "words" (denoted by [and]) appear in the syntactic specification of the language as the first and last terminals of every complete sentence. Thus any verified phrase that includes these as its extreme constituents is a complete sentence and spans the entire utterance. Such a sentence becomes a candidate for selection as the system's recognition result.

In general, the control and rating strategies do not guarantee that the first such complete spanning hypothesis found will have the highest rating of all possible spanning sentence hypotheses that might be found if the search were allowed to con-

tinue, so the system does not just stop with the first one generated.[11] However the characteristics of such a hypothesis are used by the *STOP* knowledge source to prune from further consideration other partial hypotheses which, because of their low ratings, are unlikely to be extendible into spanning hypotheses with ratings higher than the best already-discovered spanning sentence. This heuristic pruning procedure is based on the form of the ratings function (i.e., how the rating of the phrase is derived from its constituent words). The pruning procedure considers each partial phrase and uses the ratings of other word hypotheses in the time areas not covered by the phrase to determine if the phrase might be extendible to a phrase rated higher than the spanning hypothesis; if not, the partial phrase is pruned. This pruning process and the rating and halting policies are discussed in MOST77.

Of the hypotheses not discarded, extensions to the phrase [+ARE now appear as the most likely candidates to produce new and better complete phrases. This search for better complete phrases results, in steps 27 through 36, in the examination of numerous alternative extensions, each of which is promising.

Step 27. KS: PREDICT & VERIFY.
Stimulus: [+ARE+REDDY.
Action: Predict three following words. Verify CITED (60, 52:86), OR (30, 52:67), AND (25, 52:82).

Step 28. KS: CONCAT.
Stimulus: [+ARE* (phrase), HUGH (word).
Action: Create phrase: [+ARE+HUGH (83, 0:39).

Step 29. KS: CONCAT.
Stimulus: [+ARE (phrase), YOU (word).
Action: Create phrase: [+ARE+YOU (83, 0:39).

Step 30. KS: CONCAT.
Stimulus: [+ARE+REDDY (phrase), CITED (word).
Action: Create phrase: [+ARE+REDDY+CITED (80, 0:86).

Step 31. KS: PREDICT & VERIFY.
Stimulus: [+ARE+REDDY+CITED (phrase).
Action: Predict two following words. Verify BY(75, 83:98), IN(20, 86:114).

Step 32. KS: CONCAT.
Stimulus: [+ARE+REDDY+CITED (phrase), BY (word).
Action: Create phrase: [+ARE+REDDY+CITED+BY (80, 0:98).

Step 33. KS: PREDICT & VERIFY.
Stimulus: [+ARE+REDDY+CITED+BY (phrase).
Action: Predict one following word. Verify ANY (30, 105:126).

Step 34. KS: PREDICT & VERIFY.
Stimulus: [+ARE+HUGH (phrase).
Action: Predict one following word. Verify NAGEL (40, 42:63).

Step 35. KS: PREDICT & VERIFY.
Stimulus: [+ARE+YOU (phrase).
Action: Predict three following words. Reject USUALLY. Verify
REGULARLY (25, 39:116),
ALWAYS (15, 39:72).

Step 36. KS: CONCAT.
Stimulus: [+ARE+REDDY (phrase), OR (word).
Action: Create phrase: [+ARE+REDDY+OR (79, 0:67).

Step 37. KS: CONCAT.
Stimulus: [+ARE+REDDY (phrase), AND (word).
Action: Create phrase: [+ARE+REDDY+AND (78, 0:82).

Step 38. KS: STOP.
Stimulus: Stagnation
Action: Stop search and accept [+ARE+ANY+BY+FEIGENBAUM+AND+FELDMAN+]*.

The recognition processing finally halts in one of two ways: First, there may be no more partial hypotheses left to consider for prediction and extension. Because of the combinatorics of the grammar and the likelihood of finding some prediction rated at least above the absolute rejection threshold, this termination happens when the heuristic pruning procedure used by STOP and RPOL has eliminated all competitors. Such a halt occurs here as STOP decides to terminate the search process and accept the phrase [+ARE+ANY+BY+FEIGEN-

[11] An alternative control strategy based on the Shortfall Density scoring function and assumptions of the completeness of processing at the word level can guarantee that the first complete interpretation generated is the best possible (see Section 2.1). This results in a more conservative, breadth-first search, in which more alternatives are explored.

BAUM+AND+FELDMAN+] as the correct interpretation. In general there might be more than one complete sentence hypothesis at this point; STOP would select the one with the highest rating.

A second kind of halt occurs if the system expends its total allowed computing resources (time or space). (The actual thresholds used are set according to the past performance of the system on similar sentences, i.e., of the given length and over the same vocabulary and grammar.) In that case a selection of several of the highest rated phrase hypotheses is the result of the recognition process, with the selection biased toward the longest phrases which overlap (in time) the least.

Step 39. KS: SEMANT*.
Stimulus: Recognized utterance: [+ARE+ANY+BY+FEIGENBAUM+AND+FELDMAN+]*.
Action: Generate an interpretation for the database retrieval system.

The *SEMANT* knowledge source [Fox77] takes the word sequence(s) result of the recognition process and constructs an interpretation in an unambiguous format for interaction with the database that the speaker is querying. The interpretation is constructed by actions associated with "semantically interesting" nonterminals (which have been prespecified for the grammar) in the parse tree(s) of the recognized sequence(s). In our example the following structure is produced:

```
F:[U:([ARE ANY BY FEIGENBAUM AND
      FELDMAN])
   N:($PRUNE!LIST
      S:($PRUNE!LIST!AUTHOR  K:(A:
      ((FEIGENBAUM • FELDMAN)))))]
```

F denotes the total message. U contains the utterance itself. N indicates the main type of the utterance (e.g., PRUNE a previously specified list of citations, REQUEST, HELP), S the subtype (e.g., PRUNE a list according to its author). K denotes the different attributes associated with the utterance (e.g., A is the author, T is the topic).

If recognition produces more than one partial sequence, SEMANT constructs a maximally consistent interpretation based on all of the partial sentences, taking into account the rating, temporal position, and semantic consistency of the partial sentences.

The *DISCO* (discourse) knowledge source [HAYE77c] accepts the formatted interpretation of SEMANT and produces a response to the speaker. This response is often the display of a selected portion of the queried database. In order to retain a coherent interpretation across sentences, DISCO maintains a finite-state model of the ongoing discourse.

2. COMPARISON WITH OTHER SPEECH-UNDERSTANDING SYSTEMS

In addition to Hearsay-II, several other speech-understanding systems were also developed as part of the Defense Advanced Research Projects Agency (DARPA) research program in speech understanding from 1971 to 1976 [MEDR78]. As a way of concretely orienting the research, a common set of system performance goals, shown in Figure 6, was established by the study committee that launched the project [NEWE73]. All of the systems are based on the idea of diverse, cooperating KSs to handle the uncertainty in the signal and processing. They differ in the types of knowledge, interactions of knowledge, representation of search space, and control of the

FIGURE 6. DARPA speech-understanding-system performance goals set in 1971. [After NEWE73 and MEDR78.]

The system should
- Accept connected speech
- from many
- cooperative speakers of the General American Dialect
- in a quiet room
- using a good-quality microphone
- with slight tuning per speaker
- requiring only natural adaptation by the user
- permitting a slightly selected vocabulary of 1000 words
- with a highly artificial syntax and highly constrained task
- providing graceful interaction
- tolerating less than 10 percent semantic error
- in a few times real time on a 100-million-instructions-per-second machine
- and be demonstrable in 1976 with a moderate chance of success.

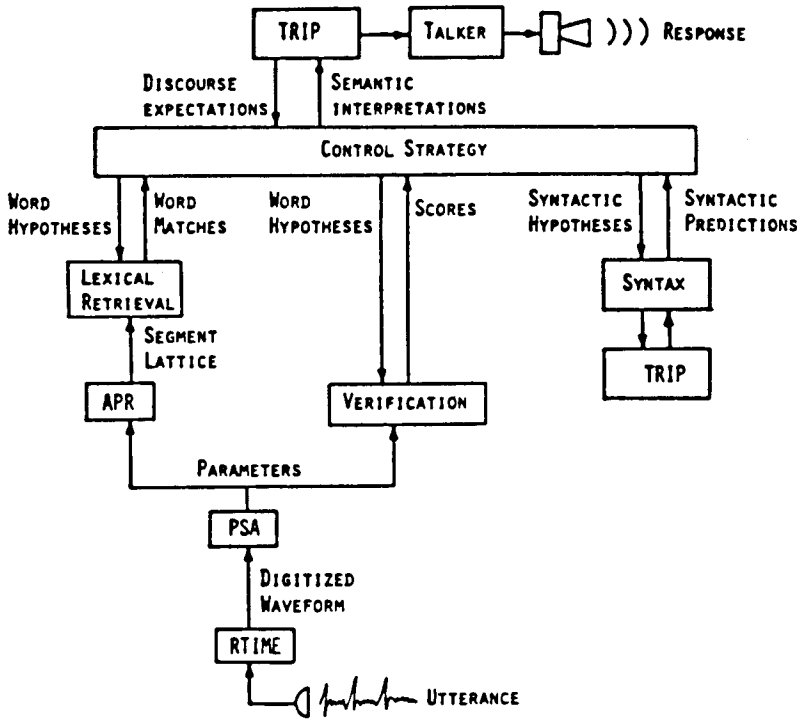


FIGURE 7. Structure of HWIM. [From WOLF80.]

search. (They also differ in the tasks and languages handled, but we do not address those here.) In this section we describe three of these systems, Bolt Beranek and Newman's (BBN's) HWIM, Stanford Research Institute's (SRI's) system, and Carnegie-Mellon University's (CMU's) HARPY, and compare them with Hearsay-II along those dimensions. For consistency we will use the terminology developed in this paper in so far as possible, even though it is often not identical to that used by the designers of each of the other systems.[12]

Although the performance specifications had the strong effect of pointing the various efforts in the same directions, the backgrounds and motivations of each group led to different emphases. For example, BBN's expertise in natural-language processing and acoustic-phonetics led to an emphasis on those KSs; SRI's interest in semantics and discourse strongly influenced its system design; and CMU's predilection for system organization placed that group in the central position (and led to the Hearsay-II and HARPY structures).

2.1 BBN's HWIM System

Figure 7 shows the structure of BBN's *HWIM* (Hear What I Mean) system [WOOD76, WOLF80]. In overall form, HWIM's general processing structure is strikingly similar to that of Hearsay-II. Processing of a sentence is bottom-up through audio signal digitization, parameter extraction, segmentation and labeling, and a scan for word hypotheses; this phase is roughly similar to Hearsay-II's initial bottom-up processing up through the MOW KS.

Following this initial phase, the Control Strategy module takes charge, calling the Syntax and Lexical Retrieval KSs as subroutines:

- The grammar is represented as an augmented transition network [WOOD70], and, as in Hearsay-II, includes semantic

[12] IBM has been funding work with a somewhat different objective [BAHL76]. Its stated goals mandate little reliance on the strong syntactic/semantic/task constraints exploited by the DARPA projects. This orientation is usually dubbed *speech recognition* as distinguished from *speech understanding*.

and pragmatic knowledge of the domain (in this case, "travel planning"). The Syntax KS combines the functions of Hearsay-II's PREDICT and CONCAT KSs. Like them, it handles contiguous sequences of words in the language, independently of the phrase structure nonterminal boundaries, as well as the merging of phrase hypotheses (i.e., island collision).

- The Lexical Retriever functions in this phase much like Hearsay-II's VERIFY KS, rating the acoustic match of a predicted word at one end of a phrase. Some configurations of HWIM also have a KS which does an independent, highly reliable, and very expensive word verification; that KS is also called directly by the Control Strategy.
- The Control Strategy module schedules the Syntax and Lexical Retrieval KSs opportunistically. To this end it keeps a task agenda that prioritizes the actions on the most promising phrase hypotheses. The task agenda is initialized with single-word phrase hypotheses constructed from the best word hypotheses generated in the bottom-up phase.

Given these similarities between HWIM and Hearsay-II, what besides the content of the KSs (which we do not address) are the differences? The most significant differences involve the mechanisms for instantiating KSs, scheduling KSs (i.e., selective attention for controlling the search), and representing, accessing, and combining KS results. These differences stem primarily from differing design philosophies:

- The Hearsay-II design was based on the assumption that a very general and flexible model for KS interaction patterns was required because the type, number, and interaction patterns of KSs would change substantially over the lifetime of the system [LESS75, LESS77b]. Thus we rejected an explicit subroutine-like architecture for KS interaction because it reduces modularity. Rather, the implicit data-directed approach was taken, in which KSs interact uniformly and anonymously via the blackboard.
- The HWIM design evolved out of an *incremental simulation* methodology [WOOD73]. In this methodology the overall system is implemented initially with some combination of computer programs and human simulators, with the latter filling the role of components (i.e., KSs and scheduling) not fully conceptualized. As experience is gained, the human simulators are replaced by computer programs. Thus by the time the system has evolved into a fully operational computer program, the type of KSs and their interaction patterns are expected to be stable. Modifications after this point aim to improve the performance of individual KSs and their scheduling, with only minor changes expected in KS interaction patterns. From this perspective, developing specific explicit structures for explicit KS interactions is reasonable.

Thus HWIM has an explicit control strategy, in which KSs directly call each other, and in which the scheduler has built-in knowledge about the specific KSs in the system. The Hearsay-II scheduler has no such built-in knowledge but rather is given an abstract description of each KS instantiation by its creator condition program.

Similarly, one KS communicates with another in HWIM via ad hoc KS-specific data structures. The introduction of a new KS is expected to occur very rarely and requires either that it adopt some other KS's existing data representation or that its new formats be integrated into those KSs that will interact with it. Hearsay-II's blackboard, on the other hand, provides a uniform representation which facilitates experimentation with new or highly modified KSs.

When one KS in a hierarchical structure like that in HWIM calls another, it provides the called KS with those data it deems relevant. The called KS also uses whatever data it has retained internally plus what it might acquire by calling other KSs. Hearsay-II's blackboard, on the other hand, provides a place for all data known to all the KSs; one KS can use data created by a previous KS execution without the creator of the data having to know which KS will use the data and without the user KS having to know which KS might be able to create the data.

The ability to embed into the HWIM system a detailed model of the KSs and

their interaction patterns has had its most profound effect on the techniques developed for scheduling. Several alternative scheduling policies were implemented in the Control Strategy module. The most interesting of these, the "shortfall density scoring strategy" [WOOD77], can be shown formally to guarantee that the first complete sentence hypothesis constructed by the system is the best possible (i.e., highest rated) such hypothesis that it will ever be able to construct. Heuristic search strategies with this property are called *admissible* [NILS71]. This contrasts with the *approximate* Hearsay-II scheduling strategy, in which there is no guarantee at any point that a better interpretation cannot be found by continued search. Thus Hearsay-II requires a heuristic stopping decision, as described in Section 1.2. In HWIM an admissible strategy is possible because the scheduler can make some strong assumptions about the nature of KS processing: in particular, the algorithms used by the Lexical Retriever KS are such that it does not subsequently generate a higher rating for a predicted word than that of the highest rated word predicted in that utterance location by the initial, bottom-up processing.

An admissible strategy eliminates errors which an approximate strategy may make by stopping too soon. However, even when an admissible strategy can be constructed, it may not be preferable if it generates excessive additional search in order to guarantee its admissibility. More discussion of this issue in speech understanding can be found in WOLF80, WOOD77, MOST77, and HAYE80. Discussions of it in more general cases can be found in POHL70, HARR74, and POHL77.

Given that hypotheses are rated by KSs, combining on a single hypothesis several ratings generated by different KSs is a problem. A similar problem also occurs within a KS when constructing a hypothesis from several lower level hypotheses; the rating of the new one should reflect the combination of ratings of its components. Hearsay-II uses ad hoc schemes for such rating combinations [HAYE77d]. HWIM takes a formal approach, using an application of Bayes' theorem. To implement this, each KS's ratings are calibrated by using performance statistics gathered on test data. This uniform scheme for calibration and combination of ratings facilitates adding and modifying KSs. The issue of evaluating the combination of evidence from multiple sources is a recurrent problem in knowledge-based systems [SHOR75, DUDA78].

2.2 SRI's System

The SRI system [WALK78, WALK80], though never fully operational on a large vocabulary task, presents another interesting variant on structuring a speech-understanding system. Like the HWIM system, it uses an explicit control strategy with, however, much more control being centralized in the Control Strategy module. The designers of the system felt there was "a large potential for mutual guidance that would not be realized if all knowledge source communication was indirect" [WALK78, p. 84]. Part of this explicit control is embedded within the rules that define the phrases of the task grammar; each rule, in addition to defining the possible constituent structure for phrases in an extended form of BNF, contains procedures for calculating attributes of phrases and factors used in rating phrases. These procedures may, in turn, call as subroutines any of the knowledge sources in the system. The attributes include acoustic attributes related to the input signal, syntactic attributes (e.g., mood and number), semantic attributes such as the representation of the meaning of the phrase, and discourse attributes for anaphora and ellipsis. Thus the phrase itself is the basic unit for integrating and controlling knowledge-source execution.

The interpreter of these rules (i.e., the Syntax module) is integrated with the scheduling components to define a high-level Control Strategy module. Like Hearsay-II and HWIM, this control module opportunistically executes the syntax rules to predict new phrases and words from a given phrase hypothesis and executes the word verifier to verify predicted words. This module maintains a data structure, the "parse-net," containing all the word and phrase hypotheses constructed, and the at-

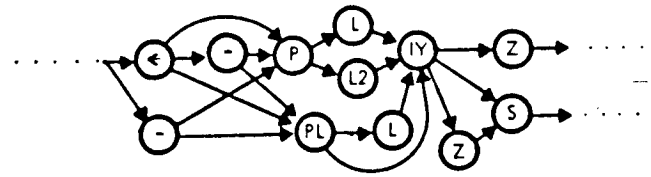


FIGURE 8. HARPY pronunciation network for the word "Please." [After LOWE80.]

tributes and factors associated with each hypothesis. This data structure is similar to a Hearsay-II blackboard restricted to the word and phrase levels. Like the blackboard, it serves to avoid redundant computation and facilitates the detection of possible island collisions.

As with Hearsay-II and HWIM, the SRI Control Strategy module is parameterized to permit a number of different strategies, such as top-down, bottom-up, island-driving, and left-to-right. Using a simulated word recognizer, SRI ran a series of experiments with several different strategies. One of the results, also substantiated by BBN experiments with HWIM, is that island-driving is inferior to some forms of left-to-right search. This appears to be in conflict with the Hearsay-II experimental results, which show island-driving clearly superior [LESS77a]. We believe the difference to be caused by the reliability of ratings of the initial islands: Both the HWIM and SRI experiments used single-word islands, but Hearsay-II uses multiword islands, which produce much higher reliability. (See the discussion at step 6 in Section 1.2 and in HAYE78b.) Single-word island-driving proved inferior in Hearsay-II as well.

2.3 CMU's HARPY System

In the systems described so far, knowledge sources are discernible as active components during the understanding process. However, if one looks at Hearsay-II, HWIM, and the SRI system in that order, there is clearly a progression of increasing integration of the KSs with the control structure. The HARPY system [LOWE76, LOWE80] developed at Carnegie-Mellon University is located at the far extreme of that dimension: Most of the knowledge is precompiled into a unified structure representing all possible utterances; a relatively simple interpreter then compares the spoken utterance against this structure to find the utterance that matches best. The motivation for this approach is to speed up the search so that a larger portion of the space may be examined explicitly. In particular, the hope is to avoid errors made when portions of the search space are eliminated on the basis of characteristics of small partial solutions; to this end, pruning decisions are delayed until larger partial solutions are constructed.

To describe HARPY, we describe the knowledge sources, their compilation, and the match (search) process. The *parameterization* and *segmentation* KSs are identical to those of Hearsay-II [GOLD77, GILL78]; these are not compiled into the network but, as in the other systems, applied to each utterance as it is spoken. As in Hearsay-II, the *syntax* is specified as a set of context-free production rules; HARPY uses the same task and grammar definitions. *Lexical* knowledge is specified as a directed pronunciation graph for each word; for example, Figure 8 shows the graph for the word "please." The nodes in the graph are names of the phonelike labels also generated by the labeler KS. A graph is intended to represent all possible pronunciations of the word. Knowledge about phonetic phenomena at *word junctures* is contained in a set of rewriting rules for the pronunciation graphs.

For a given task language, syntax and lexical and juncture knowledge are combined by a *knowledge compiler* program to form a single large network. First, the grammar is converted into a directed graph, the "word network," containing only terminal symbols (i.e., words); because of heuristics

```
⟨SENT⟩   ::= [ ⟨SS⟩ ]
⟨SS⟩     ::= please help ⟨M⟩ | please show ⟨M⟩ ⟨Q⟩
⟨Q⟩      ::= everything | something
⟨M⟩      ::= me | us
```

FIGURE 9. A tiny example grammar. [After LOWE80.]

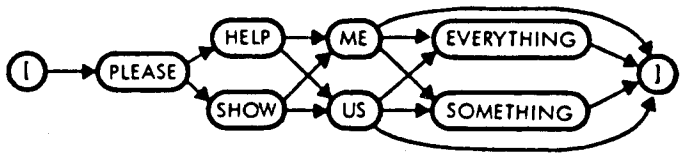


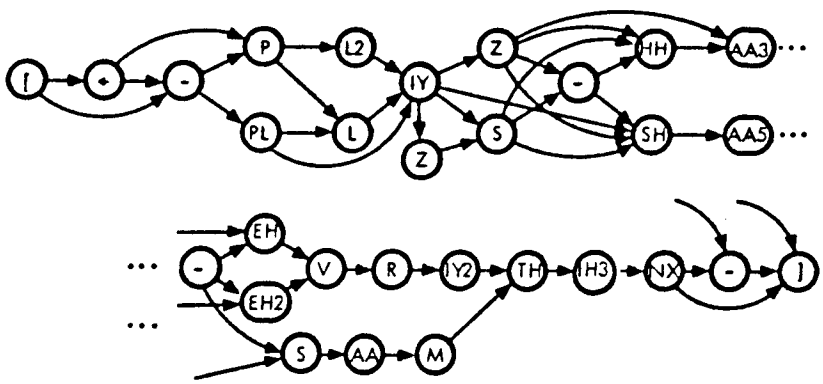
FIGURE 10. Word network for example language. [After LOWE80.]

used to compact this network, some of the constraint of the original grammar may be lost. Figure 9 shows a toy grammar, and Figure 10 the resulting word network. Next, the compiler replaces each word by a copy of its pronunciation graph, applying the word-juncture rules at all the word boundaries. Figure 11 shows part of the network for the toy example. The resulting network has the name of a segment label at each node. For the same 1011-word X05 language used by Hearsay-II, the network has 15,000 nodes and took 13 hours of DEC PDP-10(KL10) processing time to compile.

In the network each distinct path from the distinguished start node to the distinguished end node represents a sequence of segments making up a "legal" utterance. The purpose of the search is to find the sequence which most closely matches the segment sequence of the input spoken utterance. For any given labeled segment and any given node in the network, a primitive match algorithm can calculate a score for matching the node to the segment. The score for matching a sequence of nodes with a sequence of segments is just the sum of the corresponding primitive matches.

The search technique used, called *beam search*, is a heuristic form of dynamic programming, with the input segments processed one at a time from left to right and matched against the network. At the beginning of the ith step, the first $i-1$ segments have been processed. Some number of nodes in the network are *active*; associated with each active node is a path to it from the start node and the total score of the match between that path and the first $i-1$ segments of the utterance. All nodes

FIGURE 11. Partial final network for example language. [After LOWE80.]



in the network that are successors of the active nodes are matched against the ith segment and become the new active nodes. The score for a new active node is the best path score that reaches the node at the ith segment, i.e., the sum of the primitive match at the segment plus the best path score to any of its predecessor nodes.

The best path score among all the new active nodes is taken as the target, and any new active nodes with path scores more than some threshold amount from the target are pruned away. This pruning rule is the heuristic heart of the search algorithm. It reduces the number of active nodes at each step and thus reduces the amount of processing time (and storage) needed in the search; typically only about 3 percent of the nodes in the net need to be matched. Note that the heuristic does not fix the number of active nodes retained at each step but allows it to vary with the density of competitors with scores near the best path. Thus in highly uncertain regions, many nodes are retained, and the search slows down; in places where one path is significantly better than most others, few competitors are kept, and the processing is rapid. The search strategy, therefore, is automatically cautious or decisive in response to the partial results. The threshold, i.e., the "beam width," is tuned ad hoc from test runs.

There are two major concerns about the extensibility of HARPY. First, the compilation process requires all knowledge to be represented in a highly stylized form; adding new kinds of knowledge strains the developer's ingenuity. So far, however, several kinds of knowledge have been added within the basic framework of expanding a node by replacing it with a graph. For example, as mentioned previously, phonetic phenomena at word junctures are handled. Also, the expected length of each segment is stored at each node and influences the match score. The second concern is with the size and compilation cost of the compiled network; both grow very large as the task language becomes more complex. There have been proposals that the word network not be expanded explicitly, but rather that the word pronunciation graphs be interpreted dynamically, as needed. An alternative response to this concern is that computer memory and processing costs continue to decline, so that using larger networks becomes increasingly feasible.

HARPY's novel structure is also interesting in its own right and is beginning to have effects beyond speech-understanding systems. Newell has done a speculative but thorough analysis of HARPY as a model for human speech understanding, using the production system formalism [NEWE80]; Rubin has successfully applied the HARPY structure to an image-understanding task [RUBI78].

3. SYSTEM PERFORMANCE AND ANALYSIS

3.1 Overall Performance of Hearsay-II

Overall performance of the Hearsay-II speech-understanding system at the end of 1976 is summarized in Table 2 in a form paralleling the goals given in Figure 6.

TABLE 2. HEARSAY-II PERFORMANCE

Number of speakers	One
Environment	Computer terminal room (>65 dB)
Microphone	Medium-quality, close-talking
System speaker-tuning	20–30 training utterances
Speaker adaptation	None required
Task	Document retrieval
Vocabulary	1011 words, with no selection for phonetic discriminability
Language constraints	Context-free semantic grammar, based on protocol analysis, with static branching factor of 10
Test data	23 utterances, brand-new to the system and run "blind." 7 words/utterance average, 2.6 seconds/utterance average, average fanout[a] of 40 (maximum 292)
Accuracy	9 percent sentence semantic error,[b] 19 percent sentence error (i.e., not word-for-word correct)
Computing resources	60 MIPSS (million instructions per second of speech) on a 36-bit PDP-10

[a] The *static branching factor* is the average number of words that can follow any initial sequence as defined by the grammar. The *fanout* is the number of words that can follow any initial sequence in the test sentences.

[b] An interpretation is *semantically correct* if the query generated for it by the SEMANT KS is identical to that generated for a sentence which is *word-for-word* correct.

Active development of the Hearsay-II speech system ceased at the end of 1976 with the conclusion of the speech-understanding program sponsored by DARPA [MEDR78, KLAT77]. Even though the configuration of KSs at that point was young, having been assembled in August 1976, the performance described in Table 2 comes close to meeting the ambitious goals, shown in Figure 6, established for the DARPA program in 1971 [NEWE73]. This overall performance supports our assertion that the Hearsay-II architecture can be used to integrate knowledge for resolving uncertainty. In the following sections we relate some detailed analyses of the Hearsay-II performance to the resolution of uncertainty. We finish with some comparison with the performances of the other systems described in Section 2.

3.2 Opportunistic Scheduling

In earlier KS configurations of the system, low-level processing (i.e., at the segment, syllable, and word levels) was not done in the serial, lock-step manner of steps 1, 4, and 5 of the example, that is, level-to-level, where each level is completely processed before work on the next higher level is begun. Rather, processing was opportunistic and data-directed as in the higher levels; as interesting hypotheses were generated at one level, they were immediately propagated to and processed by KSs operating at higher and lower levels. We found, however, that opportunistic processing at the lower levels was ineffective and harmful because the credibility ratings of hypotheses were insufficiently accurate to form hypothesis islands capable of focusing the search effectively. For example, even at the relatively high word level, the bottom-up hypotheses created by MOW include only about 75 percent of the words actually spoken; and the KS-assigned ratings rank each correct hypothesis on the average about 4.5 as compared with the 20 or so incorrect hypotheses that compete with it (i.e., which overlap it in time significantly). It is only with the word-sequence hypotheses that the reliability of the ratings is high enough to allow selective search.

Several experiments have shown the effectiveness of the opportunistic search. In one [HAYE77a] the opportunistic scheduling was contrasted with a strategy using no ordering of KS activations. Here, all KS precondition procedures were executed, followed by all KS activations they created; this cycle was repeated. For the utterances tested, the opportunistic strategy had a 29 percent error rate (word for word), compared with a 48 percent rate for the non-opportunistic. Also, the opportunistic strategy took less than half as much processing time.[13]

In another experiment [LESS77a] the island-driving strategy, which is opportunistic across the whole utterance, was compared with a left-to-right strategy, in which the high-level search was initiated from single-word islands in utterance-initial position. For the utterances tested, the opportunistic strategy had a 33 percent error rate as compared with 53 percent for the left-to-right; for those utterances correctly recognized by both strategies, the opportunistic one used only 70 percent as much processing time.

3.3 Use of Approximate Knowledge

In several places the Hearsay-II system uses approximate knowledge, as opposed to its more complete form also included in the system. The central notion is that even though the approximation increases the likelihood of particular decisions being incorrect, other knowledge can correct those errors, and the amount of computational resources saved by first using the approximation exceeds that required for subsequent corrections.

The organization of the POM and MOW KSs is an example. The bottom-up syllable and word-candidate generation scheme approximates WIZARD matching all words in the vocabulary at all places in the utterance, but in a fraction of the time. The errors show up as poor ratings of the can-

[13] The performance results given here and in the following sections reflect various configurations of vocabularies, grammars, test data, halting criteria, and states of development of the KSs and underlying system. Thus the absolute performance results of each experiment are not directly comparable to the performance reported in Section 3.1 or to the results of the other experiments.

grammar :	05	15	F
vocabulary			
S	err = 5.9 percent	err = 20.6 percent	err = 20.6 percent
250 words	comp = 1.0	comp = 2.7	comp = 3.4
	fanout = 10	fanout = 17	fanout = 27
M	err = 5.9 percent		
500 words	comp = 1.1		
	fanout = 18		
X	err = 11.8 percent		
1011 words	comp = 2.0		
	fanout = 36		

err = = semantic error rate
comp = = average ratio of execution time to that of S05 case, for correct utterances
fanout = = fanout of the test sentences (see note a of Table 2, Section 3.1)
N = 34 utterances

FIGURE 12. Hearsay-II performance under varying vocabularies and grammars.

didate words and as missing correct words among the candidates. The POM-MOW errors are corrected by applying WIZARD to the candidates to create good ratings and by having the PREDICT KS generate additional candidates.

Another example is the WORD-SEQ KS. Recall that it applies syntactic and acoustic-phonetic knowledge to locate sequences of words within the lattice of bottom-up words and statistical knowledge to select a few most credible sequences. The syntactic knowledge only approximates the full grammar, but takes less than 1 percent as much processing time to apply. The errors WORD-SEQ makes because of the approximation (i.e., generating some nongrammatical sequences) are corrected by applying the full grammatical knowledge of the PARSE KS, but only on the few, highly credible sequences WORD-SEQ identifies.

3.4 Adaptability of the Opportunistic Strategy

The opportunistic search strategy adapts automatically to changing conditions of uncertainty in the problem-solving process by changing the breadth of search. The basic mechanism for this is the interaction between the KS-assigned credibility ratings on hypotheses and scheduler-assigned priorities of pending KS activations. When hypotheses have been rated approximately equal, KS activations for their extension are usually scheduled together. Thus where there is ambiguity among competing hypotheses, the scheduler automatically searches with more breadth. This delays the choice among competing hypotheses until further information is brought to bear.

This adaptiveness works for changing conditions of uncertainty, whether it arises from the data or from the knowledge. The data-caused changes are evidenced by large variations in the numbers of competing hypotheses considered at various locations in an utterance, and by the large variance in the processing time needed for recognizing utterances. The results of changing conditions of knowledge constraint can be seen in Figure 12, which shows the results of one experiment varying vocabulary sizes and grammatical constraints.[14]

3.5 Performance Comparisons

It is extremely difficult to compare the reported performances of existing speech-understanding systems. Most have operated in different task environments and hence can apply different amounts of constraint

[14] Note that Figure 12 shows imperfect correlation between fanout and performance; compare, for example, X05 and SF. Fanout is an approximate measure of language complexity that reflects the average uncertainty between adjacent words. While X05 has a large fanout, it may be a simpler language to interpret than SF because most of the fanout is restricted to a few loci in the language, as opposed to the lower but more uniform uncertainty of SF.

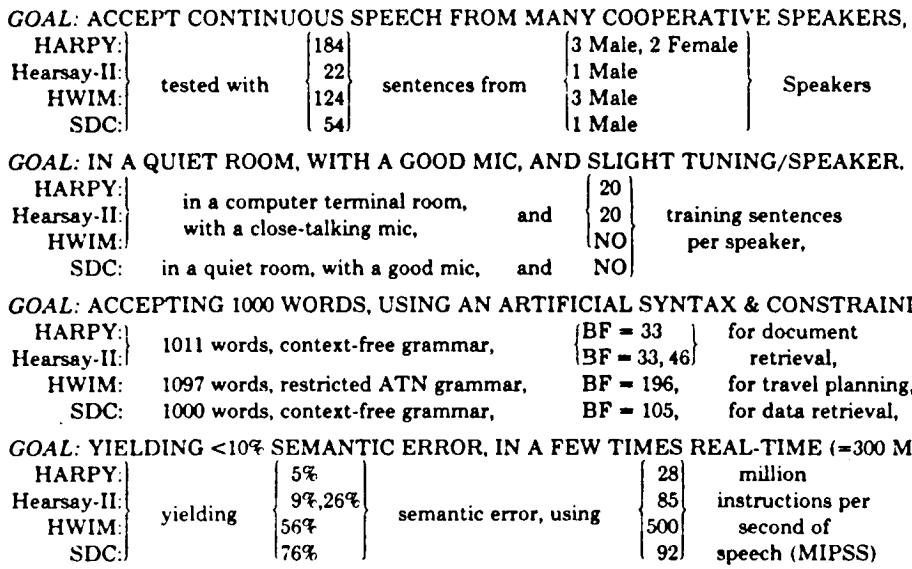


FIGURE 13. Goals and performance for final (1976) DARPA systems. [After Lea79.]

from the task language to help the problem solving. Although some progress has been made [GOOD76, SOND78, BAHL78], there is no agreed-upon method for calibrating these differences. Also, the various systems use different speakers and recording conditions. And finally, none of the systems has reached full maturity; the amount that might be gained by further debugging and tuning is unknown, but often clearly substantial.

LEA79 contains an extensive description of the systems developed in the DARPA speech-understanding project and includes the best existing performance comparisons and evaluations. Figures 13 and 14, reproduced here from that report, show some comparison of the performances of Hearsay-II, HARPY, HWIM, and the SDC system [BERN76].[15]

The Hearsay-II and HARPY results are directly comparable, the two systems having been tested on the same tasks using the same test data. HARPY's performance here dominates Hearsay-II's in both accuracy and computation speed. And, in fact, HARPY was the only system clearly to meet and exceed the DARPA specifications (see Figure 6). It is difficult to determine the exact reasons for HARPY's higher accuracy, but we feel it is caused primarily by a combination of three factors:

(1) Because of its highly compiled efficiency, HARPY can afford to search a relatively large part of the search space. In particular, it can continue pursuing partial solutions even if they contain several low-rated segments (and its pruning threshold is explicitly set to ensure this). Thus HARPY is less prone to catastrophic errors, that is, pruning away the correct path. Hearsay-II, on the other hand, cannot afford to delay pruning decisions as long and thus is more likely to make such errors.
(2) Some knowledge sources are weaker in Hearsay-II than in HARPY. In particular, Hearsay-II's JUNCT KS has only a weak model of word juncture phenomena as compared with the more

[15] Performance of the SRI system is not included because that system was run only with a simulated bottom-end. Also, there are slight differences between the Hearsay-II performance shown in Figure 13 and that of Section 3.1; the former shows results from the official end of the DARPA project in September 1976, while the latter reflects some slight improvements made in the subsequent three months.

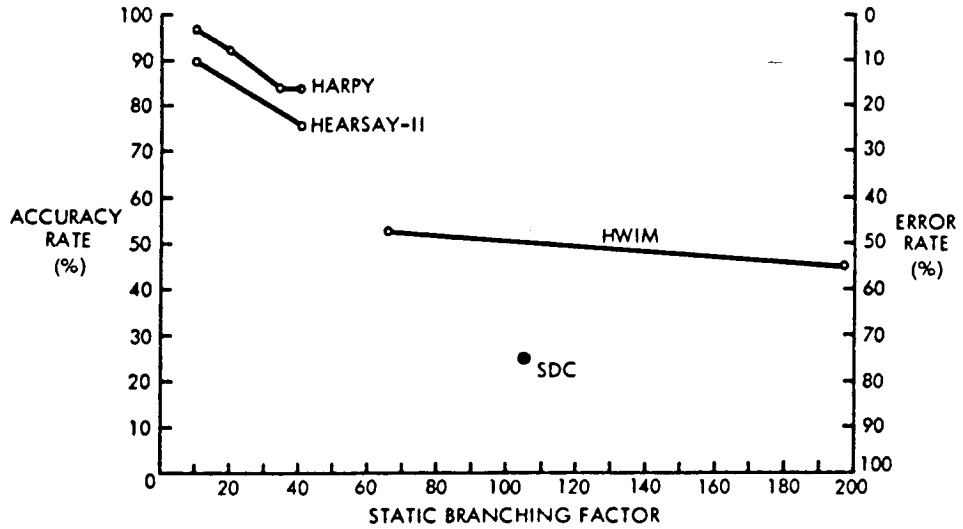


FIGURE 14 Effects of static branching factor on recognition error rate. [After LEA79.]

comprehensive and sophisticated juncture rules in HARPY. This disparity is an accident of the systems' development histories; there is no major conceptual reason why HARPY's juncture rules could not be employed by Hearsay-II.

(3) HARPY was debugged and tuned much more extensively than Hearsay-II (or any of the other DARPA SUSs, for that matter).This was facilitated by the lower processing costs for running tests. It was also helped by fixing the HARPY structure at an earlier point; Hearsay-II's KS configuration underwent a massive modification very late in the DARPA effort, as did HWIM's.

It seems clear that for a performance system in a task with a highly constrained and simply structured language, the HARPY structure is an excellent one. However, as we move to tasks that require more complex kinds of knowledge to constrain the search, we expect conceptual difficulties incorporating those kinds of knowledge into HARPY's simple integrated network representation.

4. CONCLUSIONS

Hearsay-II represents a new approach to problem solving that will prove useful in many domains other than speech. Thus far, however, we have focused on the virtues, and limitations, of Hearsay-II as a solution to the speech-understanding problem per se. In this section we consider what Hearsay-II suggests about problem-solving systems in general. To do so, we identify aspects of the Hearsay-II organization that facilitate development of "expert systems." Before concluding, we point out some apparent deficiencies of the current system that suggest avenues of further research. A more detailed discussion of these issues can be found in LESS77b.

4.1 Problem-Solving Systems

The designer of a knowledge-based problem-solving system faces several typical questions, many of which motivate the design principles evolved by Hearsay-II. The designer must first represent and structure the problem in a way that permits decomposition. A general heuristic for solving complex problems is to "divide and conquer" them. This requires methods to factor subproblems and to combine their eventual solutions. Hearsay-II, for example, divides the understanding problem in two ways: It breaks the total interpretation into separable hypotheses, and it modularizes different types of knowledge that can op-

erate independently and cooperatively. This latter attribute helps the designer address the second basic question, "How can I acquire and implement relevant knowledge?" Because knowledge sources operate solely by detecting and modifying hypotheses on the blackboard, we can develop and implement each independently. This allows us to "divide and conquer" the knowledge acquisition problem.

Two other design questions concern the *description* and *use* of knowledge. First, we must decide how to break knowledge into executable units. Second, we must develop strategies for applying knowledge selectively and efficiently. Choices for these design issues should attempt to exploit sources of structure and constraint intrinsic to the problem domain and knowledge available about it. In the current context this means that a speech-understanding system should exploit many alternative types of speech knowledge to reduce uncertainty inherent in the signal. Moreover, the different types of knowledge should apply, ideally, in a best-first manner. That is, the most credible hypotheses should stimulate searches for the most likely adjoining hypotheses first. To this end, the Hearsay-II focusing scheduler considers the quality of hypotheses and potential predictions in each temporal interval and then selectively executes only the most marginally productive KS actions. Accomplishing this type of control required several new sorts of mechanisms. These included explicit interlinked hypothesis representations, declarative descriptions of KS stimulus and response frames, a dynamic problem state description, and a prioritized schedule of pending KS instantiations.

4.2 Specific Advantages of Hearsay-II as a Problem-Solving System

This paper has covered an extensive set of issues and details. From these we believe the reader should have gained an appreciation of Hearsay-II's principal benefits, summarized briefly as follows.

Multiple Sources of Knowledge

Hearsay-II provides a framework for diverse types of knowledge to cooperate in solving a problem. This capability especially helps in situations characterized by incomplete or uncertain information. Uncertainty can arise from any of a number of causes, including noisy data, apparent ambiguities, and imperfect or incomplete knowledge. Each of these departures from the certainty of perfect information leads to uncertainty about both what the problem solver should believe and what it should do next. In such situations finding a solution typically requires simultaneously combining multiple kinds of knowledge. Although each type of knowledge may rule out only a few alternative (competing) hypotheses, the combined effect of several sources can often identify the single most credible conclusion.

Multiple Levels of Abstraction

Solving problems in an intelligent manner often requires using descriptions at different levels of abstraction. After first finding an approximate or gross solution, a problem solver may work quickly toward a refined, detailed solution consistent with the rough solution. In its use of multiple levels of abstraction, Hearsay-II provides rudimentary facilities for such variable-granularity reasoning. In the speech task particularly, the different levels correspond to separable domains of reasoning. Hypotheses about word sequences must satisfy the constraints of higher level syntactic phrase-structure rules. Once these are satisfied, testing more detailed or finely tuned word juncture relations would be justified. Of course the multiple levels of abstraction also support staged decision making that proceeds from lower level hypotheses up to higher levels. Levels in such bottom-up processing support a different type of function, namely, the sharing of intermediate results, discussed separately in the following paragraph.

Shared Partial Solutions

The blackboard and hypothesis structures allow the knowledge sources to represent and share partial results. This proves especially desirable for complex problems where no a priori knowledge can reliably foretell the best sequence of necessary de-

cisions. Different attempts to solve the same problem may require solving identical subproblems. In the sp(h domain these problems correspond to comparable hypotheses (same level, type, time). Hearsay-II provides capabilities for the KSs to recognize a hypothesis of interest and to incorporate it into alternative competing hypotheses at higher levels. Subsequent changes to the partial result then propagate to all of the higher level constructs that contain it.

Independent Knowledge Sources Limited to Data-Directed Interactions

Separating the diverse sources of knowledge into independent program modules provides several benefits. Different people can create, test, and modify KSs independently. In addition to the ordinary benefits of modularity in programming, this independence allows human specialists (e.g., phoneticians, linguists) to operationalize their diverse types of knowledge without concern for the conceptual framework and detailed behavior of other possible modules. Although the programming style and epistemological nature of several KSs may vary widely, Hearsay-II provides for all of them a single uniform programming environment. This environment constrains the KSs to operate in a data-directed manner—reading hypotheses from the blackboard when situations of interest occur, processing them to draw inferences, and recording new or modified hypotheses on the blackboard for others to process further. This paradigm facilitates problem-oriented interactions while minimizing complicated and costly design interactions.

Incremental Formation of Solutions

Problem solving in Hearsay-II proceeds incrementally through the accretion and integration of partial solutions. KSs generate hypotheses based on current data and knowledge. By integrating adjacent and consistent hypotheses into larger composites, the system develops increasingly credible and comprehensive partial solutions. These in turn stimulate focused efforts that drive the overall system toward the final goal, one most credible interpretation spanning the entire interval of speech. By allowing information to accumulate in this piecemeal fashion, Hearsay-II provides a convenient framework for heuristic problem solving. Diverse heuristic methods can contribute various types of assistance in the effort to eliminate uncertainty, to recognize portions of the sequence, and to model the speaker's intentions. Because these diverse methods exist in the form of independent, cooperating KSs, each addition to the current problem solution consists simply of an update to the blackboard.

Opportunistic Problem-Solving Behavior

Whenever good algorithms do not exist for solving a problem, we must apply heuristic methods or "rules-of-thumb" to search for a solution. In problems where a large number of data exist to which a large number of alternative heuristics potentially apply, we need to choose each successive action carefully. We refer to a system's ability to exploit selectively its best data and most promising methods as "opportunistic" problem solving [NII78, HAYE79b]. Hearsay-II developed several mechanisms to support such opportunistic behavior. In particular, its focus policies and prioritized scheduling allocate computation resources first to those KSs that exploit the most credible hypotheses, promise the most significant increments to the solution, and use the most reliable and inexpensive methods. Similar needs to focus intelligently will arise in many comparably rich and complex problem domains.

Experimentation in System Development

Whenever we attempt to solve a previously unsolved problem, the need for experimentation arises. In the speech-understanding task, for example, we generated several different types of KSs and experimentally tested a variety of alternative system configurations (specific sets of KSs) [LESS77b]. A solution to the overall problem depended on both developing powerful individual KSs and organizing multiple KSs to cooperate effectively to reduce uncertainty. These requirements necessitated a trial-and-error evaluation of alternative system designs. Throughout these explora-

tions, the basic Hearsay-II structure proved robust and sufficient. Alternative configurations were constructed with relative ease by inserting or removing specific KSs. Moreover, we could test radically different high-level control concepts (e.g., depth-first versus breadth-first versus left-to-right searches) simply by changing the focus policy KS. The need for this kind of flexibility will probably arise in many future state-of-the-art problem-solving tasks. To support this flexibility, systems must be able to apply the same KSs in different orders and to schedule them according to varying selection criteria. These requirements directly motivate KS data-directed independence, as well as autonomous scheduling KSs that can evaluate the probable effects of potential KS actions. Because it supports these needs, Hearsay-II provides an excellent environment for experimental research and development in speech and other complex tasks.

4.3 Disadvantages of the Hearsay-II Approach

We can identify two different but related weaknesses of the Hearsay-II approach to problem solving. One weakness derives from the system's generality, and the other concerns its computational efficiency. Each of these is considered briefly in turn.

Generality Impedes Specialization and Limits Power

The Hearsay-II approach suggests a very general problem-solving paradigm. Every inference process reads data from the blackboard and places a new hypothesis also on the blackboard. Thus blackboard accesses mediate each decision step. While this proved desirable for structuring communications between different KSs, it proved undesirable for most intermediate decision tasks arising within a single KS. Most KSs employed private, stylized internal data structures different from the single uniform blackboard links and hypotheses. For example, the word recognizer used specialized sequential networks, whereas the word sequence recognizer exploited a large bit-matrix of word adjacencies. Each KS also stored intermediate results, useful for its own internal searches, in appropriately distinctive data structures. Attempts to coerce these specialized activities into the general blackboard-mediated style of Hearsay-II either failed completely or caused intolerable performance degradation [LESS77b].

Interpretive Versus Compiled Knowledge

Hearsay-II uses knowledge interpretively. That is, it actively evaluates alternative actions, chooses the best for the current situation, and then applies the procedure associated with the most promising KS instantiation. Such deliberation takes time and requires many fairly sophisticated mechanisms; its expense can be justified whenever an adequate, explicit algorithm does not exist for the same task. Whenever such an algorithm emerges, equal or greater performance and efficiency may be obtained by compiling the algorithm and executing it directly. For example, recognizing restricted vocabulary and grammatical spoken sentences from limited syntax can now be accomplished faster by techniques other than those in Hearsay-II. As described in Section 2.3, by compiling all possible interlevel substitutions (sentence to phrase to word to phone to segment) into one enormous finite-state Markov network, the HARPY system uses a modified dynamic programming search to find the one network path that most closely approximates the segmented speech signal. This type of systematic, compiled, and broad search becomes increasingly desirable as problem-solving knowledge improves. Put another way, once a satisfactory specific method for solving any problem is found, the related procedure can be "algorithmetized," compiled, and applied repetitively. In such a case the flexibility of a system like Hearsay-II may no longer be needed.

4.4 Other Applications of the Hearsay-II Framework

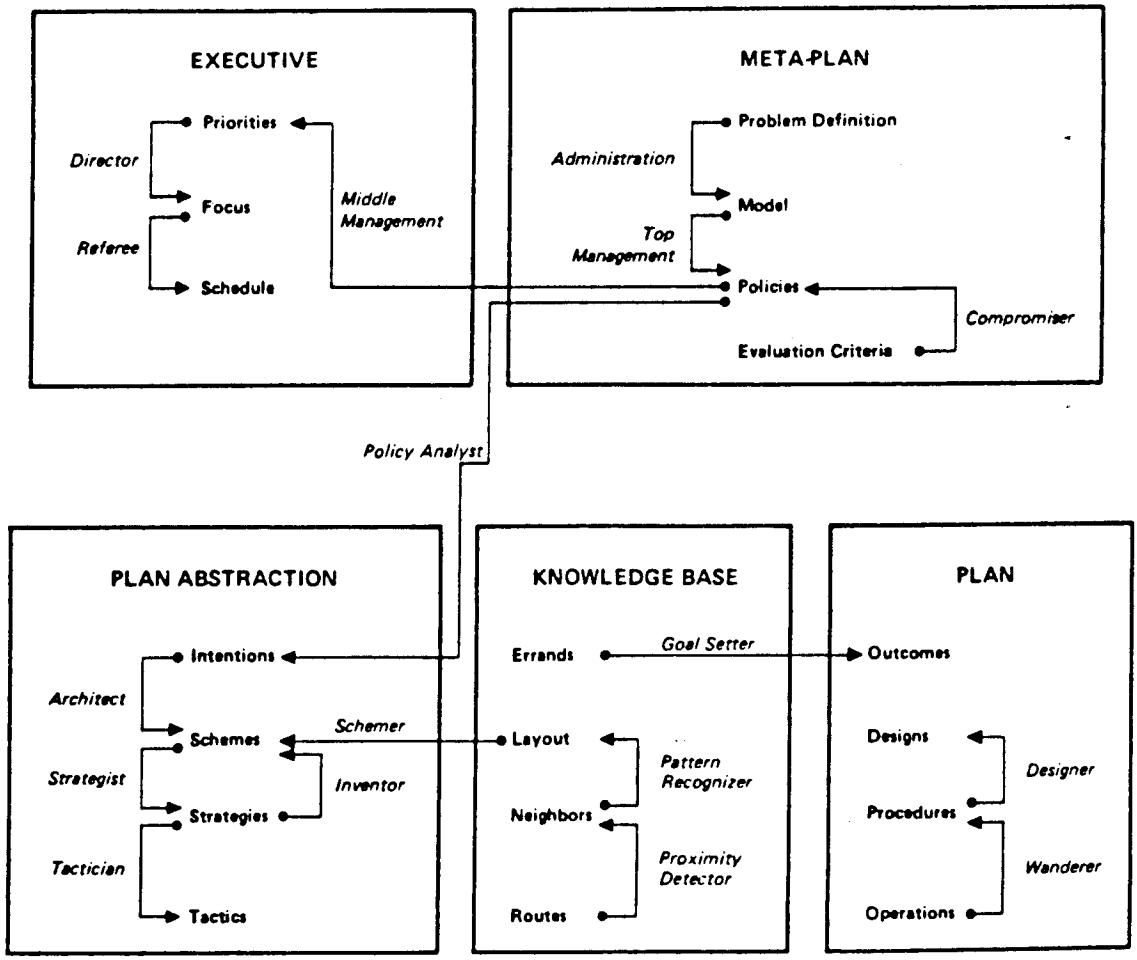
Both the advantages and disadvantages of Hearsay-II have stimulated additional research. Several researchers have applied the general framework to problems outside the speech domain, and others have begun to develop successors to the Hearsay-II sys-

tem. We will briefly discuss one of these new applications and then mention the other types of activities underway.

Although the Hearsay-II framework developed around an understanding task, B. and F. Hayes-Roth et al. have extended many of its principal features to develop a model of planning [HAYE79b, HAYE79c]. While understanding tasks require "interpretive" or "analytic" processes, planning belongs to a complementary set of "generative" or "synthetic" activities. The principal features of the Hearsay-II system which make it attractive as a problem-solving model for speech understanding also suggest it as a model of planning.

The planning application shares all the principal features of the Hearsay-II system summarized in Section 4.2, but, as Figure 15 suggests, the planning model differs from the Hearsay-II framework in several ways. In particular, the designers found it convenient to distinguish five separate blackboard "planes," reflecting five qualitatively different sorts of decisions. The Plan plane corresponds most closely to Hearsay-II's single blackboard, holding the decisions that combine to form a solution to the planning problem, i.e., what low-level operations can be aggregated to achieve the high-level outcomes of the plan. These kinds of decisions in generative tasks can be thought of as the dual of the successively higher level, more aggregated hypotheses constituting the blackboard for interpretation tasks. In the speech task, corresponding hypotheses ex-

FIGURE 15 The planning blackboard and the actions of illustrative knowledge sources. [From HAYE79b.]



press how low-level segments and phones can be aggregated to form the high-level phrases and sentences intended by the speaker. The other four planes of the planning blackboard hold intermediate decisions that enter into the planning process in various ways. For example, based on the Hearsay-II experience with selective attention strategies, resource allocation strategies were formalized and associated explicitly with an Executive plane.

Although the planning model is the only current application of the Hearsay-II framework to generative tasks, several interesting applications that transfer the approach to other interpretation problems have been made. Rumelhart [RUME76] has proposed to apply the Hearsay-II framework to model human reading behavior. In this application only one blackboard plane is used, the levels closely approximate those used in the speech-understanding task, and many additional KSs are introduced to represent how varying amounts of linguistic and semantic knowledge affect reading skills. Engelmore [ENGE77] and Nii and Feigenbaum [NII78] describe other signal-processing applications, namely, protein crystallography and acoustic signal understanding. These applications employ multiple levels and planes appropriate to their specific domains. Soloway [SOLO77] has used the framework in a learning system that develops multilevel models of observed game behaviors. Hanson and Riseman [HANS78] and Levine [LEVI78] have developed systems that mirror the Hearsay-II speech-understanding components in the image-understanding task. Arbib [ARBI79] proposes Hearsay-II-based multilevel, incremental problem-solving structures as a basis for neuroscience models, and Norman states that Hearsay-II has been a source of ideas for theoretical psychology and that it "fulfills [his]. . .intuitions about the form of a general cognitive processing structure" [NORM80, p. 383]. Finally, Mann [MANN79] has adapted the Hearsay-II structure to the task of interpreting human-machine communication dialogues.

Several researchers have focused efforts on generalizing, refining, or systematizing aspects of the Hearsay-II architecture for wider application. As previously mentioned, B. and F. Hayes-Roth have formalized some aspects of meta-planning and executive control and have treated this type of problem solving within one uniform framework. Nii [NII79] has developed a system that assists a programmer in developing a new special-purpose variant of a Hearsay-II system suitable for some particular new task. Balzer and others [BALZ80] have implemented a more formalized, domain-independent version of Hearsay-II and are applying it to an automatic-programming-like task. This system uses one blackboard for interpretation and another for scheduling decisions, in a manner akin to that proposed for the Executive decisions in the Hayes-Roth planning system. In a similar way, Stefik uses three distinct planes to record the plan, meta-plan, and executive decisions arising in a system that incrementally plans genetic experiments [STEF80].

Lesser and Erman have used Hearsay-II as a central component in a model for interpretation tasks in which the problem solving is accomplished cooperatively by distributed processors, each with only a limited view of the problem and with narrow-bandwidth intercommunication; LESS79 describes the model and some validating experiments using the Hearsay-II speech-understanding system. Hearsay-II has also influenced some attempts at developing general techniques for formal descriptions of complex systems [FOX79a, FOX79b, LESS80].

We predict that in the future the Hearsay-II paradigm will be chosen increasingly as a model of heuristic, knowledge-based reasoning. Improved compilation techniques and increased computing power will further enhance its performance. In the final analysis, however, Hearsay-II will be remembered as the first general framework for combining multiple sources of knowledge, at different levels of abstraction, into a coordinated and opportunistic problem-solving system. Such systems seem certain to play a significant role in the development of artificial intelligence.

APPENDIX. SYSTEM DEVELOPMENT

On the basis of our experience with the Hearsay-I system [REDD73a, REDD73b], at

the beginning of the Hearsay-II effort in 1973 we expected to require and evolve types of knowledge and interaction patterns whose details could not be anticipated. Because of this, the development of the system was marked by much experimentation and redesign. This uncertainty characterizes the development of knowledge-based systems. Instead of designing a specific speech-understanding system, we considered Hearsay-II as a model for a class of systems and a framework within which specific configurations of that general model could be constructed and studied [LESS75, ERMA75].

On the basis of this approach a high-level programming system was designed to provide an environment for programming knowledge sources, configuring groups of them into systems, and executing them. Because KSs interact via the blackboard (triggering on patterns, accessing hypotheses, and making modifications) and the blackboard is uniformly structured, KS interactions are also uniform. Thus one set of facilities can serve all KSs. Facilities are provided for

- defining levels on the blackboard,
- configuring groups of KSs into executable systems,
- accessing and modifying hypotheses on the blackboard,
- activating and scheduling KSs,
- debugging and analyzing the performance of KSs.

These facilities collectively form the Hearsay-II "kernel." One can think of the Hearsay-II kernel as a high-level system for programming speech-understanding systems of a type conforming to the underlying Hearsay-II model.

Hearsay-II is implemented in the SAIL programming system [REIS76], an Algol-60 dialect with a sophisticated compile-time macro facility as well as a large number of data structures (including lists and sets) and control modes which are implemented fairly efficiently. The Hearsay-II kernel provides a high-level environment for KSs at compile-time by extending SAIL's data types and syntax through declarations of procedure calls, global variables, and macros. This extended SAIL provides an explicit structure for specifying a KS and its interaction with other KSs (through the blackboard). The high-level environment also provides mechanisms for KSs to specify (usually in nonprocedural ways) information used by the kernel when configuring a system, scheduling KS activity, and controlling researcher interaction with the system.

The knowledge in a KS is represented using SAIL data structures and code, in whatever stylized form the KS developer chooses. The kernel environment provides the facilities for structuring the interface between this knowledge and other KSs, via the blackboard. For example, the syntax KS contains a grammar for the specialized task language to be recognized; this grammar is coded in a compact network form. The KS also contains procedures for searching this network, for example, to parse a sequence of words. The kernel provides facilities (1) for triggering this KS when new word hypotheses appear on the blackboard, (2) for the KS to read those word hypotheses (in order to find the sequence of words to parse), and (3) for the KS to create new hypotheses on the blackboard, indicating the structure of the parse.

Active development of Hearsay-II extended for three years. About 40 KSs were developed, each a one- or two-person effort lasting from two months to three years. The KSs range from about 5 to 100 pages of source code (with 30 pages typical), and each KS has up to about 50 kbytes of information in its local database.

The kernel is about 300 pages of code, roughly one-third of which is the declarations and macros that create the extended environment for KSs. The remainder of the code implements the architecture: primarily activation and scheduling of KSs, maintenance of the blackboard, and a variety of other standard utilities. During the three years of active development, an average of about two full-time-equivalent research programmers were responsible for the implementation, modification, and maintenance of the kernel. Included during this period were a half-dozen major reimplementations and scores of minor ones; these changes usually were specializations or selective optimizations, designed as experience with the system led to a better understanding of the usage of the various con-

structs. During this same period about eight full-time-equivalent researchers were using the system to develop KSs.

Implementation of the first version of the kernel began in the autumn of 1973, and was completed by two people in four months. The first major KS configuration, though incomplete, was running in early 1975. The first complete configuration, "C1," ran in January 1976. This configuration had very poor performance, with more than 90 percent sentence errors over a 250-word vocabulary. Experience with this configuration led to a substantially different KS configuration, "C2," completed in September 1976. C2 is the configuration described in this paper.

Implementing a general framework has a potential disadvantage: the start-up cost is relatively high. However, if the framework is suitable, it can be used to explore different configurations within the model more easily than if each configuration were built in an ad hoc manner. Additionally, a natural result of the continued use of any high-level system is its improvement in terms of enhanced facilities, increased stability, reliability, and efficiency, and greater familiarity on the part of the researchers using it.

Hearsay-II has been successful in this respect; we believe that the total cost of creating the high-level system and using it to develop KS configurations C1 and C2 (and intermediate configurations) was less than it would have been to generate them in an ad hoc manner. It should be stressed that the construction of even one configuration is itself an experimental and evolving process. The high-level programming system provides a framework, both conceptual and physical, for developing a configuration in an incremental fashion. The speed with which C2 was developed is some indication of the advantage of this system-design approach. A more detailed description of the development philosophy and tools can be found in ERMA78, and a discussion of the relationships between the C1 and C2 configurations can be found in LESS77b.

ACKNOWLEDGMENTS

The success of the Hearsay-II project depended on many persons, especially the following members of the Carnegie-Mellon University Computer Science Department "Speech Group": Christina Adam, Mark Birnbaum, Robert Cronk, Richard Fennell, Mark Fox, Gregory Gill, Henry Goldberg, Gary Goodman, Bruce Lowerre, Paul Masulis, David McKeown, Jack Mostow, Linda Shockey, Richard Smith, and Richard Suslick. Daniel Corkill, David Taylor, and the reviewers made helpful comments on early drafts of this paper.

Figure 1 is adapted from A. Newell, "A tutorial on speech understanding systems," in *Speech recognition: Invited papers of the IEEE symposium*, D. R. Reddy, Ed., Academic Press, New York, 1975. Figure 6 is adapted from M. F. Medress et al., "Speech understanding systems, Report of a steering committee," *Artif. Intell.* **9** (1978). Figure 7 is reprinted from J. J. Wolf and W. A. Woods, "The HWIM speech understanding system," in *Trends in speech recognition*, W. A. Lea, Ed., © 1980, by permission of Prentice-Hall, Inc., Englewood Cliffs, N.J. Figures 8–11 are reprinted from B. T. Lowerre and R. Reddy, "The HARPY speech understanding system," in *Trends in speech recognition*, W. A. Lea, Ed., © 1980, by permission of Prentice-Hall, Inc., Englewood Cliffs, N.J. Figure 15 originally appeared in B. Hayes-Roth and F. Hayes-Roth, "A cognitive model of planning," *Cognitive science*, 1979, **3** 275–310. Ablex Publishing Corporation, Norwood, N.J.

REFERENCES

ARBI79 ARBIB, M. A., AND CAPLAN, D. "Neurolinguistics must be computational," *Behav. Brain Sci.* **2**, 3 (1979).

BAHL76 BAHL, L. R., BAKER, J. K., COHEN, P. S., DIXON, N. R., JELINEK, F., MERCER, R. L., AND SILVERMAN, H. F. "Preliminary results on the performance of a system for the automatic recognition of continuous speech," in *1976 IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Philadelphia, Apr. 1976, pp. 425–433.

BAHL78 BAHL, L. R., BAKER, J. K., COHEN, P. S., COLE, A. G., JELINEK, F., LEWIS, B. L., AND MERCER, R. L. "Automatic recognition of continuously spoken sentences from a finite state grammar," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tulsa, Okla., Apr. 1978, pp. 418–421.

BAL280 BALZER, R., ERMAN, L. D., AND WILLIAMS, C. *Hearsay-III: A domain-independent base for knowledge-based problem-solving*, Tech. Rep., USC/Information Sciences Institute, Marina del Rey, Calif., 1980. To appear.

BARN77 BARNETT, J. A., AND BERNSTEIN, M. I. *Knowledge-based systems: A tutorial*, Tech. Rep. TM-(L)-5903/000/00 (NTIS: AD/A-044-883), System Development Corp., Santa Monica, Calif., June 1977.

BERN76 BERNSTEIN, M. I. *Interactive systems research: Final report to the Director, Advanced Research Projects Agency*, Tech. Rep. TM-5243/006/00, System Development Corp., Santa Monica, Calif., Sept. 1976.

BURT76 BURTON, R. R. *Semantic grammar: An engineering technique for constructing natural language understanding systems*, Tech. Rep. BBN Rep. No. 3453, Bolt Beranek and Newman, Cambridge, Mass., 1976.

CMU77 CMU COMPUTER SCIENCE SPEECH GROUP. *Summary of the CMU five-year ARPA effort in speech understanding research*, Tech. Rep., Computer Science Dep., Carnegie-Mellon Univ., Pittsburgh, Pa., 1977.

CRON77 CRONK, R. "Word pair adjacency acceptance procedure in Hearsay-II," in CMU77, pp. 15–16.

DUDA78 DUDA, R. O., HART, P. E., NILSSON, N. J., AND SOUTHERLAND, G. L. "Semantic network representation in rule-based inference systems," in *Pattern-directed inference systems*, D. A. Waterman and F. Hayes-Roth, Eds., Academic Press, New York, 1978, pp. 203–222.

ENGE77 ENGELMORE, R. S., AND NII, H. P. *A knowledge-based system for the interpretation of protein X-ray crystallographic data*, Tech. Rep. Stan-CS-77-589, Computer Science Dep., Stanford Univ., Stanford, Calif., 1977.

ERMA75 ERMAN L. D., AND LESSER, V. R. "A multi-level organization for problem solving using many diverse cooperating sources of knowledge," in *Proc. 4th Int. Jt. Conf. Artificial Intelligence*, Tbilisi, USSR, 1975, pp. 483–490.

ERMA78 ERMAN, L. D., AND LESSER, V. R. "System engineering techniques for artificial intelligence systems," in *Computer vision systems*, A. Hanson and E. Riseman, Eds., Academic Press, New York, 1978, pp. 37–45.

ERNS69 ERNST, G., AND NEWELL, A. *GPS: A case study in generality and problem solving*, Academic Press, New York, 1969.

FEIG71 FEIGENBAUM, E. A., BUCHANAN, B. G., AND LEDERBERG, J. "On generality and problem solving: A case study using the DENDRAL program," in *Machine intelligence 6*, D. Michie, Ed., Edinburgh Univ. Press, Edinburgh, Scotland, 1971.

FEIG77 FEIGENBAUM, E. A. "The art of artificial intelligence: Themes and case studies of knowledge engineering," in *Proc. 5th Int. Jt. Conf. Artificial Intelligence*, Cambridge, Mass., 1977, pp. 1014–1029.

FOX77 FOX, M. S., AND MOSTOW, D. J. "Maximal consistent interpretations of errorful data in hierarchically modelled domains," in *Proc. 5th Int. Jt. Conf. Artificial Intelligence*, Cambridge, Mass., 1977, pp. 65–171.

FOX79a FOX, M. S. "An organizational view of distributed systems," in *Proc. Int. Conf. Systems and Cybernetics*, Denver, Colo., Oct. 1979.

FOX79b FOX, M. S. *Organization structuring: Designing large, complex software*, Tech. Rep. CMU-CS-79-115, Computer Science Dep., Carnegie-Mellon Univ., Pittsburgh, Pa., 1979.

GILL78 GILL, G., GOLDBERG, H., REDDY, R., AND YEGNANARAYANA, B. *A recursive segmentation procedure for continuous speech*, Tech. Rep. CMU-CS-78-134, Computer Science Dep., Carnegie-Mellon Univ., Pittsburgh, Pa., May 1978.

GOLD77 GOLDBERG, H., REDDY, R., AND GILL, G. "The ZAPDASH parameters, feature extraction, segmentation, and labeling for speech understanding systems," in CMU77, pp. 10–11.

GOOD76 GOODMAN, G. *Analysis of languages for man-machine voice communication*, Tech. Rep., Computer Science Dep., Carnegie-Mellon Univ., Pittsburgh, Pa., May 1976.

HANS78 HANSON, A. R., AND RISEMAN, E. M. "VISIONS: A computer system for interpreting scenes," in *Computer vision systems*, A. Hanson and E. Riseman, Eds., Academic Press, New York, 1978, pp. 303–333.

HARR74 HARRIS, L. R. "The heuristic search under conditions of error," *Artif. Intell.* 5, 3 (1974), 217–234.

HAYE75 HAYES-ROTH, F., AND MOSTOW, D. J. "An automatically compilable recognition network for structured patterns," in *Proc. 4th Int. Jt. Conf. Artificial Intelligence*, Tbilisi, USSR, 1975, pp. 246–252.

HAYE77a HAYES-ROTH, F., AND LESSER, V. R. "Focus of attention in the Hearsay-II system," in *Proc. 5th Int. Jt. Conf. Artificial Intelligence*, Cambridge, Mass., 1977, pp. 27–35.

HAYE77b HAYES-ROTH, F., ERMAN, L. D., FOX, M., AND MOSTOW, D. J. "Syntactic processing in Hearsay-II," in CMU77, pp. 16–18.

HAYE77c HAYES-ROTH, F., GILL, G., AND MOSTOW, D. J. "Discourse analysis and task performance in the Hearsay-II speech understanding system," in CMU77, pp. 24–28.

HAYE77d 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 CMU77, pp. 19–24.

HAYE78a HAYES-ROTH, F., WATERMAN, D. A., AND LENAT, D. B. "Principles of pattern-directed inference systems," in *Pattern-directed inference systems*, D. A. Waterman and F. Hayes-Roth, Eds., Academic Press, New York, 1978.

HAYE78b HAYES-ROTH, F. "The role of partial and best matches in knowledge systems," in *Pattern-directed inference systems*, D. A. Waterman and F. Hayes-Roth, Eds., Academic Press, New York, 1978.

HAYE79a HAYES-ROTH, B., AND HAYES-ROTH, F. *Cognitive processes in planning*, Tech. Rep. R-2366-ONR, The RAND Corp., Santa Monica, Calif., 1979.

HAYE79b HAYES-ROTH, B., AND HAYES-ROTH, F. "A cognitive model of planning," *Cognitive Sci.* 3 (1979), 275–310.

HAYE79c HAYES-ROTH, B., HAYES-ROTH, F., ROSENSCHEIN, S., AND CAMMARATA, S. "Modeling planning as an incremental opportunistic process," in *Proc. 6th Int. Jt.*

Conf. Artificial Intelligence, Tokyo, 1979, pp. 375–383.

HAYE80 HAYES-ROTH, F. "Syntax, semantics, and pragmatics in speech understanding," in *Trends in speech recognition*, W. A. Lea, Ed., Prentice-Hall, Englewood Cliffs, N.J., 1980.

ITAK75 ITAKURA, F. "Minimum prediction residual principle applied to speech recognition," *IEEE Trans. Acoust., Speech, Signal Proc.* **23** (1975), 67–72.

KLAT77 KLATT, D. H. "Review of the ARPA speech understanding project," *J. Acoust. Soc. Am.* **62** (Dec. 1977), 1345–1366.

LEA79 LEA, W. A., AND SHOUP, J. E. *Review of the ARPA SUR Project and survey of current technology in speech understanding*, Final Rep., Office of Naval Research Contract No. N00014-77-C-0570, Speech Communications Research Lab., Los Angeles, Calif., Jan. 1979.

LEA80 LEA, W. A., ED. *Trends in speech recognition*, Prentice-Hall, Englewood Cliffs, N.J., 1980.

LESS75 LESSER, V. R., FENNELL, R. D., ERMAN, L. D., AND REDDY, D. R. "Organization of the Hearsay-II speech understanding system," *IEEE Trans. Acous., Speech, Signal Proc.* **23** (1975), 11–23.

LESS77a LESSER, V. R., HAYES-ROTH, F., BIRNBAUM, M., AND CRONK, R. "Selection of word islands in the Hearsay-II speech understanding system," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Hartford, Conn., 1977, pp. 791–794.

LESS77b LESSER, V. R., AND ERMAN, L. D. "A retrospective view of the Hearsay-II architecture," in *Proc. 5th Int. Joint Conf. Artificial Intelligence*, Cambridge, Mass., 1977, pp. 790–800.

LESS79 LESSER, V. R., AND ERMAN, L. D. "An experiment in distributed interpretation," in *1st Int. Conf. Distributed Computing Systems*, IEEE Computer Society, Huntsville, Ala., Oct. 1979, pp. 553–571.

LESS80 LESSER, V. R., PAVLIN, J., AND REED, S. *Quantifying and simulating the behavior of knowledge-based systems*, Tech. Rep., Dep. Computer and Information Sciences, Univ. Massachusetts, Amherst, Mass., 1980.

LEVI78 LEVINE, M. D. "A knowledge-based computer vision system," in *Computer vision systems*, A. Hanson and E. Riseman, Eds., Academic Press, New York, 1978, pp. 335–352.

LOWE76 LOWERRE, B. T. *The HARPY speech recognition system*, Ph.D. thesis, Computer Science Dep., Carnegie-Mellon Univ., Pittsburgh, Pa., 1976.

LOWE80 LOWERRE, B. T., AND REDDY, R. "The HARPY speech understanding system," in *Trends in speech recognition*, W. A. Lea, Ed., Prentice-Hall, Englewood Cliffs, N.J., 1980, Chap. 15.

LOWR80 LOWRANCE, J. *Dependence-graph models of evidential support*, Ph.D. thesis, Dep. Computer and Information Sciences, Univ. Massachusetts, 1980 (forthcoming).

MANN79 MANN, W. C. "Design for dialogue comprehension," in *17th Ann. Meeting Assoc. Computational Linguistics*, La Jolla, Calif., Aug. 1979.

MCKE77 MCKEOWN, D. M. "Word verification in the Hearsay-II speech understanding system," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Hartford, Conn., 1977, pp. 795–798.

MEDR78 MEDRESS, M. F., COOPER, F. S., FORGIE, J. W., GREEN, C. C., KLATT, D. H., O'MALLEY, M. H., NEUBURG, E. P., NEWELL, A., REDDY, D. R., RITEA, B., SHOUP-HUMMEL, J. E., WALKER, D. E., AND WOODS, W. A. "Speech understanding systems: Report of a steering committee," *Artif. Intell.* **9** (1978), 307–316.

MOST77 MOSTOW, D. J. "A halting condition and related pruning heuristic for combinatorial search," in CMU77, pp. 158–166.

NEWE69 NEWELL, A. "Heuristic programming: Ill-structured problems," in *Progress in operations research 3*, J. Aronofsky, Ed., Wiley, New York, 1969, pp. 360–414.

NEWE73 NEWELL, A., BARNETT, J., FORGIE, J., GREEN, C., KLATT, D., LICKLIDER, J. C. R., MUNSON, J., REDDY, R., AND WOODS, W. *Speech understanding systems: Final report of a study group*, North-Holland, Amsterdam, 1973.

NEWE75 NEWELL, A., "A tutorial on speech understanding systems," in *Speech recognition: Invited papers of the IEEE symposium*, D. R. Reddy, Ed., Academic Press, New York, 1975, pp. 3–54.

NEWE77 NEWELL, A., MCDERMOTT, J., AND FORGIE, C. *Artificial intelligence: A self-paced introductory course*, Computer Science Dep., Carnegie-Mellon Univ., Pittsburgh, Pa., 1977.

NEWE80 NEWELL, A. "HARPY, production systems and human cognition," in *Perception and production of fluent speech*, R. Cole, Ed., L. Erlbaum, Hillsdale, N.J., 1980, Chap. 11.

NII78 NII, H. P., AND FEIGENBAUM, E. A. "Rule-based understanding of signals," in *Pattern-directed inference systems*, D. A. Waterman and F. Hayes-Roth, Eds., Academic Press, New York, 1978.

NII79 NII, H. P., AND AIELLO, N. "AGE (Attempt to Generalize): A knowledge-based program for building knowledge-based programs," in *Proc. 6th Int. Jt. Conf. Artificial Intelligence*, Tokyo, Feb. 1979, pp. 645–655.

NILS71 NILSSON, N. *Problem-solving methods in artificial intelligence*, McGraw-Hill, New York, 1971.

NORM80 NORMAN, D. A. "Copycat science or does the mind really work by table look-up?," in *Perception and production of fluent speech*, R. Cole, Ed., L. Erlbaum, Hillsdale, N.J., 1980, Chap. 12.

POHL70 POHL, I. "First results on the effects of error in heuristic search," in *Machine in-*

telligence 5, B. Meltzer and D. Michie, Eds., Edinburgh Univ. Press, Edinburgh, Scotland, 1970.

POHL77 POHL, I. "Practical and theoretical considerations in heuristic search algorithms," in *Machine intelligence 8*, E. Elcock and D. Michie, Eds., Ellis Horwood, Chichester, England, 1977.

REDD73a REDDY, D. R., ERMAN, L. D., AND NEELY, R. B. "A model and a system for machine recognition of speech," *IEEE Trans. Audio and Electroacoustics* **AU-21** (1973), 229-238.

REDD73b REDDY, D. R., ERMAN, L. D., FENNELL, R. D., AND NEELY, R. B. "The Hearsay speech understanding system: An example of the recognition process," in *Proc. 3rd Int. Jt. Conf. Artificial Intelligence*, Stanford, Calif., 1973, pp. 185-193.

REDD75 REDDY, D. R., ED. *Speech recognition: Invited papers presented at the 1974 IEEE Symposium*, Academic Press, New York, 1975.

REDD76 REDDY, D. R. "Speech recognition by machine: A review," *Proc. IEEE* **64** (Apr. 1976), 501-531.

REIS76 REISER, J. F. *SAIL*, Tech. Rep. AIM-289, AI Lab., Stanford Univ., Stanford, Calif., 1976.

RUBI78 RUBIN, S. *The ARGOS image understanding system*, Ph.D. thesis, Computer Science Dep., Carnegie-Mellon Univ., Pittsburgh, Pa., 1978.

RUME76 RUMELHART, D. E. *Toward an interactive model of reading*, Tech. Rep. 56, Center for Human Information Processing, Univ. California, San Diego, 1976.

SACE74 SACERDOTI, E. E. "Planning in a hierarchy of abstraction spaces," *Artif. Intell.* **5** (1974), 115-135.

SHOR75 SHORTLIFFE, E. H., AND BUCHANAN, B. G. "A model of inexact reasoning in medicine," *Math. Bio. Sci.* **23** (1975).

SHOR76 SHORTLIFFE, E. *Computer-based medical consultation: MYCIN*, Elsevier, New York, 1976.

SMIT76 SMITH, A. R. "Word hypothesization in the Hearsay-II speech system," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Philadelphia, Pa., 1976, pp. 549-552.

SMIT77 SMITH, A. R. *Word hypothesization for large-vocabulary speech understanding systems*, Ph.D. thesis, Computer Science Dep., Carnegie-Mellon Univ., Pittsburgh, Pa., 1977.

SMIT81 SMITH, A. R., AND ERMAN, L. D. "NOAH: A bottom-up word hypothesizer for large-vocabulary speech understanding systems," *IEEE Trans. Pattern Anal. Mach. Intell.* (1981), to be published.

SOLO77 SOLOWAY, E. M., AND RISEMAN, E. M. "Levels of pattern description in learning," in *Proc. 5th Int. J. Conf. Artificial Intelligence*, Cambridge, Mass., 1977, pp. 801-811.

SOND78 SONDHI, M. M., AND LEVINSON, S. E. "Computing relative redundancy to measure grammatical constraint in speech recognition tasks," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tulsa, Okla., Apr. 1978.

STEF80 STEFIK, M. *Planning with constraints*, Ph.D. thesis, Computer Science Dep., Stanford Univ., Stanford, Calif., Jan. 1980.

WALK78 WALKER, D. E., ED. *Understanding spoken language*, Elsevier North-Holland, New York, 1978.

WALK80 WALKER, D. E. "SRI research on speech understanding," in *Trends in speech recognition*, W. A. Lea, Ed., Prentice-Hall, Englewood Cliffs, N.J., 1980, Chap. 13.

WOLF80 WOLF, J. J., AND WOODS, W. A. "The HWIM speech understanding system," in *Trends in speech recognition*, W. A. Lea, Ed., Prentice-Hall, Englewood Cliffs, N.J., 1980, Chap. 14.

WOOD70 WOODS, W. A. "Transition network grammars for natural language analysis," *Commun. ACM* **13**, 10 (Oct. 1970), 591-606.

WOOD73 WOODS, W. A., AND MAKHOUL, J. "Mechanical inference problems in continuous speech understanding," in *Proc. 3rd Int. Jt. Conf. Artificial Intelligence*, Stanford, Calif., 1973, pp. 73-91, also *Artif. Intell.* **5**, 1 (Spring 1974), 73-91.

WOOD76 WOODS, W., BATES, M., BROWN, G., BRUCE, B., COOK, C., KLOVSTAD, J., MAKHOUL, J., NASH-WEBBER, B., SCHWARTZ, R., WOLF, J., AND ZUE, V. *Speech understanding systems: Final technical progress report*, Tech. Rep. 3438, Bolt Beranek and Newman, Cambridge, Mass., Dec. 1976 (in five volumes).

WOOD77 WOODS, W. A. "Shortfall and density scoring strategies for speech understanding control," in *Proc. 5th Int. Jt. Conf. Artificial Intelligence*, Cambridge, Mass., 1977, pp. 13-26.