NOTICE WARNING CONCERNING COPYRIGHT RESTRICTIONS:
The copyright law of the United States (title 17, U.S. Code) governs the making of photocopies or other reproductions of copyrighted material. Any copying of this document without permission of its author may be prohibited by law.

SYNTAX AND SEMANTICS
IN A DISTRIBUTED SPEECH UNDERSTANDING SYSTEM

Frederick Hayes-Roth and David J. Mostow
Computer Science Department[1]
Carnegie-Mellon University
Pittsburgh, Pa. 15213

ABSTRACT

The Hearsay II speech understanding system being developed at Carnegie-Mellon University has an independent knowledge source module for each type of speech knowledge. Modules communicate by reading, writing, and modifying hypotheses about various constituents of the spoken utterance in a global data structure. The syntax and semantics module uses rules (productions) of four types: (1) recognition rules for generating a phrase hypothesis when its needed constituents have already been hypothesized; (2) prediction rules for inferring the likely presence of a word or phrase from previously recognized portions of the utterance; (3) respelling rules for hypothesizing the constituents of a predicted phrase; and (4) postdiction rules for supporting an existing hypothesis on the basis of additional confirming evidence. The rules are automatically generated from a declarative (i.e., non-procedural) description of the grammar and semantics, and are embedded in a parallel recognition network for efficient retrieval of applicable rules. The current grammar uses a 450-word vocabulary and accepts simple English queries for an information retrieval system.

INTRODUCTION: THE PROBLEM

The fundamental problem facing the syntax and semantics component of a speech understanding system is uncertainty. The system is uncertain about a variety of questions, including: whether a given word is really uttered by the speaker; when a recognized word begins and ends; whether a particular interval of the utterance contains a silence, a filled pause ("er," "um," "uh"), an informationless interjection ("y'know," "I mean"), or an information-bearing word or phrase; whether a recognized word or phrase is used in a particular sense; etc. Any decisions made on the basis of such uncertain information are potentially incorrect and must therefore be reversible. The classical method of reversing decisions is backtracking. Backtracking and best-first evaluation of alternative parses are the primary strategies employed by the Hearsay I speech understanding system (Reddy, et al., 1973a, 1973b).

In Hearsay II (Lesser, et al., 1975) multiple alternatives are represented explicitly in a global data structure ("blackboard") and considered in parallel rather than one at a time as in Hearsay I. Processing is driven by independent data-directed knowledge source modules (KSs) which create, examine, and revise hypotheses, stored on the blackboard, about the utterance. One dimension of the blackboard is level of representation: an interval of speech may be simultaneously represented at the acoustic, phonetic, phonemic, syllabic, word, phrasal, and conceptual levels. The KSs translate from one level to another with the ultimate objective of representing the utterance at the conceptual level, i.e., understanding it. Hearsay II is a distributed logic system in that control of processing is distributed heterarchically among the KSs rather than organized hierarchically. Each KS is responsible for deciding when it has useful information to contribute to the analysis of the input.

The syntax and semantics KS in Hearsay II is called SASS, and deals with hypotheses representing words and phrases perceived or expected in the utterance. From SASS's viewpoint, the blackboard can be viewed as a chart of hypothesized words as in Figure 1, which represents the word hypotheses generated by lower-level KSs in response to the utterance "Tell me about beef." In the figure, time goes from left to right and the vertical dimension represents hypothesis credibility on a scale from -100 to 100, as estimated by other KSs. SASS's problem is to find the most plausible sequence of temporally adjacent words. Plausibility is defined by the credibility of the individual word hypotheses and the grammaticality and meaningfulness of the sequence. The concept of temporal adjacency is generalized to tolerate fuzzy word boundaries, overlap between successive words, silences in the middle of word sequences, and unintelligible intervals. Since some of the uttered words may not have been hypothesized, SASS must be able to expand the solution space by inferring the likely presence of a missing word on the basis of existing word hypotheses. Such inferences are relatively weak since several predictions may be plausible in a given context. In the example of Figure 1, SASS hypothesizes the missing word "tell" in the interval preceding "me about beef." Since SASS is uncertain as to which word hypotheses are correct, it also makes several incorrect word predictions. Figure 2 shows the words predicted by SASS on the basis of the words shown in Figure 1. The figures do not reflect the fact that the various hypotheses are generated at different times and SASS starts generating predictions prior to completion of the word recognition process.

In order to control the potentially explosive search through this combinatorial and expanding solution space, SASS must be able to reflect the variable reliability of its inference rules and to relax its plausibility criteria dynamically so as to stimulate processing on unrecognized portions of the utterance. SASS must be able to use partial information to guide further processing in useful directions. To avoid duplicated computation, SASS must store and use partial parses, which are intermediate computations (plausible subsequences) common to many potential parses. SASS must combine these partial parses into plausible complete parses, select the best complete parse, interpret the meaning of the recognized utterance, and respond appropriately.

The problems faced by SASS -- uncertainty, combinatorial search, fuzzy pattern-matching, strong and weak inferences, and the need to exploit partial information -- are common to many large knowledge-based systems. Efficient solution of these problems appears to require a system organization in which the scheduling of inferential processes is sensitive to various cooperative and competitive relationships among the inferred hypotheses. For example, processing should be facilitated on an hypothesis supported cooperatively by multiple sources of information. Conversely, processing should be inhibited on an hypothesis which competes -- i.e., is inconsistent with -- a strongly credible hypothesis. Inhibition in an environment of uncertainty must be implemented non-deterministically, since the weaker hypothesis may in fact be correct. Non-deterministic inhibition is effected in Hearsay II by a focus of attention mechanism which allocates computational resources so as to consider the most promising hypotheses before others (Hayes-Roth & Lesser, 1976).

The approach used in SASS is relevant to pattern recognition for its fuzzy pattern-matching; to problem solving for its flexible combination of bottom-up, top-down, forward inferencing, and problem reduction mechanisms; and to information retrieval and the problem of pattern-directed function invocation for its efficient mechanism for continuously monitoring a data base for occurrences of any of a large number of relational patterns or templates.

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

OVERVIEW OF METHOD

Given a declarative (i.e., non-procedural) description of the target language which our system is to understand, we need to convert it into behavior which is adequate to understand utterances in the language efficiently and robustly. Our approach has been to automate this conversion as much as possible. Syntactic and semantic knowledge about the target language is expressed in a compact, readable grammar. A compiler converts the grammar into precondition-response productions. The productions are embedded in a recognition network to enable efficient continuous monitoring of the blackboard for stimuli matching production preconditions. In general, many productions will be invocable at any given time. Various scheduling policies serve to hasten the invocation of productions which are considered likely to generate useful (correct, relevant, and necessary) results and to inhibit or defer less promising invocations.

LINGUISTIC KNOWLEDGE

The grammar describing the target language is expressed using parameterized structural representations (PSRs), which are sets of attribute-object pairs. We use a PSR to define a class of words and phrases which can fulfill the same syntactic or semantic function in the target language. The current target language consists of simple English queries for a news retrieval program. For example, the PSR

```
    ($CLASS: $QUERY, $PNAME: "PARSED QUERY",
           ϵ: $GIMME+$WHAT,
           ϵ: TELL+$ME+$RE+$TOPICS,
           ϵ: WHAT+HAPPENED+$ANYWAY,
           ϵ: WHAT+$BE+THE+$NEWS+$RE+$TOPICS,
           ϵ: $BE+THERE+$ANY+$PIECES+$RE+$TOPICS,
           $ACTION: PASS,
           $LEVEL: 300)
```

defines the class "$QUERY" of possible queries in terms of its alternative syntactic realizations. The attribute "ϵ" denotes membership in the class. Each member of the class is a sequence template whose constituents, separated by "+", are words or phrases. Phrasal constituents are prefixed by "$" and defined in turn by other PSRs. Additional attributes of the class are defined by other components of the PSR. "$ACTION: PASS" means that SASS's response upon recognizing an instance of any of the five templates in the class should be to treat it as an instance of $QUERY. The $LEVEL attribute estimates the relative completeness of the partial parse underlying the hypothesized phrase. The PSR

```
    ($CLASS: $TOPICS,
           ϵ: $PLACE,
           ϵ: $FOOD,
           ϵ: $TECHNOLOGY,
           ϵ: $SCIENCE,
           ϵ: $GOVERNMENT,
           ϵ: $POLITICS,
           ϵ: $PEOPLE,
           ϵ: $TOPICS+$CONJUNCTION+$TOPICS,
           $ACTION: PASS, $LEVEL: 40)
```

defines the class of possible topics in the news in terms of its semantic subclasses. The grammar for the current 450-word target language consists of 113 PSRs.

TYPES OF BEHAVIOR RULES

SASS has a repertoire of strong and weak methods, represented by different types of behavior rules used in understanding.

A recognition rule generates a phrase hypothesis in response to sufficiently credible hypotheses for the phrase's constituents. SASS considers an hypothesized constituent to be recognizable if its credibility rating, determined by other KSs, exceeds a minimum threshold for plausibility. The hypothesized constituents may also have to satisfy some structural condition such as temporal adjacency between sequential constituents of a phrase. A recognition rule represents a strong inference; its strength is the probability that the recognized constituents can be interpreted as an instance of the phrase. For example, "beef" can be interpreted as a food or as a complaint, depending on context. Recognition rules drive processing upward toward a complete parse of the utterance from plausible partial parses. Recognition behavior can be thought of as bottom-up parsing.

A prediction rule hypothesizes a word or phrase which is likely to occur in the context of a previously recognized portion of the utterance. Prediction rules drive processing outward in time from "islands of plausibility," and are necessary since not all words in a spoken utterance may be recognized bottom-up by lower-level KSs. Predictive behavior can be thought of as forward inferencing. The strength of a predictive inference is the conditional probability that the predicted constituent occurs, given that its predictive context has been recognized. This strength is inversely related to the number of constituents which can plausibly occur in the given context.

A respelling rule enumeratively hypothesizes the constituents of a predicted phrase, by subdividing an hypothesized sequence into hypotheses for its sequential constituents, or by splitting an hypothesized class into alternate hypotheses for its various members. Respelling rules drive processing downward toward the word level, so that high-level phrasal predictions can ultimately be tested word-by-word by lower-level KSs. Respelling can be thought of as top-down behavior or generation of subgoals from goals.

Finally, a postdiction rule solicits post hoc support for (i.e., serves to increase the credibility ratings of) existing hypotheses from other hypotheses in whose context they are plausible. Postdiction rules include prediction and respelling rules which are too weak to justify creation of hypotheses, but can contribute useful information when the hypotheses already exist. For example, an expectation for an instance of $TOPICS following the word "about" should not be respelled into hypotheses for all the nouns in the vocabulary, since to do so would explode the search space. However, once the word "beef" is hypothesized in the correct time interval on the basis of other knowledge, the hypothesis should receive support from the expectation for a topic word.

Postdiction rules serve three functions: they allow cooperation between inferences which support the same hypothesis on the basis of different evidence; they allow words and phrases hypothesized with initial low credibility ratings to be recognized on the basis of their contextual plausibility; and they help focus attention in productive directions by increasing the ratings of hypotheses which are contextually plausible (and thus relatively likely to be correct) so that processing on them is scheduled sooner. In the sense that postdiction responds to weakly-rated hypotheses by seeking causal antecedents (predictors) for them, postdiction can be thought of as post hoc inferencing or "twenty-twenty hindsight."

CONVERSION OF STATIC KNOWLEDGE TO BEHAVIOR RULES

Most of the information necessary for understanding the target language is implicit in the grammar which describes it. The automatic conversion of this static information into a usable procedural form is effected by a simple compiler called CVSNET, which translates the PSRs into recognition, prediction, respelling, and postdiction rules. A few rules hand-coded in explicitly procedural form are then added, for example a rule that prints a message when a sentence is recognized. The only linguistic knowledge in CVSNET itself is an elementary understanding of sequences and classes. CVSNET decomposes the sequence templates $c_1+c_2+...+c_n$ into pairs of subsequence templates. For example, from the sequence template TELL+$ME+$RE+$TOPICS, CVSNET generates the new templates $ME+$RE+$TOPICS and $RE+$TOPICS.

CVSNET then generates the appropriate rules for each template. The recognition rule for a sequence is to concatenate its hypothesized subsequences provided they are temporally adjacent and sufficiently credible. The respelling rule respells a predicted sequence into its two subsequences. Prediction rules are generated to predict the remaining constituents of the sequence when a subsequence of it has been recognized. Similarly, CVSNET generates rules for recognizing an instance of a class from an hypothesized constituent of the class and for respelling a predicted class into its constituents. CVSNET estimates the strength of each such rule as an inverse function of class size. CVSNET also generates the relevant postdiction rules. Some of the rules generated from the PSRs are shown below; rule type is indicated by the type of arrow separating stimulus and response ("→" for recognition, "=>" for prediction, "+>" for respelling, and "<=" for postdiction) and rule strength is shown in parentheses.

```
TELL & $ME → TELL+$ME  < CONCATENATE (100) (100) >

TELL & $ME <= TELL+$ME  < POSTDICT!SEQ (100) (100) >

TELL+$ME +> TELL & $ME  < RESPELL!SEQ (100) (100) >

$ME => TELL  < PREDICT!LEFT (50) >

TELL <= $ME  < POSTDICT!LEFT (50) >

TELL => $ME+$RE+$TOPICS  < PREDICT!RIGHT (100) >

$ME+$RE+$TOPICS <= TELL  < POSTDICT!RIGHT (100) >

$FOOD → $TOPICS  < PASS (100) >

$TOPICS +> $FOOD  < RESPELL!CLASS (70) >

$FOOD <= $TOPICS  < POSTDICT!ELEMENT (88) >
```

The linguistic knowledge expressed compactly in the grammar is represented highly redundantly in the generated rules. This redundancy provides the basis for robust performance in the errorful domain of speech: in regions of the utterance where strong inferences (recognition rules) are inadequate (for example, because lower-level KSs have failed to hypothesize some of the uttered words), weaker inferences must be applied in order for the utterance to be understood.

IDENTIFICATION OF INVOCABLE RULES

All of the rules described have the form [precondition($x_1,x_2,\ldots,x_n$) $\Rightarrow^f$ response($x_1,x_2,\ldots,x_n$)], signifying that a specified response can be inferred with strength f from the objects x_1, x_2, ..., x_n whenever these objects are in the relationships described by the associated precondition. The large number of rules required even in a relatively simple system (over 3000 rules for a 450-word vocabulary) necessitates an efficient means of continuously monitoring the blackboard to determine which rules are currently invocable because of data satisfying their preconditions.

This problem is solved by embedding the rules in an automatically compilable recognition network (ACORN), as discussed elsewhere (Hayes-Roth & Mostow, 1975). In brief, each grammatical constituent (word or phrase) is assigned a unique node in the network. Rules whose preconditions refer to the constituent are stored at the node. Whenever an hypothesis for the constituent is created or revised, its node is activated and the relevant rules become invocable.

PRINCIPLES OF CONTROL

The rule preconditions are defined in terms of various thresholds for plausibility, temporal adjacency, etc. These thresholds can be given values specific to a particular region of the utterance and are dynamically modifiable. Thus rules are invoked not only in response to new hypotheses but also in response to local threshold changes. This mechanism allows flexible matching of rule preconditions. Thresholds can be relaxed in unrecognized regions of the utterance to permit localized application of methods whose weakness would cause combinatorial explosion if they were applied uniformly throughout the utterance.

Hypotheses are explicitly linked in the data base to hypotheses which support them inferentially, and the links are marked with the strengths of the inferences. A rating policy module (RPOL) rates the plausibility of new hypotheses on the basis of the ratings of the hypotheses which support them and the strengths with which they do so. RPOL updates these ratings when an hypothesis receives new support or when the rating of one of its supporting hypotheses is changed. Hypotheses are rated separately on their contextual plausibility and on the extent to which they are supported by lower-level hypotheses.

The combinatorial search can be controlled by modifying the appropriate threshold values. For example, the search can be broadened or narrowed by relaxing or tightening criteria for recognizability, since the solution space consists only of sequences of recognizable words. A best-first search policy can be implemented simply by ordering rule invocations according to the strengths of the rules and the plausibility ratings of the hypotheses matching the rules' preconditions. The search can be further focussed by inhibiting low-level processing within a region already accounted for by a credible high-level hypothesis. Of course this policy must be pursued with caution since the high-level hypothesis may be incorrect. Cautious inhibition is implemented as deferred processing. A similar policy of procrastination can be used to defer application of weak inferences in a region until strong methods fail. An inferential process can be deferred by scheduling it with low priority (so that it may never in fact be executed), or by scheduling it only when the relevant thresholds are relaxed. The latter mechanism permits reconsideration of previously rejected alternatives.

Discourse rules can also help to focus the search. For example, an hypothesis that the current topic of conversation is food increases the a priori probability that the word "beef" will be uttered. If we can predict subject matter or syntax from any one of many knowledge elements (e.g., a recognized cue word in the same utterance, semantic analysis of previous utterances, knowledge of the particular speaker's interests), we can create such an hypothesis. This form of semantic and syntactic priming is non-restrictive in that it does not preclude recognizing an utterance which is inconsistent with an hypothesized topic of conversation or an expectation for a particular grammatical construction. The mechanism is also graceful in that it does not impose a strict hierarchy of topical domains, and in fact tolerates ambiguity and uncertainty in the expectations generated by previous discourse.

Inexact matching can also be carefully controlled with thresholds. An interval of silence in the middle of an utterance can be accepted by relaxing temporal adjacency thresholds in the region of the silence so that hypothesized sequence constituents temporally separated by the silence will be considered temporally adjacent. For example, if the speaker says "Tell me about . . . beef," this mechanism allows the words "about" and "beef" to be considered temporally adjacent. Interjections and unclear intervals of speech can be nondeterministically ignored by treating them as silences. Sometimes the uttered words cannot be recognized by lower-level KSs even after SASS hypothesizes them on the basis of surrounding context. In such cases, partially-matched phrases can be recognized by lowering credibility thresholds in unintelligible intervals so that unfulfilled expectations for missing constituents are treated as if they had been fulfilled. These mechanisms can even be used to tolerate some variation from the target language by ignoring extra verbiage not accounted for in the grammar and by filling in omitted constituents required by the grammar.

PERFORMANCE EVALUATION

The contribution of each KS in Hearsay II is highly dependent upon the behavior of the others. Consequently, SASS's performance is difficult to evaluate. For instance, SASS's prediction of the missing word "tell" in the previous example may have been critical to recognition of the utterance. On the other hand, the word-hypothesizer KS might eventually have lowered its own thresholds enough to have weakly hypothesized the missing "tell." In this case, SASS's postdiction of the hypothesized "tell" from its surrounding context might have been critical in increasing its credibility rating sufficiently to permit it to be recognized.

Despite the complex dynamics of the integrated system, we do have an evaluation methodology for SASS which will be pursued in the next year. Basically, our strategy is to generate a variety of artificial problems, each defined by a set of hypothesized words, and measure the elapsed time until SASS parses the utterance. In particular, we should be able to evaluate the relative efficacy of the four types of behavior rules in overcoming various kinds of error in the artificial input. If we can then estimate the relative frequencies of different kinds of errors generated by lower-level KSs, we can attempt to optimize SASS's behavioral profile.

CONCLUSION

There are many functions to be performed by a syntax and semantics knowledge source within a speech understanding system. In addition to simply parsing a sentence, the knowledge source must use a variety of strong and weak inferencing methods to hypothesize missing constituents and adduce support for existing hypotheses found in appropriate contexts. A production system using four types of rules has been developed to implement such desirable "knowledgeable" behaviors, which are automatically inferred from a simple declarative representation of the language to be understood. By making the invocation of a rule be dependent upon both the credibility of the data matching the rule's preconditions and the estimated strength of the rule as a useful inference, the entire search process may be controlled so as to pursue dynamically modifiable global and local processing objectives. In sum, such a production system provides a general framework for representing "knowledgeable" syntactic and semantic behaviors. Moreover, the fine computational grain of the behavior rules makes possible the flexible and precise control needed to avoid a combinatorial explosion in the search for a plausible interpretation of continuous speech.

REFERENCES

Hayes-Roth, F. and Lesser, V. R. Focus of attention in a distributed logic speech understanding system, 1976. Appears in this volume.

Hayes-Roth, F. and Mostow, D. J. An automatically compilable recognition network for structured patterns. Proc. Fourth Inter. Joint Conf. Artificial Intelligence, 1975, 246-251.

Lesser, V. R., Fennell, R. D., Erman, L. D., and Reddy, D. R. Organization of the HEARSAY II speech understanding system. IEEE Trans. Acoustics, Speech, and Signal Processing, 1975, ASSP-23(1), 11-23.

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, 1973a, AU-21(3), 229-238.

Reddy, D. R., Erman, L. D., Fennell, R. D., and Neely, R. B. The HEARSAY speech understanding system: an example of the recognition process. Proc. Third Inter. Joint Conf. Artificial Intelligence, 1973b, 185-193.

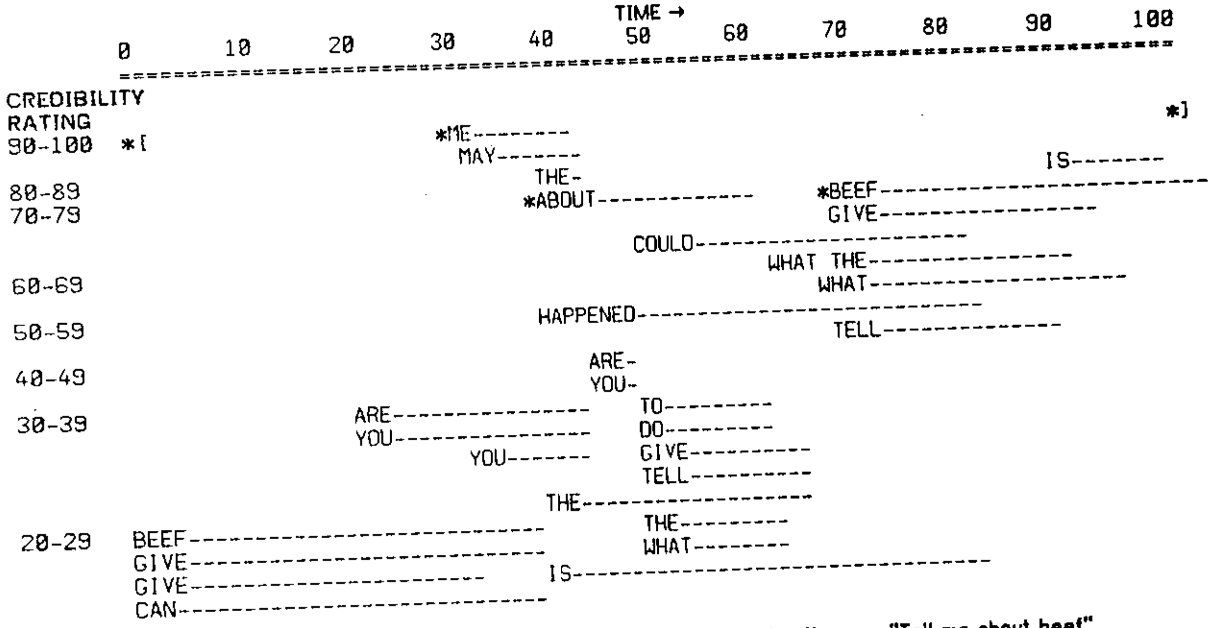

Figure 1. Words hypothesized bottom-up in response to utterance "Tell me about beef" "*" marks correct hypothesis; "[" and "]" denote hypothesized beginning and end of utterance

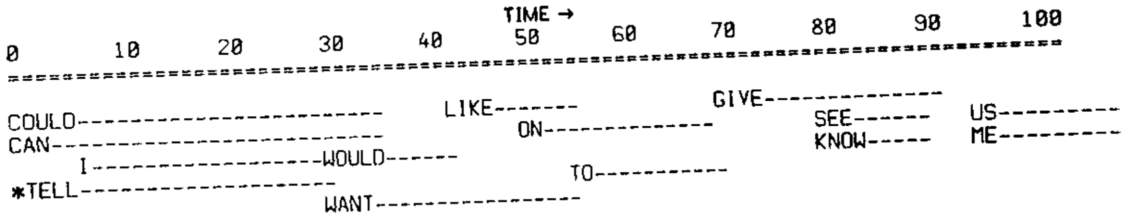

Figure 2. Words predicted by SASS on the basis of the hypotheses shown in Figure 1