An Integrated Speech and Natural Language Dialog System: Using Dialog Knowledge in Speech Recognition

Sheryl R. Young, Alexander G. Hauptmann, and Wayne H. Ward

April, 1988
CMU-CS-88-128(2)

Abstract

This article describes an integrated architecture for combining natural language processing with speech understanding in the context of a dialog system. We use constraints derived from dialog knowledge, pragmatics, semantics and syntax to restrict the words which can be recognized in a large vocabulary, speaker independent continuous speech recognition task. These knowledge sources are derived by analyzing user goals, strategies, objects in focus and database responses. Their use by the speech recognition system is enabled by combining the traditionally independent modules of a word matcher and parser. The integration of these knowledge sources allows the system to significantly reduce the search space for words in the speech signal in a dynamic manner without imposing unnatural restrictions on the user.

We wish to acknowledge Edward T. Smith, Philip Werner and Zongge Li. This research would not have been possible without their assistance.

This research was sponsored by the Defense Advance Research Projects Agency (DOD), ARPA Order No. 4976, monitored by the Air Force Avionics laboratory under contract F33615-84-K-1520.

Table of Contents

List of Figures

List of Tables

1. Introduction: The Need to Integrate Speech and Natural Language

For many years, problems in speech recognition and natural language processing have been studied independently of one another. For the most part, the work done on dialogs, intentions, goals and problem solving behavior has never been applied to speech. This is surprising, since current speech recognition technology is far from perfect and could much benefit from more knowledge based constraints. In this manuscript, we will both describe a recently implemented speech understanding system which uses knowledge based constraints early in the speech recognition process and report the resulting performance improvements.

The primary problem in analyzing connected speech is the enormously large and complex search space. In connected speech, word boundaries are not clearly identifiable. Additionally, word pronounciations are influenced by context, and frequently syllables are combined and omitted. Problems are compounded by the fact that words have many alternate pronounciations and different speakers display different accents. Thus, when using a large lexicon, or more than a few hundred words, one muyst deal with an enormous search space. Feature based recognition systems, such as the ANGEL system at CMU (Adams and Bisiani, 1986) locate features of the signal to form a network of possible segmentations. These segments are then associated with phonemic labels. Finally, a network composed of alternate word pronounciations is matched against all parts of the phonemic net for all words in the lexicon. This results in the generation of many hundreds of words candidates for every word actually spoken. A second class of recognition systems relies on time based processing to analyze the input speech signal. Again, the search space is enormous. To evaluate the many choices generated at each point in processing, speech systems assign likelihood values to their hypotheses. Due to lack of constraints, incorrect possibilities are often chosen over corect ones.

The search space problem is further complicated when people speak naturally. Normal speech is imperfect. It contains misspoken words, incomplete sentences, restarts and both silent and noisy pauses. For this reason, current speech processing efforts have focused on recognizing isolated sentences read by a speaker. Unfortunately, even speaker dependent systems[1], which generally exhibit better performance than speaker independent systems, currently have significant error rates when evaluated on prepared sentences read aloud by a speaker.

Fortunately, the speed and accuracy of speech recognition systems is a function of the size of their search space. As the search space is constrained, both speed and accuracy improve (Kimball, 1986; Erman and Lesser, 1980; Lowerre and Reddy, 1980) . Hence, it seems only reasonable to exploit natural, knowledge based constraints which exist in a spoken environment to reduce search space and improve speech recognition accuracy.

Work in the natural language community has shown that natural communication is highly structured and contains many natural constraints at the level of dialogs and problem solving. Tasks are conducted with a particular set of goals or problems in mind. The structure of plans and problem solving spaces has been long studied (Newell and Simon, 1972; Sacerdoti, 1974; Fikes and Nilsson, 1971). Hierarchical planning and problem solving is a powerful mechanism for representing the importance of goals and their interrelations, and for understanding unexpected information in text and dialog (Wilensky, 1978, 1983; Cohen and Perrault, 1979; Allen and Perrault, 1980; Litman and Allen, 1987; Grosz and Sidner, 1986). Grosz (1977) has shown how the notion of a user focus in problem solving dialogs is related to a semantic partitioning of a problem solving space. Such partitioning can assist in disambiguating input, particularly referent determination and pronominal anaphora (Sidner, 1979) .

Given these natural constraints on goal directed, problem solving behavior, it is only logical to examine ways in which these knowledge sources can be used to intelligently reduce search space and improve speech recognition accuracy. This entails looking at spoken language in a realistic problem solving context. We will describe a portion of a working speech understanding system,

[1] Speaker dependent recognition systems are trained on many utterances spoken by a single speaker.

MINDS, where users must solve naval resource management problems with the aid of a database. The system uses natural constraints to dynamically limit the search space used to recognize spoken utterances. These constraints are computed following each utterance and database response and applied early in the signal processing procedures to provide maximal effectiveness.

1.1. Current Speech Recognition Research: Use of Knowledge Based Constraints

The speech recognition literature shows several different approaches to limiting search space. The most commonly applied constraints are syntactic and semantic restrictions on individual sentences . These constraints have been shown to significantly reduce search space and improve recognition performance (Lowerre and Reddy, 1980; Erman and Lesser, 1980). Most speech recognition systems use semantic and syntactic constraints in some form of semantic network (Lea, 1980; Kimball, Price, Roucos, Schwartz, Kubala, Chow, Haas, Krasner and Makhoul, 1986; Borghesi and Favareto, 1982). This network is the basis for a parsing module and does not change from one utterance to the next. All reasonable constraints about the structure and content of single sentences are embedded into the network.

A second approach emphasizes semantic structure over syntactic constraints, as exemplified in caseframe parsing of speech (Hayes, Hauptmann, Carbonell and Tomita, 1986). The results of this work provide two important insoghts into the speech recognition problem. are demonstrated in this work. First, like other systems emphasizing semantic structure over syntactic constraint (Gatward, Johnson and Conolly, 1986) this system leaves too much ambiguity in the syntactic combination possibilities and consequently shows poor recognition results. However, more importantly, their work demonstrates the need to apply constraints earlier in the recognition process than at the parsing level, as the bottom up processing of speech signal input resulted in the generation of far too many word hypotheses for effective parsing.

Pragmatics and dialog level constraints have been mostly ignored in speech recognition systems. While several speech recognition systems claim to have dialog, discourse or pragmatic components (Lea, 1980), they only use knowledge above the sentence level like a typed natural language system to disambiguate and interpret a parse. The knowledge sources are not used to constrain the speech recognition process. Two notable systems use knowledge beyond the level of single sentences.

Barnett (1973) describes a speech recognition system which uses a "thematic" memory. It keeps track of previously recognized content words and predicts that they are likely to reoccur. In addition, Barnett refers to a dialog structure which limits possible sentence structures in different dialog states. Unfortunately, no actual results are reported.

Fink and Biermann (1986) and Biermann, Rodman, Ballard, Betancourt, Bilbro, Deas, Fineman, Fink, Gilbert, Gregory and Heidlage (1983) implemented a system that used a "dialog" feature to correct errors made by a small vocabulary, commercial speech recognition system. Their system, like Barnett's, was strictly history based. It remembered all sequences of previously recognized sentence meanings and builds a finite state dialog network. If the currently analyzed utterance looked similar to one of the stored sentence meanings, the stored meaning was used to correct any recognition errors in the new utterance. Significant improvements in sentence and word error rates were found when a history based prediction could be applied. The history constraint was only applied after a word recognition module had processed the speech, in an attempt to correct possible errors. It was not used to aid in the initial recognition process.

1.2. Innovations of the MINDS System

The MINDS system represents a radical departure from the principles of most other speech recognition systems and attempts to move beyond the limits of other systems. We believe that we can exploit the knowledge about users' problem solving strategy, their goals and focus as well as the general structure of a dialog to constrain speech recognition down to the signal processing level. In contrast to Fink and Biermann's (1986) system, we do not only correct misrecognition errors after they happen, but apply our constraints as early as possible during the analysis of an utterance. Our approach uses "predictions" derived from the problem-solving dialog situation to limit the search space at the lower levels of speech processing. At each point in the dialog, we assess the possible states a user could logically progress toward in the next utterance. These states include both subdialogs about database responses as well as further progress towards a goal. Associated with each state is a static set of concepts which may or may not be applicable during a specific problem solving session with the system. Thus, the concepts associated with each state are evaluated in light of the current context and the constraints derived from previous dialog information to dynamically create a set of concepts that may be expressed in the next utterance. For example, a state may permit a user to ask about any ship capabilitily. However, prior constraints limit the ship to be a frigate and the concepts to include only radar and sonar. We call this set of concepts a "prediction". The list of concepts in the prediction set is combined with a set of syntactic networks for possible sentence structures. The result is a dynamically constructed semantic network grammar, which reflects all the constraints derived from all our knowledge sources. Thus, a new grammar is created after each utterance/database response pair.

Since prior research has pointed out the importance of constraining the search space as early as possible, we have developed an intergrated parser and word matcher. As described earlier, word matchers are responsible for hypothesizing words from the acoustic phonetic input. The integration of the word matcher with the parser permits the parser to use the dynamic semantic network grammar for guiding the word matching process. Specifically, the parser requests the word matcher to only search for words in specific semantic categories at given points in the speech signal. Thus, only a very restricted set of word choices is possible at each point in the signal. This reduces the amount of search necessary and cuts down on the possibility of recognition errors due to ambiguity and confusion between words.

In the next section, we present an overview of the entire MINDS system indicating the domain and the flow of control in the system. The following sections focus on the generation and use of predictions. Finally, we report the results of an experiment to evaluate the effectiveness of our predictions in both reducing search space and improving speech recognition accuracy.

2. Overview of the MINDS System

Our research on integrating speech and natural language work is being done in the context of a multimedia interactive dialog system called MINDS. The MINDS system provides a robust user interface to a database which is used to solve limited resource problems in the naval domain. It is capable of:

- interpretting ambiguous user queries;
- interpretting queries using a model of the database;
- performing multiple database queries and evaluations in response to a single question;
- intelligently filtering database output.

Users of the system have important advantages primarily related to the ability of MINDS to process information from three input media (speech, typing or pointing). This provides a natural and habitable environment for the user (Martin, 1973; Shneiderman, 1980; Legett and Williams, 1984). Additionally, use of the input media can be interspersed. For example, a user may make references to displayed information using a mouse while speaking. The system then resolves this into a complete query. The system also has many means of communicating with the user (displays, speech, menus, questions), and can choose the media most appropriate.

In using MINDS, system users perform tasks such as finding least impact solutions to problems. For example, in dealing with the problem of broken equipment in the naval domain, user must assess the impact of the problem and determine whether to delay a mission to repair the vessel or to find replacement ship which will least impact other on-going missions. The database used by the problem solver is an unclassified version of the United States Navy's FRESH database. It contains information about task forces, ships, and their dynamic and static characteristics such as missions, fuel, current location, speed, radar, weapons, ability to deal with various threats, any current problems (CASREPS), etc. Additionally, the domain also covers screen manipulation commands.

As depicted in Figure 1, the flow of information in the MINDS system begins with input being passed to an *input manager*. Spoken and typed inputs are sent to the *parsing module*, while pointing information is sent directly to the *completion module*. The completion module takes the parser output and information about items pointed to and integrates these into a meaningful representation. However, as it is not always possible to unambiguously interpret input, the completion module can initiate a clarification dialog with the user. The completion module communicates with the *focus module*. Once a representation is complete it is passed to the focus module which updates goals and focus and creates abstract database queries when appropriate. Queries are passed to an "expert" *database interface* which contains a representation of the database and has the ability to take abstract queries, transform them into the required number of database queries and intelligently filter the database output to provide useful query responses. The database interface creates SQL statements and communicates with the *Informix database management system* (Informix, 1986). Database output is passed to the focus module, which evaluates the answer to derive implications for further user queries. Finally the focus module generates predictions to guide processing of further input and calls the multi-media *output manager* to communicate the information to the user.

3. An Overview of Predictions

Our approach to limiting the search space in the processing of speech relies upon the dynamic construction of semantic network grammars and lexicons which reflect the constraints derived from assessing the user's specific (sub)goals, options and strategies. These constraints are used to create five types of predictions. The predictions are described below along with the important features of the modules responsible for their creation and use in the speech recognition system.

3.1. Types of Predictions

The predictions are derived from the set of concepts which the user could logically mention to further either progress toward their current (sub)goal or their understanding of prior answers. From this information, we try to infer as much information as possible so as to maximally reduce the search space for words in the speech signal. Hence, we have come up with five types of predictions: dialog or problem solving stage, semantics and pragmatics of dialog phase, restrictions on active concepts, anaphoric restrictions, and ellipsis restrictions.

- *Dialog* or *Problem solving stage* predictions define the current place within a general dialog script. In our domain this implies assessing the problem, finding a replacement ship, changing the screens, etc. These constraints are very general and analogous to the constraints used in prior dialog systems such as GUS (Bobrow, Kaplan, Kay, Norman, Thompson and Winograd, 1977). Dialog stage information can eliminate some complete categories of semantic and syntactic information. Each of the dialog or problem solving stages has an associated set of goal trees, which are used to derive the semantic and pragmatic dialog phase predictions.

- *Semantic and pragmatic dialog phase* predictions are characterized as concepts. These concepts are contained in the AND-OR trees associated with a dialog stage. These trees define alternate goals and subgoals as well as traversal options. States in the trees can be optional, required, single or multiple use. The traversal options as well as the concepts associated with any one state change as function of the constraints discovered during the problem solving session. These constraints are propagated from prior queries and database responses, the paths and strategies selected by the user as well as what subproblems the user has solved. The concepts associated with a problem solving phase are represented in static domain knowledge

Figure 1: The Components of the MINDS System

base which contains information about the concepts and their interrelations.

- *Restrictions on active concepts* are used to create additional predictions for limiting search space. These predictions restrict values of specific concepts which are active at a particular goal tree node. Our predictions only list restrictions on goal tree nodes the user could "visit" in their next utterance. These restrictions originate from both users and database responses in prior dialog phases. For example, once a user concentrates on solving a problem involving a certain ship's impairment, we can be certain all statements using shipnames in the damage assessment stage will refer to the damaged ship.

- *Anaphoric* predictions restrict the kinds of anaphoric referents available at each dialog phase. The possible anaphoric referents are determined by user focus. From the current dialog phase, focus selects previously mentioned dialog concepts and database answers which are important at this point. These concepts are the referential content of anaphora in the next utterance. Similarly, these restrictions apply to references available for pointing. The referential predictions include distinctions between plural and singular pronoun referents.

- *Elliptic* predictions limit the kinds of elliptic substitutions we can expect at a given point. Elliptic utterances are predicted when we expect the user to ask about several concepts of the same type, after having seen a query for the first concept. Thus the ellipsis predictions indicate whether an ellipsis would be appropriate and if so, the concepts which could be mentioned elliptically.

Initially, the system establishes a set of predictions which expect the user to either state some problem or find out the most recently reported problems in some area of the world. Within a few question/answer cycles, this information allows us to infer the user's top level goal. Throughout the dialog session, predictions are constantly updated as events occur.

3.2. Generation of Predictions: An Overview

Three system modules are responsible for generating predictions, expanding them into potential linguistic surface forms and employing them to limit the search space for words when processing the speech input signal.

1. *The focus module* generates all conceptual predictions by tracking and modifying focus, dialog phase, and user strategies. This module receives two types of input: a semantic representation of the user's query from the completion module and database responses from the database module. When user input is received from the completion module it updates focus, dialog phase, and user strategy. Then it creates a database query. When database responses are received the module updates history and evaluates the response relative to the goal states, modifying the goal trees when appropriate. Finally, it creates a new set of predictions and passes them to the completion module. For example, when a user query is received, the module records the path selected by the user and infers the information the user wishes from the database from the active concepts associated with the user's current dialog state. When the database response is received, the focus module evaluates the response in search of additional constraints. These are propagated and the module finally assembles a list of possible next states and applicable concepts associated with the states.

2. *The completion module* is responsible for creating a new grammar and lexicon from the predictions. It takes the concepts from the predictions, finds the nets

associated with the conepts and generates a new grammar for the parser module. The grammar it creates is expressed as a set of recursive transition networks containing semantic word categories on their transition arcs. The completion module is the primary interface to the parser module. It also has the task of integrating mouse clicks into a query, resolving pronominal references and dealing with incomplete or confusing input.

3. *The parser module* uses the dynamically constructed lexicon and grammar to control the search for words in the acoustic phonetic latice of alternate segmentations and features. Thus its input is both the network of phonemes produced by the front end, ANGEL (Adams and Bisiani, 1986) and the lexicon and grammar. It uses the grammar to compute the semantic word categories which could appear in a certain segment of speech and expanding these concepts into words which are contained in the dynamically generated lexicon. Then the parser accesses all the word models for those words and selectively calls the word matcher with word models and locations in the speech signal. In this manner, the parser forms a rank-ordered set of phrase hypotheses to span the utterance.

The algorithms employed by the three modules important for the generation, expansion and use of predictions are described in the next three sections.

4. The Focus Module

Two functions performed by the focus module are central to the generation of predictions for the parsing module. They are:

1. Tracking goal states
2. Identifying objects and classes currently in focus

These two functions are generally responsible for generating the semantic/pragmatic and syntactic predictions, respectively.

4.1. Representing and Tracking Goal States: Dialog stage, Dialog phase and Restrictions of Active Concept Predictions

The focus module identifies a user's goal and tracks the user's progress and paths through a problem solving space (Newell and Simon, 1972). The use of plans and goal trees has been widely studied in the context of problem solving, planning (Sacerdoti, 1974; Fikes and Nilsson, 1971; Sussman, 1975) learning (Cheng and Carbonell, 1986, Laird and Newell, 1983; Rosenbloom and Newell, 1986), text (Young, 1984, 1985) and dialog (Grosz, 1977; Robinson, 1978; Levy, 1979; Allen, 1979; Allen and Perrault, 1978; Hobbs and Evans, 1980). Like most goal trees, ours are formulated as AND-OR trees. States can be optional, required, mutually exclusive, ordered, single or multiple use. It is important to note that our states are not independent of each other. Like most limited resource problems (Fox, 1983; Sacerdoti, 1974) constraints derived from prior activities are propagated among later states. In our implementation, these constraints are represented at multiple levels of abstraction which enables us to represent the users actions in the manner of least commitment planning techniques (Fox, 1983). Each state in our goal trees has a set of associated concepts which are limited and modified by the constraints propagated earlier. These concepts are represented in a separate domain knowledge base. As the user progresses through a task, further constraints are inferred, the goal trees are dynamically modified by restricting the concepts assocaited with any one state as well as by generating new states based upon the constraints derived previously.

Four elements of the goal tracking process allow us to narrow the semantic content of utterances the user could speak next:

1. Paths through the goal tree
2. Semantics/pragmatics of the goal itself
3. Interaction of goal state with knowledge base semantics
4. Constraints placed by earlier goals on a current goal

To illustrate how each of these attributes restricts what may come later in a dialog, we will use an example problem which is commonly pursued by users of the MINDS system. The exemplary problem is to find replacements for a disabled ship. This goal has a number of subgoals:

- determining if the disability affect the ship's ability to perform its mission
- determining the important attributes of the disabled vessel, particularly in relationship to its mission
- finding other ships which have the requisite capabilities and which are available
- determining which replacement possibilities will least affect other missions (finding the least impact solution)
- issuing orders to a particular ship or set of ships

4.1.1. Constraints derived from the goal tree

By maintaining representations of goal trees, states visited and current state, we can determine the states a user could visit next. Since each state has an associated set of concepts, which may or may not be restricted from prior information, it is rather straightforward to list these concepts for use in creating a new grammar. However, the task is not quite so simplistic, because the constraints derived from database responses, prior information and context restrict permissible paths. Furthermore, user strategy can influence choice of path when multiple paths are available. For example, if a user encounters an impasse where no available ships have a required attribute, the user might try to substitute a similar attribute or decide to use a combination of replacement ships. Either alternative is available, however, it is possible that prior experience with a user indicates the user is most likely to "go for the next best thing". Similarly, if a user is focusing on ship capabilities and has already addressed a number of them, and the database responses have not eliminated the need to ask for more capability information, we can predict a constrained set of capabilities the user is likely to ask about next. If the database response eliminates the need to ask for more information about capabilities, we can predict the other types of information the user will ask by moving up a level in the search tree and looking for information not yet covered, e.g. mission ratings or ability to perform on different types of missions.

4.1.2. Use of semantic/pragmatic information about a goal state in creating predictions

Goals at different levels of the goal hierarchy have semantic information and constraints associated with them. For example, if a user is pursuing a subgoal of inquiring about the purpose of a disabled ship to a particular mission, all queries will involve either the single ship or the mission. No other ships will be relevant. Similarly, only a certain set of the damaged ship's attributes will be important, namely those having to do with its ability to perform its role in the mission. However, if we are considering a set of ships as possible replacements for the damaged ship, attributes such as availability, length of current deployment and importance of current mission are relevant. Thus pragmatic and semantic information is stored both in the domain knowledge base as well as in the goal tree states. Since these states are hierarchically organized, the information is appropriately inherited.

4.1.3. Use of knowledge base information in creating predictions

While the structure of the goal tree and its associated semantic and pragmatic information place constraints on what is reasonable to ask, a second class of constraints are propagated from the knowledge base information which further restrict the content of utterances.

The knowledge base constraints are applied when concepts are activated by either user mention or being included in a database response. One of the clearest examples of a knowledge base constraint comes from problem identification. Each problem is unique in some ways. Differences in specific problems or specialization of a more general problem can be used to dynamically modify the goal tree and hence the predictions generated. For example, ship capabilities are a large class. However, this class is semantically limited by ship type. Only carriers have airplanes. If a frigate (i.e. a relatively small ship) is disabled, the capabilities associated with airplanes would be eliminated from any searches for that ship's critical capabilities. Other problem specific constraints act similarly. For example, the responsibilities of a disabled ship also limit which capabilities are important. A search and rescue mission has different needs than a space craft recovery or surveillance mission. Hence, problem specific information narrows or modifies the general structure of the goal tree by restricting the concepts associated with specific states and by generating new states which contain the appropriate constraints.

4.1.4. Constraints formed by the dialog sequence

Earlier we described how constraints derived from the knowledge base are propagated in the goal trees. Similarly, goal trees are dynamically modified by constraints discovered in earlier states. One example in the problem scenario described is the constraint imposed by a damaged ship's critical capabilities on the search for replacement ships. These constraints, like those from the knowledge base reduce or modify the set of concepts which would otherwise be associated with a particular state in a goal tree. For example, if a disabled ship has three important capabilities which enable it to perform its mission, any replacement ship will need to have either the same three capabilities or an equivalent set. Hence, when generating states associated with the search for required capabilities for a replacement ship these constraints will be used to generate states and evaluate database responses. Similarly, other solutions to the problem can be derived from prior information. For example, if a user knows that a certain class of ships has all the required capabilities, the user need only ask about those ship classes and can omit any discussion of specific required capabilities.

4.2. Identifying Focus: Syntactic Predictions

The MINDS system not only generates predictions about the content most likely to be expressed in following utterances, it also generates certain syntactic predictions. These predictions enable us to place restrictions on concepts and their bindings, determine objects available for definite reference, mouse clicking and pronominal reference and to determine whether concepts can be refered to with elliptic utterances. This information is determined by maintaining a representation of the objects currently in focus (Sidner, 1979; Grosz, 1977; McCue and Lesser, 1983) .

To illustrate the types of predictions generated from the focus mechanism consider the following examples:

- The user is informing the database about a disabled ship: *The focus mechanism would predict that no anaphoric referents could be used since no context has been provided.*

- The user is inquiring about a disabled ship: *The focus mechanism predicts that only single anaphoric referents can be used to refer to ships since a single ship is in focus. Also, only shipnames with the value of the ship in focus can be used during this dialog stage.*

- The user has already asked about capabilities and the database response indicates that the user will continue seeking information about capabilities: *The focus module predicts that the user could use an elliptical utterance to refer to one of the set of active capabilities.*
- A set of ships satisfying all the requirements has been identified (perhaps they are not deployed and are frigates). The user could ask additional things about the set or an individual ship. *The focus module would predict that plural anaphoric referents or definite reference to specific ships could be used but that elliptical utterances would be unlikely. Also, the module predicts shipnames to be limited to the ships in the set.*
- The user has isolated a group of ships with certain required capabilities and availability, and needs to choose the best of these. Furthermore, the user has just asked about the speed of one of them. *The focus module predicts that the user could ask about the speed of the remaining ships in the group (plural anaphora) or could ask about the speed of a particular ship but not by using a single anaphoric referent. Also, the user could use an ellipsis and specify only a shipname. Again, shipnames are limited to those in the group of ships.*

As illustrated by the above examples, determining the objects and attributes which are in focus and generating predictions about possible linguistic forms requires access to knowledge about the current goal, objects which have been identified and the context in which the identification was made. Furthermore, one must also consider the user's strategy. User strategy interacts with maintaining focus. The answer to a question has entirely different implications for focus with two opposing strategies. For example, a user may first determine the necessary attributes for a replacement ship and then successively apply these constraints to narrow the set of possible replacements. We call this a a *successive narrowing strategy*. A set of ships with one of the capabilities of interest is defined. Ships are successively eliminated as each of the remaining capabilities is specified. Hence, a successive narrowing strategy implies that previously suitable ships which do not have a capability currently in focus need to be eliminated from the objects under consideration. In contrast, a user could begin by applying all possible constraints and then gradually loosen restrictions.

Our algorithm for determining what is "in focus" and can thus be used for definite reference is based on the interaction of user strategy with location in the goal tree. We keep track of the goal state active when a concept is first mentioned, either by the database or the user, like most other systems. If a state is still active, the object referenced in that state can be used for definite reference. By maintaining more than one active state, we can usually deal with user strategies in this manner. However, the user strategy can also place restrictions on what would otherwise be available for reference, and these constraints restrict what would otherwise be admissible.

To determine the attributes and objects which can be referenced elliptically, we must keep track of both focus and prior utterances. More specifically, we need to know what level in the goal tree was accessed by a previous question and the possible goal tree states the user could move to on the subsequent utterance. If both the prior and subsequent goal states are both "children" of some parent state, then elliptical utterances would be expected by the prediction module.

Thus, the generation of syntactic predictions requires that the focus module keep track of all inputs. A dialog task implies constant change and the MINDS system in particular allows a user to input information in many distinct input modes. Proper interpretation of input and database output is a function of goals, history and any user strategies as well as the input material itself.

5. The Completion Module: Expanding Predictions into Dynamic Networks

Once conceptual predictions are generated by the focus mechanism, they must be expanded into possible linguistic surface forms by the completion module for use by the parser. Since the predictions are abstract representations, they must be translated into word sequences with that conceptual meaning, in effect reversing the classic understanding process by unparsing the conceptual representations into all possible word strings which can denote the concept. It also uses the information about objects and attributes in focus to place restrictions on the semantic word categories which are contained in the predicted subnets.

5.1. Use of Semantic Subnets

In addition to the individual concepts, which usually expand into noun phrases, we also have a complete semantic recursive transition network grammar that has been partitioned into subnets. A subnet defines allowable syntactic surface forms to express a particular semantic content. For example, all ways of asking for the capabilities of ships are grouped together. The arcs in the network constrain semantic categories instead of just words. Certain categories are "variable", the membership in the category may vary with predictions. For example, ship capabilities may be limited to various types of radar only, due to the focus predictions. The parser is then told that for the *variable* ship-capabilities only the *fillers* associated with radar types are allowed. While each subet constrains a set of semantic information, the subnets themselves are grouped according to semantic categories. Hence, there is more than one subnets for each combination of semantic categories. This permits multidimensional indexing of subnets. Subnets are precompiled for efficiency.

5.2. Syntactic Subnet Partitioning

In addition to partitioning by semantic content, the task grammar is also partitioned along the syntactic dimensions addressed by the focus module. Thus we have separate subnets for ellipses and expressions involving anaphora. This grouping is orthogonal to the semantic subnet clusters mentioned above. Hence, we classify the semantic content as well as the syntactic form of each grammar subnet.

5.3. Dialog State Subnets

Each of the dialog states in a script is associated with a series of semantic/syntactic subnets. While individual subnets are frequently associated with multiple dialog states, by eliminating certain subnets from consideration during a dialog phase, we reduce ambiguity. The subnet restrictions placed by dialog state generally do not restrict the objects which can be talked about, but limit the sentence types used during this dialog phase. Thus declarative sentences are not relevant when we expect queries about ships. They are relevant when information is given to the database within certain parts of the script. Interrogatives and imperatives are also matched with some dialog states. Thus, dialog state places general restrictions on the subnets which correspond with to specific features in a dialog state. For example, commands to alter the organization of the display windows are only appropriate in certain dialog states. Furthermore, natural use of language dictates the use of fairly rigid kinds of syntactic structures to refer to the screen objects and their manipulation. You would never hear a passive request such as *"This window should be enlarged by me"* but rather something like *"Enlarge this window"* or *"I want to remove this ship from the display"*.

5.4. Translating Predictions into Surface Forms

As illustrated above, the grammar is multidimensionally segmented into subnets. Our algorithm for using this information to translate the predictions from the focus module into a form usable by the parser is as follows. First we examine only the subnets that are relevant to a particular dialog state. Next we restrict this set to include only those subnets which contain one or more of the predicted semantic concepts. Forms that violate predictions on ellipsis or anaphora are pruned from this set. Once the set of subnets is defined, we look for all the "variable" semantic concept categories, and check if their membership has been reduced by the predictions from the focus module. The module then forms an active lexicon list and grammar based on the resulting subnets and restrictions derived from this algorithm.

6. The Parser Module: Speech Parsing with Strong Constraints

Once the parser is notified that a new prediction set has been released from the completion module, it processes the active lexicon and grammar for use with subsequent input. The parsing module deals with both spoken and typed input. Since typed input presents fewer of the problems we have to deal with in speech understanding, we will focus on the processing of spoken input.

The parsing module is composed of an integrated set of parsers and word matchers. When a user speaks an utterance, the ANGEL (Adams and Bisiani, 1986) front-end produces a network of phonetic labels from the speech signal. A set of independent locators is used to segment the speech. These locators look for features in the signal indicating events like stops, fricatives, closures, etc. The segments form a network, in that there are alternate paths through the same area of speech. Once the speech is segmented, the segments are labelled. A vector of phoneme labels with a probability for each is assigned to the segment.

The parser module takes as input the network of phonemes produced by the front end and eventually forms a set of phrase hypotheses. The phonemes allow words to start and end at just about any point in an utterance. These points do not necessarily correspond to the start and end points of real words in the utterance. The word matcher must take the lattice of phonemes and create a lattice of words. This is a difficult network to network matching problem. For each word in the lexicon, the word matcher has a number of different word models representing alternate pronounciations. These alternate pronounciations are generated by applying a set of rules a baseform phonemic transcription of the word (Rudnicky, 1987). These word models are represented as a network of phonemes. Normally, the word matcher will try to find match all word models against all portions of the acoustic phonemic network. Then the parser takes the lattice of words generated by the word matcher and tries to form meaningful phrases from the most likely word candidates (Ward, Hauptmann, Stern and Chanak, 1988; Stern, Ward, Hauptmann and Leon, 1987).

In contrast, the MINDS system integrates the traditionally separate modules of parser and word matcher. By integrating the parsers and word matcher and dynamically generating new grammars and lexicons, we can significantly reduce the search space for words in the acoustic phonetic signal and hence reduce the complexity of the speech recognition problem. There are two important reasons why the search space for words is greatly reduced. First, only the words that are contained in the currently active lexicon are considered by the word matching routine. As a result the word matcher generates far fewer word candidates in the word lattice than if predictions were not used to limit the words which could be matched. Secondly, the parser uses the subnet grammars to direct the operation of the word matcher. It does this by telling it which words to look for in a specific portion of the phoneme lattice, giving it either a start or end position. More specifically, in the process of extending phrases to span an utterance, the parser produces a set of categories that could extend a specific phrase. These categories are derived from the predicted set of subnets. These categories are expanded into words, but only the words in the lexicon are included in the list (recall concepts can have restrictions on their "fillers"). The parser then asks the word module to search for the set of word models in the adjacent area of the phoneme network. Thus, the parsing module uses the semantic and syntactic predictions to restrict which words can be matched in a specific region of speech. These restrictions not only eliminate many alternatives, improving the probability of a successful match, but they also frequently eliminate acoustically confusable words from consideration.

In order to produce a rank ordered set of possible phrase hypotheses, all possible utterances must be scored. The word module evaluates each hypothseized word by giving it a score based on the scores for the individual phonemes matches as well as the overall goodness of the word model match. Phrase hypotheses are assigned a score based on the scores of their component words.

Those below a relative threshold quality are pruned. The extensions continue until all phrases span the utterance (or were pruned). Thus, the parser produces a rank-ordered set of phrase hypotheses that span the utterance.

6.1. Two Alternate Parsing Strategies: Island Driving vs Left-to-Right

We are currently experimenting with two basic parsing strategies, Left-to-Right and Island-Driving. The left-to-right parser starts with a partial hypothesis which is the start token, since neither the beginning or end of a speech signal is clearly identifiable. The candidates for the start token are derived from using the word matcher to detect any word which could begin a phrase included in a predicted subnet in the region of speech. The left-to-right parser forms new hypotheses by adding words to the end of current ones, extending only to the right. As described above, the parser looks for the semantic categories of words which could follow any of the reasonable scoring word hypotheses. It expands these categories into words contained in the active lexicon and then calls the word matcher on the list of words.

The island-driven parser forms a set of words which are all words that can appear in any position in the currently active subnets. These words are constrained in the currently active lexicon, derived from the predictions. The phoneme net is scanned for matches of these words. A threshold match score is set. Word matches that have a score better than the threshold are kept as islands, and others are pruned. All islands that can juncture, or could reasonably follow one another given considerations of the speech signal, are joined to form larger islands. Thus, islands are joined when the same word is added to the right of one and left of another. The parser then seeks to extend islands to the right and left, as explained above.

The two parsing strategies described above have complementary strengths and weaknesses. The left-to-right parser is more efficient than the island-driving parser in that it can use all grammatical and lexical constraints in a single pass. The minimal set of words is used at each point. However, it is less robust than the island-driven parser when words are misrecognized. Word matches can be missed because of mispronunciations, noise, front end errors, incorrect word models, etc. In this case, the correct word has a very bad score while other word hypotheses in the same position may have much better scores. A left-right beam-search parser will exhibit garden path behavior here if it prunes tightly enough that the extension with the misrecognized word is deleted, while an the island-driven parser will not.

6.2. Dealing with Ill-Formed Input: Bad Parses and Misrecognized Words

Thus far, we have concentrated on deriving predictions which constrain what a user could reasonably say, and the use of these predictions by a large vocabulary, speaker independent speech recognition system. However, an interactive dialog does not merely consist of repeated question answer cycles. We define a **cycle** as a satisfactory, complete pair of user request and system response (i.e. the system receives a correct, legitimate request and provides an answer or action in response). However, due to both the complexity of the speech recognition problem and the tendencies of users to be incomplete, ambiguous and misspeak, the system must deal with ill-formed input (Weischedel and Black, 1983) . Thus, before processing a user query, there is often a need to engage in a a series of interactions between the system and user to clarify, disambiguate or correct the request.

Our system uses three techniques for dealing with ill-formed user input:

1. Evaluating the semantic content of questionable information
2. Requesting user verification
3. Initiating a clarification dialog

6.2.1. Evaluating Possibly Incorrect Information

One feature of the parsing module is that scores are assigned to words and phrases which reflect the system's confidence that the item is correct. Parses with bad scores must be treated as questionable by the system. Fortunately, the system can determine the semantic category of poor scring information and determine its relative importance to the interpretation of the utterance by using the subnet grammars. If poor scoring words do not contain important semantic content, system may proceed without interruption. Since the partitioning of networks and interpretation of input is semantic, all words need not be correctly recognized for the system to fulfill the request. The appropriate subnet must be determined and the content words must be correctly recognized. Misrecognition of other words in the surface forms does no harm. The system must recognize that the user asked about a specification for a ship, that the specification was SPEED and that the ship was KENNEDY. It is irrelevant whether the user said "Please show me" or "What's the" as the request form. Grouping of subnets according to meaning allows us to focus speech recognition on these content words and de-emphasize filler words.

6.2.2. Verification

When the parser detects that an important content word is questionable but is the most likely to have been said by the user, the verification strategy is employed. Before the state of the system changes and an time-intensive database query is initiated, we verify that the speech was correctly parsed by displaying a paraphrase of the system's interpretation of the utterance as text, as well as speaking it. The user may acknowledge this by clicking on a "pop-up" menu, typing or speaking a confirmation or disconfirmation. The parser also uses a subnet to parse this. Alternatively, unless the user rejects the interpretation of the input within a few seconds, the system assumes an affirmative response.

6.2.3. Clarification

A clarification dialog is initiated when the parser cannot form a good guess about what the user said. This will happen when a content word could not be recognized from the speech signal, when a content word was mistyped or when an utterance was produced which does not match one of the currently predicted subnets. It could also occur if anaphoric referents could not be resolved. The standard procedure there is to ask the user a question about the intended utterance. In addition, a "pop-up" menu appears on the screen. The items in the "pop-up" menu come from the parser. The parser produces a set of phrase hypotheses instead of a single best one. Thus the user must select a phrase interpretation taken from alternate phrase hypotheses or reject the parse completely. The user is then free to speak, type or click the clarification. The phrase hypotheses contained in the menu are derived using many strong constraints. These constraints originate from both the predictions as well as from partially recognized input. Thus, there are usually very few in the menu alternatives to choose from. Should there be many alternatives, the system also contains precompiled canonical expressions used for querying the user about specific concepts or intended meanings of utterances.

7. Evaluation

Prior research indicated that if we could effectively reduce the search space involved in processing a speech signal, we would be able to improve recognition performance. Therefore, we attempted to derive a circumscribed set of concepts which could be mentioned in a subsequent utterance. This set of concepts excluded information which appeared unreasonable, repetitive or non-productive. The set was derived from tracking a user's progress through a problem solving task.

To evaluate the effects of integrating dialog, goals and focus into a speaker-independent, continuous speech recognition system, we assessed

- the size of the search space for words
- recognition accuracy

with and without the use of our added knowledge based constraints. Thus we evaluate the system using only syntactic and semanitc constraints at the individual sentence level and contrast these findings with the system using both the individual sentence constraints and all of the dialog level knowledge described earlier

7.1. System Impact on Search Space

To measure the effectiveness of the predictions in reducing the search space of the parser/word matcher module we employed two measures, average branching factor and perplexity. Average branching factor indicates how many chioces the speech recognition system is faced with when trying to identify a word. Generally, lower branching factor indicates higher constraint and better recognition because the system has fewer choices to make. This results in fewer errors in the speech recognition process. Perplexity is another related measure derived from information theory. It is calculated as 2 raised to the power of the entropy of the grammar. Entropy is the average of the log of the number of branches or the possible words which could occur at each point in a sentence (Roucos, 1987; Brown, 1987). Perplexity is the standard measure used when evaluating the performance of speech recognition systems. The identical system will increase its recognition performance as it employs grammars which decrease perplexity. While a system grammar may have a large branching factor or perplexity, a more meaningful measure is often the test set branching factor or perplexity. These measures evaluate the difficulty of the actual sentences parsed. A test set branching factor is computed by tracing the path of each utterance through the nets and averaging the branching possibilities encountered during a correct parse. Test set perplexity is the perplexity for the nodes actually traversed during a particular utterance.

To measure the perplexity reduction provided by the various dialog knowledge sources we created three dialog scenarios which varied greatly in the types of questions asked and the level of detail employed. These dialogs shared few common words and visited different states in the task goal trees. One dialog was constructed to ask very high level information which yielded much information. This dialog thus provided less constraining context than the other two. The second dialog intermixed both high level and low level questions. High level questions were generally used except in places where more in-depth information would be more useful for finding an optimal solution. The third dialog asked many detailed questions, but by no means exhausted all the information which could be asked. The number of sentences in each of the dialogs was 10, 21 and 30, for dialogs 1, 2 and 3, respectively.

The test set branching factor and test set perplexity were computed for each of the three dialogs. The test set measures did not include information from either verification states or clarification dialogs. Only the actual sentences produced as queries by the user were used for calculations.

Complexity of the Recognition Task: Branching Factors		
Constraints used:	sentence level	dialog knowledge
Combined Test Set B.F.	63.8	14.2
Dialog 1 Test Set B.F.	63.2	14.4
Dialog 2 Test Set B.F.	61.0	14.4
Dialog 3 Test Set B.F.	66.0	14.1

Table 7-1: Average test set branching factor for the actual utterances used in the evaluation dialogs

As seen in Tables 7-1 and 7-2, the overall result of applying the dialog level constraints was to reduce average test set branching factor from an average of 63.8 to 14.2, and to reduce average test set perplexity from an average of 31.5 to 9.5. These averages were derived by multiplying the actual test set numbers for each dialog by the number of sentences in the dialog and dividing

Complexity of the Recognition Task: Perplexity		
Constraints used:	sentence level	dialog knowledge
Combined Test Set Perpl.	31.5	9.5
Dialog 1 Test Set Perpl.	32.0	10.7
Dialog 2 Test Set Perpl.	29.1	9.7
Dialog 3 Test Set Perpl.	33.0	9.0

Table 7-2: Test set perplexity for the actual utterances used in the evaluation dialogs

by the total number of sentences in all three dialogs. The results show that search space is most significantly reduced when dialog knowledge sources are used. The branching factor is cut to less than one fourth its value when sentence level syntax and semantics are used alone. Similarly, the dialog level knowledge is responsible for reducing the perplexity measure by a factor of three. Furthermore, these effects show little variability. When perplexity is measured using all predictions to constrain the search set, perplexity averages at 9.5, and ranges from 9 to 10.7.

To further assess the effects of the various types of predictions on the reduction in perplexity, we computed additional statistics. First we evaluated the effects of using dialog or problem solving stage alone. Since this type of knowledge has been used in the past, evaluating its effects will give us an idea of how much of the observed reduction is search space is attributable to the to the other four knowledge sources. When dialog state predictions are added to the restrictions placed by the grammar, perplexity is decreases to an average of 24.66, a reduction of 6.84. Thus, the other four knowledge sources jointly reduced perplexity from 24.66 to 9.5, or by 15.16.

In order to determine the relative contributions of each of the four additional knowledge sources, we took each sentence and looked at the information predicted to occur by the grammar constrined by the dialog or problem solving stage knowledge and the information predicted by selectively turn on and off the other sources of prediction. Unfortunately, two effects complicate these measures. First, the knowledge sources used for prediction are not orthogonal. Different knowledge sources exclude the same information from inclusion in the dynamically generated grammar. This effect is most pronounced in the later stages of each dialog or problem solving stage, when there are few options left to the user, enabling the semantic and proagmatic knowledge of dialog phase to eliminate most possibilities. The second complication arises in sentences where there all forms of anaphora and most ellipses are permitted. Because we wished to evaluate the effectiveness of each knowledge source independent of specific dialogs and problem solving situations, we selected sentences where all five types of knowledge placed restrictions on the grammar.

Our results show that all knowledge sources except predicting the types of pronominal references which could be used, significantly reduced perplexity. The semantic and pragmatic knowledge about dialog phases reduced perplexity by an average of 5.6. The restrictions on active concept bindings which scope objects for definite reference reduced perplexity by an average of 3. Restrictions on pronominal referents reduced perplexity by only 0.6. However, since pronominal references are most acoustically confusable, we believe these restrictions did lead to an overall imporvement in recognition performance. Last, predicting when elliptical utterances can be used and the objects which can be referred to elliptically reduced perplexity by 5.4.

Finally, as seen in the tables, the three dialogs have roughly equivalent test set branching factors and perplexity measures. This implies that the three dialogs are equally difficult to recognize in a speech understanding system.

Thus, the ability to intelligently circumscribe the set of semantic information which could sensibly follow a given utterance results in significantly reducing the number of words a speech recognition system considers when processing a signal.

7.2. System Impact on Speech Recognition Accuracy

To test the effectiveness of the use of this knowledge in MINDS, 5 speakers (3 male, 2 female) spoke to the system. To assure a controlled environment for these evaluations, the subjects only spoke the sentences prepared in three sample dialog scripts, which contained 30, 21 and 10 sentences each. As discussed earlier, the three dialogs differed in the number and specificity of the questions asked. Each speaker spoke all sentences in all three dialogs, yielding 61 sentences per speaker, or a total of 305 sentences. To prevent confounding of the experiment due to misrecognized words, the system did not use its own speech recognition result to change state. Instead, after producing the speech recognition result, the system read the correct recognition from a file which contained the complete dialog script. Thus the system always changed state according to a correct analysis of the utterance.

The system was only tested with a vocabulary of 205 words, even though the complete vocabulary is 1029 words. Since we were using an older, experimental version of the ANGEL front-end [1], our recognition results where substantially worse than for the current official CMU speech system. However, the point we wish to make concerns the relative improvement due to our knowledge sources, not the absolute recognition performance of the total speech system. Thus we present comparisons of speech recognition accuracy performance using two different levels of constraints: using sentential knowledge constraints only and using all the power of the dialog predictions. Thus each utterance was parsed with two different levels of constraint.

- The **"sentential level"** constraints used the grammar in its most general form, without partitioning. The constraints found in the combined semantic network of all possible sentence structures were used. The network grammar was the same for all utterances in all dialogs. This only allowed recognition of syntactically and semantically correct sentences, but ignored any user goals, focus or dialog knowledge. In addition, we used all the word level constraints. These include knowledge of word pronunciation and coarticulation rules. The sentential level is the equivalent of all the knowledge employed by most existing speech systems, as discussed earlier.

- Using all **"dialog knowledge"** constraints, we applied all the knowledge built into the system at every level. In particular all applicable dialog knowledge was added to improve performance of the system. The grammar was dynamically reconstructed for each utterance, depending on the dialog situation, user focus and goals. Thus the grammar was different for almost every utterance. Of course, the word and sentential level knowledge was also used.

Tables 7-3 and 7-4 show the actual parsing results for each dialog, averaged across speakers, for each mode. Word accuracy refers to the percentage of spoken words which were recognized accurately by the system. Sentence accuracy refers to the percentage of sentences where the system reacted as if all the words had been understood correctly. Further analysis revealed that 1.5% of these sentences contained small misrecognized words (such as "give" as opposed to "show"), but the resulting meanings and all references were correct.

As seen in Table 7-4, overall average sentence accuracy for the 305 sentences increased dramatically, from 32.1 to 58.4 percent, when the dialog predictions limited the words which could be matched by speech recognition system. Similarly, as seen in Table 7-3 the dialog constraints yielded a significant increase in the accuracy of word recognition, increasing recognition rates from 44.6 to 66.3 percent overall. These increases in accuracy are also reflected in all individual dialogs.

While the actual recognition accuracy numbers are dependent on the particular recognition

Speech Recognition Accuracy: Word Recognition		
	Constraints	
	Sentence Level	Dialog Knowledge
Dialog 1	43.9	66.6
Dialog 2	49.7	68.8
Dialog 3	36.3	60.1
All dialogs combined	44.6	66.3

Table 7-3: Recognition results are shown as percentage of words correct with and without dialog constraints from the MINDS system

Speech Recognition Accuracy: Sentences Correct		
	Constraints	
	Sentence Level	Dialog Knowledge
Dialog 1	31.2	58.1
Dialog 2	38.0	61.9
Dialog 3	22.0	52.0
All dialogs combined	32.1	58.4

Table 7-4: Recognition results are shown as percentage of sentences correct with and without the dialog constraints from the MINDS system

system used, the increased recognition accuracy due to the higher level constraints would be noticeable in any system.

8. Conclusions and Future Directions

Conceptualized and tested in this paper was a system designed to incorporate various forms of dialog level knowledge into a speech recogntion system. The system was designed to decrease search space and improve recognition performance. The most salient feature of this system was introducing dialog level knowledge early in the recogntion process. This was enabled by the use of dynamically generated grammars. A new grammar was dynamically generated after each cycle of user request and system response. The test was successful in that it significantly reduced the search space for words in a speech signal and also significantly imporved speech recognition accuracy.

Furthermore, it is important to note that the dynamically generated grammars used by the MINDs system are applicable to other speech recognition technologies which are not feature based. We initially began our experiements using a commercially available, speaker dependent, isolated word recognition system. While our approach to creating new grammars worked quite well, two problems discouraged us from pursuing our research with these tools. First, the commercially available system created new lexicons very slowly. This was true with mere creation of an active lexicon. Hence, we did not even try to provide new word lists for each individual word, which is one of the strengths of a network grammar. Secondly, using an isolated word recognition system prevented us from being able to study more natural speech patterns. Thus, we opted for a system which would ultimately enable us to continue our research without major adaptation. Currently, we are also adapting a speaker independent, continuous speech system which uses hidden Markov modelling techniques (a time based system) to accept predictions from the MINDS system. Since hidden markov models currently provide higher levels of recognition than feature based models[2], we expect that our performance will enable us to use the system on many untrained users. Thus, we claim that the techniques discussed in this manuscript are applicable to most all speech recognition technologies.

The ability to intelligently circumscribe search space in a speech recognition task is also important because it will make investigating more natural speech phenomenon, such as misspeaking, restarting utterances, silent and filled pauses, more tractable. To date, connected speech recognition system aviod processing signals incorporating such phenomenon because the search space is already enormous.

Based on these findings, we are currently investigating three important issues which emerged. First, we are investigating using additional knowledge sources to imporve our ability to predict what a user is likely to say. Thus, we are currently adding information about user background knowledge (semantic, not episodic) which should enable us to more efficiently partition the goal trees. For example, in our domain, there are many classes of ships within each ship type. Ships within these classes tend to have similar functionality, although some may have newer versions of equipment than others. Persons with significant background knowledge will know this information and will not have to inquire whether its feasible to substitute a ship in one class for another in the same class. Similarly, the requirements of certain missions or operations are well known to experienced seamen. These users will know the effect of a malfunctioning piece of equipment on a mission have will ask different types of questions than a person without such information. On the other hand, a student with little exposure to a naval domain will not be familiar with the different types of equipment within a class and will be much more likely to ask about database answers. We believe that knowledge about user experience will enable us to further restrict the types of utterances likely to be asked. Recent research has demonstrated that systems can automatically determine a user's level of expertise (Chin, 1988) and we believe models of individual users can be automatically developed to refine the expertise predictions.

Secondly, we are investigating ways to eliminate any rigidity which may result from an over zealous use of predictions. Thus, we are currently implementing a version of the system which employs a multi layered set of predictions. The advantage of such a system is that predictions can constrain the search space more than in the current system by using knowledge of user expertise and strategy preferences while avioding the rigidity which could result. With this model we can gradually relax some of the constraints in the face of bad evidence in the spoken input. The idea is that the system first parses with the most restrictive set of constraints possible. Should the parse fail, we try again, looking beyond the immediate predictions and allowing other alternative expressions to occur.

Finally, for this domain, the goal trees were hand-coded based upon detailed analyses of transcriptions of problem solving sessions. We are now looking into automating ways to analyze a preferred way of solving problems from user transcripts without strong predictions in place initially. As the patterns emerge clearly, the predictions also fall into place.

We believe that the system introduced herein offers a substantial advantage to furthering speech recognition technology. The ability to intelligently circumscribe the search space for words will facilitate the study of both very large vocabulary systems and natural speech phenomenon such as pausing and restarted utterances.

[2] at least as of the October 1987 DARPA meeting

References

1. Adams, D. A. and Bisiani, R. "The Carnegie-Mellon University Distributed Speech Recognition System". *Speech Technology* (March/April 1986).

2. Allen, J. F. *A Plan Based Approach to Speech Act Recognition.* Ph.D. Th., University of Toronto, 1979.

3. Allen, J. F. and Perrault, C. R. "Analyzing Intention in Utterances". *Artificial Intelligence 15*, 3 (1980), 143-178.

4. Allen, J. and Perrault, C. Participating in Dialogs: Understanding via Plan Deduction. Proceedings of the Second National Conference of the Canadian Society for Computational Studies of Intelligence, Toronto, 1978.

5. Barnett., J. "A Vocal Data Management System". *IEEE Transactions on Audio and Electroacoustics AU-21*, 3 (June 1973), 185 - 186.

6. Biermann, A., Rodman R., Ballard B., Betancourt, T., Bilbro, G., Deas, H., Fineman, L., Fink, P., Gilbert, K., Gregory, D. and Heidlage, F. Interactive natural language problem solving: A pragmatic approach. Conference on Applied Natural Language Processing, 1983, pp. 180 - 191.

7. Bobrow, D.G., Kaplan, R.M., Kay, M., Norman, D.A., Thompson, H. and Winograd, T. "GUS: A Frame Driven Dialogue System". *Artificial Intelligence 8* (1977), 155 - 173.

8. Borghesi, L. and Favareto, C. Flexible Parsing of Discretely Uttered Sentences. COLING-82, Association for Computational Linguistics, Prague, July, 1982, pp. 37 - 48.

9. Brown, P. *The Acoustic Modelling Problem in Automatic Speech Recogntion.* Ph.D. Th., Carnegie-Mellon University, 1987.

10. Cheng, P. W. and Carbonell, J. G. The FERMI System: Inducing Iterative Macro-operators from Experience. Proceedings on the Fifth National Conference on Artificial Intelligence, AAAI-86, 1986, pp. 490-495.

11. Chin, D.N. *Intelligent Agents as a Basis for Natural Language Interfaces.* Ph.D. Th., University of California, Berkeley, 1988.

12. Cohen, P. R. and Perrault, C. R. "Elements of a Plan-Based Theory of Speech Acts". *Cognitive Science 3* (1979), 177-212.

13. Erman, L.D. and Lesser, V.R. The Hearsay-II Speech Understanding System: A Tutorial. In Lea, W.A., Ed., *Trends in Speech Recognition*, Prentice-Hall, Englewood Cliffs, NJ, 1980, pp. 340 - 360.

14. Fikes, R. E. and Nilsson, N. J. "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving". *Artificial Intelligence 2* (1971), 189-208.

15. Fink, P. K. and Biermann, A. W. "The Correction of Ill-Formed Input Using History-Based Expectation With Applications to Speech Understanding". *Computational Linguistics 12* (1986), 13-36.

16. Fox, M. S. *Constraint-Directed Search: A Case Study of Job-Shop Scheduling.* Ph.D. Th., Carneige-Mellon University, 1983.

17. Gatward, R.A., Johnson, S.R. and Conolly, J.H. A Natural Language Processing System Based on Functional Grammar. Speech Input/Output; Techniques and Applications, Institute for Electrical Engineers, 1986, pp. 125 - 128.

18. Grosz, B. J. *The Representation and Use of Focus in Dialogue Understanding*. Ph.D. Th., University of California at Berkeley, 1977. SRI Tech. Note 151.

19. Grosz, B. J. and Sidner, C. L. "Attention, Intentions and the Structure of Discourse". *Computation Linguistics 12* (1986), 175-204.

20. Hayes, P. J., Hauptmann, A. G., Carbonell, J. G., and Tomita, M. Parsing Spoken Language: a Semantic Caseframe Approach. COLING86, Bonn, Germany, August, 1986.

21. Hobbs, J.R. and Evans, D.A. "Conversation as Planned Behavior". *Cognitive Science 4* (1980), 349-377.

22. *Informix-ESQL/C Programmer's Manual*. 1986.

23. Kimball, O., Price, P., Roucos, S., Schwartz, R., Kubala, F., Chow, Y.-L., Haas, A., Krasner, M. and Makhoul, J. Recognition Performance and Grammatical Constraints. Proceedings of the DARPA Speech Recognition Workshop, Science Applications International Corporation Report Number SAIC-86/1546, 1986, pp. 53 - 59.

24. Laird, J. E. and Newell, A. A Universal Weak Method. Proceedings of the Eight Joint Conference on Artificial Intelligence, 1983.

25. Lea, W.A. (Ed.). *Trends in Speech Recognition*. Prentice-Hall, Englewood Cliffs, NJ, 1980.

26. Leggett,J. and Williams, G. "An empirical investigation of voice as an input modality for computer programming". *International Journal of Man-Machine Studies 21*, 6 (December 1984), 493 - 520.

27. Levy, D. Communicative Goals and Strategies: Between Discourse and Syntax. In *Syntax and Semantics, Vol. 12*, T. Givon, Ed., Academic Press, New York, 1979.

28. Litman, D. J. and Allen, J. F. "A Plan Recognition Model for Subdialogues in Conversation". *Cognitive Science 11* (1987), 163-200.

29. Lowerre, B. and Reddy, R. The Harpy Speech Understanding System. In Lea, W.A., Ed., *Trends in Speech Recognition*, Prentice-Hall, Englewood Cliffs, NJ, 1980, pp. 340 - 360.

30. Martin, J.. *Design of Man-Computer Dialogues*. Prentice-Hall, Englewood Cliffs, 1973.

31. McCue,D. and Lesser, V. Focusing and Constraint Management in Intelligent Interface Design. Tech. Rept. COINS Tech report 83-36, University of Massachusetts at Amherst, 1983.

32. Newell, A. and Simon, H. A.. *Human Problem Solving*. New Jersey: Prentice-Hall, 1972.

33. Robinson, J. Purposeful Questions and Pointed Answers. 7th International Conference on Computations Linguistics, Bergen, Norway, 1978.

34. Rosenbloom, P. S. and Newell, A. The Chinking of Goal Hierarchiers: A Generalized Method of Practice. In *Machine Learning: An Artificial Intelligence Approach*, Michalski, R. S., Carbonell, J. and Mitchell, T., Ed., Morgan Kaufman, 1986.

35. Roucos, S. Measuring Perplexity of Language Models used in Speech Recognizers. Unpublished Manuscript, 1987.

36. Rudnicky, A.I. The Lexical Access Component of the CMU Continuous Speech Recognition System. Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, 1987.

37. Sacerdoti, E. D. "Planning in a Hierarchy of Abstraction Spaces". *Artificial Intelligence 5*, 2 (1974), 115-135.

38. Shneiderman, B.. *Software Psychology: Human Factors in Computer and Information Systems*. Winthrop, Cambridge, MA, 1980.

39. Sidner, C. L. *Towards a Computational Theory of Definite Anaphora Comprehension in English Discourse*. Ph.D. Th., MIT, 1979. AI-TR 537.

40. Stern, R.M., Ward, W.H., Hauptmann, A.G. and Leon, J. Sentence Parsing with Weak Grammatical Constaints. ICASSP-87, 1987.

41. Sussman, G.. *A Computer Model of Skill Acquisition*. Elsevier, New York, 1975.

42. Ward, W.H., Hauptmann, A.G., Stern, R.M. and Chanak, T. Parsing Spoken Phrases Despite Missing Words. ICASSP-88, 1988.

43. Weischedel, R.M. and Black, J.E. "Responding Intelligently to Unparsable Inputs". *American Journal of Computation Linguistics 9* (1983), 161-177.

44. Wilensky, R. *Understanding Goal-Based Stories*. Ph.D. Th., Yale University, Sept. 1978.

45. Wilensky, R.. *Planning and Understanding*. Addison Wesley, Reading, MA, 1983.

46. Young, S. R. *A Theory and Simulation of Macrostructure*. Ph.D. Th., University of Colorado, Sept. 1984.

47. Young, S. R. "How to Simulate Cognitive Processes". *Behavioral Research Methods, Instrumentation and Computers 17* (1985), 20-31.