PHRAN:

A KNOWLEDGE-BASED APPROACH TO NATURAL LANGUAGE ANALYSIS

by

Robert Wilensky and Yigal Arens

Memorandum No. UCB/ERL M80/34

12 August 1980

Run blank

PHRAN

A KNOWLEDGE-BASED APPROACH TO NATURAL LANGUAGE ANALYSIS

by

Robert Wilensky and Yigal Arens

Memorandum No. UCB/ERL M80/34

12 August 1980

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

PHRAN
A Knowledge-Based Approach to Natural Language Analysis*

Robert Wilensky and Yigal Arens

Computer Science Division
Department of EECS
University of California, Berkeley
Berkeley, California 94720

Abstract

We have developed an approach to natural language processing in which the natural language processor is viewed as a knowledge-based system whose knowledge is about the meanings of the utterances of its language. The approach is oriented around the phrase rather than the word as the basic unit. We believe that this paradigm for language processing not only extends the capabilities of natural language systems, but handles those tasks that previous systems could perform in a more systematic and extensible manner.

We have constructed a natural language analysis program called PHRAN (PHRasal ANalyzer) based in this approach. PHRAN reads texts in English and produces structures representing their meanings. PHRAN's knowledge of English is not confined to the word level. Instead, the system has knowledge about language constructs of varying levels of specificity, from canned literal phrases to general verb-oriented patterns. Associated with each language pattern is a corresponding meaning component. As PHRAN reads a sentence, it searches its data base for the language patterns that best interpret the incoming text. Then the associated meaning components of those patterns are used to create a meaning representation for that utterance.

This model has a number of advantages over existing systems, including the ability to understand a wider variety of language utterances, increased processing speed in some cases, a clear separation of control structure from data structure, a knowledge base that could be shared by a language production mechanism, greater ease of extensibility, and the ability to store some useful forms of knowledge that cannot readily be added to other systems.

---

*Research sponsored in part by the Office of Naval Research under contract N00014-80-C-0732.

Table of Contents

1.0 INTRODUCTION

Most intelligent behavior seems to require large amounts of world knowledge. Thus the "knowledge-based systems" approach has recently emerged as a central theme in Artificial Intelligence. For a system to perform a task intelligently, the researcher must first be able to specify the kinds of knowledge that the system must have. A way of representing the knowledge must be found. Then the knowledge must be organized so that it can be applied to the task at hand.

The need to cope with large quantities of knowledge has led to the emergence of "knowledge engineering" issues. It is desirable in practice for a system to be robust, modular, extensible, and easy to modify. This is rare enough for any sizable computer system, much more so for extremely complex intelligent programs with large and ornery data-bases. A good deal of research interest has therefore been paid to the problem of designing knowledge-based systems that are amenable both to the user and the researcher.

The problem of constructing a natural language processing system may be viewed as a problem of constructing a knowledge-based system. From this orientation, the questions to ask are the following: What sort of knowledge does a system need about a language in order to understand the meaning of an utterance or to produce an utterance in that language? How can this knowledge about one's language best be represented, organized and utilized? Can these tasks be achieved so that the resulting system is easy to add to and modify? Moreover, will the system be a good model of a human language user?

Existing natural language processing systems vary considerably in the kinds of knowledge about language they possess, as well as in how this knowledge is represented, organized and utilized. However, most of these systems are based on ideas about language that do not come to grips with the fact that a natural language processor needs a great deal of knowledge about the meaning of its language's utterances. As a result, the knowledge possessed by most natural language processing systems has not been subjected to the sort of analysis that the knowledge of other knowledge-based systems has undergone.

For example, most of the knowledge used to determine the meaning of an utterance in a language should be usable by a system that wishes to express that meaning in the language. However, most systems encode most of their language knowledge implicitly in procedures so that this knowledge must be duplicated if it is to be used elsewhere. The exception to this in some systems is their syntactic knowledge, which is often encoded declaratively in the form of a grammar. But as we shall argue, this constitutes a miniscule portion of the knowledge about language needed by a system.

Part of the problem is that most current natural language systems assume that the meaning of a natural language utterance can be computed as a function of the constituents of the utterance. The basic constituents of utterances are assumed to be words, and all the knowledge the system has about its language is stored at the word level. However, many natural language utterances have interpretations that cannot be found by examining their components. Idioms, canned phrases, lexical collocations, and structural formulas are instances of large classes of language utterances whose interpretation require knowledge about the entire phrase independent of its individual words. In addition, some language utterances that are interpretable in terms of their constituents seem nevertheless to be stored as phrases. Stored along with these phrases are associations to contexts in which the phrase is commonly found or is appropriate to use.

Attaching all knowledge in a system at the word level is restrictive because it causes processing issues and representational issues to be merged together. Thus the need to represent some knowledge about the language may force a decision about processing strategies that need not be correct. Control structure information must be included along with the knowledge about the meaning of each word, making the addition of new information tedious and difficult. Moreover, since the control structure of understanding and production are very different, knowledge about the meaning of an utterance must be stored once in a form amenable to the understanding mechanism, and again in a form amenable to the production mechanism.

We propose as an alternative a model of language use that comes from viewing language processing systems as knowledge-based systems that require the representation and organization of large amounts of knowledge about what the utterances of a language mean. This model has the following properties:

1. It has knowledge about the meaning of the words of the language, but in addition, much of the system's knowledge is about the meaning of larger forms of utterances.

2. This knowledge is stored in the form of pattern-concept pairs. A pattern is a phrasal construct of varying degrees of specificity. A concept is a notation that represents the meaning of the phrase. Together, this pair associates different forms of utterances with their meanings.

3. The knowledge about language contained in the system is kept separate from the processing strategies that apply this knowledge to the understanding and production tasks.

4. The understanding component matches incoming utterances against known patterns, and then uses the concepts associated with the matched patterns to represent the utterance's meaning.

5. The production component expresses itself by looking for concepts in the data base that match the concept it wishes to express. The phrasal patterns associated with these concepts are used to generate the natural language utterance.

6. The data-base of pattern-concept pairs is shared by both the understanding mechanism and the mechanism of language production.

7. Other associations besides meanings may be kept along with a phrase. For example, a description of the situations in which the phrase is appropriate or most commonly used may be stored, as might a person strongly associated with the phrase. This information could be used in understanding to infer facts about the situation in which the utterance is used that could not be inferred just from its meaning. Alternatively, this same information might be used in producing an utterance appropriate to the context in which the speaker finds himself.

2.0 PHRAN

PHRAN (PHRasal ANalyzer) is an English language understanding system based on this model of language use. PHRAN reads a text and produces structures that represent its meaning. These meaning representations may then be passed to other programs that may reason about them or perform other understanding functions. PHRAN currently serves as the natural language front end to PAM (Plan Applier Mechanism), a story understanding program that uses PHRAN's meaning structures as input into its inference generating processes (Wilensky 1978).

PHRAN integrates both productive and non-productive language abilities to provide a relatively flexible and extensible natural language understanding facility. While PHRAN does have knowledge about individual words, it is not limited to such knowledge, nor is its processing capability constrained by a word-based bias.

Here are some examples of sentences PHRAN can understand:

Oilmen are encouraged by the amount of natural gas discovered in the Baltimore Canyon, an undersea trough about 100 miles off the New Jersey coast. (Newsweek, March 1980)

Tenneco, one of 39 companies engaged in drilling in the area, thinks its leased tract contains a marketable supply of gas.

Mary wanted to talk to the man who brought her son home.

The young man was told to drive quickly over to Berkeley.

John has gotten into another argument with his boss.

The man rewarded Bill with a million dollars for saving his life.

The book marker wanted by your mother is in the red box.

Willa's best friend is a bum who lives in madison square.

If John gives Bill the big apple then Bill won't be hungry.

The school bus was driven by Mary's friend to The Big Apple.

John has kicked the bucket.

John kicked the red bucket.

The bucket was kicked by John.

The old French man's brother picked the book up.

If Mary brings John we will go to a Chinese restaurant.

Willa gives me a headache.

At the center of PHRAN is a knowledge base of phrasal patterns. A phrasal pattern is a description of an utterance that may be at many different levels of abstraction. For example, it may be a literal string such as "so's your old man"; it may be pattern with some flexibility such as "<nationality> restaurant", or "<person> <kick> the bucket"; or it may be a very general phrase such as "<person> <give> <person> <object>".

Associated with each phrasal pattern is a conceptual template. A conceptual template is a piece of meaning representation with possible references to pieces of the associated phrasal pattern. Each phrasal pattern-conceptual template association encodes one piece of knowledge about the semantics of the language. For example, associated with the phrasal pattern "<nationality> restaurant" is the conceptual template denoting a restaurant that serves <nationality> type food; associated with the phrasal pattern "<person1> <give> <person2> <object>" is the conceptual template that denotes a transfer of possession by <person1> of <object> to <person2> from <person1>.

The understanding process reads the input text and tries to find the phrasal patterns that apply to it. As it reads more of the text it may eliminate some possible patterns and suggest new ones. At some point it may recognize the completion of one or more patterns in the text. It may then have to chose among possible conflicting patterns. Finally, the conceptual template associated with the desired pattern is used to generate the structure denoting the meaning of the utterance.

2.1 Advantages Of PHRAN

In PHRAN, both productive and non-productive language is handled by a single mechanism. Knowledge about non-productive or phrasal language is encoded in the same type of pattern as is knowledge about productive forms - the patterns are simply of different levels of specificity. Thus the system handles phrasal language constructs as well as the more productive constructs in a uniform and unobtrusive manner.

Also, the system addresses "knowledge engineering" problems in that it has a clean separation of knowledge about language meaning from language processing strategies: The language knowledge is kept in the phrase-concept pairs, while the processing knowledge is embedded in the understanding mechanism.

This separation has several advantages.

1. The knowledge base of knowledge about the language is declarative, and thus is potentially sharable by a language production mechanism (Such a mechanism is currently under construction).

2. Since the control structure of the mechanism is separate from the knowledge base, representational and processing issues do not compound one another. Language knowledge can be represented independently from how it may be accessed by a given routine.

3. It is relatively easy to add new information to the system. Since control structure and data are now well separated, it is only necessary to enter a new phrase-concept pair into the data-base. No new routines need be constructed, nor does the interaction of these routines with previous routines need be considered.

4. Since the knowledge base is potentially usable by a language production mechanism as well, additions to the data-base should extend the capabilities of both components simultaneously.

5. Looking up the meaning of a "previously understood" phrase may be more economical in the long run than recomputing meanings from scratch each time.

6. Although PHRAN does not currently possess other kinds of associations besides the connection of patterns to concepts, it would not be difficult to add them to the existing declarative structures.

In addition we also feel that PHRAN is a better model of human language understanding. It has the ability to use language knowledge of various levels of specificity without treating the less productive forms as bizarre special cases. This seems more plausible to us than the hypothesis of a separate language understanding (and production) mechanism for each of the many kinds of language constructs, which we will describe in some detail below. Most other systems that bother with these forms at all deal with them as an afterthought, not as part of the essential structure of the model.

PHRAN's single store of language knowledge seems intuitively correct to us. Differences in people's understanding and production abilities could be accounted for by differences in the procedures that apply this knowledge to each task, or in how the phrasal-concept pairs are indexed for each task. In addition, people seem to chuck their processing into phrases when they speak or when they read. This might be a product of manipulating a particular phrasal unit.

The next section of this paper consists of a critique of previous approaches to natural language analysis. The problem of phrasal language constructs is discussed in particular. In the following section, these constructions and their implication for natural language processing are examined in some detail. The next section contains a lengthy description of how PHRAN works; it is compared to other approaches in the following section. Finally, there is an appendix describing how phrasal language has been treated in linguistics, and how our approach might help to address some of the issues that have raised there.

3.0 PREVIOUS APPROACHES

Systems that use natural language are concerned with the twin problems of language understanding and language production. A system with natural language capabilities must be able to determine the meaning of a natural utterance in order to understand its implications and take appropriate actions. When the system has some content it wishes to communicate to the user, it must be able to encode this content as a comprehensible natural language utterance.

There have been a number of attempts to build natural language understanding systems and natural language front ends. However, much of this work has been concerned with using natural language in severely restricted domains, or as the front end to systems which only need to process individual sentences in isolation. As a result, many of these solutions work only within such restricted domains and cannot be easily extended to new domains or more general applications.

In addition, many of these systems are able to use "procedural semantics" for the particular task at hand. For example, if the system is meant to be a front end of a data base manager, then it is usually sufficient to translate a natural language utterance into a command understandable to the data base system. However, this approach does not appear suitable if the system is meant to be a general natural language understander that will be understanding connected text or coherent conversation. In these cases, one needs to make inferences from the utterances after they have been understood. This would not appear to be possible unless the meaning of the utterance is represented declaratively in some formalism that could be manipulated by subsequent processes.

Since we are interested in the more general problem of understanding unrestricted and coherent natural language texts, we will concern ourselves here only with those systems that produce declarative meaning representations. A more general comparison of our system with other natural language processing systems appears toward the end of this paper.

Meanings in a program using declarative representations are usually expressed by a formal system, such as Conceptual Dependency (Schank, 1975) or some version of a semantic network (e.g., Simmons, 1974). Given a formalism for meaning representation, the problems of language understanding and language production may be posed as the problem of mapping into and out of a formal structure: Determining the formal structures that best represent the meaning of a language utterance is the problem of language understanding; finding a natural language utterance that conveys the content of a formal structure is the problem of language production.

While there have been many attempts to build language understanding systems, there are very few language production systems. One such system is Goldman's (1974), He describes a program called BABEL that can produce a number of alternative English sentences from some meaning structures. His work was used as the basis of the natural language generation components of a number of systems. For example, the SAM system (Cullingford, 1978) uses a set of production procedures modeled after Goldman's to produce story summaries and paraphrases in several different languages, including Chinese and Spanish as well as English.

Goldman's system uses a discrimination net to select a verb that is suitable for expressing a given conceptualization. Once the verb has been chosen, a case structure (Fillmore, 1968) is used to relate noun phrases and embedded sentences to the verb. That is, the verb, once chosen, is largely responsible for determining how the rest of the conceptualization will be expressed.

Riesbeck (1974) developed a conceptual analyzer that maps English language utterances into Conceptual Dependency meaning structures. A recent incarnation of this system (Schank and Riesbeck, 1976) is called ELI (English Language Interpreter) has been extended to include additional language constructs such as complicated noun groups and certain prepositional phrases (Gershman, 1977).

Riesbeck's system (and the various systems that have descended from it) works by attaching routines to individual words. These routines are generally responsible for building pieces of a meaning representation. When a word is read by the system, the routines associated with that word are used to build up a meaning structure that eventually denotes the meaning of the entire utterance.

While our design of a language analyzer is quite different from Riesbeck's, his system has a number of general features that we wish ours to have as well. As was mentioned above, ELI produces declarative meaning representations. It also has no separate syntactic and semantic phases; instead, both kinds of processing go on at once. Moreover, whatever syntactic analysis is done occurs because it is needed in order to understand the sentence, and not for its own sake. The program is also in principle integrated into a large system that is capable of retrieving facts and making inferences. Thus analysis does not go on as an independent process, but requires interaction with memory and other intelligent components.

One of the salient features of Riesbeck's work is that some of the routines associated with words are not executed immediately. Rather, these routines wait for an expected event to happen and then take some action. For example, a routine associated with the word give might wait to see if the verb is followed by a direct object that denotes a physical object or one that denotes an action. If a physical object is spotted, a meaning representation is assembled denoting a transfer of possession (as in "John gave Mary the ball"), but if a direct object denoting an action is spotted, a structure denoting that action will be created (as in "John gave Mary a beating"). Thus Riesbeck uses word-based routines to generate expectations in order to compute the meaning of an utterance.

Many of our criticisms of existing natural language processing programs are directed at Riesbeck's system. This is not because we find his system the most objectionable, but rather, because the goals of Riesbeck's system come closest to our own. Most of the criticisms we have to make are equally applicable to other systems, as will be emphasized in the section on comparisons below.

3.1 Problems

There are both practical and theoretical difficulties inherent in the approach just described. For example, in Riesbeck's conceptual analyzer, specific understanding routines are needed for each word known to the system. Thus extending the system's vocabulary requires the creation and debugging of new code, a fairly difficult and time consuming task. In addition, these routines function only in the understanding process. The knowledge they embody is inaccessible to other mechanisms, in particular, to the production procedures. The mechanism for producing language utterances must therefore have its own source of knowledge about the meaning of the language's utterances. Even though this knowledge is already present in the language understanding routines, it must be duplicated in a form that is usable by language production procedures.

For example, the fact that the word give could mean "transfer of possession" would have to be encoded in two places: once in a language production routine that describes how to use give to express the "transfer of possession" concept, and once in a language understanding routine that tells how to assemble the "transfer of possession" meaning representation from the English language utterance containing the word give. In fact, since these are independent routines, one may be missing and the other present for any particular word. Thus the system may exhibit the strange behavior of using the word give, say, in a sentence, and then failing to comprehend the user's next utterance because it contains the unfamiliar word give in it (In practice, this scenario is in fact not uncommon).

Thus the lack of knowledge shared by the production and comprehension mechanisms poses the practical problem of having to add all this knowledge twice, and the theoretical difficulty that the model of language use shows no connection at all between these components. However, the approach involves an even more troublesome problem. This problem is based on a view of language that assumes the knowledge people have about their language exists primarily on the word level. According to this model, the language user, in its function as a language understander, combines word meanings to produce sentence meanings. As a speaker of the language, the language user uses this knowledge about the language's words to chose those words that best communicate some idea.

What is missing in this model is the fact that a great deal of the knowledge people have about their language is not found at the word level. Rather, much of a language user's knowledge about the meaning of utterances is knowledge about much larger units of speech. Such knowledge is necessary in order to understand the meaning of these utterances or to produce these utterances in the contexts in which they are appropriate. For in general, the meaning of such an utterance is not computable from its subparts. This knowledge may also shortcut the understanding and production processes, since it allows the direct interpretation of some large language units without much analysis. In addition, knowledge about these larger structures is often necessary for suggesting contexts which help to understand subsequent parts of a natural language text.

This is not to say that people do not have knowledge about the words of their language or about how to combine them; this of course is an important part of any language processing system. However, much of what goes on in language processing seems to go on at other levels. More importantly, as the processes that operate on these other levels are examined, they suggest that the processes that have been assumed to operate on the word level are actually quite different from those embodied in previous systems.

4.0 PHRASAL LANGUAGE CONSTRUCTS

In this section, we examine some of the kinds of constructs that appear to be present in English. The list is not meant to be complete. In fact, we are not claiming that there are a small number of different kinds of constructs, each with there own set of special purpose processing rules. We suspect that just the opposite is true, i.e., that there are many kinds of patterns which can be handled relatively uniformly by the proper mechanism. The purpose of this section is to demonstrate the pervasiveness and importance of the phenomenon, which we feel has unjustly been relegated to second class citizen status in linguistics and Artificial Intelligence.

By the term "phrasal language constructs" we refer to those language units of which the language user has specific knowledge. We mean to include a variety of different forms under this heading. For example, non-productive language constructs are a large class of phrasal utterances in which the meaning of the utterance is not determined by the meaning of its parts. Idioms like "kick the bucket", "by and large", "bury the hatchet", and "throw the book at" are exemplary members of this class. So are discontinuous dependencies like "look ... up" (as in "look the word up in the dictionary") and "throw ... up" (meaning regurgitate). Some lexical collocations also qualify, for example, "the Big Apple" (meaning New York City), "eye dropper", and "weak safety" (a defensive back in football who plays on the side away from the tight end).

These constructs are phrasal in that the language user must know the meaning of the construct as a whole in order to use it correctly. However, we also wish to include as phrasal language constructs those utterances whose meanings may be determinable from their parts, but which are known as wholes for some other reason. Some utterances are appropriate only in certain situations; some have strong associations to particular people, ideologies, or circumstances. For example, the phrase "expletive deleted" now conveys some additional information than it did before the transcriptions of Nixon's tapes were made public.

We also wish to include in the category of phrasal constructs particular instances of more general cases that a language user happens to know for no other reason than that they are used frequently. Thus commonly used expressions may be stored redundantly simply because it is easier to process them this way than it is to recompute them from scratch each time.

Finally, as we will argue below, the most reasonable way to talk about the meaning of "ordinary" language constructs (i.e., ones whose meanings are computed from their parts, which have no glaring restrictions on their use and no special associations, and are not previously memorized by the user) is in terms of phrasal constructs. That is, phrasal language constructs admit different degrees of variability. The constructs corresponding to our knowledge about "ordinary" language are structured and function exactly like the more restricted cases above, except that their constituents admit progressively more general categories.

4.1 Types Of Phrasal Language Constructs

We will now examine a number of common types of phrasal language constructs. There is nothing particularly special about the classes presented here in that they are all handled more or less the same way in the model we espouse. However, they demonstrate a number of kinds of common phrasal utterances and kinds of variability. They also correspond to what seem to be intuitively real categories.

Formulaic utterances constitute one class of phrasal constructs. For example, consider the sentence

How do you do?

used as a greeting. It seems most probable that a human listener does not process this utterance by synthesizing the meanings of a number of individual words, but rather, that the sentence is processed as a whole. That is, the meaning of the utterance, as well as its interpretation as a superficial greeting, are associated with the entire phrase rather than computed as a function of the phrase's constituents. Knowledge of the specific phrase is particularly important here as a literal interpretation of the meaning of the utterance will not distinguish this formulaic request from a sincere request about one's well being.

Lexical collocations are word sequences that have come to be distinctive. They are similar to formulaic utterances in that their meanings seem to be stored with the collocation rather than produced as a necessary result of the juxtaposition of words. Noun-noun compounds often function in this manner. For example, eye dropper and name dropper, have rather different semantic structures, as do bird seed and grass seed - in the first case, the object dropped is mentioned in the second but not first compound; in the second pair, the first member specifies who eats the seed, the second what it grows.

Most importantly, the meaning of each utterance does not appear to be determinable from its components via some general procedure. Otherwise, ear dropper would be readily understandable as an instrument for applying medication to an ear, and address dropper as a person who shows off by stating the addresses of impressive people at a cocktail party.

Lexical collocations function like entries in a dictionary. While there may be some semblance of a relationship of the meanings of the components to the meanings of the collocation, the meaning of the collocations cannot necessarily be derived from the meanings of its constituents. Rather, their meaning is established by a direct correspondence of this meaning to the entire phrase.

Many language forms that can be understood in terms of their components nevertheless have what appear to be arbitrary restrictions on their usage. Some collocations are restricted in terms of categories of some type. This is called "colligation." For example, "divulge" is a synonym for "disclose" and "reveal". However, it cannot be used with a "that" clause, although its two synonyms can. Thus "he revealed that he erased the tape" is acceptable, but "he divulged that he erased the tape" is bothersome to many. For further discussion of this point see the appendix.

Related to lexical collocations and formulaic speech instances of "canned phrases". For example, the phrase "for example" seems to be understood as a whole, rather than as a combination of the meaning of the word "for" and the word "example". Thus while "a case in point" means about the same thing as "an example", "for case in point" is probably not acceptable to a native speaker of English. In any case, whether or not the phrase is or is not in principle derivable from its subparts is not so much the point. Rather, it is that the generation or understanding of the phrase almost certainly does not involve such an analysis.

Becker (1975), who suggests the use of a <u>phrasal lexicon</u> for language production, finds six discernible classes of phrases. Most of these fall under the heading of canned phrases. Interestingly, many of Becker's examples are in principle understandable in terms of their components. Yet they are at the same time a distinct part of a user's knowledge about its language. For example, Becker's <u>situational utterances</u> include expressions such as "It only hurts when I laugh!" which expresses (often ironically and implying just the opposite) the unimportance of an affliction. Verbatim texts constitute another set of examples. For instance, "99 and 44/100 percent pure" is an advertising slogan that is comprehensible without any special phrasal knowledge. Nevertheless, the phrase suggests a context (and a product) that would not otherwise be associated with the phrase as a function of its individual words.

Some canned phrases that are not productive (i.e., whose meaning is not strictly a function of their parts) are sometimes related to an aspect of a situation to which they apply. For example, "in the red", meaning in debt, is related to the fact that debts are often recorded in red ink in ledgers. Similarly, "in a rut" describes a negative state of affairs, a meaning that is probably related to the literal meaning via metaphor. However, whatever metaphor it is that is responsible for this correspondence does not determine the meaning of the phrase. For example, "in a groove" is semantically close enough to the previous expression that the same metaphorical relationship should apply. However, the usual semantics associated with this colloquialism describes a positive rather than negative state of affairs. While the meaning of these phrases may be consistent with a possible meaning computable from the constituents, the meaning cannot be determined from such a computation,

(However, as Lakoff and Johnson (1979) point out, knowledge of the metaphors that motivate these phrases is crucial for certain language understanding tasks. For example, "frozen" metaphors can come alive in sentences like "They buried the hatchet, but then they dug it up again." Certainly, a mechanism beyond our phrasal understanding is needed here. Our point is simply that such a mechanism will not replace the need for specific phrasal knowledge).

Another class of canned phrase are usually called cliches. For example, "hot as hell", "sly as a fox", "dressed to kill", "strong as an ox", "patently absurd", and "blow sky high" all have become such highly stylized ways of expressing a high degree of something that they do not convey the intensity they would otherwise.

The preceding phrasal constructs are of a rather rigid construction. They admit little variation in either the words used, the morphology of these words, or the word order. They are essentially literal strings. Many idioms are also subject to this constraint. For example, "by and large", "red herring", "tit for tat", "on the other hand" admit to no variation in their use. However, many idiomatic phrasal patterns can occur in a variety of forms. For example, the meaning of an idiom like "kicked the bucket" is not computable from its components. However, it is not strictly a function of a literal string either. The subject of the sentence is free to be any noun group that designates a person, and the form of the verb kick is free to change morphologically depending on tense, mode, etc. Thus one might encounter "John kicked the bucket last week", and "The old man would have kicked the bucket by now if it hadn't been for that doctor."

It is interesting to note, as Chafe (1968) points out, that some idioms, such as "by and large" and "trip the light fantastic" would be ungrammatical if they were not idiomatic constructions. We feel that this is evidence that phrasal constructions never undergo further analysis once a pattern is found to interpret them. That is, in the understanding process, large, specific patterns are chosen in preference to smaller, more general patterns, and these are used directly to determine the meaning of the utterance. Thus "by and large" matches an identical pattern before it matches a more general pattern for the conjunction "and". The finer structure of idioms is never considered by the understander .

Discontinuous dependencies are language patterns that may contain intervening words or phrase. For example, "look up" can occur either as "John looked the word up in the dictionary" or "John looked up the word in the dictionary." These are considered to be idioms since the meaning of the phrase is not a function of a sense of "look" or "up" that occurs elsewhere. Other examples include "put on some weight" (as in "John put some weight on"), "blow off some steam" (as in "John blew some steam off"), "bear a grudge" (as in "John bears Bill a grudge") and "make one's mind up" (as in "John made his mind up to quit school"). "Put up" (as in "John put the money up for the campaign") seems to work this way, although "put up with" as in "I won't put up with this crap anymore!") does not.

Another language form that is not quite literal but not entirely understandable in terms of its parts is what Fillmore has termed a structural formula (Fillmore, 1979). While some constructs must be treated as literal strings, and some allow some morphological or syntactic freedom, structural formulas allow a class of substitutions constrained by certain considerations. Fillmore gives as examples "Someone played Something to Someone's Something", as in "She played Desdemona to my Othello", and "X in and X out", as in "Day in and day out" and "Morning in and morning out."

We would be would include some more complicated examples in this class as well. For example, consider the somewhat slang phrase "to get <blank> out", as used in the following conversation: "You must have enjoyed eating all that trout you caught." "Well, I actually got trouted out before my fishing trip was over." The meaning of the phrase is "to do so much of the activity normally associated with <blank> as to not want to do it anymore", which would probably qualify it as a complex structural formula.

Fillmore suggests that there may be a large number of such formulas, each with its own rules for semantic interpretation. If so, then a great deal of revision of current semantic theories may be in store. Since these theories suffer from the assumption that the meanings of all phrases can be determined directly from their constituents, the idea of specific rules for interpretation have tended to be ignored. However, if these in fact constitute the majority of our knowledge about the meaning of utterances, then the theories will account for very little.

4.2 More General Phrasal Constructs

Our notion of a phrasal language construct is similar to a structural formula in that it allows for classes of words in addition to variations on morphology and word order. However, we intend our concept to be more general. For example, we wish to include semantic generalizations like "<nationality> restaurant", where the object in angle brackets denotes a category of concepts. In this case, the phrase is meant to include "Chinese restaurant", "Latin American restaurant", etc. The meaning associated with this phrase denotes a restaurant that prepares food in the manner of that nationality (as opposed to one that only served people of that nationality, or was owned by someone of that nationality).

In the most general case, a phrase may express the "normal" word sense associated with a word. For example, to express the "normal" meaning of the verb kick, the phrase "<person> <kick-form> <object>" is used. This denotes some verb form involving kick (e.g., kick, kicked, would have kicked") followed by some utterance denoting an object (Since the notion for the various forms of a verb is tedious, we usually just use the verb alone for expository purposes). Associated with this phrasal pattern is a concept that denotes applying a force to an object by striking it with one's foot. Along with this association is the correspondence between the conceptual cases and the items in the phrasal pattern. In this case, the object to which the force is applied is the <object> in the phrasal pattern; the actor who does the kicking is the person denoted by the subject of the sentence.

As we have seen, phrasal constructs vary widely in how specific or general a set of utterances they correspond to. Our criterion for determining whether a set of forms should be accommodated by the same phrasal pattern is essentially a conceptual one. That is, if the meaning of the phrases is different, they should be matched by different patterns. If the surface structure of the phrases is similar and they seem to mean the same thing, then they should be accommodated by one pattern.

This criterion has a number of consequences. First, each verb has one or more phrasal patterns corresponding to it, one for each word sense. For example, give might have one pattern that will match a syntactic direct object that is a physical object, and another that matches a syntactic direct object that is an action. The conceptual component associated with the first pattern denotes a transfer of possession of the object denoted by the syntactic direct object, while that associated with the second denotes the action designated by the syntactic direct object.

Note that while there are a number of syntactic regularities across these patterns, it does not appear useful to express all this syntactic knowledge separately. This is because not all the generalizations that can be made from these forms are useful for computing a meaning. For example, "<person> <give> <object>" and "<person> <receive> <object>" have a similar form, but their associations with conceptual patterns are different. Namely, the subject of the sentence becomes the actor of the conceptualization associated with the first pattern, but becomes the recipient of the object in the second. Since the goal of the system is to express the relation of language patterns to meanings, two separate patterns are needed here. It does not appear to be useful to express this syntactic regularity, since it seems to play no role in language use.

It is in this respect that our phrasal patterns differ most from conventional grammars. Most grammars are explicitly concerned with expressing syntactic regularities. Crammar-based understanding systems usually use a grammar to generate a syntactic parse, and then use semantic routines to determine the meaning of the grammatical structure. The model we propose assumes that there is little separation of this knowledge; the patterns known to the language user stem directly from semantic considerations. Thus those syntactic generalizations that are useful may be expressed explicitly, but those that are not may only appear implicitly in the more semantically oriented patterns of the system.

We consider the status of any particular generalization, semantic or syntactic, to be an empirical question, and not a crucial part of our model. It is our belief that those forms that are known to language users are those to which they can attach meaning. For example, if the passive construct generally has the same semantic interpretation in all cases, then it would merit explicit expression in the model. However, if we find some special cases in which it functions differently, then the explicit expression of these special passive forms would simply be added to the model.

Another feature of these phrasal patterns is that they may contain optional components. For example, in the phrasal pattern for the verb <u>leave</u>, the location left may not be expressed. We might find "John left New York" as well as "John left". Likewise, for the verb "go", the "from <location>" is often omitted. Thus we find "John went to New York" and John went to New York from Boston". Sometimes the options define mutually exclusive sets. For example in the case of <u>give</u>, we find "<person> <give> <object> to <person>" and "<person> <give> <person> <object>", both of which seem to mean about the same thing.

Since the meaning of these forms is the same, it is within the spirit of our model to allow for the expression of options within a pattern rather than express each of these cases separately. For example, "<person> <give> {(<object> to <person>),(<person> <object>)}" could be used to define the pattern for <u>give</u>; "<person> <leave> {<location>,∅}" the pattern for <u>leave</u>. In some cases, the conceptual component of the pattern can describe a default to use when an optional part is not found in the sentence being analyzed.

4.3 Implication For AI Natural Language Processing

It appears that the number of structural formulas, lexical collocations, idioms, canned phrase, etc., is actually quite large, and moreover, that each one comes with its own rules for interpretation. In fact, it may be that this form of language use is just as basic as the compositional form assumed by the models discussed above. If so, then the basic form of our natural language understanding and generation programs is in need of a great deal of overhauling.

The assumption that the meaning of an utterance can be determined simply by examining its parts has had a number of adverse effects. The most important of these is the orientation of most understanding systems around the word as the basic unit. This is true both as the unit of processing and as the unit of knowledge organization. For example, Riesbeck's system (like its descendents) organizes its knowledge about individual words. This has a number of negative consequences. First, consider how some of the examples of phrasal language given above would have to be processed by such a system. Take for example the meaning of the idiom "John kicked the bucket." To find the meaning of this idiom using Riesbeck's conceptual analysis method, the verb <u>kick</u> would have to have a routine associated with it that looked forward for the word <u>bucket</u>, and which would compute the idiomatic meaning if this prediction is fulfilled. The problem, of course, is that this routine would be looking for the word <u>bucket</u> whenever the verb <u>kick</u> appears.

In general, when all knowledge is associated at the word level, phrasal utterances must be handled by a routine attached to each word in the utterance that looks forward or backward for another word of the utterance. Since it cannot be known in advance whether the phrase is to appear, these routines must be checked whenever the individual words involved appear, even if they appear in quite different contexts.

This situation seems even less intuitively plausible when one considers how certain noun phrases need to be processed. For example, Gershman (1977) discusses the understanding of the noun phrase "Chinese restaurant". (The noun phrase is problematic because it is not obvious a priori that the phrase should mean "a restaurant that serves Chinese food" rather than "a restaurant owned by a Chinese" or "a restaurant that serves Chinese customers"). Since he is working within Riesbeck's system, Gershman must either attach a routine to the word Chinese that always looks ahead to see if it is being followed by the word restaurant, or he must attach a routine to the word restaurant that always looks back to see if it has been preceeded by a word that denotes a nationality.

While this latter alternative is chosen as one that best captures the semantic generality encompassed by this phrase, this decision is forced by constraints imposed by the processing mechanism, regardless of whether or not it is the best knowledge representation solution. That is, the processing mechanism requires that knowledge be attached to individual words. However, the correct generalization about the meaning of an utterance may be in terms of elements that are more abstract than individual words. In the above case, the more abstract category nationality is needed. Since this is not a particular word, the knowledge needed to compute the meaning of the phrase has to be attached to restaurant. Thus the indexing issue of how to store and retrieve some knowledge and the representational issue of what generalization should be stored mutually constrain one another. The problem arises out of the premature commitment to index all knowledge on the word level, thus imposing some untenable constraints upon the system.

Attaching all knowledge to individual words also does not leave any place to associate situations or connotations that might be keyed in by the phrase. For example, the saying "Winning isn't everything. It's the only thing." has strong associations to a particular individual, sport and philosophy. This association is a function of the phrase itself much more so than its meaning (which is difficult to compute in any case). Since a natural language processing system must be able to make such associations, it must be provided with a level at which these associations can be stated.

Another problem with such a system is that it cannot take computational advantage of the preconditioning that goes into canned phrases. Computational understanding is a difficult and cpu-consuming process. If a system could be provided with the pre-processed meaning of a large number of canned phrases, then some of this processing time can be traded off for storage. This trade-off is advantageous for two reasons. First, it probably does not cause the total number of phrases to be stored to go up significantly. The system requires the storage of non-productive phrases anyway in order to determine their meanings, and of many productive phrases because it needs them for associations such as those described previously. Second, it seems that people really do have such a vast store of phrasal utterances and that this model is psychologically plausible. Moreover, it is consistent with the findings in many areas of AI that large quantities of knowledge are necessary for any intelligent system.

Adding information to such a system has also proven to be a cumbersome task. Organizing knowledge around individual words forces this knowledge to be encoded procedurally, as the knowledge about a word may have to specify what to do if another word is seen. Thus new procedures must be written when new words or word senses are added to the systems repertoire. Often, old definitions have to be rewritten because a new choice point is introduced into the process dealing with a previously known word. In general, entering new knowledge requires writing new code and changing old.

The procedural encoding of knowledge forced by attaching routines to individual words poses another difficulty. Since the meaning of an utterance is generated by these routines, there is no knowledge shared between the understanding system and a system for language production. That is, the knowledge about the meanings of utterances must be stored redundantly in two separate places: In routines for understanding, and again in some other routines involved in the generation process. While it is clear that language production and language understanding are in fact very different processes, it would seem strange and inefficient that they share no knowledge of the meaning of their language.

In sum, word-based systems have the following disadvantages: They impose difficulties in dealing with non-productive language constructs whose meaning is not computable from its components, and it is difficult for them to take advantage of known meanings of canned phrases. There is no place in such systems to attach associations from utterances to the contexts they suggest. Such systems require that knowledge be indexed via individual words, thus causing indexing and representational issues to mutually constrain one another. Also, the requirement that knowledge be indexed by individual words forces the encoding to be procedural. This prevents a sharing of knowledge between the generation and understanding routines, and also makes the addition of new information to the system difficult and tedious.

5.0 HOW PHRAN WORKS

Having discussed in detail some of the motivations for phrasal-based language processing, let us examine some of the details of our implementation. The basic points of this section are to show how some of the claims for our model have been realized, how we have solved some of the problems that arise once language processing issues are formulated in this model, and what problems we feel remain to be solved to our satisfaction. While our program is not complete, we feel it is far enough along to demonstrate that our model does indeed have something to offer.

5.1 Overall Algorithm

PHRAN is made up of three parts - a database of pattern-concept pairs, a set of comprehension routines, and a routine which suggests appropriate pattern-concept pairs. PHRAN takes as input an English sentence, and as it reads it from left to right, PHRAN compares the sentence against patterns from the database. Whenever a matching pattern is found, PHRAN interprets that part of the sentence that matched the pattern as describing the concept associated with the pattern in the pattern-concept pair. Periodically throughout this process, the pattern suggesting mechanism offers PHRAN patterns that enable the analysis to continue.

PHRAN's knowledge base contains many of the types of patterns described previously. They range in specificity from literal strings to sequences of concepts described in a certain order. Patterns vary in size from those that match individual words to those that match entire sentences.

As PHRAN reads an input sentence, the words appearing in the sentence, along with patterns that have already matched parts of it, drive the pattern suggesting routine in offering PHRAN patterns to be considered for matching fragments of the sentence. The patterns PHRAN considers "active" (i.e. those that have matched up to the point where PHRAN has read) are used to interpret the remaining words in the sentence by giving PHRAN a reasonable idea of what may conceivably appear later on. The process of suggesting new patterns and matching them against the input continues until the end of the sentence is reached and all suggested patterns have been tried. At this point PHRAN uses the pattern that matches the whole sentence to determine the meaning of the entire utterance.

5.1.1 Overview Of PHRAN Patterns

A pattern-concept pair consists of a specification of the phrasal unit, an associated concept, and some additional information about how the two are related. PHRAN instantiates the concept, i.e., fills in the specifics from the matched pattern, when the pattern match is successful. It is often necessary to carry around more information, however. For example, it may be important to know that a concept denoting a person came from a noun phrase. Thus when PHRAN instantiates a concept, it actually creates an item called a <u>term</u> that includes both the concept as well as some additional information.

A pattern is a sequence of conditions that must hold true for a sequence of terms. The conditions on a term may refer to lexical, syntactical, and conceptual categories. They range from the requirement that the word giving rise to the term be identical to a given word, to having the term represent a particular kind of action.

A pattern may specify optional terms too, the place where these may appear, and what effect (if any) their appearance will have on the properties of the term formed if the pattern is matched. For example, consider the following informal description of one of the patterns suggested by the mention of the verb 'to eat' in certain contexts.

```
{  pattern to recognize -
        [<first term: represents a person>

         <second term: an active form of EAT>
```

```
          <OPTIONAL third term: represents food>
         ]

   term to form -
         (INGEST (ACTOR <first term>)
                 (OBJECT <third term, if present, else FOOD>))
}
```

This pattern directs PHRAN to ask if the term preceding the verb is a person, if the term following that is a form of <u>eat</u>, and if the following term could be an item of food. If so, then the first term will be used to fill the ACTOR slot of the INGEST conceptualization, and the third term to fill the OBJECT slot. The third term is marked as optional. If it is not present in the text, PHRAN will fill the OBJECT slot with a default representing generic food.

Determining whether the verb matches is a simple lexical check, while determining whether something is a person or a food may require a call to an elaborate memory routine. Such routines are not part of PHRAN proper, but of some program PHRAN is in principle running in conjunction with. For convenience, many of these checks are simply canned into PHRAN at present.

5.1.1.1 Pattern Generation

Not all patterns PHRAN needs are explicitely present in its database. For instance, consider the pattern of the previous example,

[<person> <EAT> <food (optional)>]

This pattern will match sentences like "John ate the apple". But it seems as if the same information should be usable in sentences like "John wanted to eat the apple" as well, in conjunction with a pattern like [<person> <WANT> <EVENT>]. The problem is that here the verb appears preceded by 'to' and not a <person> as before.

PHRAN does not have each of these cases stored explicitly in its database, but rather it has only the 'basic' patterns (i.e. '<subject> <verb> <object>' type patterns) stored, and the pattern suggesting routine is able to generate a new pattern to use to match the input. PHRAN can generate patterns that recognize passive forms of verbs, various types of relative clauses, phrase of the form "<person> <do> <verb>", "<person> <do> not <verb>", "<person> <modal-verb> <verb>", etc., and phrases with infinitive forms of verbs (e. .g, "John likes to go fishing"). In all, PHRAN can generate ten types of modified patterns, although each form is not applicable to all verb-based patterns.

5.2 Processing Overview

When PHRAN analyzes a sentence, it reads the words one at a time, from left to right. It does a little morphological analysis on the sentence while it is being read, just enough to recognize contractions and "'s"s. The pattern suggesting routine determines if any new patterns should be tried, and PHRAN checks all the new patterns to see if they agree with that part of the sentence already analyzed, discarding those that don't. A word's meaning is determined simply by its matching a pattern consisting of that literal word. Then a new term is formed with the properties specified in the concept associated with the word, and this term is added to a list PHRAN maintains. PHRAN then checks if the term it just added to the list completes or extends patterns that had already been partially matched by the previous terms. If a pattern is matched completely, the terms matching that pattern are removed and a new one, specified by the concept part of the pattern-concept pair, is formed and replaces the terms the pattern matched.

When PHRAN finishes processing one word it reads the next, iterating this procedure until it reaches the end of a sentence. At this point, it should end up with a single term on its list. This term contains the conceptualization representing the meaning of the whole sentence.

5.2.1 Simple Example

The following is a highly simplified example of how PHRAN processes the sentence "John dropped out of school":

First the word "John" is read. "John" matches the pattern consisting of the literal "John", and the concept associated with this pattern causes a term to be formed that represents a noun phrase and a particular male person named John. No other patterns are suggested. This term is added to *CONCEPT*, the list of terms PHRAN keeps and which will eventually contain the meaning of the sentence. Thus *CONCEPT* looks like

< [JOHN1 - person, NP] >

"Dropped" is read next. It matches the literal "dropped", and an appropriate term is formed. The pattern suggesting routine instructs PHRAN to consider the 'basic' pattern associated with the verb 'to drop', which is:

{ [<person> <DROP> <object>] [...] }

Its initial condition is found to be satisfied by the first term in *CONCEPT* -- this fact is stored under that term, and succeeding ones will be checked to see if this partial match continues. The term that was formed after reading "dropped" is now added to the list. *CONCEPT* is now:

< [JOHN1 - person, NP] , [DROP - verb] >

PHRAN now checks to see if the pattern stored under the first term matches the term just added to *CONCEPT* too, and it does. This new fact is now stored under the last term.

Next the word "out" is read. The pattern suggestion mechanism is alerted by the occurrence of the verb 'drop' followed by the word 'out', and at this point it instructs PHRAN to consider the pattern

{ [<person> <DROP> "out" "of" <school>] [...] }

The list in *CONCEPT* is checked against this pattern to see if it matches its first two terms, and since that is the case, this fact is stored under the second term. A term associated with 'out' is now added to *CONCEPT*:

< [JOHN1 - person, NP] , [DROP - verb] , [OUT] >

The two patterns that have matched up to DROP are checked to see if the new term extends them. This is true only for the second pattern, and this fact is stored under the next term. The pattern [<person> <DROP> <object>] is ignored from this point on.

Now the word "of" is read. A term is formed and added to *CONCEPT*. The pattern that matched up to OUT is extended by OF so the pattern is moved to the next term.

The word "high" is read and a term is formed and added to *CONCEPT*. Now the pattern under OF is compared against HIGH. It doesn't satisfy the next condition. PHRAN reads "school", and the pattern suggestion routine presents PHRAN with two patterns:

```
1. { [ "high" "school" ] [ representation denoting a
                            school for 10th through 12th
                            graders] }

2. { [<adjective> <noun>] [ representation denoting
                             noun modified by adjective] }
```

Both patterns are satisfied by the previous term and this fact is stored under it. The new term is added to *CONCEPT*, now:

```
< [JOHN1 - person, NP] , [DROP - verb] , [OUT] ,
  [OF] , [HIGH - adj] , [SCHOOL - school, noun] >
```

The two patterns are compared against the last term, and both are matched. The last two terms are removed from *CONCEPT*, and the patterns under OF are checked to determine which of the two possible meanings we have should be chosen. Patterns are suggested such that the more specific ones appear first, so that the more specific interpretation will be chosen if all patterns match equally well. Only if the second meaning (i.e. a school that is high) were explicitly specified by a previous pattern, would it have been chosen.

A term is formed and added to *CONCEPT*, which now contains

```
< [JOHN1 - person, NP] , [DROP - verb] , [OUT] ,
              [OF] , [HIGH-SCHOOL1 - school, NP] >
```

The pattern under OF is checked against the last term in *CONCEPT*. PHRAN finds a complete match, so all the matched terms are removed and replaced by the concept associated with this pattern.

CONCEPT now contains this concept as the final result:

```
< [ ($SCHOOLING (STUDENT JOHN1)
                (SCHOOL HIGH-SCHOOL1)
                (TERMINATION PREMATURE)) ] >
```

5.3 Pattern-Concept Pairs In More Detail

5.3.1 The Pattern

The pattern portion of a pattern-concept pair consists of a sequence of predicates. These may take one of several forms:

1. A word; which will match only a term representing this exact word.

2. A class name (in parentheses); will match any term representing a member of this class (e.g. "(FOOD)" or "(PHYSICAL-OBJECT)").

3. A pair, the first element of which is a property name and the second is a value; will match any term having the required value of the property (e.g. "(Part-Of-Speech VERB)").

In addition, we may negate a condition or specify that a conjunction or disjunction of several must hold.

The following is one of the patterns which may be suggested by the occurrence of the verb 'give' in an utterance:

```
[(PERSON) (ROOT GIVE) (PERSON) (PHYSOB)]
```

Each condition aside from the second is a call to memory to determine if the object denoted by the term belongs to the specified category.

5.3.1.1 Optional Parts

Not all terms a pattern calls for are required to be present for PHRAN to consider the pattern successfully matched. To indicate the presence of such optional terms, a list of pattern-concept pairs is inserted into the pattern at the appropriate place. These pairs have as their first element a sub-pattern that will match the optional terms. The second part describes how the new term to be formed if the main pattern is found should be modified to reflect the existence of the optional sub-pattern.

The concept corresponding to the optional part of a pattern is treated in a form slightly different from the way we treat regular concept parts of pattern-concept pairs. As usual, it consists of pairs of expressions. The first of each pair will be placed as is at the end of the properties of the term to be formed, and the second will be evaluated first and then placed on that list, which will be described in the next section.

For example, another pattern suggested when 'give' is seen is the following:

```
[(PERSON) (ROOT GIVE) (PHYSOB) ([(TO (PERSON))
                                 (TO (OPT-VAL 2 CD-FORM))])]
```

The terms of this pattern describe a person, the verb give, and then some physical object. The last term describes the optional terms, consisting of the word to followed by a person description. Associated with this pattern is a concept part that specifies what to do with the optional part if it is there. Here it specifies that the second term in the optional pattern should fill in the TO slot in the conceptualization associated with the whole pattern. This associated conceptualization is not shown here, but will be described below.

It should be noted that the particular pattern shown above need not be a separate pattern in PHRAN from the one that looks for the verb followed by the recipient followed by the object transferred. We often show patterns without all the alternatives that are possible for expositional purposes. Sometimes it is simpler to write the actual patterns separately, although we attach no theoretical significance to this disposition.

5.3.2 The Concept

When a pattern is matched, PHRAN removes the terms that match it from *CONCEPT* and replaces them with a new term, as defined by the second part of the pattern-concept pair. For example, here is a complete pattern that may be suggested when the verb 'eat' is encountered:

```
([(PERSON) (ROOT EAT) ([((FOOD))
                        (FOOD (OPT-VAL 1 CD-FORM))])]

 [P-O-S 'SENTENCE
  CD-FORM '(INGEST (ACTOR ?ACTOR) (OBJECT ?FOOD))
  ACTOR (VALUE 1 CD-FORM)
  FOOD 'FOOD)])
```

The concept portion of this pair describes a term covering an entire sentence, and whose meaning is the action of INGESTing some food. The next two descriptors specify how to fill in variable parts of this action. These begin with '?'. The expression (VALUE n prop) specifies the 'prop' property of the n'th term in the matched sequence of the pattern (not including optional terms). OPT-VAL does the same thing with regards to a matched optional sub-pattern. Thus the concept description above specifies that the actor of the action is to be the term matching the first condition. The object eaten will be either the default concept FOOD, or, if the optional sub-pattern was found, the term corresponding to this sub-pattern.

Some parts of a conceptualization may be left empty, to be filled at a later point; a slot in the conceptualization can be filled by a term in a higher level pattern of which this one is an element. For example, when analyzing "John wanted to eat a cupcake" a slight modification of the previous pattern is used to find the meaning of "to eat a cupcake". Since no subject appears in this form, the higher level pattern specifies where it may find it. That is, a pattern associated with "want" looks like the following:

```
{ [<person> <WANT> <infinitive>]
  [infinitive-subject (VALUE 1 CD-FORM)
                  .....]  }
```

This specifies that the subject of the clause following <u>want</u> is the same as the subject of <u>want</u>.

5.4 Extended Notions Of Pattern And Concept

Phrasal units involving adverbs, relative clauses, and prepositional phrases are recognized through the use of patterns in the manner just described. However, these forms differ from those we have seen so far both in how they may appear in an utterance and in their associated semantic function. We therefore require a slightly modified notion of a pattern-concept pair in order to handle them.

The semantic function associated with the patterns previously encountered is to denote a concept. The semantic function of an adverb or the like is to modify some existing concept rather than simply denote one of its own. We therefore refer to the former patterns of speech as concept builders, and the latter as concept modifiers.

When a pattern matches a concept builder, the concept template associated with the pattern is used to create a structure denoting a specific instance of that concept. The resulting concept may match a constituent of another, higher-level pattern, and eventually become part of the conceptual structure that that pattern is responsible for denoting. For example, "home run" is a concept builder that denotes a certain kind of baseball feat. If it is matched in a sentence, a structure is built denoting this idea. If the phrase occurred in the sentence "John hit a home run", the home run structure will match a constituent of a pattern for hit, and thus end up as part of a concept denoting something John did.

Concept modifiers do not work this way. For example, consider the sentence "John ran quickly." If quickly just built a structure denoting a faster than normal activity, then the run pattern would have to have a constituent that matches it in order for the structure to find its way into the structure that the run pattern builds. The problem is that concept modifiers may occur in too many places in too many different phrasal patterns. For example, quickly may appear in any position in any phrase of the form "<person> <transitive verb> ... ": "John ate quickly", "Quickly, John left", "John quickly spoke up", etc. It would therefore be unwise to require that patterns matching modifiers explicitly appear as optional constituents in all the places in all patterns in which they might appear. This would be unmanagable and would fail to capture a useful generalization.

To handle concept modifiers, therefore, an extended notion of a pattern-concept pair is used. Rather than specify what to build, the concept part associated with a concept modifier pattern specifies where to look for a concept to modify, and how to modify it. The pattern parts are matched just like any other pattern. But once it has been determined that a pattern has been matched, PHRAN uses the associated concept part to find and change a structure created from some other pattern-concept pair.

5.4.1 Adverbs And Adverbial Phrases

Adverbs are generally concept modifiers, and do not as a rule appear as constituents in patterns. Instead, upon recognizing an adverb, PHRAN is instructed to search within the active patterns for an action that it can modify. When such an action is found the concept part of the pair associated with the adverb is used to modify the concept of the original action. An adverb will be used to modify the next appropriate term seen if a modifiable one has not occurred yet. In this manner PHRAN is able to correctly interpret adverbs appearing in a variety of places within a pattern.

For example, the adverbs "slowly" and "quickly" operate in this way, and PHRAN can handle them in constructs like the following:

John ate slowly.

Quickly, John left the house.

John left the house quickly.

John slowly ate the apple.

John wanted slowly to eat the apple.

Some special cases of negation are handled by specific patterns. For example, the negation of the verb want is usually interpreted as meaning "want not" - "Mary didn't want to go to school" means the same thing as "Mary wanted not to go to school". Thus PHRAN contains the specific pattern [<person> <do> "not" <want> <inf-phrase>] which is associated with this interpretation.

In general, most of the difficulty in handling adverbs is representational rather than procedural - PHRAN can figure out what the adverb is modifying, but it is often difficult to describe how the appearance of the adverb affects the meaning of the sentence.

5.4.2 Relative Clauses And Prepositional Phrases

When analyzing utterances containing relative clauses, PHRAN must also search for the term which the clause refers to. PHRAN must find the appropriate term preceeding the clause, a task which requires consulting a special list of the objects mentioned in the sentence. While reading a sentence, whenever a new object or person is recognized, PHRAN adds it to a list it maintains. Then, when the need arises, PHRAN can quickly determine whether a term referred to actually exists.

For example, when processing the sentence "John saw the man who was walking towards him" PHRAN first processes "John saw the man" and forms a term denoting its meaning. When the relative clause "who was walking towards him" is processed, PHRAN must find the person that the clause refers to. This is done by checking the last term on the above mentioned list to see if it indeed represents a person.

Following is an example of a pattern-concept pair of the previous type. It is suggested when an utterance such as "the man who saw the gun" is encountered. If the required referent is not found when this pattern is suggested, the pattern is immediately discarded.

```
{   [ (*REFERENT IMMEDIATELY-PRECEEDING (PERSON))
       WHO (ROOT SEE) (PHYSOB) ]

    [ P-O-S 'REL-CLAUSE
      CD-FORM '(ATTEND (ACTOR ?ACTOR) (OBJECT EYES) (TO ?TO))
      ACTOR (VALUE 1 CD-FORM)
      TO (VALUE 4 CD-FORM)]   }
```

The previous pattern-concept pair does not exist explicitely in the system. Instead, when the pattern suggesting mechanism recognizes the sequence "<person> who <verb>", PHRAN generates a pattern-concept pair of this type, from the "basic" patterns associated with the verb.

Prepositional phrases often appear as constituents in larger phrasal patterns, such as the one associated with the verb "to give" that was seen earlier. When prepositional phrases appear and they are not part of another pattern, PHRAN is capable of processing them in a manner similar to that in which it deals with relative clauses.

An example of such a phrase is the following:

```
{  [ (*REFERENT IMMEDIATELY-PRECEEDING (PHYSOP)) IN (CONTAINER) ]

   [ P-O-S 'PREP-PHRASE
     CD-FORM '(CONTAINS (CONTAINER ?CONTAINER) (OBJECT ?OBJECT))
     CONTAINER (VALUE 3 CD-FORM)
     OBJECT (VALUE 1 CD-FORM) ]   }
```

Phrases of the types described in this section may refer to either an <u>immediately</u> <u>preceeding</u> term or to the <u>most</u> <u>recent</u> term of a certain type encountered. A term referred to in this manner will be searched for only within the set of the active patterns.

5.5 Pattern Manipulation In More Detail

5.5.1 Reading A Word

After a new word is read, PHRAN receives a list of relevant patterns from the pattern suggesting mechanism. These patterns include patterns that are used to interpret the word just read -- since words in PHRAN are just small patterns and like any other patterns they may be associated with a concept. PHRAN compares the suggested patterns with the list *CONCEPT* discarding those that conflict with it and retaining the patterns that match the list up to the point PHRAN has read. The partially matched patterns are stored in a 'pattern-list' associated with the last term in *CONCEPT*. Now PHRAN uses the associated concepts in the pattern-concept pairs whose pattern consists of the literal word to construct a new term, denoting the word's meaning. If there is more than one associated concept (i.e. more than one possible meaning) PHRAN tries to determine which meaning is appropriate in the current context, using the "active" patterns (those that have matched up to the point where PHRAN has read). It checks if there is a particular meaning that will match the next slot in some pattern or, if no such definition exists, if there is a meaning that might be the beginning of a sequence of terms whose meaning, as determined via a pattern-concept pair, will satisfy the next slot in one of the active patterns. If this is the case, that meaning of the word is chosen. Otherwise PHRAN defaults to the first of the meanings of the word.

A new term is formed and if it satisfies the next condition in one of the active patterns, the appropriate pattern is moved to the pattern-list of the new term. If the next condition in a pattern indicates that the term specified is optional, then PHRAN checks for the optional terms, and if it is convinced that they are not present, it checks to see if the new term satisfies the condition following the optional ones in the pattern. If, at a later time, there is an indication that the decision that the optional terms were not present may have been erroneous, PHRAN will undo some of its processing and search for them again.

5.5.2 A Pattern Is Matched

At some point in this process PHRAN will check an incoming term against the active pattern-list and will find that, not only does the new term satisfy the next condition of some pattern, but that the condition is the last one specified. When a pattern has matched completely, PHRAN still continues checking all the other patterns on the pattern-list. When it has finished, PHRAN will take the longest pattern that was matched and will consider the concept of its pattern-concept pair to be the meaning of the sequence. If there are several patterns of the same length that were matched PHRAN will group all their meanings together. New patterns are suggested and a disambiguation process follows, exactly as in the case of a new word being read.

For example, the words "the big apple", when recognized, will have two possible meanings; one being a large fruit, the other being New York City. PHRAN will check the patterns active at that time to determine if one of these two meanings satisfies the next condition in one of the patterns. If so, then that meaning will be chosen. Otherwise 'a large fruit' will be the default, as it is the first in the list of possible meanings.

When a meaning is chosen, PHRAN replaces the terms that matched the pattern with a term denoting this meaning.

5.6 Indexing And Pattern Suggestion

Retrieving the phrasal pattern matching a particular utterance from PHRAN's knowledge base is an important problem that we have not yet solved to our complete satisfaction. We find some consolation in the fact that the problem of indexing a large data base is a necessary and familiar problem for all knowledge based systems.

We have tried two pattern suggestion mechanisms with PHRAN:

1. Keying patterns off individual words or previously matched patterns.

2. Indexing patterns under ordered sequences of cues gotten from the sentence and phrasal patterns recognized in it.

The first indexing mechanism works, but it requires that any pattern used to recognize a phrasal expressions be suggested by some word in it. This is unacceptable because it will cause the pattern to be suggested whenever the word it is triggered by is mentioned. The difficulties inherent in such an indexing scheme can be appreciated by considering which word in the phrase "by and large" should be used to trigger it. Any choice we make will cause the pattern to be suggested very often in contexts when it is not appropriate. In this form, PHRAN's processing roughly resembles ELI's (Riesbeck et al, 1975).

We therefore developed the second mechanism. The patterns-concept pairs of the database are indexed in a tree. As words are read, the pattern suggesting mechanism travels down this tree, choosing branches according to the meanings of the words and of larger units that have been recognized. It suggests to PHRAN the patterns found at the nodes it has arrived at. The list of nodes is remembered, and when the next word is read the routine continues to branch from them, in addition to starting from the root. In practice, the number of nodes in the list is rather small.

For example, whenever a noun-phrase is followed by an active form of some verb, the suggesting routine instructs PHRAN to consider the simple declarative forms of the verb. When a noun-phrase is followed by the verb 'to be' followed by the perfective form of some verb, the routine instructs PHRAN to consider the passive uses of the last verb. The phrasal pattern that will recognize the expression "by and large" is found at the node reached only after seeing those three words consecutively. In this manner this pattern will be suggested only when necessary.

This mechanism is designed so that it also suggests patterns that would match the sentence under a re-interpretation of some of the words already read. In other words, PHRAN may become aware, at certain points, that a different interpretation of fragments of the sentence analayzed previously will lead to a different meaning being given to a larger segment of the text. It should be possible for PHRAN to decide, when such a situation arises, whether to re-analyze this part of the sentence. However, such a procedure has not yet been implemented.

The main problem with this scheme is that it does not lend itself well to allowing contextual cues to influence the choice of patterns PHRAN should try. This is one area where future research will be concentrated.

5.7 A Detailed Example

Following is the detailed PHRAN analysis of the sentence

John gave Mary a piece of his mind.

This example is a fairly typical demonstration of how PHRAN works. However, it illustrates only the more basic features of the program. For example, the example contains no instances of pattern generation, relative clause or prepositional phrase attachment, or handling of complicated noun phrases. In addition, this particular example was chosen to minimize the number of patterns suggested by PHRAN that later end up being rejected.

In this example we will refer to several patterns with symbols P1, P2, .., and OPT. The correspondence between symbols and pattern-concept pairs is given at the end of the example.

Word Read	Term Formed (some of its properties)	*CONCEPT*	Patterns Suggested	Results of Checking Patterns & Comments
JOHN	T1 (JOHN1,person noun-phrase)	(T1)	none	
GAVE	T2 (root give, verb)	(T1,T2)	P1,P2,P3	all are consistent with T1,T2.

** In the pattern suggestion tree, PHRAN is on a branch that will
** eventually suggest the correct pattern to interpret the
** sentence (P8). It will only be suggested after more information
** is available, however.

Word Read	Term Formed (some of its properties)	*CONCEPT*	Patterns Suggested	Results of Checking Patterns & Comments
MARY	T3 (MARY1,person noun-phrase)	(T1,T2,T3)	none	P1,P3 satisfied by T3. P2 is not, and is left on T2's pattern-list.
A	T4 (article,	(T1,T2,T3, T4)	P4	P4 consistent with *CONCEPT*. P1,P3

	indefinite)			left on T3.
PIECE	T5 (measure)	(T1,T2,T3 T4,T5)	P5	P4 fails. PHRAN now expects, e.g., "a piece of rock"
OF	T6 (of, pre- position)	(T1,T2,T3, T4,T5,T6)	OPT (optional sub-pattern from P5)	OPT satisfied, in part, by T6.
HIS	T7 (possessive pronoun)	(T1,T2,T3, T4,T5,T6, T7)	P6,P7	both ok on T7.

** Up to this point PHRAN believes (i.e. has a main active pattern
** whose associated concept indicates) that the sentence is something
** of the form "person gave person some object". During the course
** of analyzing the input PHRAN's suggesion mechanism has been moving
** down the nodes of the indexing tree. A particular pattern will be
** triggered after reading the next word that will supply PHRAN with
** the ^&real\& meaning of the utterance.

MIND	T8 (organ, mind,noun)	(T1,T2,T3, T4,T5,T6, T7,T8)	P8	P7,P8 are both matched completely. P8 is longer and so it dominates. All terms matching P8 are removed and replaced by a new one, T9.
---	T9 (sentence, MTRANS)	(T9)	none	

** The end marker is seen and there are no further patterns to
** consider, so PHRAN's work is complete.

PHRAN now outputs information it has collected about objects mentioned in the sentence -

```
((PERSON (OBJECT JOHN1))
 (PERSON (OBJECT MARY1)))
```

And finally it outputs its representation for the complete utterance -

```
(MTRANS (ACTOR JOHN1)
        (MOBJECT
           (CAUSATION (ANTECEDENT (DO (ACTOR MARY1)))
                      (CONSEQUENT (STATE-CHANGE
                                         (ACTOR JOHN1)
                                         (STATE-NAME ANGER)
```

```
                                        (TO (LEVEL 7))))))
        (FROM JOHN1)
        (TO MARY1))
```

The following are patterns and the optional sub-pattern used in the example. The 'concept' parts of the pattern-concept pairs are omitted when they are deemed unnecessary or clear from the context.

Patterns:

```
P1:     { [ (PERSON) (ROOT GIVE) (PERSON) (PHYSOB) ]

          [ P-O-S 'SENTENCE
            CD-FORM '(ATRANS (ACTOR ?ACTOR) (OBJECT ?OBJECT)
                             (FROM ?FROM) (TO ?TO))
            ACTOR (VALUE 1 CD-FORM)
            OBJECT (VALUE 4 CD-FORM)
            FROM '?ACTOR
            TO (VALUE 3 CD-FORM) ] }


P2:     { [ (PERSON) (ROOT GIVE) (PHYSOB)
                     ([(TO (PERSON))
                       (TO  (OPT-VAL 2 CD-FORM)]) ]

          [ .... ] }


P3:     { [ (PERSON) (ROOT GIVE) (PERSON) (ACTION) ]

          [ P-O-S 'SENTENCE
            CD-FORM (VALUE 4 ACT)   ~ this will have slots to be
            ACTOR (VALUE 1 CD-FORM)       ~ filled by ACTOR and
            TO (VALUE 3) ] }              ~ TO.


P4:     { [ A (P-O-S NOUN) ]

          [ .... ] }


P5:     { [ A PIECE ([(OF (PHYSOB))
                      (SUBSTANCE (OPT-VAL 2 CD-FORM))]) ]

          [ P-O-S 'NOUN-PHRASE
            CD-FORM (NEWSYM PIECE)
            DO (ADD-TO-*SUPPLEMENTARY-CONCEPTS*
                 `(AMOUNT-OF (SUBSTANCE ?SUBSTANCE)
                             (OBJECT @(OLDSYM PIECE)))) ] }


P6:     { [ (P-O-S POSSESSIVE-PRONOUN) (PHYSOB) ]
```

```
          [ P-O-S 'NOUN-PHRASE
            DESCRIPTION (VALUE 2 DESCRIPTION)
            CD-FORM (VALUE 2 CD-FORM)
            DO (ADD-RELATION-TO-*SUPPLEMENTARY-CONCEPTS*
                 'POSS (VALUE 1 PERSON) (VALUE 2 CD-FORM)) ] }

P7:       { [ (P-O-S POSSESSIVE-PRONOUN) (OR (ORGAN) (LIMB)) ]

          [ P-O-S 'NOUN-PHRASE
            DESCRIPTION (VALUE 2 DESCRIPTION)
            CD-FORM (VALUE 2 CD-FORM)
            DO (ADD-RELATION-TO-*SUPPLEMENTARY-CONCEPTS*
                 'PART-OF (VALUE 1 PERSON) (VALUE 2 CD-FORM)) ] }

P8:       { [ (PERSON) (ROOT GIVE) (PERSON) A PIECE OF
                        (P-O-S POSSESSIVE-PRONOUN) MIND ]

          [ P-O-S 'SENTENCE
            CD-FORM '(MTRANS (ACTOR ?ACTOR) (MOBJECT ?MOBJECT)
                             (FROM ?ACTOR) (TO ?TO))
            ACTOR (VALUE 1 CD-FORM)
            TO (VALUE 3 CD-FORM)
            MOBJECT '(CAUSATION
                        (ANTECEDENT (DO (ACTOR ?TO)))
                        (CONSEQUENT (STATE-CHANGE
                                        (ACTOR ?ACTOR)
                                        (STATE-NAME ANGER)
                                        (TO (LEVEL 7))))) ] }
```

<u>Optional</u> <u>Sub-Pattern</u>:

```
OPT:      { [ OPTION       ~ in P5
              OF (PHYSOB) ]

            [ .... ] }
```

5.8 Technical Data

In its current state PHRAN knows about 305 "basic" patterns of varying length and abstraction. About 170 of these patterns are individual words. Of these, about 40 are verbs. PHRAN knows both the roots of these verbs, as well as all the morphological variations in which each verb may be found. Of the 135 patterns containing more than one word, about 60 are verb based, i. e., they indicate the way a particular verb is used. This latter group of patterns can be used by the program, when the need arises, to generate an additional 400 patterns, approximately.

PHRAN is written in UCI-LISP and runs on a DEC-20/40 at the University of California at Berkeley. When loaded into a LISP core, the program takes up 9K words of memory, and the database of patterns takes up an additional 18K words. The total core image, including the UCI-LISP interpreter, is 60K, plus free storage.

Sentences of the type described in this paper take from 3 to 15 seconds of CPU time to analyze, and the processing uses up 500 - 2500 words from free storage and 20 - 105 words from the free word list. A typical sentence will be analyzed in 6 seconds, using 1100 words of free storage and 40 from the free word list. PHRAN is currently interpreted code, and no attempt has been made to produce as efficient a program as possible. The usual speed-up gained through compiling is about 8-10 to 1, although storage generally remains about the same. In addition, the DEC 20/40 is reported to have about a third the through-put of a DEC 20/60. We therefore find these statistics to be reasonably encouraging.

Adding knowledge of more words to PHRAN will not increase processing time, since words and concepts that are not mentioned in a sentence have no bearing on its processing. What will cause an increase in processing time is an increase in the number of patterns suggested when the same words or concepts are encountered. Such an increase will cause PHRAN to have to consider more patterns and will require more time to be spent checking those patterns against the sentence being analyzed. The amount of additional time spent may not be very significant, however, because most of the irrelevent patterns will be ruled out by the time only a few more words are read. For example, whether the system knows about 1 or 4 basic patterns associated with "give" makes only a 12% difference in the time spent on analyzing the sentence "John told Bill to give the book to Mary". Of course, this will make no difference in the time needed to analyze a sentence not containing any mention of "give".

5.9 Other Versions Of PHRAN

An alternative version of PHRAN is being implemented by Fred Mueller using the INGRES data base system. INGRES is a data base management system based on the relational model, and was develop by Held, Stonebraker, and Wong (1975). It runs on top of the UNIX operating system on a VAX 11/780 at Berkeley.

Since a great deal of effort was put into separating PHRAN's knowledge of language from its procedural knowledge, it is possible to move PHRAN's entire pattern-concept knowledge base out of LISP and into the relational structures provided by INGRES. The rest of PHRAN is still written in LISP, using Franz LISP on the VAX, although this may be re-written in C in the future. In either case, accessing the appropriate patterns is just a matter of making a data base query to INGRES. Some of PHRAN's processing have been changed somewhat to accomodate and take advantage of INGRES' searching and querying facilities. For example, the INGRES version tends to suggests more patterns at fewer points than does the standard LISP implementation.

A great deal of effort has been put into INGRES to assure efficient searching and querying of large volumes of data in real time. However, it is usually used to manage data of a much different sort than is found in PHRAN, or most other AI programs, for that matter. It will therefore be interesting to see how this sort of system compares in both speed and ease of expression to a more convenional LISP implementation. The INGRES version is not yet at the point where a meaningful comparison can be made. However, we believe that this is the first time that a data base system has been used as a back-end to a nature language processing program.

5.10 Some Comments And Speculations

5.10.1 Principles Of Language Comprehension

While the kind and number of language patterns PHRAN must have do not appear to have interesting constraints, there are a number of generalizations about how these patterns are used for comprehension that may be significant. The following is a list of some of these principles. These principles are heuristic in nature. That is, we do not mean that each can never be violated, but rather, than they contribute to an interpretation, together with contextual and semantic factors. They emerge as principles <u>ceteris paribus</u>.

1. More specific patterns are preferred over less specific ones.

For example, idiomatic patterns like "kick the bucket" are more specific than patterns like "kick <object>", so the idiomatic pattern is chosen over the pattern that would lead to a literal interpretation. This rule is manifest in PHRAN, via the indexing mechanism, which will generally suggest more general patterns before more specific ones, as it must wait for more words to be read before the specific pattern can be suggested. Thus patterns suggested later are given priority over those suggested earlier.

This behavior appears to be part of a general knowledge application principle that is not language specific, namely "always use as specific a piece of knowledge as you have for any given situation."

2. Longer patterns are preferred over shorter ones.

For example, the sentence "John gave up the hill" could be interpreted as meaning either that John surrended the hill or that John did some act of giving toward the top of the hill. In the first case, the pattern matched is "<person> <give> up <object>", while in the second, one pattern is required for "<person> <give> {<object>}" (i. e., where the optional object is omitted in the utterance), and another for the prepositional phrase "up <location>". Ceteris paribus, this principle states that the former interpretation will be chosen over the latter as the former uses fewer patterns.

3. Optional parts of a pattern are preferred over new patterns.

For example, consider the sentence "John arrived in New York". The correct interpretation of this sentence is that John arrived in New York from some place outside of New York. An alternative interpretation is that John went from one place in New York to another, and therefore arrived in New York (this is structurally similar to "John died in New York"). The first interpretation is reached by matching the pattern "<person> arrive {in <location>}", including matching the optional "in <location>" subpattern to "in New York". The second interpretation is reached by assuming that the optional part is missing, and that "in New York" is a loose prepositional phrase in search of a conceptualization to be attached to.

This principle states that, <u>ceteris</u> <u>paribus</u>, the first interpretation is to be chosen of the second, as the first is arrived at by using an optional component rather an by using a new pattern.

5.10.2 Prediction

Currently, disambiguation is done by considering which of the meanings of the word, or phrase, would extend one of the active patterns. Of course, this does not always work, since it is possible that at a given point there are no active patterns that can help. None may have been suggested yet, or because all the active patterns may be irrelevant (i.e. none of the possible meanings satisfy the next condition of any pattern).

Riesbeck (1975) emphasizes the importance of being able to make predictions to disambiguate word senses. More generally, his point is that understanding involves making a prediction about what might happen, checking to see if the prediction has been confirmed, and if it is, taking an appropriate action. Thus when his analyzer sees the word give, for example, it makes one prediction that says, "if the direct object denotes a physical object, build a concept designating a transfer of possession". Another example of a prediction is the rule "if the direct object denotes an action, build a concept representing that action".

Our work is very much influenced by this idea, but our model differs from Riesbeck's in one important respect. Riesbeck's predictions are explicit objects in his system. To implement a prediction, a special entity denoting that prediction is set up within his system. Much of what his system is concerned with is the management of these explicit prediction objects - deciding when to set them up, when to get rid of them, when to stack some of them for a while and when to restore them to active use.

Predictions in PHRAN, on the other hand, are just as real as far as their effect is concerned, but they are implicit in nature. That is, a prediction in PHRAN is a side product of having suggested one of more patterns that have not yet been completely matched to the input. The yet unmatched parts of the pattern serve to predict what might be heard next. The concept attached to the pattern describes what to do when the prediction is confirmed. Subsequent parts of the pattern describe what to predict next.

Thus we also feel that the role of prediction in language processing is important. But we feel it is a side product of the process that finds chunks of knowledge that may be applicable to the text, and then tries to apply them. In this view, predictions are not entities that perform processing, as they are in Riesbeck's system, but manifestations of the understanding process.

Viewing predictions as manifestations of other processes rather than as entities in themselves is not just an argument about their ontological status. It has both psychological and practical implications. For example, Gershman (1979) describes an extension of Riesbeck analyzer that has predictions that look backwards as well as forwards. Not only is it unintuitive to think of these as predictions anymore, but it becomes difficult to decide when a certain task should be accomplished by a request associated with one word looking forward, or by one associated with another word looking backward. These problems do not arise in PHRAN, since predictions are not entities in and of themselves. PHRAN's patterns do not have directional constraints, so no special kind of processing need be specified as a function of when then pattern is suggested (See Birnbaum and Selfridge (1979) for a description of some of the problems involved in word disambiguation in a system where predictions are directional).

5.10.3 Indexing

We have addressed the problem of indexing the pattern-concept pairs in the database in a previous section, and we noted there that it would be desirable to have a system that would take contextual cues into consideration when suggesting patterns to the comprehension routines. We know of no implemented scheme that does this to our satisfaction. In particular, no scheme, including our own, accounts adequately for how a certain word sense is selected by a context. That is, how is it that people will assume a word sense in a context without even realizing that a word is ambiguous? To do so implies that they can leap to the correct word sense without even considering inappropriate senses. Since all the schemes we are aware of require some kind of search through possible word senses, it is difficult to see how they could account for this phenomenon.

While it is extremely speculative at the moment, it may be that the basic model of computation assumed to underly this process is wrong. Some alternative schemes that may have more desirable properties have recently been suggested by Anderson and Hinton (1980), Minsky (1979), and Feldman (1979). Basically, these schemes all make use of highly parallel, neural-like mechanisms in which it is easy to maintain associations between the components of individual experiences. That is, given that sufficient parts of the memory of an overall experience have been activated, the rest of that experience automatically is recalled.

In terms of language processing, these schemes may help as follows: Both a language pattern and contextual elements may constitute one part of a previous understanding experience, which also includes, among other things, the meaning of the utterance in that context. When both that pattern and context are active again, the rest of that experience is also activated, including the meaning of the utterance. Thus, given that the basic processing model can be made to work, it may be possible to get disambiguation "for free".

In a sense, the whole problem of indexing goes away in these models. However, we do not believe that any of the models have been developed sufficiently at this point to know whether or not they will constitute a powerful alternative to existing information processing techniques.

5.10.4 Assymmetries And The Shared Knowledge Base

One argument that could be advanced against the psychological validity of a common knowledge base for the understanding and production mechanisms is that there are observable assymmetries between what people produce and what they are capable of understanding. For example, a person may be able to understand an utterance using a certain word or phrase, although he would never use that phrase himself.

There are a number of natural ways to account for such assymmetries within our model. The simplest involves indexing. Our model suggests that the understanding and production mechanisms use different indexing schemes: The understander must be able to react to any phrase it hears, as it has no control over its input. In contrast, the language generator need only have a way of producing some utterance for each idea it wishes to express; if there is more than one phrasal pattern that can be used to express this idea, only one of them needs to be accessible to the producer for it to function adequately. For example, the generator need only index the "<person> kill <person>" phrasal-concept pair in order to have a way of expressing that someone died. However, the understander must index both this pattern and "<person> kick the bucket", as it has no way of knowing which it will hear.

This predicts that a person should able to understand (or at least recognize) what he produces, but that he may be able to understand many utterances he would not produce. This is precisely what is observed.

This kind of assymmetry in task requirements carries over into other aspects of production and understanding. For example, in the case of concept modifiers, PHRAN must often search around to find an appropriate concept to modify, whereas the analogous generation task need only find a place in a phrase to insert a modifier. Thus the generator need not have all the flexibility of placement required by the understander. This would be consistent with a finding that individuals tend to insert modifiers in fewer places than they are capable of understanding them in.

6.0 COMPARISON TO OTHER SYSTEMS

While very little work has been done in the area of language production, there are a number of language understanding systems or systems with natural language front ends. We will not attempt to survey all these systems here. Rather, we will compare other work primarily to those systems that make some use of a "patterns of speech" type approach to natural language processing.

One of the earliest such systems is Colby's PARRY. PARRY is a simulation of a paranoid mental patient that contains a natural language front end (Parkinson et al, 1977). It receives a sentence as input and analyzes it in several separate "stages". PARRY makes several passes over the input in which it identifies and brackets off noun phrases, replaces idiomatic phrases with simpler one word expressions, replaces verbs used in the passive with active ones, and may even rearrange the sentence using the same, or another, verb, and performs other tasks as well.

In effect, PARRY replaces the input with sentences of successively simpler form. In the simplified sentence PARRY searches for patterns, of which there are two basic types: patterns used to interpret the whole sentence, and those used only to interpret parts of it (relative clauses, for example). In matching these patterns, PARRY is allowed to ignore unrecognized words.

For PARRY, the purpose of the natural language analyzer is only to translate the input into a simplified form that a model of a paranoid person may use to determine an appropriate response. No attempt is made to model the analyzer itself after a human language user, as we are doing, nor are claims made to this effect. A system attempting to model human language analysis could not permit several unrelated passes, the use of a transition network grammar to interpret only certain sub-strings in the input, or a rule permitting it to simply ignore parts of the input.

This additional theoretical shortcoming of PARRY - having separate grammar rules for the complete sentence and for sub-parts of it - is shared by Hendrix's LIFER (Hendrix, 1977). LIFER is designed to enable a database to be queried using a subset of the English language. This subset can be enlarged rather easier by the user, who can type in new patterns and instruct the system how to interpret them.

As is the case for PARRY, the natural language analysis done by LIFER is not meant to model humans. Rather, its function is to translate the input into instructions and produce a reply as efficiently as possible. Thus nothing resembling a representation of the meaning of the input is ever formed. While we do not wish to breathe new life into the procedural-declarative controversy, we take for granted that a declarative representation of meaning is necessary for the inference processes that begin once natural language analysis is completed.

Of course, the purpose of LIFER is not to be the front end of a system that understands coherent texts, and which must therefore perform subsequent inference processes. While LIFER provides a workable solution to the natural language problem in a limited context, many general problems of language analysis are not addressed in that context. Other problems have simple solutions that are adequate here but do not seem to be extensible. Hendrix's treatment of "and" and pronomial references would seem to fall into this category.

SOPHIE (Burton 1976) was designed to assist students in learning about simple electronic circuits. It can conduct a dialogue with the user in a restricted subset of the English language, and it uses knowledge about patterns of speech to interpret the input. SOPHIE accepts only certain questions and instructions concerning a few tasks. As is the case with LIFER, the language utterances acceptable to the system are restricted to such an extent that many natural language processing problems need not be dealt with and other problems have solutions appropriate only to this context.

For example, consider SOPHIE's handling of the pronoun "it". "It" is always taken to mean either a measurement or an element of the circuit being debugged, independent of the previous dialogue between the program and the user. This seems to be satisfactory in SOPHIE's case, but it provides neither a solution to the general problem, nor does it give any indication of what such a solution might be.

SOPHIE also does not produce any representation of the meaning of the input. For example, the sequence "the voltage at the collector of Q5" is always interpreted as a call to a function that measures the appropriate voltage. In addition, SOPHIE makes more than one pass on the input and ignores unknown words, practices that have already been criticized.

The augmented finite state transition network (ATN) has been used by a number of researchers to aid in the analysis of natural language sentences (for example, see Woods 1970). The ATN itself is a very general formalism, and in recent years has been extended to include enough alternative control structures that it may be considered more of a programming language than a theory of language understanding (e.g., see Waltz et al (1976).

However, most systems that use ATN's incorporate one feature which we find objectionable on both theoretical and practical grounds. This is the separation of analysis into syntactic and semantic phrases. Natural language analyzing programs utilizing ATNs parse an input according to given grammar rules. The only role semantics plays is in the classification of grammatical nodes. Later on, special semantic routines are used to analyze content of the parse created by the ATN. It would seem unlikely that such a separation occurs in human understanders; and there is no compelling evidence we are aware of to justify such a separation.

Of course, most ATN advocates do not make claims about the psychological validity of their model. As to the efficacy of the separation of syntactic and semantic processing, this has been argued at length elsewhere (see Schank 1975 for example). Let it suffice to say that using all forms of knowledge together would seem to have a better chance of constraining the entire process that would using these forms of knowledge in separate phrases. In addition, most ATN based systems (for example Woods' LUNAR program) do not produce representations, but rather, run queries of a data base.

None of the previously mentioned systems contains a component that generates English sentences from a representation of a concept. The responses they do give are either completely canned or else arrived at by filling slots in some fixed pattern. While these system were not designed with this task in mind, it would seem that a reasonable model of human natural language understanding should in principle be capable of sharing most of its knowledge about the meaning of the utterances of its language with a language generation mechanism. However, it would not seem possible to use most of the knowledge in the data bases of the analysis programs described above for the generation task. This is particularly true in systems espousing some form of procedural semantics - if the meaning of the utterance is thought to be the execution of some routine, then it is hard to see how semantic knowledge that results in a process being executed can be used to produce a natural language utterance.

In contrast to the systems just described, Wilks' English-French machine translator does not share several of their shortcomings (Wilks 1973). It produces a representation of the meaning of an utterance, and it attempts to deal with unrestricted natural language. The main difference between Wilks' system and system we describe is that Wilks' patterns are matched against concepts mentioned in a sentence. To recognize these concepts he attaches representations to words in a dictionary.

The problem is that this presupposes that there is a simple correspondence between the form of a concept and the form of a language utterance. However, it is the fact that this correspondence is not simple that leads to the difficulties we are addressing in our work.

For example, Wilks' systems is forced to make an initial pass using "fragmentation functions" which separate the sentence into segments using key words (and sometimes rearrange the segments) so that the concepts in the sentence may be identified. This solution would appear to be entirely unsatisfactory for processing non-productive language forms in which meanings cannot be closely associated with individual words. In general, since the correspondence of words to meanings is complex, it would appear that a program like Wilks' translator will eventually need the kind of knowledge embodied in PHRAN to complete its analysis.

One recent attempt at natural language analysis that radically departs from pattern-based approaches is Rieger and Small's system (Small 1978). This system uses word experts rather than patterns as its basic mechanism. The idea of putting as much information as possible under individual words is about as far from our conception of language analysis as one can get, and we would argue, would exemplify all the problems we have described in word-based systems.

However, there are actually a number of important similarities between the two systems. First, Rieger and Small seem to combine syntactic and semantic knowledge in their experts. Since more traditional pattern-based systems are generally pattern-based insofar as syntactic analysis is concerned, the word expert idea is closer to our approach in this respect. In addition, Rieger and Small's system acknowledges the fact that a natural language analyzer must contain a great deal of knowledge about its language, and that representing and organizing this knowledge is a major problem. Thus while we do not feel that word experts is the best way to resolve this problem, it at least seems to address what we feel is the crucial issue.

Appendix

Phrasal Language Constructs and Linguistics

It may be surprising, given the apparent pervasiveness of phrasal language constructs in everyday language, that the study of phrasal language forms has received very little attention in this country. This may be due to the dominance of the generative model in American linguistics and to this model's emphasis on an abstract notion of linguistic competence. In this approach, the actual knowledge people possess about their language is regarded as insignificant in comparison with various generalizations that can be made about this knowledge.

Within the generative paradigm, the fact that language users have memories and may in fact possess a great deal of very specific, possibly even redundant, knowledge about language forms and their associated meanings is treated as an irrelevant issue; the great number of language forms that seem not to fit the model are regarded as unimportant or irritating exceptions.

However, this view of language (or perhaps more accurately, of language study), is by no means universally held. For example, Chafe (1968) describes idiomaticity as an important enough anomaly within the generative paradigm to justify the replacement of that model with an approach that is more closely connected to human cognitive processes. Attempts to account for idioms within the transformational model, he notes, reveal only the impoverishment of that paradigm.

For example, in transformational grammar, an idiom like "kick the bucket" is problematic because it shares some properties with other sentences of similar structure, but not all these properties. Thus the verb kick can change to reflect the tense and have agreement with the subject without losing the idiomatic meaning of the phrase. However, the passive form of the utterance does not have this property, i.e., "The bucket was kicked by John" probably doesn't mean that John died.

This is problematic to the transformationalists since their goal is to characterize a language in some minimal way. The transformationalist solution to the problem is, in effect, to put some filter in the generative mechanism to prevent the passive transformation from being applied successively to produce the unwanted form (Fraser, 1970). However, the solution hardly leaves one with the satisfying feeling that the reason that the passive form of this utterance is not used has been accounted for. Chafe points out, however, that a more satisfying solution is obtainable in a model that takes issues of cognition into account. That is, since the notion of a bucket is not cognitively present in the meaning the user is trying to express, the user would have no motivation to stress "bucket" by using a form that puts it at the front of the sentence. This explanation is not available in the generativist paradigm since the notion of whether something is there cognitively does not arise.

Another approach to language in which the importance of phrasal constructs is not trivialized is given by Bolinger (1976). He states a position very similar to our own, in which more specific utterances play as important a role as abstractions from these utterances. As Bolinger puts it, not every language structure should be viewed as having been built up from scratch; rather, language involves the large scale use of a lot of prefabricated material.

Among his many examples, the following are of particular interest. There appear a number of strong restrictions on the use of the words ago and else in addition to those restrictions imposed by the words' syntactic and semantic properties. For example, ago is always suffixed. We say "a year ago" and not "an ago year". It is never used independently, as in "He got there ago", meaning, he got there some time ago. It is subject to peculiar context restrictions. For example, "a long time ago" and "a short time ago" are fine. So is "long ago", although "short ago" doesn't make it.

"Else" is also always suffixed. "Someone else", "some place else", "who else", and "nobody else" are all fine. But "sometime else" and "some person else" are peculiar, that is, "else" seems restricted to indefinite pronouns but excludes those referring to time. One exception is "or else", as in "do what I say or else". Restrictions on manner as well as on temporal and indefiniteness apply. For example, "we'll do it somewhere" and "we'll do it somewhere else" are both understandable, as are "where else" and "how else" and "we'll do it somehow". But "we'll do it somehow else" is strange; most language users would express this idea as "we'll do it some other way".

Why do people not use "ago" and "else" to express the ideas that could be embodied in expressions like "sometime else" or "short ago"? Certainly people are not without the mechanism for producing new utterances. Bolinger's explanation is "that at least in part we do not do it because we have not heard it done. We have no memory of it." Thus Bolinger is challenging the reductionist models of language that seek to eliminate variety. The accuracy as well as the psychological validity of that approach seems doubtful. Bolinger notes that his "heterogeneous" view of language suggests that linguists take into account the vast store of knowledge people seem to have about their language, rather than try to reduce this wealth to as small a set as possible. While this may be unsettling to some, it may in fact fit the situation most accurately. The human brain, he points out, is not a vestigial organ.

Fillmore (1979) has recently characterized a number of models of language as suffering from an idealization that he has termed innocence. The innocent language user knows the meaning of the various morphemes of its language, and the rules by which they interact. As a listener it computes the meaning of an utterance from the meaning of its parts, and as a speaker, it decides how best to express something by choosing the individual words that combine to convey the desired message. Thus the models of language analysis and language production described above, as well as most formulations of generative linguistics, are models of innocence. The innocent language user is incapable of dealing with most kinds of phrasal constructs, as the additional information they require is not available to it.

While the idealization of innocence may have prevented much attention from being paid to phrasal constructs by linguists in this country, the situation has been somewhat better elsewhere. For example, the British linguists have paid a great deal of attention to collocations - expressions whose meaning can be computed from its parts, but is nevertheless particularized (Mitchell, 1971). In the Soviet Union, a field called phraseology has developed as a branch of linguistics and seems to be thriving. See for example Arnold (1973).

Although it is useful for expositing a wide variety of phrase constructs, this work does not address the question of how these constructs might be stored or represented in a human or other cognitive system, or how they might be associated with meanings. However, it is our contention that considering these cognitive issues is not only essential for our own tasks, but also for understanding the kinds of phenomena linguistics are interested in.

As an example, a good deal of the work on idioms that has gone on in linguistics concerns the kinds of syntactic forms that the same idiom can take - whether words can be inserted in the middle of the phrase, whether the idiom occurs in the passive, whether it can be nominalized, etc. For example, "kick the bucket" is never found in the passive; "the bucket was kicked by John" has only a non-idiomatic meaning. On the other hand "keep close tabs on" may occur both ways: "The press kept close tabs on Kissinger" and "Close tabs were kept on Kissinger by the press" both seem plausible.

While we do not feel that we have resolved all the issues here, some of the explanations are fairly straightforward in our model. For example, in the case of passive forms for expressions like "kick the bucket", the solution involves the meaning of the passive form. That is, if the passive form performs some function, like focusing attention or topicalizing, then it is not at all clear what a speaker of the language would have been trying to communicate by using this idiom in the passive. Thus a speaker would not produce this form because it does not correspond to anything he would want to say.

Similarly, a listener would not interpret this form idiomatically because the words "the bucket" are not a good index into the phrasal pattern. This may be the case simply due to frequency of occurrence - i.e., "the bucket" may occur frequently enough in by itself in forms in which is taken literally so that it is not useful to index the idiomatic sense under this sequence alone. Not only would an interpretation be hard to find because of the passive voice - but the specific pattern needed to find the idiomatic interpretation probably is difficult to get to considering the order in which the words are read.

On the other hand, putting "close tabs" in the front may be a reasonable way to focus on the surveillance rather than the person who was surveyed. Also, this sub-phrase seems to be so closely associated with the whole pattern that it should provide a good search key. This might also explain why some people find "Tabs were kept on Kissinger" a bit annoying - "tabs" alone is not as good a key to the pattern as it is when followed by "close".

Acknowledgments

We wish to thank Chris Riesbeck and Drew McDermott for their helpful comments on an earlier version of this paper.

References

1] Anderson, James and Hinton, Geoffrey (1980) Models of Information Processing in the Brain. In press.

2] Arnold, I. V. (1973). Set expressions. From The English Word, Moscow, pp. 140.163.

3] Becker, Joseph D. (1975). The phrasal lexicon. In Theoretical Issues in Natural Language Processing. R. Schank and B.L. Nash-Webber (eds.). Cambridge, Mass.

4] Birnbaum, L. and Selfridge, M. (1979). Problems in conceptual analysis of natural language. Yale University Department of Computer Science Research Report #168.

5] Bolinger, Dwight (1976). Meaning and Memory. Forum Linguisticum 1.1.1-14.

6] Burton., Richard R. (1976). Semantic Grammar: An Engineering Technique for Constructing Natural Language Understanding Systems. BBN Report No. 3453, Dec 1976.

7] Chafe, Wallace (1968). Idiomaticity as an anomaly in the Chomskyan paradigm. Foundations of Language 6.1.22-42. Understanding of newspaper stories. Yale University Research Report #116.

8] Cullingford, R. E. (1978). Script Application: Computer Understanding of newspaper stories. Yale University Research Report #116.

9] Feldman, J. A. (1979). University of Rochester, Department of Computer Science Research Report TR52.

10] Fillmore, C.J. (1968). The case for case. In E. Bach and R. Harms (eds.), Universals in Linguistic Theory. Holt, Rinehart and Winston, New York.

11] Fillmore, C.J. (1979). Innocence: A Second Idealization for Linguistics. In Proceedings of the Fifth Berkeley Language Symposium, Berkeley, California.

12] Fraser, Bruce (1970). Idioms within a transformational grammar. Foundations of Language 6.1.22-42.

13] Goldman, Neil (1975). Conceptual generation. In R. C. Schank, Conceptual Information Processing. American Elsevier Publishing Company, Inc., New York.

14] Held, G. D., Stonebraker, M. R., and Wong, E. (1975). INGRES - A relational data base system. AFIPS Conference Proceedings vol. 44, NCC.

15] Hendrix, Gary G. (1977). The Lifer Manual: A Guide to Building Practical Natural Language Interfaces. SRI International: AI Center Technical Note 138, Feb 1977.

16] Kay, Martin (1975). Syntactic Processing and Functional Sentence Perspective. In Theoretical Issues in Natural Language Processing, R. C. Schank and B. Nash-Webber (eds.). Cambridge, Mass.

17] Lakoff, G. and Johnson, M. (1980). Metaphors We Live By. In press.

18] Minsky, M. (1979). K-Lines: A Theory of Memory. AI Memo 516, MIT AI Lab.

19] Mitchell, T. F. (1971). Linguistic "Goings On"; Collocations and Other Matters Arising on the Syntactic Record. Archivum Linguisticum 2 (new series) 35-69.

20] Parkinson, R.C., Colby, K.M., and Faught, W.S. (1977). Conversational Language Comprehension Using Integrated Pattern-Matching and Parsing. Artificial Intelligence 9, 111-134. q.inc ref

20] Riesbeck, C. K. (1975). Conceptual analysis. In R. C. Schank, Conceptual Information Processing. American Elsevier Publishing Company, Inc., New York.

21] Riesbeck, C. K. and Schank, R. C. (1975). Comprehension by computer: expectation-based analysis of sentences in context. Yale University Research Report 78.

22] Schank, R. C. (1975). Conceptual Information Processing. American Elsevier Publishing Company, Inc., New York.

23] Schank, R. C., Lebowitz, M. and Birnbaun, L. (1980). An Integrated Understander. In American Journal of Computational Linguistics, vol. 6 no. 1, January-March 1980.

24] Simmons, R. F. (1973). Semantic networks: Their computation and use for understanding English sentences. In R. C. Schank and K. M. Colby (eds.), Compute Models of Thought and Language. W. H. Freeman and Company, San Francisco.

25] Small, S. (1978). Conceptual language analysis for story comprehension. Technical Report No. 663, Dept. of Computer Science, University of Maryland, College Park, Maryland.

26] Waltz, D. L., Finin, T., Green, F., Conrad, F., Goodman, B., and Hadden, G. (1976). The PLANES system: natural language access to a large data base. Coordinated Science Lab., University of Illinois, Urbana, Tech. Report T-34.

27] Wilensky, Robert (1978). Understanding Goal-Based Stories. Yale University Department of Computer Science Technical Report #140.

28] Wilks, Yorick (1973). An AI Approach to Machine Translation. In Computer Models of Thought and Language, R.C. Schank and K.M. Colby (eds.), W.H. Freeman and Co., San Francisco, 1973.

29] Woods, W. A. (1970). Transition Network Grammars for Natural Language Analysis. CACM 13, 591-606.

Run blank