THE IMPATIENT TUTOR:
AN INTEGRATED LANGUAGE UNDERSTANDING SYSTEM

Brian Phillips & James Hendler

Texas Instruments Inc.
Dallas, Texas 75265, USA

We describe a language understanding system that uses the techniques of segmenting the computation into autonomous modules that communicate by message passing. The goal is to integrate semantic and syntactic processing to achieve greater flexibility and robustness in the design of language understanding systems.

Introduction

This paper addresses the control problem in language understanding systems. Many formalisms have evolved for representing the syntactic, pragmatic, and semantic data of language, but the ability to access them in a flexible and efficient manner has not proceeded apace. This delay is understandable: one needs to know what to control before one can control it. Although the isolation of the subproblems is a valid methodology, there comes a time when a deeper understanding of the language system requires that the data and control aspects of the problem be considered together.

Linguistic theory has not offered much insight in the control of linguistic processes; Chomsky (1965) finessed the problem by creating "competence" as the proper view for theoretical linguistics, rather than the study of "performance". In fact, it is this study of process that is one of the contributions of computational linguistics to the study of language (Hays, 1971).

An overview of control strategies

Within automated language understanding systems we find a variety of strategies:

Linear control.

A logical approach is to adopt a linear control strategy in which syntactic analysis is followed by semantic interpretation (Woods, 1971). Unfortunately, this places an overwhelming burden on semantic processing which has to interpret each complete parse when the ambiguity may only lie in part. Further, there are cases where syntactic relations cannot be determined by syntactic analysis alone, for example, the role of "tree" in (1).

John was hit by the tree. (1)

Semantic grammars.

Faced with a need to access semantic information during syntactic analysis, one suggestion is to construct a "semantic grammar" (Hendrix, 1977) in which some categories in the syntactic rules are replaced by semantically based categories of the domain, e.g., verbs may be subclassified as verbs of movement, containment, excitement, etc. (Sager, 1975). The disadvantage of this approach is that the domain becomes an integral part of the grammar, with the result that either the number of syntactic rules is considerably enlarged, or the rule set has to be rewritten to move to another topic area.

Semantic parsing.

Other approaches have managed to achieve success by avoiding the problem of integration completely: the systems have essentially one component. Schank (1975) has systems based on the hypothesis that language understanding is driven from the semantics with minimal use of any syntactic analysis. But such systems can go astray because of their high semantic expectation. For example, the word "escape" carries with it the prediction that it is an action of terrorists (Schank, Lebowitz, & Birnbaum, 1978); this causes an erroneous analysis of a sentence such as "The policeman escaped assassination..."

Others have proposed procedural systems built around semantic knowledge (Rieger & Small, 1980). In the Rieger and Small system the knowledge is on the word level. Their main drawback is an inability to easily change domains.

Design Features

The power of syntax diminishes as more complex constituents are encountered. Syntax can give good descriptions for the structure of phrases, becomes less detailed when describing the role of phrases within clauses, has relatively little to say about the clause structure of sentences, and even less about sentences in discourse. As syntactic forces diminish, semantic relations describe the structure -- discourse cohesion is semantic (Halliday & Hasan, 1976). Consequently we believe that a language understanding system should have the ability to bring syntactic and semantic knowledge to bear on the analysis at many points in the computation in order to prevent the flow of extraneous analyses to later steps in the analysis.

We agree with Schank (1975) that the goal of analysis is not to produce a parse tree. It should not even be a subgoal, as is the case in systems that first produce a parse tree then perform semantic interpretation. The parse tree should be considered as a data structure that should either be constructed incidentally to the analysis, or be capable of being constructed should it be needed. But syntax cannot be ignored. Often it may not appear to be contributing much, but it is clear that syntactic structure is of use in determining antecedents of proforms, for example.

Schank's (1975) hypothesis of semantic prediction appears to us to be a good approach. The goal is certainly to build a meaning representation of the linguistic act and top-down analysis can lead to greater efficiency. Top-down systems tend to leave open the question of what to do when there is no prior knowledge to guide the analysis. We envisage a system that can flow into a predictive mode when the situation is appropriate, but otherwise has a default control structure of syntax-then-semantics. In short, we want a data-driven control structure.

Message passing

To achieve the design goals mentioned above, we are segmenting the problem into autonomous processes that communicate by passing messages to each other. This is Hewitt's (1976) view of computation as a society of cooperating experts.

We have experts that know about the organizing principles of syntax and of semantics. The experts are then interpretive, which gives flexibility in changing to another language, or to a new domain. We have experts for case-frames, scripts, clauses, subjects, and the like.

The experts will at points in time become associated with domain knowledge, i.e., the grammar of a language, or world knowledge for a problem area.

The job of an expert can be to instantiate a model that it has been given (top-down analysis), or if it was not given a model, then to find a model (bottom-up analysis). The process of instantiation is performed by eliciting information from other experts who can use their expertise on the problem; they of course may have to consult further experts. Some experts are not instantiators, rather they are processes that are common to several other experts; for parsimonious representation we give them expert status.

The output of the system is a semantic description of the input as instantiated case-frames. The novelty of the situation is captured by the way in which the case-frames are linked and by their spatio-temporal settings. The semantic description augments the encyclopedia and is thus available as pragmatic knowledge in the continuing analysis of the input.

The impatient tutor.

This initial project is a study of message flow in the system. As each word of the input is processed we are trying to disseminate its effect throughout the system. In particular we wish to have the analysis rapidly reaching the overall semantic description of the task so that it can be checked against the prescribed actions and any divergence noted. If a deviation is apparent, the system will interrupt the student. We are not proposing the system as a serious tutor;

it's shortcomings are quite apparent: if a student intended to say "I will get the hammer before I get the wrench ..." the impatience of the system would cause an interjection after hammer because of an expectancy of a wrench.

The advantages of message passing

Efficiency.

Without prediction, linguistic analysis can only be a uni-directional search of the problem space, which is exponential in complexity. If a goal is known or predicted, then bidirectional searching, from input and goal, reduces the complexity. Yet greater efficiency can be achieved if the prediction can be directly associated with the input.

In other schemes for processing language, the flow of control is constrained to follow the organization of the data.

The ability of any expert to communicate with any other expert is how we achieve the greater efficiency. If an expert is instantiating a case-frame, for example, it can be in direct communication with a phrase expert that is trying to instantiate some syntactic rule. The findings of the phrase expert are transmitted directly to the case-frame expert, which may check the suggestion by calling upon the taxonomic expert. As each message carries with it a return address, it can be returned directly to the originator of the query without being chained through any intermediate experts.

We are using the addresses of messages to achieve our desired perspective on syntax. Although the information necessary to build a parse tree is in messages, the information can be returned directly to the expert that initiated the query, bypassing other experts who were intermediaries in the answering process. The omitted experts may include those that build syntactic structure. However, a message also has a trace of its route and, should the need arise, the longer path can be followed to build structure.

Robustness.

It is apparent that there is a certain amount of redundancy in language. This is probably why apparently inadequate systems have been able to process well-formed discourse. But real people do not speak with perfection. Eventually natural language systems will have to be able to process the normal language of people. A user will not be enamored of a system that demands more care and attention be given to the language of his interaction than is usual for his other conversational activities.

To progress to a systematic study of robustness we need to examine schemes by which all of linguistic knowledge may be flexibly invoked; thus we believe that the systems that contain less than this knowledge will not be a suitable vehicle. Linear control structures are equally not the answer. If the erroneous item is first encountered, there is no way of using later components. The flexibility of the message passing scheme will allow other knowledge to be accessed.

Organization of the data

The data of our system is divided into three parts: the syntactic rules, the semantic knowledge, and the definitions of words. The syntactic rules are contained in the "grammar", the semantic rules in the "encyclopedia" and the word definitions in the "dictionary."

Grammar

The grammar consists of a set of rules of the form shown in Figure 1.

```
Clause == Subj Verb Object
Clause == Subj Stative Compl
Subj == NP
NP == Det Adj* Noun
NP == N Clause
Compl == State
etc.
```

Figure 1: Grammar

The rules are written to allow the presence of a "subject" expert between the "clause" expert and the "NP" expert as it is the subject expert that knows about subject-verb agreement. Agreement rules (not shown) are written in terms of syntactic features such as "number". The experts for syntax use these rules to determine what parts of speech to expect next. The rules are language specific and are therefore not encoded into the syntactic experts. Only the universal categories have corresponding experts.

Dictionary.

The dictionary consists of word definitions that include the syntactic properties of the word. Thus the word "left" would have information that it could be an adjective (as in "left foot") , a verb ("left home") and a noun ("the new left"). The description of the sense of each word is reached by a pointer from the dictionary into the encyclopedia. For example, that as a noun it refers to a group of people, as an adjective refers to a positional referent, and that as a verb it can build the case frame associated with leaving.

Encyclopedia.

The encyclopedia consists of a network of case frames linked by relations of causality, taxonomy, instance and equivalence (Phillips, 1978).

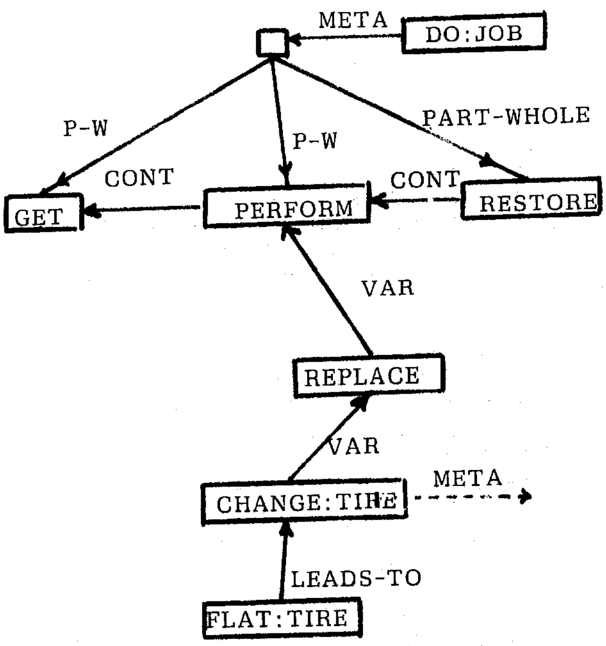

Figure 2: Simplified version of semantic network with information about changing a tire.

In Figure 2 we see knowledge about changing a tire. The CONTingency links represent causal dependencies. The META links show the equivalence of concepts, one concept having an equivalent description by a set of concepts. For example "replace" represents "removing an old object and putting on a new one". If concepts in the resulting description also have meta-links, the decomposition can be continued. Schank's (1979) MOP's are similar to our meta-organization. The VARiety link is used to show taxonomic classication. Thus "Change-tire" is a kind of "replace". Common knowledge need only be represented once; it is inherited by concepts lower in the taxonomy than the point of representation. The INSTance relation captures the episodic nature of memory by storing specific instances as instantiations of intensional descriptions: "That time I changed my tire in front of Mom's house." is one instantiation of the general changing a tire event.

Anatomy of an expert

Each expert in the system knows how to use specific types of links and to perform operations using local data. An expert also keeps track of its message activity. As an example, take the "Chronology" expert, Figure 3.

```
Static component:

   Name: CHRONOLOGY
   Link types: CONT, LEADTO,
               SEQ, ENABLE
   Process:
      (a) If NEXT-EVENT requested
            then trace LEADTO or SEQ
      (b) If LAST-EVENT requested
            then trace SEQ or CONT
            of node and of VAR(node)

Dynamic component:

   Memory: (record of already
             traced links)
   Status: (waiting for another
             expert to complete, or
             finished)
```

Figure 3: The CHRONOLOGY expert

There are two parts to each expert. The static part which is not changed during processing, and the dynamic part which is. The dynamic component contains a memory, which keeps track of all processing done by this expert so far. This is primarily included for efficiency, since it saves the expert from having to repeat computations. It also contains a "Message Center", which tells whether it is waiting for an answer from another expert (is a Client to another expert) or has other experts waiting for replies (has Customers). It also has default Customers to whom messages should be sent even if they have not been requested.

The static component has a name, a list of the link types which the expert knows about, and a set of process rules. These rules are the heart of the experts, since they contain information on what processes to call to get information and what other experts to call. In the case of the Chronology expert shown in Figure 3 it uses the process "trace" to follow links, and can call the taxonomy expert to get superior nodes. In the case of the syntactic experts these process rules include information about using the syntactic grammar rules to find the next expert to call.

Translation

As experts have vocabularies that are peculiar to their domains, messages -- in particular from semantic to syntactic experts -- may require translation from the terminology of the sender to that of the receiver.

For example, messages between clause experts (CLE) and case-frame experts (CFE). The former uses the concepts of subject, object, verb, etc., whereas the latter has events, states, and agents, instruments, etc. Let us consider a scenario in which a CLE has analyzed a "subject" and wants to convey this information to a CFE. It could send the role-labelled concept to the CFE. However, to attribute a CF role to the concept, the CFE needs to know the mood of the sentence. This it can only determine by sending messages back to the CLE. The overall effect would be to transfer information available to the CLE to the CFE. It is obviously more efficient to have the translation process as part of the resources available to the CLE and to have it send off a possible "agent", say, to the CFE. The CFE can verify or reject the hypothesis using the semantic resources available to it.

If the CFE is predicting a certain "instrument", say, it could have available to it information on the realizations of instruments and remit to the CLE the prediction. Again this is putting knowledge of syntax and of forms into the CFE; it seems better to have the CFE send "instrument" and the word concept to the CLE which decides upon likely realizations.

All in all the translation process resides more naturally with the CLE. In general, it is taken that the translation resides in the experts on the syntactic side of the system.

Other semantic phenomena that can have correlates in syntax are contingency, sequence, and decomposition. For example, chronological ordering may be realized by "then". In general there are many possible realizations; they can be single words or even clauses. A little-understood "connective" expert has the job of watching for the syntactic clues.

An Example of Experts in Action

In this section we will outline how the system uses the knowledge Figure 2 to process input about changing a tire, for example, (4) and (5).

The left front tire is flat. (4)
I will change it. (5)

The goal of the system is to create a meaning representation by instantiating a CF. Through meta-links, a CF can be equivalent to a complex of CF's; thus the top-level instantiation may be achieved by instantiating the lower rank CF's.

A CFE normally has a model of a CF that it is trying to instantiate. Initially this cannot be the case and the system has to revert to a bottom-up approach. The CFE sends a message to the CLE requesting that it be sent a translation of a syntactic analysis of a clause. The CLE has to find a clause using the rules of the grammar in Figure 1. The clause rules show that a "subject" expert has to be invoked. In turn it sends a request to a "NP" expert. The NP expert finds the rules that describe its constituent structure. Given the many many rules that could be used, it would be inefficient to examine them all, so input is used to guide its choice. The expert gets the word by asking an "input" expert, which prompts the user. The NP expert selects those rules that can be part of a model consistent with the input. The syntactic instantiation is similar to a chart parse (Kaplan, 1973) showing the hierarchical arrangement of constituents. At this point, the CLE has not recognized any of the entry points to the translator and so cannot yet respond to the CFE. The next input word is taken by the CLE. The input will instantiate some of the analysis paths and possibly eliminate some. And so on until a constituent that can fulfill the subject expert's request is recognized. Omitting a number of steps, the response is "the left front tire". The subject expert cannot truthfully forward this phrase as it cannot be certain that it is a subject until the mood of the clause is known. We are still considering what to do in this situation. We could wait or could send the concept off without annotation to see if the CFE can make any use of it. (The latter would be profitable if there are only a limited number of semantic possibilities in the context.) Let us assume that we wait. The subject expert interrogates the CLE for information on its mood, which require that the clause expert continue the analysis. Once the verb expert has functioned, the information is available and so the the stative verb. The grammar then predicts that a "state" will follow. This is confirmed by the word "flat". After receiving the response from the CLE, the CFE has the following instantiation:

```
CF1:  ( Agent - TIRE1
        Act   - STATE
        Obj   - FLAT)
```

This episode becomes part of the encylopedia.

A CFE contains the knowledge that when a state is found, a request should be passed to Chronology asking for the NEXT-EVENT. Chronology traces the LEADTO link from CF1 and predicts that Change:tire will be the next act. It passes this information back to the CFE. The CFE now has the prediction that the following CF

```
CF2:
    (Agent - (unknown)
     Act   - Change:tire
     obj   - from TIRE1 to SPARE)
     Instr - TOOL6
```

will be found. TOOL6 is a token representing a group consisting of a jack and a wrench. For the sake of brevity in this example this information is made explicit, in the actual program it can be determined by tracing other links. The CFE has now processed the first case frame to the best of its abilities and sets out to instantiate the prediction. As the CFE has CF2 as its model, it can work in a top-down manner. When the prediction is passed to the CLE and translated, "tire" will be available as a match for the pronoun "it".

The instantiation of the model produces CF3:

```
CF3:
  ( Actor - self    (from "I")
    Act   - Change:tire
    obj   - Tire1     (from "it") )
```

The CFE seeks to set up more predictions for the dialogue. It looks to see if this action is contingent on any others. To do this it calls up chronology and requests the LAST-EVENT for CF3. Chronology calls upon taxonomy which ascends variety links to the "perform" act in " do:job". The taxonomy expert also checks to see if the meta-node has any contingencies, but in this case it doesn't. If it did, that would also be returned to chronology. It finds CF4:

```
CF4:
  ( Agent - self
    Act   - GET
    Obj   - TOOL6 ).
```

This is then passed back to the CFE to serve as a prediction for the next input. And so the cycle of prediction and instantiation continues.

References

Chomsky, N. Aspects of the Theory of Syntax. Cambridge: MIT Press, 1965.

Halliday, M.A.K., & Hasan, R. Cohesion in English. London: Longman, 1975.

Hays, D.G. "The field and scope of computational linguistics." Proceedings of the International Conference on Computational Linguistics. Debrecen, 1971.

Hendrix, G.G. "Human engineering for applied natural language processing." Proceedings of the 5th International Joint Conference on Artificial Intelligence. Cambridge, 1977.

Hewitt, C. "Viewing control structures as patterns of passing messages." (MIT AI Memo 410.) Cambridge: MIT AI Laboratory, 1976.

Kaplan, R.M. "A general syntactic processor." In R. Rustin (Ed.), Natural Language Processing. New York: Algorithmics Press, 1973.

Phillips, B. "A model for knowledge and its application discourse analysis." American Journal of Computational Linguistics, 1978, Microfiche 82.

Sager, N. "Computerized discovery of semantic word classes in scientific fields." In R. Grishman (Ed.), Directions in Artificial Intelligence: Natural Language Processing. (Courant Computer Science Report #8). New York: New York University, 1975.

Schank, R.C. Conceptual Information Processing. New York: American Elsevier, 1975.

Schank, R.C. "Reminding and Memory organization: An Introduction to MOPs." (Yale University Research Report #170.) New Haven: Yale University, 1979.

Schank, R.C., Lebowitz, M., and Birnbaum, L.A. "Integrated Partial Parsing." (Yale University Research Report #143). New Haven: Yale University, 1978

Small, S.L., & Rieger, C. Conceptual analysis with the Word Expert parser. Annual Meeting of the Cognitive Science Society. New Haven, 1980.

Woods, W.A., & Kaplan, R.M. The Lunar Sciences natural language information system. (BBN Report No. 2265.) Cambridge: Bolt Beranek & Newman, 1971.