An Adaptive Natural Language Parser

PERRY L. MILLER
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

ABSTRACT

When a user interacts with a natural language system, he may well use words and expressions which were not anticipated by the system designers. This paper describes a system which can play TIC-TAC-TOE, and discuss the game while it is in progress. If the system encounters new words, new expressions, or inadvertent ungrammaticalities, it attempts to understand what was meant, through contextual inference, and by asking intelligent clarifying questions of the user. The system then records the meaning of any new words or expressions, thus augmenting its linguistic knowledge in the course of user interaction.

1. INTRODUCTION

A number of systems are being developed which communicate with users in a natural language such as English. The ultimate purpose of such systems is to provide easy computer access to a technically unsophisticated person. When such a person interacts with a natural language system, however, he is quite likely to use words and expressions which were not anticipated. To provide truly natural interaction, the system should be able to respond intelligently when this happens.

Most current systems, such as those of Winograd [10] and Woods [11], are not designed to cope with such "linguistic input uncertainty." Their parsers fail completely if an input sentence does not use a specific, built-in syntax and vocabulary. At the other extreme, systems like ELIZA [9] and PARRY [2] allow the user to type anything, but make no attempt to fully understand the sentence. The present work explores the middle ground between these extremes: developing a system which has a great deal of knowledge about a particular subject area, and which can use this knowledge to make language interaction a flexible, adaptive, learning medium.

In pursuing this goal, the present work is most closely related to work being done in the various speech recognition efforts [5, 7, 8, 12] which are studying how linguistic and semantic constraints can help deal with the ACOUSTIC error and uncertainty of speech. The adaptive system, however, is designed to deal with a much more LINGUISTIC type of uncertainty.

When people use unfamiliar words or expressions in conversation, we can usually deduce from context what is meant, and if not, we can at least ask intelligent clarifying questions. To allow the machine to do the same, there must be a very flexible interaction of syntax and semantics in the parsing/understanding process. There must be a different parser organization, and a more flexible use of linguistic and semantic constraints, than is present in current natural language systems.

The adaptive system is a step towards this goal The current implementation is a prototype, designed to illustrate many of these ideas, and to tie them together in a restricted system that is complete but not too complex. The system's domain of discourse is TIC-TAC-TOE: it plays a game and discusses the game in progress. It has a (currently limited) set of semantic concepts, and attempts to interpret the user's sentences in terms of these. The set of semantic concepts is FIXED. The system does not attempt to learn new concepts, merely new ways of describing concepts that are already familiar.

When the system encounters a new word or a new phrasing, it uses contextual information to determine the possible meaning, and to ask clarifying questions which are as intelligent as possible. It then remembers the meaning of the new expression, and will use it itself when it talks.

2. SAMPLE SESSION

The best way to introduce the system is to show it in operation. In the sample session that follows, user input is preceded by "U:", machine output by "M:", and comments added by the author are in small type, enclosed in brackets.

(MACHINE: I PLACE AN X IN THE TOP LEFT SQUARE)

```
(X - -)
(- - -)
(- - -)
```

[The machine makes its opening move.]

(USER: PLAY A ZERO IN THE CENTER SQUARE)

(M: IS ZERO A NOUN MEANING O)

(U: YES)
(M: OK)

```
(X - -)
(- O -)
(- - -)
```

[The new word "zero" is added to the machine's vocabulary.]

(M: I PLACE AN X IN THE BOTTOM RIGHT SQUARE)

(X - -)
(- O -)
(- - X)

(U: <u>SLAP AN O IN THE BOTTOM LEFT CORNER</u>)

(M: IS (TO SLAP SOMETHING SOMEWHERE) A WAY OF SAYING (TO PLACE SOMETHING SOMEWHERE))

(U: YES)
(M: OK)

(X - -)
(- O -)
(O - X)

[A new surface frame (as described in section 4) is added to the set of surface frames which express the concept of making a move.]

(M: I SLAP AN X IN THE TOP RIGHT SQUARE)

(X - X)
(- O -)
(O - X)

[In this sample session, the machine uses its newly acquired vocabulary at every opportunity.]

(U: <u>HOW COME YOU PLACED AN X IN THE TOP RIGHT SQUARE</u>)

(M: IS (HOW COME SOMEONE DOES SOMETHING) AN EXPRESSION MEANING (WHY DOES SOMEONE DO SOMETHING))

(U: YES)
(M: OK)

(M: TO BEAT YOU)

[A new surface frame is added to those expressing the concept 'WHY'. Syntactic features are associated with each such frame indicating such things as whether subject-verb inversion is to be performed.]

(U: CAN YOU CURRENTLY BEAT ME)

(M: IS CURRENTLY AN ADVERB REFERRING TO THE TIME OF A MOVE)

[These are the only adverbs the system knows about. In fact, at present, the system ignores adverbs in its semantic processing. It asks the question because it wants to account for all the words in the sentence.]

(U: YES)
(M: OK)

(M: YES)

(U: AFTER YOU PLAYED AN X IN THE TOP WEST SQUARE COULD I BEAT YOU IF I PLAYED AN O IN THE BOTTOM RIGHT SQUARE)

(M: IS WEST AN ADJECTIVE MEANING)
EITHER
(1: CORNER)
(2: LEFT)

(U: 2)
(M: OK)

(M: NO)

[Notice that two top squares contain an x. However, only the left one was played while the bottom right square was free. Therefore the system concluded that the top left square was being described. This example illustrates how contextual information can be used to ask as intelligent a clarifying question as possible.]

(U: HOW COME YOU PLACED AN X IN THE TOP EAST SQUARE)

(M: IS EAST AN ADJECTIVE MEANING)
EITHER
(1: RIGHT)
(2: WEST)

(U: 1)
(U: OK)

(M: TO BEAT YOU)

[Here, either top square might have been referred to.]

(U: WHEN I PLAYED AN O IN THE BOTTOM LEFT SQUARE COULD YOU BEAT ME)

(M: IS (WHEN SOMEONE DOES SOMETHING) AN EXPRESSION MEANING (AFTER SOMEONE DOES SOMETHING))

(U: YES)
(M: OK)

(M: YES)

[In the domain of this TIC-TAC-TOE discussion, "when" and "after" have the same meaning.]

[The next two sentences each contain two problems for the system to resolve.]

(U: WHEN YOU PLAYED AN X IN THE TOP RIGHT CELL COULD I KEEP YOU FROM BEATING ME)

(M: IS CELL A NOUN MEANING SQUARE)

(U: YES)
(M: OK)

(M: IS (TO KEEP SOMEONE FROM DOING SOMETHING) A WAY OF SAYING)
EITHER
(1: (TO PREVENT SOMEONE FROM DOING SOMETHING))
(2: (TO ALLOW SOMEONE TO DO SOMETHING))
(3: (TO FORCE SOMEONE TO DO SOMETHING))

(U: 1)
(M: OK)

(M: NO)

(U: IF I LAY AN O ON THE RIGHT SIDE SQUARE HOW CAN YOU POLISH ME OFF)

(M: IS (TO LAY SOMETHING SOMEWHERE) A WAY OF SAYING (TO SLAP SOMETHING SOMEWHERE))

(U: YES)
(M: OK)

(M: IS (TO POLISH SOMEONE OFF) A WAY OF SAYING)
EITHER
(1: (TO BEAT SOMEONE))
(2: (TO LOSE TO SOMEONE))
(3: (TO TIE SOMEONE))

(U: 1)
(M: OK)

(M: BY LAYING AN X ON THE TOP CENTER CELL)

[If this same session were subsequently repeated, of course, the system would not need to ask any questions of the user.]

2.1 Comments on Current Limitations

There are a number of limitations to the adaptive system as it now stands. Some of these may be apparent in the sample session, but an introduction to the system is not complete without discussing them explicitly.

(1) The number of concepts available to the system at present is very small. This, in fact, is why the system's first guess is usually the correct one. If the sentence is at all within the system's comprehension, the options as to its meaning are currently quite limited.

(2) The range of expressive devices presently recognized is quite limited as well. For instance, the system does not recognize relative clauses, conjunctions, or pronouns (except for I and you).

(3) The system currently deals only with TOTALLY UNFAMILIAR words and expressions in this adaptive fashion. It will not correctly handle familiar words which are used in new ways (such as a noun used as a verb, as in "zero the center square").

(4) The system tries to map the meaning of new words and expressions into its specified set of underlying concepts. It then displays its hypotheses to the user, giving him only the option of saying yes or no. The user cannot say "no, not quite, it means ...". (Thus concepts like "the 'northeast' square" or "the 'topmost' square" would be confusing and not correctly understood.)

The present simple system has been developed with two goals in mind: (1) to explore the techniques required to achieve adaptive behavior, and (2) to help formulate the issues which will have to be faced when incorporating these techniques into a much broader natural language system.

3. OVERVIEW

Fig. 1 shows the various stages that the Adaptive System goes through in understanding a sentence. In this section, we shall watch while the system processes the sentence "How come you placed an x in the top right square."

(1) Local Syntactic Processing:
In this first stage, the system scans the entire sentence looking for local constituents. These include "simple" noun phrases (NPs) and prepositional phrases (PPs), ("simple" meaning "up to the head noun but not including any modifying clauses or phrases"), and verb groups (VGs) consisting of verbs together with any adjoining modals, auxilliaries, and adverbs. In this instance, the system finds the two NPs, "you" and "an x", the PP "in the top right square", and the VG "placed".

(2) Semantic Clustering:
At this stage, the clause-level processing starts. Unlike most systems, this clause-level processing is driven by SEMANTIC relationships, rather than by syntactic form. It uses a semantics-first "clustering", with a secondary use of syntax for comments and confirmation. In this example, all the local constituents found can be clustered into a description of a single concept: that of making a move. Section 4 describes the mechanics of this stage in more detail.

(3) Cluster Expansion and Connection:
During this stage an attempt is made to account for each word in the sentence by expanding the concept clusters, and if there is more than one, by joining them together to form an entire multiclausal sentence. In this case, the concept cluster might be expanded in two ways.
a) One possibility might be that it is a "HOW" type question, and that "come" is some sort of adverb. However this possibility violates a semantic constraint, since the system is not set up to answer how a move is made; only how to win, how to prevent someone from winning, etc. Therefore this possibility is ignored.
b) The other possibility is that "how come" is a new way of describing some other clause function.

(4) Contextual Inference; Clarification; and Response:
During this final stage, any contextual information available is brought to bear on areas of uncertainty, any necessary clarifying questions are asked, and the system responds to the sentence. In this example, the only uncertainty is the meaning of "how come". Since this is the main

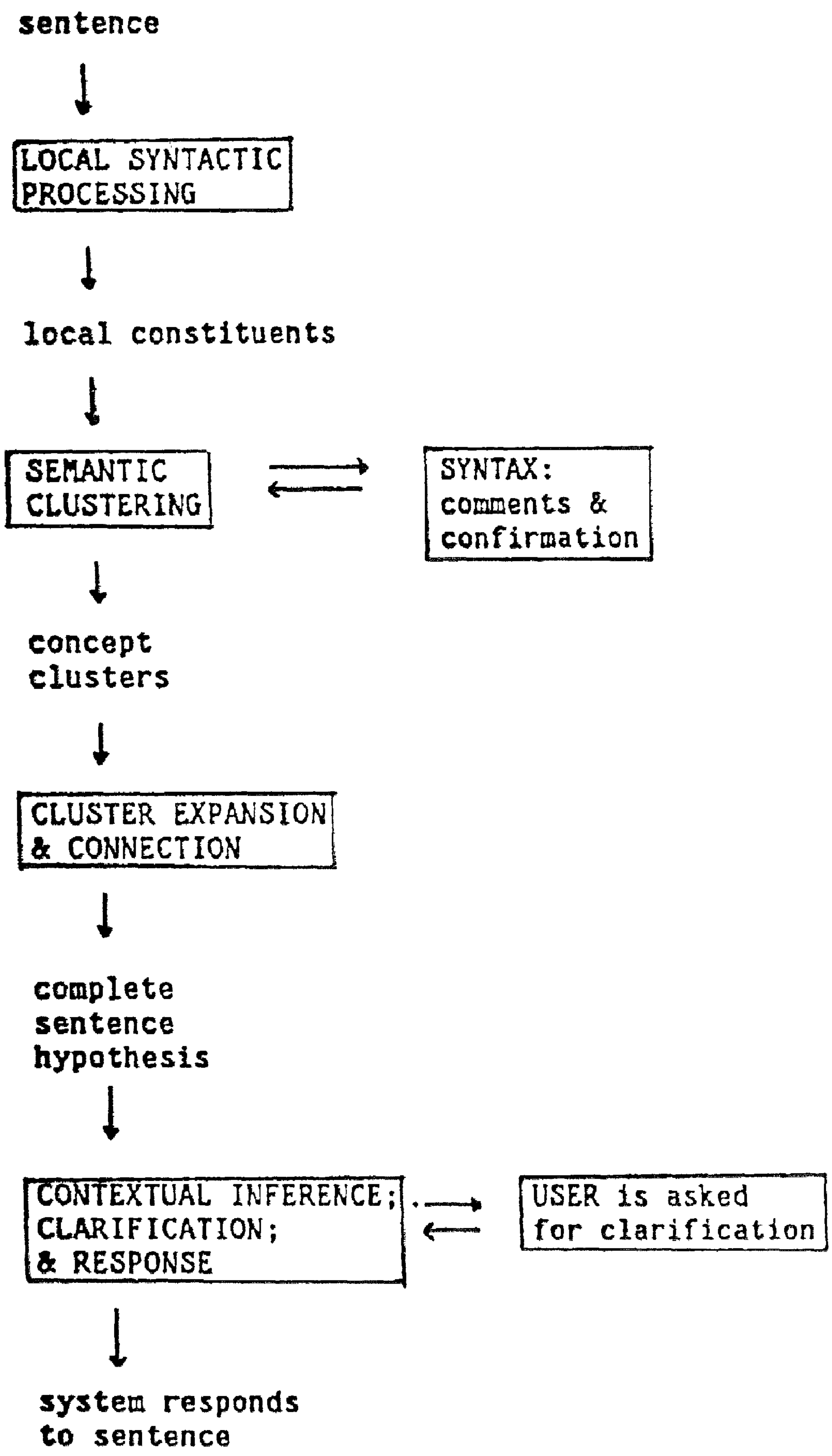

Fig. 1: Adaptive System Overview

clause of the sentence, the possibility of its being an "if" or "after" clause are discarded. The remaining possibilities are "imperative", "how", "why", and "can". The system does not answer "how" and "can" questions in relation to making moves. Similarly, "imperative" does not make sense since the action described is a previously made move. Therefore the system asks if "How come someone does something" means "Why does someone do something". The user answers "yes", so the system stores this new way of asking "why", and proceeds to answer the question.

4. SEMANTICS-FIRST CLAUSE-LEVEL PROCESSING

One of the major differences between this approach to parsing and that of a top-down, syntax-driven system (such as Woods' or Winograd's) is the order in which syntactic and semantic processing is done at the clause level.

In a top-down system, a sentence must exactly match the built-in syntax before semantics can even be called and given the various constituents of a clause. This is clearly undesirable when one is dealing with input uncertainty, since one cannot be sure exactly how the user will phrase his sentence. One would prefer to let semantics operate first on any local consituents present, so that it can make a reasonable guess as to what is being discussed.

As semantically-related clusters of local constituents are found, syntax can be consulted and asked to comment on the relative grammaticality of the various clusters. If there are two competing semantic interpretations of one part of a sentence, and syntax likes one much better than the other, then the "syntactically pleasing" interpretation can be pursued first. Later, if this does not pan out, the syntactically irregular possibility can be looked at as well. In this way, syntax can help guide the system, but is not placed in a totally controlling position.

A by-product advantage of this semantics-first approach is that the system can handle mildly ungrammatical input without any extra work. In addition, the semantics-first clustering approach lends itself quite naturally to handling sentence fragments.

In the remainder of this section, we describe how the adaptive system organizes its linguistic knowledge to implement this semantics-first approach. As we shall see, there are three components of this knowledge.

(a) The local recognizers which initially find local constituents. These recognizers are represented in Augmented Transition Network [11] form, are quite simple, and are not described further in this paper.
(b) Clause-level knowledge of how actions and clause-functions are described. This knowledge is expressed in a descriptive fashion which makes it easily manipulable, and easy to add to.
(c) Clause-level syntactic knowledge which is expressed in a domain-independent form.

4.1 Knowledge of how Actions are Described

Figure 2 illustrates how the system stores its knowledge of how actions (or events) are described. This knowledge is stored at two levels: the conceptual level, and the surface (or expressive) level

As shown in Fig. 2, the concept PLACE represents the act of making a TIC-TAC-TOE move.

(a) On the CONCEPTUAL level, there are three "conceptual slots" indicating the actors which are involved in the action: a player, a mark, and a square.

(b) On the SURFACE, or expressive, level there is a list of surface frames each indicating one possible way that the concept can be expressed. Each surface frame consists of a verb plus a set of syntactic case frames to be filled by the actors.

(Notice that neither the conceptual slots nor the surface frames indicate explicitly the order in which the various constituents are to appear in a sentence.)

When the system processes a sentence, it fills the conceptual slots with local constituents found in the sentence If it has found a familiar verb, then it also gets any surface frame(s) associated with that verb. At this point it calls syntax, asking for comments.

For instance, if the input sentence is "I place an x in the corner", then all the conceptual slots of #PLACE would be filled, and the system would pass the following string to syntax "agent verb obj pp". As a result, clause-level syntax does not see the actual constituents of the sentence, only the labels specified in the surface case frame, plus information indicating number, tense, etc.

An interesting aspect of this approach is that the clause-level syntax is entirely domain-independent. It knows nothing about TIC-TAC-TOE, or even about the words used to talk about TIC-TAC-TOE. The surface frames allow semantics to talk to syntax purely in terms of syntactic labels. As a result, one could write a single syntactic module, and then insert it unchanged into many domains.

4.1.1 Using this Information

In this section, we describe in more detail how this knowledge can be used when processing a sentence.

(1) If the verb and constituents are familiar:

If there is no uncertainty in a clause, then each constituent can be put into one of the conceptual slots, and any surface frames associated with the verb can be examined The frame indicates the case (agent, object, etc.) associated with each constituent when that verb is used. The frame is used to create a string of case labels that are sent to syntax for comments.

For instance, if the sentence is "I place an x in the center

CONCEPT: PLACE

CONCEPTUAL SLOTS:

```
P: player
M: mark
S: square
```

SURFACE FRAMES:

```
VERB: place    (as in:
AGENT: P       "I place an x in the center")
OBJ: M
in: S

VERB: play     (as in:
AGENT: P       "I play an x in the center")
OBJ: M
in: S

VERB: play     (as in:
AGENT: P       "I play the center")
OBJ: S
```

Fig. 2: Linguistic Knowledge about Actions

square", the string passed to syntax is "agent verb obj pp". Syntax replies that the sentence follows normal order. Had the string been "verb obj pp", syntax would reply that the subject had been deleted. If the string was "do agent verb obj pp", syntax would reply that subject-verb inversion had taken place. Given "agent obj verb pp", syntax would reply that the object was out of position.

Thus syntax is set up to notice both grammatical and ungrammatical permutations in constituent order, and to comment appropriately. The system must then decide how to interpret these comments.

For instance, if syntax replies that the object is out of position in the clause, or that there is incorrect agreement in number between subject and verb, the system may decide that the user has made a minor grammatical error, and allow the sentence to be processed anyway, especially if there is no better interpretation of the sentence. In this way, clause-level syntax plays an assisting role rather than a controlling role in the analysis of a sentence.

(2) If a constituent is unknown:

If an unknown constituent is present, then both the frame and slot information can be used to help resolve its meaning. For instance, suppose the sentence is "I place a cross in the center square", and the word "cross" is unfamiliar.

Here, during the semantic clustering, the conceptual slots for a player and a square can be filled by "I" and "in the center square", but the slot for a mark is unfilled. In addition, there is the unknown constituent "a cross".

A natural hypothesis, therefore, is that the unknown constituent refers to a type of mark. Since the verb is familiar, a surface frame is available. Next, assuming the unknown constituent is a mark, the string "agent verb obj pp" can be passed to syntax. When syntax approves, this offers additional confirmation that the hypothesis is probably right.

Subsequent evaluation of this hypothesis indicates that the sentence makes sense only if the mark referred to is an x, so the system asks if "cross" is a noun meaning "x".

(3) If the verb is unknown:

If an unfamiliar verb is used, then there is no surface frame available to help guide the analysis. Instead, syntax must be used in a different mode to propose what the surface frame should be.

Suppose the sentence is "I plunk an x in the center square". Here, all the constituents can be clustered into the concept #PLACE, but there is an unknown word, and no verb. The logical hypothesis is that the new word is a verb. A special syntactic module is therefore passed the following string "NP(P) verb(plunk) NP(M) PP(in,S)". This module examines the string and produces a new frame:

```
VERB: plunk
AGENT: P
OBJ: M
in: S
```

The system can then ask if "to plunk something somewhere" means "to place something somewhere", and upon getting an affirmative reply, can add the new frame to those associated with the concept PLACE.

Since the system uses the surface frames to generate its own replies, it can now use this new frame itself when it talks. When the system wants to generate a clause, it passes a selected frame, the constituents, and a list of syntactic features to a clause generator which outputs the specified form. (Thus, clause-level syntax can be used by the system in three different modes: (1) to comment on the grammaticality of a string of case markers, (2) to construct a new surface frame, and (3) to generate clauses when the system itself replies.)

4.2 Knowledge of how Clause-Functions are Described

As illustrated in Fig. 3, knowledge of how clause-function concepts are described is also expressed as two levels.

CONCEPT: #WHY

CONCEPTUAL SLOTS:

ACTION: #PLACE

SURFACE FRAMES:

Why ACTION(SVINV) (as in:
"Why does someone do something")

How come ACTION() (as in:
"How come someone does something")

Fig. 3: Linguistic Knowledge about Clause Functions

Each clause function has a conceptual slot indicating what types of action can be used with that clause type (in this case, the action #PLACE), and a list of surface frames indicating different ways in which the concept can be expressed.

A clause-type frame currently includes any special words which introduce the clause (ie. "why" or "how come"), together with a list of syntactic properties which should be present in the clause. This list of syntactic properties might include SVINV, "subject-verb inversion" (as in "why does someone do something"), or "subject deletion", "ING form", and "use of a particular preposition" (as in "from doing something").

These syntactic features, however, need not be inflexible rules. Sentence understanding can still proceed even if the syntactic features found by syntax do not exactly match those specified by the clause-function frame. Thus, an inadvertent ungrammaticality can readily be recognized as such, and processing can continue.

4.2.1 Using the Clause Function Knowledge

In this section we examine how this clause function knowledge can be used.

(1) With no uncertainty:

If the input sentence is "Why did you place an x in the center square", then during the semantic clustering the string "do agent verb obj pp" is passed to syntax, which replies that subject-verb inversion has taken place.

When examining the whole clause, the system sees that it exactly matches one of the surface frames for a #WHY-type question, since it starts with the word "why" and contains subject-verb inversion.

Suppose, however, the sentence had been "Why you place an x in the center square", or "How come did you place an x in the center square". Each of these sentences matches a surface frame for a #WHY-type question, except that in both cases subject-verb inversion is incorrect. In such a case, the system can, if it chooses, decide that the user has made a minor error, and allow the sentence to be processed anyway. The locally-driven semantics-first approach lets this happen in a natural way.

(2) A new surface frame:

Another problem arises when a new clause introducer is encountered, as in: "Wherefore did you place an x in the center square". Here, as described in section 3, the system hypothesizes that this may be a new way of asking a #WHY-type question. Since syntax reports that subject-verb inversion has taken place, the system can therefore create a new surface frame:

Wherefore ACTION(SVINV)

to be added to the frames associated with #WHY.

4.3 Comments

In summary, the adaptive system stores its linguistic knowledge in a very accessible form. It is not embedded in the parsing logic. Knowledge of how actions and clause-functions are described is represented in a descriptive, manipulable format. Syntax is domain independent, and is used only to make comments, with semantics playing the guiding role. This organization allows the parsing/understanding process to proceed in a flexible fashion.

5. CONCLUSION

Language communication is an inherently adaptive medium. One sees this clearly if one takes a problem to a lawyer and spends time trying to assimilate the related "legalese". One also sees it in any conversation where a person is trying to convey a complicated idea, expressed in his own mental terms, to someone else. The listener must try to relate the words he hears to his own set of concepts. Language has, presumably, evolved to facilitate this sort of interaction. Therefore it is reasonable to expect that a good deal of the structure of language is in some sense set up to assist in this adaptive process. By the same token, studying language from an adaptive standpoint should provide a fresh perspective on how the various levels of linguistic structure interact.

REFERENCES

[1] Davies, D.J.M., and Isard, S.D., 'Utterances as Programs,' presented at the 7th International Machine Intelligence Workshop, Edinburg, June 1972.

[2] Enea, H., and Colby, K.M., 'Ideolectic Language Analysis for Understanding Doctor-Patient Dialogs', Proceedings of the 3rd IJCAI, Stanford, August 1973.

[3] Fillmore, C.J., 'The Case for Case', in 'Universals in Linguistic Theory', Bach and Harms (Eds.), Holt, Rinehart, and Winston, Inc., Chicago 1968.

[4] Joshi, A.K., and Weischedel, R.M., 'Some Frills for the Modal TIC-TAC-TOE of Isard and Davies: Semantics of Predicate Complement Constructions,' Proceedings of the 3rd IJCAI, Stanford, August 1973.

[5] Miller, P.L., 'A Locally Organized Parser for Spoken Input', Comm. ACM 17, 11 (Nov. 1974), 621-630.

[6] Miller, P.L., 'An Adaptive System: for Natural Language Understanding and Assimilation', RLE Natural Language Memo No. 25, MIT, February 1974.

[7] Reddy, D.R., Erman, L.D., Fennell, R.D., and Neeley, R.B., 'The HEARSAY Speech Understanding System', Proceedings of the 3rd IJCAI, Stanford, August 1973.

[8] Walker, D.E., 'Speech Understanding through Syntactic and Semantic Analysis', Proceedings of the 3rd IJCAI, Stanford, August 1973.

[9] Weizenbaum, J., 'Eliza- a Computer Program for the Study of Natural Communication between Man and Machine', CACM 9, 1972.

[10] Winograd, T. Procedures as a Representation of Knowledge in a Computer Program for Understanding Natural Language, MAC-TR-84, Project MAC, MIT, Cambridge, Mass., February 1971.

[11] Woods, W.A., and Kaplan, R.M., 'The Lunar Sciences Natural Language Information System', BBN Report No. 2265, Bolt, Beranek, and Newman Inc. September 1971,

[12] Woods, W.A., and Makhoul, J., 'Mechanical Inference Problems in Continuous Speech Understanding , Proceedings of the 3rd IJCAI, Stanford, August 1973.