GENERATING AND ANSWERING LITERAL ENGLISH QUESTIONS FROM TEXT

Technical Report NL-21

Rob't. F. Simmons & Michael K. Smith

March 1974

NATURAL LANGUAGE RESEARCH FOR CAI

Sponsored by

THE NATIONAL SCIENCE FOUNDATION

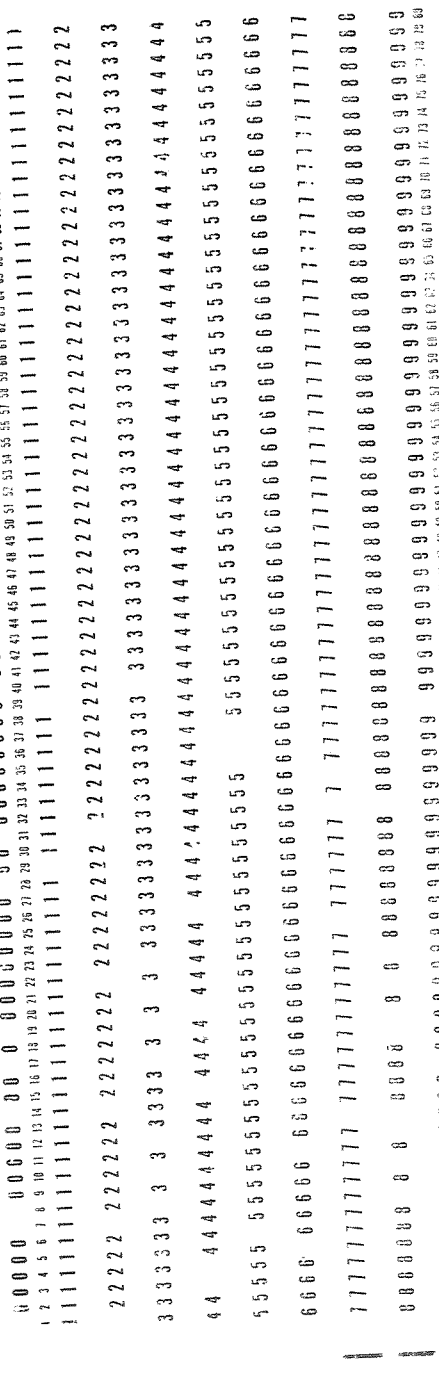
Grant GJ 509X

Department of Computer Science

University of Texas, Austin

ABSTRACT

Methods for generating and answering literal questions from deep case structures for English are described. Since these methods require large dictionaries and grammars an alternative approach is presented that substitutes an editing and indexing program for structuring English text for question generation and answering. An application to AMI, an augmented mixed initiative text teaching system, is developed. Protocols from AMI are presented.



GENERATING AND ANSWERING LITERAL ENGLISH QUESTIONS FROM TEXT

I Introduction

Despite notable progress in automatic analysis for subsets of natural languages, the goal of developing a grammar and semantic system for unrestricted text is still far from achievement. We conceive of a system that reads and analyzes English text, answers questions about it, generates essays or questions and is usable by a teaching or discussion program to support conversation based on its content. Such a system, though still unattainable, would have important applications to automated teaching systems, information retrieval and verbal data management. The purpose of this paper is to develop an approximation to such a system that may have some practical utility.

Generally, our approach to computer understanding and question answering in English has been one of syntactically and semantically analyzing an English string to obtain a deep case structure in the form of a verb, its modality and its arguments. A simple example of this structure can be seen for the following sentence:

E1: The old man often ate rancid fish on Sundays.

(VB EAT, MODALITY (PAST INDIC ACTIVE), AGENT (THE OLD MAN),
THEME (RANCID FISH), TIME (ON SUNDAYS),
FREQ (OFTEN))

An English question such as E2, is analyzed to a comparable form.

E2. What did the old man eat?

(VB EAT, MODALITY (PAST, ACTIVE, INTERROG), AGENT (THE OLD MAN),
THEME WHAT)

Question answering in its simplest form is a Match algorithm that replaces the queried argument in the question, (e.g. THEME), with the corresponding argument in that statement which satisfactorily matches the rest of the question. If the Match algorithm can use rules of inference to determine that two apparently different terms such as "human" in a question and "man" in the answer match satisfactorily then it is a deductive algorithm. If it is limited to literal matching of character strings it can be called a literal Q-A algorithm.

When supported by a generation grammar and a lexicon that shows inflectional variants of words, the deep case structure can be used to generate English strings. By changing the modality values -- active, passive, imperative, interrogative, etc. -- variations of the sentence may be produced. Fragments of sentences sufficient to answer a question may be generated as well.

In recent papers we have explored the automatic analysis of sentences, the accumulation of a data base of deep case structures, and its use for answering questions and generating English sentences. (See Simmons 1973, Hendrix, Slocum & Thompson 1973.) The prominent weakness of the approach is that it requires the use of a complete dictionary and grammar to transform from text strings into deep case structures and no such body of linguistic data is available for English. Many problems in the development of such a linguistic system are still unsolved -- e.g. resolution of pronoun reference and anaphora, selection of relevant word sense, resolution of syntactic and semantic ambiguity, -- so it is not clear that an adequate syntactic and semantic system could yet be constructed for understanding unrestricted text.

Existing approaches are simply not complete enough to believe that expanding them to include dictionaries of twenty thousand words and increasingly large grammars would lead to a text understanding system. Sager's String Analysis Project has been concerned for about a decade with developing sufficient grammar and dictionary to support the analysis of a few scientific articles. (Sager 1972). Hers has been a fairly successful effort, but it does not promise early applications and it requires massive amounts of linguistic data and program reflecting long term efforts by project members.

Other projects concerned with the development of large grammars and dictionaries include those of Petrick(1972), Heidorn (1971), Woods (1970), and Stachowitz (1972), to mention a few. A discouraging fact about these efforts is that the resulting linguistic data, though general in applicability, are highly specialized in format to the requirements of the local system. The consequence is that the existence of several large bodies of linguistic data is not of great help to researchers elsewhere. And these "large" collections of linguistic data in fact account for only relatively tiny subsets of English -- not enough for text. In addition, the systems cited above require on the order of 100K words of program and data and are so far uneconomical for most applications.

These observations imply to me the desirability of searching for an alternate method of structuring textual materials to achieve the ability to answer questions, discuss and teach. The alternate method explored in this paper is to substitute an easily definable and teachable text-editing process for the linguistic tasks of developing grammars, dictionaries, parsers and generators. Section two of this paper presents a method for generating English questions from deep case structure analyses of sentences and outlines the literal Q-A procedure on them. Section 3 develops an alternate method of obtaining a surface query-case analysis via an editing method and its use in generating and answering questions. The concluding section 4 discusses application of the technique to the use of edited text in teaching and question-answering systems.

II Questions & Case Structures

A case structure analysis represents a sentence as a predicate and a set of labelled arguments. A logical analysis of case structure systems by Bruce (1973) suggests quite strongly that the labels are a semantic classification system for the predicates and arguments whose nature is dependent on the purpose for which the system is designed. For a question-answering system, semantic categories for predicates and arguments can serve to classify synonymous paraphrase sets of sentences into a common deep case structure (Schank 1973, Wilkes 1973). A surface syntactic categorization that is generally useful for defining agreement relations and voice and mood transformations is the Subject-Verb-Indirect Object-Direct Object-Complement system that is generally taught in English courses.

Let's examine some analyses of sentence E1.

E1. The old man often ate rancid fish on Sundays.

A1 Fillmore: *

(EAT, MODALITY (INDIC, PAST, ACTIVE, OFTEN, ON SUNDAYS),
AGENT (THE OLD MAN), OBJECT (RANCID FISH))

A2 Surface Cases:

(VB (OFTEN ATE), SUBJECT (THE OLD MAN), OBJECT (RANCID FISH),
COMPLEMENT (ON SUNDAYS))

A3 Schank: *

(INGEST, ACTOR (THE OLD MAN), OBJECT (RANCID FISH)
TIME (ON SUNDAYS, OFTEN) FROM OUTSIDE, TO (IN THE OLD MAN)

A4 Simmons:

(TAKE, VB EAT, MODALITY (INDIC, PAST, ACTIVE),
CAUSAL ACTANT (THE OLD MAN), THEME (RANCID FISH),
TIME (ON SUNDAYS), FREQUENCY (OFTEN))

The surface syntactic analysis offers minimal semantic information while Schank's case analysis not only semantically categorizes the verb and its arguments, it adds the information that the fish came from outside to the old man and is in him. The analyses can thus be ordered on semantic depth: Surface, Fillmore, Simmons, Schank. The deeper the analysis, the more computationally efficient it is for answering questions. If we ask E3:

E3. Who gobbled the fish?

Only the Simmons and the Schank analyses can be expected to find a match between "gobble" and "eat" since they both classify these terms under a

---

* My approximations to Fillmore & Schank

more inclusive verb. If we consider some literal questions on the sentence--Who ate..., What did the old man eat..., When, How often... -- we find that all analyses are equally good for Who and What in that they mark corresponding categories in the analyzed sentence as follows:

(VB ATE, SUBJ WHO, OBJECT FISH)
(EAT, MODALITY (PAST INTERROG ACTIVE), AGENT WHO, OBJECT FISH)
(TAKE, VB EAT, MODALITY (PAST INTERROG ACTIVE)
CASUAL ACTANT WHO, THEME FISH
(INGEST, ACTOR WHO, OBJECT FISH, ...)

In the case of When and How often, we can see that only the two deeper structures furnish the possibility of a direct match by argument name, while the surface and Fillmore structures require searching through the complement and verb or through the modality to discover the match of some element with the question phrase. Schank's analysis is the only one deep enough to answer a How question although with some labor of gathering up the FROM, TO and IN arguments to present an answer of "from outside to inside the man".

Generating Questions: An algorithm for generating a question about any constituent can be defined over each of these case analyses. The algorithm will be simpler to the extent that the constituents are clearly marked. Thus it will be more difficult to generate a How Frequently type question for those analyses that do not include a FREQUENCY case, or a When question for those without an explicit TIME case.

Let's see what is involved in generating questions from analysis A4. First we need a function that can be told what to query, what content to generate in what form. Calls to this function will specify a structure name to indicate the content, a list of changes to the Modality for designing the question form, and the phrase to be queried.

Table 1 shows a property list representation of A4, some lexical structure for "eat", and the LISP function GEN. GEN is called with three arguments: a structure, A4; a list of modality changes, in this case MOOD Interrogative; and Q, an indicator of what is to be queried. Q may take any case symbol as a value to show which argument is to be questioned. Figure 1 shows the top level of an Augmented Transition Net (ATN) grammar, R, that will substitute a question word for the argument Q by using the function WH, and which will front the questioned argument when it occurs. Detail of how such ATN grammars are used to generate sentences, phrases and questions are found in Simmons (1973), Hendrix, Thompson & Slocum (1973). The ATN system expects to be scanning a sentence, but for the generation

```
A4    PRED TAKE
      VB EAT
      MODALITY M1
      CA      C1 (THE OLD MAN)
      TH      C2 (RANCID FISH)
      TIME    C3 (ON SUNDAYS)
      FREQ    OFTEN
```

```
M1    VOICE ACTIVE
      TENSE PAST
      MOOD INDIC
      ASPECT NIL
```

1a Property list representation of A4

```
EAT   W/C VB
      CLASS TAKE
      CASE ((ACTIVE CA V TH OPTS)
           (PAS TH V "by CA OPTS))
      PAST ATE
      PARTC EATEN
```

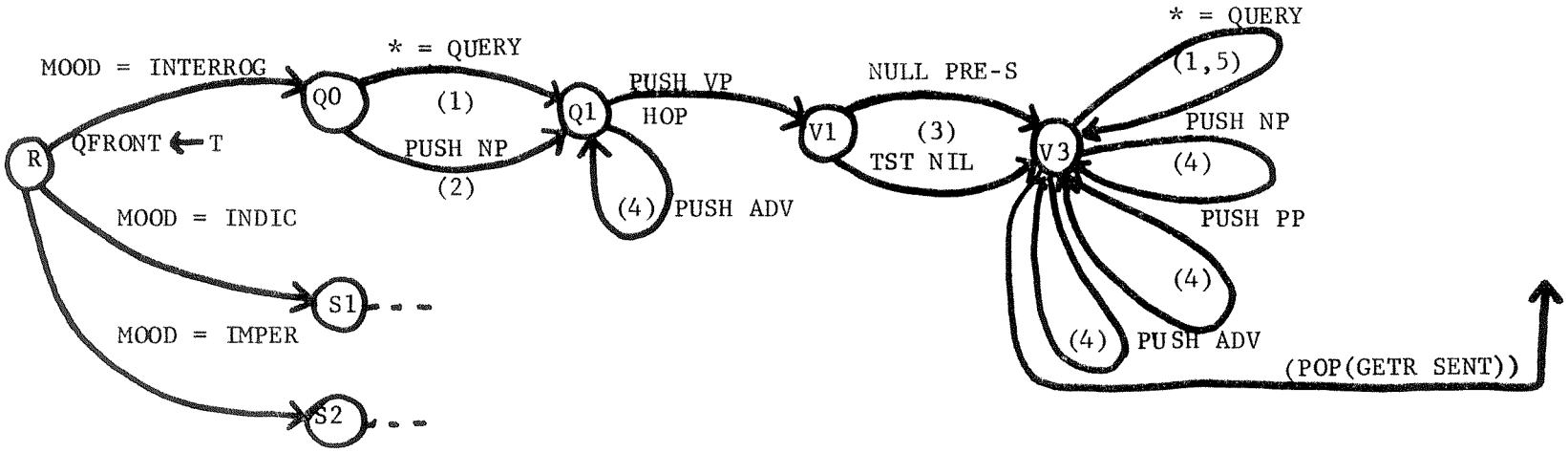
1b Some Lexical Structure for "eat"

Table 1 Generating Questions

(DEFINE((GEN (LAMBDA (ST CHANGES Q) (PROG (ST1 J) (COND ((NULL (SETQ ST1(GET ST MODALITY))) (RETURN NIL))	GEN (A4 (MOOD INTERROG) CA) ST = A4 Q = CA Set ST1 to Modality of ST ST1 = M1
(T (CHANGE1 CHANGES ST1)))	Changes M1 MOOD to INTERROG
(SETQ J (GET (GET ST VB) CASE))	Gets case generation patterns from verb "eat"
(COND ((NULL J)(RETURN NIL))) A1 (COND ((NULL J)(RETURN NIL)) ((NOT (EQ(CAAR J)(GET ST1 VOICE)))) (T (SETQ J (CDR J))(GO A1))	Checks first element of case generation pattern to see if it applies to the voice of M1, else tries another rule
(T (SETQ SNT (CDAR J))))	Put remainder of case generation rule in SNT which will be used in the ATN grammar R
(SENDR Q) (SENDR ST)	Send the values of Q & ST to grammar R
(RETURN (PUSH R SNT)))))	Return what R generates
(CHANGE1 (LAMBDA (LST ST) (COND ((NULL LST) NIL) (T (CONS (PUT ST (CAAR LST)(CDAR LST)) (CHANGE1 (CDR LST) ST)))))))	CHANGE1 changes the modality according to changes

1c The LISP Function GEN

Table 1 Generating Questions (con't.)



Operations:

(1) (SETR PRE-S (WH QUERY))

(2) (SETR SENT (LIST *))

(3) (SETR SENT (CONS (AUX *) (GETR SENT)))

(4) (SETR SENT (LIST (GETR SENT) *))

(5) (SETR SENT (LIST (GETR PRE-S)(GETR SENT)))

Note: Test are shown above the arcs; operations by numbers or expressions below.

Figure 1 Top Level of ATN Grammar for Generating Questions

application it requires a case pattern for the scanner to operate on and a semantic structure to which that pattern refers. If GEN pushes to R with SNT set to (CA V TH OPTS), and ST=A4 and Q=TH, the ATN sets its scanner to CA of the pattern which is the content of a register named *, the star register. The first arc leaving R tests if MOOD of A4 is INTERROGative. The operation below this arc is to set a flag, QFRONT, to the value True and then to transfer control to state Q0. Leaving Q0 we first test to see if the value of the argument in the * register --CA-- is equal to the value of the register Q -- TH. Since they are not this test fails, and the next arc, PUSH NP is tried. The NP subroutine develops the string "THE OLD MAN" and returns it to the top in the * register. The operation under this arc is (2), which sets a register SENT to "THE OLD MAN", and transfers to Q1, moving the scanner to V in the generation pattern.

At Q1, PUSH VP is successfully crossed and returns "ATE" in *. Generally VP will return a string of auxiliaries and a verb when successful, so from a different structure it might have returned, "HAD BEEN EATING". Without changing the scanner, control is HOPped to V1. The first exiting arc checks the register PRE-S to see if the subject of the sentence has been queried; if not, it applies the function AUX to what VP returned in the *register to obtain an auxiliary verb with which to front the sentence. For "ATE", AUX returns "DID" and changes "ATE" to "EAT"; for "HAD BEEN EATING", it returns "HAD". Control is transferred then to V3 and the Scanner is moved to TH in the pattern. Exiting V3 we discover Query = TH matches *, so PRE-S is set to "WHAT" and concatenated to SENT. SENT now contains "WHAT DID THE OLD MAN EAT". Since V3 loops back on itself, the scanner moves to OPTS in the pattern and the rest of the sentence is generated by PUSHing NP,PP and ADVerb.

Since the SNT pattern was (CA V TH OPTS) we would end up generating, "WHAT DID THE OLD MAN EAT ON SUNDAYS OFTEN". To get the original sentence the pattern would have to be (CA FREQ V TH OPTS) and the grammar R of Figure 1 would have to be modified to include an adverb loop at state Q1. The system of Table 1 and Figure 1 will accept the following calls and generate questions as shown below.

GEN (A4 (MOOD INT)CA) → WHO OFTEN ATE RANCID FISH ...?
GEN (A4 (MOOD INT)TH) → WHAT DID THE OLD MAN OFTEN EAT ...?
GEN (A4 (MOOD INT)FREQ) → HOW FREQUENTLY DID THE OLD MAN ...?
GEN (A4 (MOOD INT)TIME) → WHEN DID THE OLD MAN OFTEN EAT FISH?
GEN (A4 (MOOD INT)VB) → DID THE OLD MAN OFTEN EAT ...?

Phrases: It is convenient to consider a noun phrase or noun group as an ordering of case labels. Winograd (1972) for example specifies a noun group as follows:

DET ORD NUM ADJ* CLASF* NOUN Q*

Thus a case structure for "the old man from Spain" can be represented as:

(N MAN,NBR SING, ADJ OLD, DET THE PP (from Spain))

To generate a noun phrase we first seek a case label, DET and generate its value as an English word, then unsuccessfully seek an ORD and a NUM, then an ADJ string which finds OLD, then fail on a CLASF, then find a noun MAN, then a Q which is satisfied by a PP which then calls a PP pattern to generate one or more prepositions followed by a noun phrase.

Adjective groups are not simply strings as indicated by the ADJ* above, they actually have their own case ordering rules as in "the big heavy red firetruck" which is dominated by a case ordering rule such as (SIZE WEIGHT COLOR). A detailed linguistic analysis of adjective ordering is presented by ( ).

To question constituents of the noun group requires a more complex fronting transformation than is used at the top level of the sentence. For example:

Where was the man from who ate rancid fish on Sundays?

What age was the man from Spain who ate ...? Although this can be fairly easily accomplished in an ATN grammar, it is much simpler not to front, and so generate:

The old man from Where ate rancid fish...

What kind of man from Spain ate What kind of fish...

Verb strings have their own complex ordering rules which are signalled in our system by the case values for TENSE, ASPECT, FORM, MOOD, MODAL, NEG, etc. Since the verb string is often composed of discontinuous constituents and inflected roots, this generation pattern is best expressed in an ATN rather than as a case ordering pattern. We have discussed it in some detail in Simmons & Slocum (1972).

Answering Questions: A literal question answering algorithm for a case system must first retrieve one or more sentence analyses from a set of analyses that represent a text -- usually a semantic network representing the discourse. This is accomplished by associating with each dictionary entry a U/I (used in) attribute with a list of structure names -- such as A4 -- as its value. A function CANDS (candidates) is defined to order the set of answer structures according to the number of lexical entries they have in common with the question.

The next step is to match the question structure against the structure of each candidate sentence until one or more candidates -- or none -- prove to include the question structure. The final step is to test the value of the questioned case for correspondence with the constraints imposed by the question, word, or phrase; e.g. singular, plural, definite, indefinite, possessive, person, place, thing, sentence, etc. If this match is successful the value of the questioned case is known to be an answer. This algorithm and its generalization to deductive question answering is published and described in more detail elsewhere (Simmons 1973).

III An Alternate Approach

The previous section described methods for generating and answering questions from a case structure analysis of sentences and expressed a pessimistic view for the near future about the possibility of constructing a system to automatically analyze unrestricted text. Frequently applications researchers such as Wexler (1970) and Carbonell (1970) have constructed by hand semantic representations to be used in their dialogue systems, and there is an already hoary tradition in Q/A research of avoiding text analysis by just such methods. Our alternate approach is in this tradition. It substitutes human editing of text with a text editor for the use of an automatic parser, and allows for partial analysis of text sentences and paragraphs. It generates questions and answers them to the extent that the editing instructions allow. If it fails to identify a direct answer to a question, it provides the text string that is most relevant to the question as a default value.

The heart of this approach is a method for editing the text into useful question answering and generation structures. A fluent English speaker finds it very easy to question almost any constituent in an English sentence. Edit provides him with a string substitution language to communicate a question structure to the program. A variation of example E1 can be used again to demonstrate.

E1. The old man from Spain often ate rancid fish on Sundays.

The form of the Edit language is as follows:

(Label(String1 : String2)(Answer))

(E1.1 (WHO : THE $ SPAIN)(OLD MAN))

E1.1 is interpreted by a question generator to modify E1 as follows:

Who often ate rancid fish on Sundays?

If this question is presented to a student, the system will evaluate his response as correct if it contains both "old" and "man". If a user asks the question, "Who ate fish?", the system will use its retrieval logic to find E1 and other relevant sentences, then the question Match program can recognize the correspondence of the WHO query to E1.1 and return as answer, "the old man from Spain".

We can systematically question each constituent of the sentence with the following editing instructions:

(E1.1 (WHO: THE $ SPAIN)(OLD MAN))
Who often ate fish on Sundays?

(E1.2 (WHERE: SPAIN)())
The old man from where often ate fish on Sundays?

(E1.3 (WHAT AGE: THE OLD)(OLD))
What age man from Spain often ... Sundays?

(E1.4 (HOW FREQUENTLY:OFTEN)())
The old man from Spain how frequently ate ... Sundays?

(E1.5 (WHAT: FISH)())
The old man ... often ate what on Sundays?

(E1.6 (WHAT KIND OF: RANCID)())
The old man ... ate what kind of fish on Sundays?

(E1.7 (WHEN: ON SUNDAYS)(SUNDAYS))
The old man ... ate rancid fish when?

(E1.8 (WHY IS IT THAT: NIL)(A DOCTOR PRESCRIBID IT))
Why is it that the old man ... on Sundays?

(E1.9 (HOW DID HE EAT FISH: $) (WITH A FORK))
How did he eat fish?

(E1.10 (DID WHAT: ATE $ SUNDAYS)((OR INGEST EAT TAKE) FISH))
The old man ... often did what?

Several features are apparent in the above examples. First, there is no query fronting so some questions such as "The old man ... how frequently ate fish on Sundays?" are stylistically awkward. To avoid this, a more redundant instruction can be used:

(E1.11 (HOW FREQUENTLY DID THE OLD MAN EAT: THE $ ATE)(OFTEN))
How frequently did the old man eat fish on Sundays?

or as we chose in our prototype implementation, the query term can be marked in the presentation as follows:

THE OLD MAN * HOW FREQUENTLY * ATE FISH ON SUNDAYS.

The symbol, $, stands for any intervening string. In the answer portion of the instruction a list of words (A B C) is taken to require A and B and C. A parenthesis followed by an OR, followed by a list as, (OR A B C) will accept any member as an answer. Where nothing is in the answer parenthesis, the string to the right of the colon is scanned for a list of roots required in the answer. A period to the right of the colon means to add Stringl to the end of the sentence; a $ to the right, means substitute Stringl for the

entire sentence, and NIL in that position means write String1, then write the sentence.

Instructions E1.7 and E1.8 introduce new material answering the Why and How of the sentence. E1.9 queries the verb and its object and allows; "ingest" or "eat" or "take"; and "fish" as the answer.

It is possible to apply sequences of instructions, such as 1.3, 1.2, 1.4, 1.6, 1.5 and 1.7 to generate:

WHAT AGE MAN FROM WHERE HOW FREQUENTLY ATE WHAT KIND OF WHAT WHEN?

Questions that simultaneously query two or three elements of a sentence may have some use as complex operators in a teaching program.

The question answering algorithm (RESPOND) that uses the edited text first scans the user's input string to discover an initial question word or phrase. If none is found it takes the input as a comment and returns the string of text most relevant to the comment. In asking questions the user is required to begin his question with a question word; WHO, WHAT, WHEN DID, IS, etc. Thus, in a question such as:

Who ate fish which was rancid?

the scanner picks up "Who" and ignores the "which".

The question answering algorithm corresponds closely to that described earlier for application to case structures, but it has available less information on which to base a decision that a direct answer is present. Like the MATCH algorithm it is supported by an indexing system and a CANDidates function. The indexing system takes each content word of text, strips it to root form and creates for each root form a list of used-in, (U/I), values that show each paragraph and sentence number in which the root was used. The CANDS function transforms the input question to a question phrase and a set of content root forms. Each root is looked up in the index to obtain the list of text strings in which it was used. The and/or union and intersections of these lists are computed to order the text strings with respect to the number of question root forms in each. The strings with the greatest number of intersections with the question are returned as the best candidates for answers.

The Q-A Match algorithm is called if the best candidate accounts for more than fifty percent of the question roots. Otherwise the best candidate is returned. Q-A Match examines the edited text to determine if the query word of the question has been used in marking the string of text. An example will show the method.

What did the old man eat?

The query register contains WHAT; the root content forms are "old" "man" and "eat". The function CANDS retrieves E1 with a score of 3/3 hits on the question roots. From our previous editing example the following questions are associated with E1:

```
(E1.3(WHAT AGE : THE OLD)())
(E1.5(WHAT : FISH)())
(E1.6(WHAT KIND OF : RANCID)())
(E1.10(DID WHAT : ATE $ SUNDAYS)...)
```

Two more heuristics are used in deciding if a direct answer exists. The first is that the query word or phrase of the question must exactly match

a query term in the edited string. The second is that the answering phrase may not contain root forms present in the question. According to these criteria, E1.3, E1.6 and E1.10 are immediately rejected as not matching the query term of the question. E1.5 is acceptable as a matching "WHAT" and its answering term, "fish" is not present in the question. The short answer, FISH is returned as the answer.

It is immediately apparent that this combination of heuristics may fail on occasion. Suppose the example sentence had been,

E1. The old man from Spain often ate rancid fish on the wharf.

and that E1.7 generated the following WHAT question:

(E1.7 (WHAT : WHARF)(OR WHARF DOCK))

This WHAT question cannot be rejected by the heuristics given in the last paragraph. The system must return the default value, E1 with some such comment as "Can't distinguish between two whats". Although with experience additional heuristics may become apparent to resolve this problem, the difficulty can often be foreseen and avoided during editing. A more useful editing would be the following:

(E1.7 (WHERE : ON $ WHARF)(OR WHARF DOCK))

With this instruction the failure is avoided.

Theoretically, we can derive editing conventions that will almost invariably identify a constituent by some combination of question words and auxiliaries or prepositions. Then if the questioner is sophisticated with respect to the editing conventions, the system will be correspondingly more precise in its answering capability. But in application to general users such as students or information retrievalists, we have no right to assume understanding of the editing conventions and can only expect an appropriate use of the query words. Experience with actual linguistic habits of ordinary users can be expected to result in additional heuristics based on auxiliary terms related to the question, but these must wait on the accumulation of data.

The addition of a dictionary entry for each root form indexed can allow the system to accomplish some levels of inference beyond the direct answer match. If for each root form, superset chains are available through the dictionary we can define algorithms that can detect the correspondence between

What person devoured a fish?

and

The old man ate fish on Sundays.

The prospect, however, for introducing dictionary entries is clouded by the usual difficulties of accumulating data and distinguishing among sense meanings of root forms. It suggests an attractive but not necessarily immediately productive line of research.

IV An Application

In accordance with our aim to develop a useful text-based conversational teaching system, we have embedded the literal question generation and answering techniques in an experimental program. From Carbonnell (1970) we adopted the notion of a mixed initiative teaching system and we augment his control scheme with an explicit text-based teaching program. The result is called AMI for Augmented Mixed Initiative.

AMI consists of four routines, Index, Respond, Edit and Teach. The logic of Edit, Index and Respond was described in the previous section. Teach is a system that interprets a language that is used to construct a branching teaching or discussion program that specifies a series of operations such as Tell, Query, Garble, Puzzle, etc. on strings of edited text. It allows interruptions by student questions or comments to which it responds with direct answers or relevant text strings. It evaluates and scores student answers to the questions it generates, accumulates his experience and reports his accomplishments. Teach differs but little from the run of instructional languages except that its operations are defined over a textual data base.

A Teach program is a list of statements that have the following format:

(label) Operator (Identifier) (Answer)

Ex. STRT QGEN E1 WHEN AND- MARCH 1933

The parentheses indicate optionality.

In LISP the example statement has the following appearance:

STRT (QGEN E1 (WHEN) (AND- MARCH 1933))

It is an instruction to generate from text string E1 a question, substituting WHEN in string E1 for the portion of the string marked with the label, WHEN, and to require of the student's answer both MARCH and 1933.

The syntax for the present implementation of the Teach language is as follows:

STATEMENT	:=	(LABEL) & OPERATOR & (IDENTIFIER)
LABEL	:=	ATOM
OPERATOR	:=	TELL/QGEN/SCORE/S = /...
IDENTIFIER	:=	SPECIFIER & (ANSWER)
SPECIFIER	:=	STRING / TEXT ADDRESS & (QELEMENT)
ANSWER	:=	STRING / BOOLEAN & ANSWER*

```
BOOLEAN        :=   AND- / OR-
QELEMENT       :=   QWORD / STRING
STRING         :=   ATOM*
QWORD          :=   WHO / WHAT / WHERE / WHEN / ...
```

The symbols used above are defined as:

:=	-	defined as
()	-	optional argument
&	-	and
/	-	or
*	-	one or more

The present implementation is a first prototype programmed in UT LISP by Michael Smith. Our aim in this computational experiment is to study the gross behavior of the system in presenting text questions under programmed control and to study the effectiveness of the question answering and retrieval logic on small texts.

This prototype uses an ordinary program editor to mark the text strings instead of the program EDIT, that we described earlier. The convention of allowing the keywords in the questioned phrase to be required of the answer is used unless either an answer is marked in the phrase or in the teaching program. These features will become clear in the examples below. *

E5 THE PRESIDENT OUTLINED HIS NEW DEAL PROGRAM IN A CRISP INAUGURAL ADDRESS.

The editor is used to modify E5 as follows:

E5 (WHO THE PRESIDENT (= ROOSEVELT)) OUTLINED (WHAT HIS NEW DEAL PROGRAM) IN (WHAT A CRISP INAUGURAL ADDRESS)

Notice that (= ROOSEVELT) is the reference for this contextual use of "THE PRESIDENT" and also the answer to the WHO question.

A teaching instruction involving E5 is as follows:

```
(QGEN E5 (WHO))
```

This generates:

WHO OUTLINED HIS NEW DEAL PROGRAM IN A CRISP INAUGURAL ADDRESS?

The answer is ROOSEVELT. If we wish to override this answer the following instruction is used:

```
(QGEN E5 (WHO)(AND- FRANKLIN ROOSEVELT))
```

Now both FRANKLIN and ROOSEVELT are required in the correct answer. From

---

* For clarity of discussion we prefix E to the numbers the system actually uses.

the same example we can generate other questions.

(QGEN E5 (NEW DEAL))

ROOSEVELT OUTLINED *WHAT* IN A CRISP INAUGURAL ADDRESS/

The specifier (NEW DEAL) in the instruction identified the marked phrase to be questioned. Since no answer string was given in the instruction or in the editing, the content keywords, NEW DEAL PROGRAM, are required in the correct answer. In a similar fashion we may generate:

(QGEN E5 (INAUGURAL)(AND- INAUGURAL (OR- SPEECH ADDRESS)))

ROOSEVELT OUTLINED HIS NEW DEAL IN *WHAT*

The answer is INAUGURAL SPEECH or INAUGURAL ADDRESS.

An example segment of a Teach program and some edited text sentences it refers to are shown in Tables 3 & 4. The initialization conditions are shown in the function, STRATEGY. If *FILL were set to T, then each time the student missed part of an answer, a fill-in-the-blank would be generated for the remainder of the answer. In the last question above, for example, the student might reply, SPEECH. Since this is only half the answer, the program would generate:

HIS _____ SPEECH

The second strategy option, *CRATIO, is the proportion of keywords required in an answer for it to be scored correct when no specific answer has been specified. The value .49 means that if half or more of the required keywords are given, the system will respond, OK and go on.

Additional features to be seen in the program in Table 3 include the use of (=AP), a special function for use with appositional lists, and the operators, S= & F= which are predicates that test the last answer as Satisfactory or Failed conditionally transferring control to the statement on the right. The operator SCORE prints the accumulated score of the student for this session. REVIEW is designed to hold the student responsible for the results of his questions to the system. The answers retrieved are saved in the variable REVIEWLIST. The operator REVIEW runs through this list generating successive questions and evaluating responses.

```
(
(STRATEGY
 (*FILL T)
 (*CRATIO 0.49))

(PROGRAM
 (TELL (THIS IS A REVIEW OF PAGES 410 TO 411 IN
         *THE RISE OF THE AMERICAN NATION*))
 (QGEN 3 (WHO))
 (QGEN 1 (WHEN) (AND= MARCH 1933))
 (QGEN 5 (NEW DEAL))
 (QGEN 7 (=AP))
 (TELL 10)
 (TELL 11)
 (QGEN 12 (WHAT) (AND=(OR= ASSIST ASSISTANCE)
                       (OR= JOBLESS HUNGRY) AMERICANS))
 (S=   (GOTO A))
 (QGEN 13 (HOW-MANY) (AND= (OR= 13 14 15) MILLION))
A (QGEN (DO YOU WANT TO SKIP AUTOMATIC
         GENERATION) (YES))
 (S=   (GOTO B))
 (QGEN 14)
 (QGEN 15)
 (QGEN 16)
 (QGEN 17)
 (QGEN 18)
B (TELL 21)
 (TELL 22)
 (QGEN 24 (BOONDOGGLING))
 (TELL 25)
 (TELL 26)
 (QGEN 27 (WPA) (AND= WPA))
 (TELL 30)
 (QGEN 32)
 (QGEN 33 (WHEN))
 (SCORE)
 (REVIEW)

END))
```

Table 3. An Example Teaching Program

```
(CHAP17

( 1   (WHO PRESIDENT FRANKLIN DELANO ROOSEVELT) TOOK OFFICE
      (WHEN ON MARCH 4 1933) IN THE MIDST OF (WHAT THE GREAT
      DEPRESSION))
( 2   (WHO HE (= ROOSEVELT)) BEGAN HIS ADMINISTRATION WITH
      (WHAT A RINGING SUMMONS TO THE AMERICAN PEOPLE TO
      FACE THE FUTURE WITH COURAGE AND FAITH))
( 3   (QUOTE THE ONLY THING WE HAVE TO FEAR IS FEAR ITSELF)
      (WHO HE (= ROOSEVELT)) CONFIDENTLY (VPA STATED))
( 4   (WHAT HIS (= ROOSEVELTS) CALM WORDS) HELPED TO LIFT THE
      NATION FROM (WHAT ITS DESPAIR) AND HELPED TO RALLY THE
      PEOPLE (WHERE BEHIND THE GOVERNMENT))


( 5   (WHO THE PRESIDENT (= ROOSEVELT)) OUTLINED (WHAT HIS
      NEW DEAL PROGRAM) IN (WHAT A CRISP INAUGURAL ADDRESS))
( 6   (WHO HE (= ROOSEVELT)) THEN PRESENTED HIS REFORM PROPOSALS
      WITH (WHAT RECOMENDATIONS FOR IMMEDIATE ACTION) TO
      (WHAT A SPECIAL SESSION OF CONGRESS) THAT HE CALLED SOON
      AFTER TAKING OFFICE)


( 7   (WHAT THE NEW DEAL) HAD IN GENERAL THREE AIMS - (=AP)
      (WHAT RELIEF RECOVERY AND REFORM))
( 8   BECAUSE (WHY AMERICANS WERE CLAMORINNG FOR ACTION) (WHAT
      THE THREE AIMS (= RELIEF REFORM AND RECOVERY)) WERE
      OFTEN MIXED TOGETHER AS (WHAT OBJECTIVES OF A SINGLE ACT
      OF CONGRESS))
( 9   SOMETIMES MEASURES ADOPTED TO REALISE ONE OF THE AIMS
      INTERFERED WITH (WHAT OTHER MEASURES DESIGNED TO
      ACHIEVE THE OTHER AIMS))


( 10  BUT FOR PURPOSES OF ANALYSIS IT IS CONVENIENT TO
      DIVIDE THE NEW DEAL INTO (WHAT ITS THREE ESSENTIAL
      PARTS) (=AP) (WHAT 1 MEASURES TO PROVIDE RELIEF FOR THE
      UNEMPLOYED 2 MEASURES TO SPEED THE RECOVERY OF AGRICULTURE
      AND 3 MEASURES TO REMEDY CERTAIN WEAKNESSES IN THE
      ECONOMIC SYSTEM))

(E11  SUCH WAS THE NATURE OF THE GREAT EXPERIMENT THAT (WHO
      PRESIDENT ROOSEVELT AND CONGRESS) LAUNCHED (WHEN IN 1933))
)
```

Table 4. LISP Edited Text

The following several pages present a protocol of AMI's operation on pages 410 - 413 of "Rise of the American Nation" a high school level history text. The teaching program is the one illustrated in Table 3. Under conditions of fewer than 50 users on the UT TAURUS timesharing system, AMI operates with one or two second response times. It -- and for this version its index -- reside in about 20k decimal core in UT LISP. A strong sense of its behavior can be obtained by reading the protocol.

In the following protocol the system begins by presenting questions to the student and evaluating his answers according to the teaching program in Table 3. At various points the student interrupts by typing START QMODE and typing in his own question. For our convenience in analyzing its behavior, the system introduces its response with the values of three variables: QLENGTH shows the number of roots in the question, MATCHES shows the number of hits in the answer, and PSET shows the list of text sentences with that number of hits. The student occasionally types MORE to obtain additional answers from the PSET. Occasional comments from LISP tell of garbage collections and, in one case, the time-sharing system, TAURUS, warns that it is going off the air.

```
(YOUR RESPONSES TO THIS TEACHING PROGRAM MUST BE OF THE FOLLOWING FORM
 - A STRING OF WORDS ENDED BY A SPACE AND THEN A COMMA * IN ORDER TO
GET INTO QUESTION ASKING MODE SIMPLY TYPE IN START QMODE , * TO END
QMODE TYPE END QMODE , * IF YOU WANT TO RUN SOME LISP FUNCTIONS TYPE F
 , - THEN TYPE IN THE FUNCTIONS AS YOU NORMALLY WOULD IN LISP * IF YOU
 WANT THE RESULTS PRINTED YOU NEED TO TYPE (PRINT (FUNCTION)) * TO END
 THE FUNCTION MODE TYPE END *)

*********

(THIS IS A REVIEW OF PAGES 410 TO 411 IN *THE RISE OF THE AMERICAN
NATION*)

(THE ONLY THING WE HAVE TO FEAR IS FEAR ITSELF *WHO* CONFIDENTLY
STATED)
*
ROOSEVELT ,

CORRECT

(PRESIDENT FRANKLIN DELANO ROOSEVELT TOOK OFFICE *WHEN* IN THE MIDST
OF THE GREAT DEPRESSION)
*
MARCH 1933 ,

CORRECT

(ROOSEVELT OUTLINED *WHAT* IN A CRISP INAUGURAL ADDRESS)
*
THE NEW DEAL ,

///// GARBAGE COLLECTION: 654 2128

(OK - FULL ANSWER FROM TEXT = (HIS NEW DEAL PROGRAM))

(THE NEW DEAL HAD IN GENERAL THREE AIMS - - WHAT WHERE THEY)
*
RELIEF REBUILDING ,

(FILL IN THE BLANKS)
(RELIEF RECOVERY AND -----)
*
RECONSTRUCTION ,

(RELIEF RECOVERY AND REFORM)
```

Table 5. Protocol of Session with AMI

```
(PUT FOR PURPOSES OF ANALYSIS IT IS CONVENIENT TO DIVIDE THE NEW DEAL
INTO ITS THREE ESSENTIAL PARTS 1 MEASURES TO PROVIDE RELIEF FOR THE
UNEMPLOYED 2 MEASURES TO SPEED THE RECOVERY OF AGRICULTURE AND 3
MEASURES TO REMEDY CERTAIN WEAKNESSES IN THE ECONOMIC SYSTEM)

(SUCH WAS THE NATURE OF THE GREAT EXPERIMENT THAT PRESIDENT ROOSEVELT
AND CONGRESS LAUNCHED IN 1933)

(THE MOST URGENT TASK FACING ROOSEVELT WHEN HE TOOK OFFICE WAS *WHAT*)
*
START QMODE .

ASK-AWAY
WHAT WAS THE MOST URGENT TAK-SK FACING ROOSEVELT WHEN HE TOOK
OFFICE .
(QLENGTH= 8)
(MATCHES= 8)
(PSET= (140))
(TO PROVIDE ASSISTANCE FOR MILLIONS OF JOBLESS HUNGRY AMERICANS)

NEXT-QUESTION
END QMODE .

(BACK TO THE OLD QUESTION)
(THE MOST URGENT TASK FACING ROOSEVELT WHEN HE TOOK OFFICE WAS *WHAT*)
*
PROVIDE ASSISTANCE FOR MILLIONS OF JOBLESS HN-UNGRY AMERICANS .


///// GARBAGE COLLECTION: 620 1969

CORRECT

(DO YOU WANT TO SKIP AUTOMATIC GENERATION)
*
YES .

CORRECT

(THE FEDERAL GOVERNMENT ATTACKED THE PROBLEM OF PROVIDING JOBS IN
SEVERAL DIFFERENT WAYS)

(FOR INSTANCE DURING 1933 - 1934 IT PAID NEARLY $ 1 BILLION IN WAGES
TO MEN AND WOMEN DRAWN FROM THE RELIEF ROLLS WHO WERE GIVEN JOBS ON
MADE WORK PROJECTS)

(CRITICS OF THE NEW DEAL CALLED MADE WORK *WHAT*)
*
BOONDOGGLING .

CORRECT
```

```
(PRESIDENT ROOSEVELT AND OTHER NEW DEALERS KNEW THAT FEDERAL CHARITY
AND MADE WORK WERE AT BEST NECESSARY EVILS)

(WHAT AMERICANS NEEDED AND WHAT THE NEW DEALERS WANTED TO PROVIDE WAS
SOCIAILY USEFUL WORK)

(TO PROVIDE SOCIALLY USEFUL WORK A NEW AGENCY WAS CREATED IN 1935 WITH
 HARRY L HOPKINS AS ITS HEAD - CALLED WHAT)
*
WPA ,

CORRECT

(THE WPA PROGRAM HELPED PEOPLE IN MANY WALKS OF LIFE)

(UNEMPLOYED ACTORS MUSICIANS AND WRITERS ENRICHED AMERICAN LIFE BY
PROVIDING *WHAT*)
*
START QMODE ,

ASK-AWAY
WHAT WERE TH -E AGENCIES CREATED TO BRING WORK TO YOUNG PEOPLE ,
(QLENGTH= 6)
(MATCHES= 3)
(PSET= (330 450 530))
(TO PROVIDE SOCIALLY USEFUL WORK A NEW AGENCY THE WORKS PROGRESS
ADMINISTRATION - WPA - WAS CREATED IN 1935 WITH HARRY L HOPKINS AS ITS
 HEAD)
(CCC AND NYA WERE CREATED TO BRING IMMEDIATE WORK RELIEF TO THE
NATIONS YOUTH)

NEXT-QUESTION
WHAT ADMINISTRATION LAUNCHED THE NEW DEAL ,


///// GARBAGE COLLECTION: 584 1942

(QLENGTH= 4)
(MATCHES= 3)
(PSET= (660 720 740))
(FEDERAL HOUSING PROGRAMS)
(THE NATIONAL NEGRO CONGRESS)
(NEGROES)

NEXT-QUESTION
END
MORE ,
(FINALLY RESPONDING TO PRESSURES SUCH AS THESE AND TO THE PRODDING OF
ELEANOR ROOSEVELT THE PRESIDENTS WIFE THE NEW DEAL ADMINISTRATION
APPOINTED PROMINENT NEGROES TO IMPORTANT FEDERAL POSITIONS)
```

```
WHAT ASSOCIATION DID TENANT FARMERS BELONG TO ,

T
TM TIME :: 9.538 SEC.
(QLENGTH= 4)
(MATCHES= 2)
(PSET= (650 710 710))
(BLACK TENANT FARMERS AND SHARECROPPERS WERE ESPECIALLY HARD HIT AND
THEY OFTEN SUFFERED DISCRIMINATION IN THE ALLOCATION OF BENEFITS FROM
NEW DEAL AGRICULTURAL AGENCIES)
(IN THE SOUTHERN TENANT FARMERS UNION TENANT FARMERS BLACK AND WHITE
OFTEN JOINED TOGETHER AGAINST WEALTHY LANDLORDS)

NEXT-QUESTION
END QMODE ,

(BACK TO THE OLD QUESTION)
(UNEMPLOYED ACTORS MUSICIANS AND WRITERS ENRICHED AMERICAN LIFE BY
PROVIDING *WHAT*)
*
PLAYS AND RECREATION ,

(FILL IN THE BLANKS)
(PLAYS ----- GUIDEBOOKS AND OTHER ----- OF RECREATION)
*
,

(PLAYS CONCERTS GUIDEBOOKS AND OTHER FORMS OF RECREATION)


(AT THE PEAK OF ITS ACTIVITY *WHEN* NEARLY 4 MILLION AMERICANS WERE
WORKING FOR THE WPA)
*
1938 ,

WRONG
(FILL IN THE BLANKS)
(IN MARCH -----)
*
1933 ,

(IN MARCH 1936)


(NUMBER CORRECT = 7)
(NUMBER WRONG = 2.00)
(PERCENT = 77.78)
```

(THIS IS A REVIEW OF THOSE QUESTIONS THAT YOU ASKED ME)

///// GARBAGE COLLECTION: 621 1975

(THE MOST URGENT TASK FACING ROOSEVELT WHEN HE TOOK OFFICE WAS *WHAT*)
*
FINDING JOBS FOR MILLIONS OF HUNGRY AMERICANS ,

(OK - FULL ANSWER FROM TEXT = (TO PROVIDE ASSISTANCE FOR MILLIONS OF JOBLESS HUNGRY AMERICANS))

(TO *WHAT* A NEW AGENCY THE WORKS PROGRESS ADMINISTRATION - WPA - WAS CREATED IN 1935 WITH HARRY L HOPKINS AS ITS HEAD)
*
PROVIDE RELIEF ,

(FILL IN THE BLANKS)
(PROVIDE ----- USEFUL -----)
*
SOCIALLY WORK ,

(PROVIDE SOCIALLY USEFUL WORK)

(DISCRIMINATION ALSO EXISTED IN THE ADMINISTRATION OF *WHAT* AND ESPECIALLY AT THE LOCAL LEVELS IN OTHER NEW DEAL PROGRAMS)
*
FEDERAL HOUSING PROGRAMS ,

CORRECT

(*WHAT* LAUNCHED IN 1935 WITH SOME SUPPORT FROM WHITE LIBERALS AND RADICALS PROMOTED THE INTEREST OF BLACK AMERICANS WITHIN THE NEW DEAL AGENCIES)
*
THE NATIONAL NEGRO CONGRESS ,

CORRECT

(FINALLY RESPONDING TO PRESSURES SUCH AS THESE AND TO THE PRODDING OF ELEANOR ROOSEVELT THE PRESIDENTS WIFE THE NEW DEAL ADMINISTRATION APPOINTED PROMINENT *WHAT* TO IMPORTANT FEDERAL POSITIONS)
*
NEGROES ,

CORRECT

(BLACK TENANT FARMERS AND SHARECROPPERS WERE ESPECIALLY HARD HIT AND THEY OFTEN SUFFERED *WHAT* IN THE ALLOCATION OF BENEFITS FROM NEW DEAL AGRICULTURAL AGENCIES)

*

DISCRIMINATION ,

CORRECT

*

NIL ,

///// GARBAGE COLLECTION: 622 1960

CORRECT

LESSON-OVER

*VALUE:

NIL

*TIME: 8326

Discussion: In our first experiments with this system we noticed that it was not always easy for a student to recognize a question and identify the questioned element. The question generator was modified to mark the question word with asterisks. Our original function word list was too short and frequently answer retrieval occurred with too much emphasis on function words. This was easily remedied by adding to the function word list all the words in the text that could not signify the content of a sentence. The scoring formula for retrieval is still simply a ratio of hits by candidate sentences on the roots of the question. Improvements to this formula include weighting the words as an inverse function of their frequency and weighting each candidate according to the correlation between the order of its roots with those of the question.

We noticed very soon that editing is an art that requires experience with the type of questions that are typically asked. We obtained a set of questions from two high school students who had read the text and tried them out in QMODE. Typical results are shown in Protocol 2 on the following pages.

```
WHY WAS THE WPA CREATED ,
(QLENGTH= 2)
(MATCHES= 2)
(PSET= (330))
(TO PROVIDE SOCIALLY USEFUI WORK A NEW AGENCY THE WORKS PROGRESS
ADMINISTRATION - WPA - WAS CREATED IN 1935 WITH HARRY L HOPKINS AS ITS
 HEAD)

NEXT-QUESTION
WHAT WAS THE NAME OF THE ORGANIZATION IN WHICH TENANT FARMERS BOTH
BLACK AND WHITE HOINED TOGETHER AGGINST WEALTHY LANDLORDS ,
(QLENGTH= 13)
(MATCHES= 7)
(PSET= (710))
(IN THE SOUTHERN TENANT FARMERS UNION TENANT FARMERS BLACK AND WHITE
OFTEN JOINED TOGETHER AGAINST WEALTHY LANDLORDS)

NEXT-QUESTION
WHAT WAS THE MAIN PURPOSE OF THE NT-ATIONAL NEGRO CONGRESS ,
(QLENGTH= 5)
(MATCHES= 3)
(PSET= (720))

///// GARBAGE COLLECTION: 586 1816

(NEW DEAL AGENCIES)

NEXT-QUESTION
MORE ,
(THE NATIONAL NEGRO CONGRESS LAUNCHED IN 1935 WITH SOME SUPPORT FROM
WHITE LIBERALS AND RADICALS PROMOTED THE INTEREST OF BLACK AMERICANS
WITHIN THE NEW DEAL AGENCIES)

NEXT-QUESTION
WHY DID THE NEW DEAL HELP FARMERS ,
(QLENGTH= 4)
(MATCHES= 3)
(PSET= (650))
(BLACK TENANT FARMERS AND SHARECROPPERS WERE ESPECIALLY HARD HIT AND
THEY OFTEN SUFFERED DISCRIMINATION IN THE ALLOCATION OF BENEFITS FROM
NEW DEAL AGRICULTURAL AGENCIES)

NEXT-QUESTION
WHAT WERE TWO AGENCIES CREATED TO BRING WORK RELIEF TO THE NATIONS
YOUTH ,


///// GARBAGE COLLECTION: 524 1714

(QLENGTH= 8)
(MATCHES= 6)
(PSET= (450))
(CCC AND NYA)
```

Table 6. Protocol of actual students questions

```
NEXT-QUESTION
HOW DID THE CCC AND NYA BRING WORK RELIEF TO THE NATIONS YOUTH ,
(QLENGTH= 7)
(MATCHES= 7)
(PSET= (450))
(CCC AND NYA WERE CREATED TO BRING IMMEDIATE WORK RELIEF TO THE
NATIONS YOUTH)

NEXT-QUESTION
HOW DID THE CCC AND NYA ACCOMPLISH THEIR PURPOSE ,


///// GARBAGE COLLECTION: 579 1850

(QLENGTH= 5)
(MATCHES= 2)
(PSET= (450 520 620))
(CCC AND NYA WERE CREATED TO BRING IMMEDIATE WORK RELIEF TO THE
NATIONS YOUTH)
(YOUTHS OF THE CCC RECEIVED FOOD CLOTHING AND SHELTER THEY WERE PAID
WAGES WHICH THEY WERE EXPECTED TO SHARE WITH THEIR FAMILIES AND THEY
WERE OFFERED OPPORTUNITIES FOR RECREATION AND EDUCATION)

NEXT-QUESTION
WHY DID ROOSEVELT CALL A SPECIAL SESSION OF CONGRESS SOON AFTER HIS
INAUGURATION ,
(QLENGTH= 7)
(MATCHES= 5)
(PSET= (60))
(ROOSEVELT THEN PRESENTED HIS REFORM PROPOSALS WITH RECOMENDATIONS FOR
 IMMEDIATE ACTION TO A SPECIAL SESSION OF CONGRESS THAT HE CALLED SOON
< OM >TAURUS OFF IN 2 MIN.  SAVE FILES NOW.......
 AFTER TAKING OFFICE)

NEXT-QUESTION
WHAT WERE THE THREE GENERAL AIMS OF THE NEW DEAL ,
(QLENGTH= 5)
(MATCHES= 5)
(PSET= (70))
(RELIEF RECOVERY AND REFORM)

NEXT-QUESTION
WHY DID ROOSEVELT START A BLACK CABINET ,


///// GARBAGE COLLECTION: 578 1837

(QLENGTH= 4)
(MATCHES= 2)
(PSET= (760))
(BETHUNE MCLEOD AND WEAVER AND OTHERS MADE UP AN INFORMAL GROUP OF
ADVISERS OFTEN CALLED THE BLACK CABINET)
```

The students were given no direction as to question type, other than to limit them to the first section of chapter 17. Of the twenty questions they produced AMI answered eleven correctly, two of these with explicit strings. Two questions were not answered at all by the text and two others required inference over more than one sentence. Performance on the other five incorrectly answered would be improved by weighting and ordering heuristics and attention to intersentential interaction. Given that the aim of the program is to teach rather than answer questions it does not seem critical to respond to queries with hundred percent accuracy so long as the student has a good chance of an informative response.

Conclusions: As a computational experiment, AMI shows that an Editing and Indexing system can indeed support a question generation and answering program that offers some promise as a text reviewing and teaching device. As a heuristics based system, much experimentation is required to tune up weighting algorithms and to develop sharper logic for recognizing correct direct answers and rejecting false ones. Measuring its effectiveness as a teaching device requires experimentation in a practical teaching environment with cooperation of interested teachers. Both of these are high priorities on our research list.

REFERENCES

1. Bruce, B.C., "Case Structure Systems", Proc. 3rd Int. Jt. Conf. Art. Intell., Stanford Res. Inst., Menlo Park, Calif., 1973.

2. Carbonell, J.R., "AI in CAI: An Artificial Intelligence Approach to Computer Assisted Instruction", IEEE Trans. on Man-Machine Systems, MSS-11, 1970.

3. Fillmore, C.J., "The Case for Case" in Bach E, & Harms T., (Eds.), Universals in Linguistic Theory, Holt, Rinehart & Winston, New York, 1968.

4. Heidorn, George E., "Natural Language Inputs to a Simulation Programming System", Naval Postgraduate School, Monterrey, Calif., Dec. 1971. (also IBM Yorktown Hts., N.Y.)

5. Hendrix, G.G., Thompson, C. & Slocum, J., "Language Processing via Canonical Verbs and Semantic Models", Proc. 3rd Int. Jt. Conf. Art. Intell., Stanford Res. Inst., Menlo Park, Calif., 1973.

6. Petrick, S.R., "A Recognition Procedure for Transformational Grammars", in Rustin R. (Ed.), Natural Language Processing, Prentice Hall, Englewood Cliffs, N.J., 1972.

7. Rise of the American Nation, Vol 2, Harcourt, Brace, Jovanovich, Inc., New York, 1972, pp 410-413.

8. Sager, N., "The String Parser for Scientific Literature", in Rustin R. (Ed.), Natural Language Processing, Prentice Hall, Englewood Cliffs, N.J., 1972.

9. Schank, R. & Colby, K. (Eds.), Computer Models of Thought and Language, W.H. Freeman & Co., San Francisco, 1973.

10. Schank, R.C., "Identifications of Conceptualizations Underlying Natural Language", in Schank & Colby.

11. Simmons, R. & Slocum, J., "Generating English Discourse from Semantic Nets", CACM, Oct., 1972.

12. Simmons, R.F., "Semantic Networks: their Computation and Use for Understanding English Sentences", in Schank & Colby.

13. Stachowitz, R., "Requirements for Machine Translation", Linguistics Res. Ctr., Univ. of Texas, Austin, 1972.

14. Wexler, J.D., "A Generative Teaching System that Uses Information Nets and Skeleton Patterns", Ph.D. Dissertation, Univ. of Wisc., Dept. of Comp. Sci., Madison, Wisc., 1970.

15. Wilks, Y., "An Artificial Intelligence Approach to Machine Translation", in Schank & Colby.

16. Winograd, T., "A Procedural Model of Language Understanding", in Schank & Colby.

17. Woods, W.A., "Transition Network Grammars for Natural Language Analysis", CACM, Oct., 1970.