41P.

OPERATION OF A SEMANTIC QUESTION-ANSWERING SYSTEM

by Bertram Raphael

November 1, 1963

Operation of a Semantic Question-Answering System

by Bertram Raphael

I. Introduction

A. Preliminary Remarks

A computer program has been written in the LISP programming language which accepts information and answers questions presented to it in a restricted form of natural English language. The program achieves its effects by automatically creating, adding to, and searching a relational model for factual information. The purpose of this memo is to describe and explain the behavior of the program.

The remainder of this section briefly describes the structure of the model. Section II presents sample conversations illustrating various features of the program, and describes the implementation of those features. Section III is a brief survey of conclusions drawn from this research. It is assumed throughout that the reader is at least somewhat familiar with the LISP programming system (and its meta-language notation), the concept of property (description) lists, and the usual notations of Mathematical Logic.

B. Structure of the Model

The program usually interprets a simple declarative sentence as expressing the fact that a certain relation holds between certain objects; these objects are usually defined by words which name them.

Every word has associated with it a list of pairs called its property list (or description list). The first element of each pair is considered

to be the name of an attribute of the word, and the second element the value of that attribute. If a statement asserts that relation R holds between words x and y, then this relationship is represented by placing attribute-value pairs which express this fact on the property-lists of both x and y, and these links are considered to be the model for the relation.

Since in general relations are not commutative, relation R must be factored into two relations R1 and R2 so that if relation R holds between x and y, I can say that y stands in relation R1 to x, and x stands in R2 to y. R1 and R2 are used as attributes on the property lists of x and y, respectively.

If one and only one object can be in relation R1 to any word x, then the value of attribute R1 can be simply the name of that object, y. In this case I say that a type-1 link exists from x to y following the attribute R1. If R holds between x and y and also between x and z, type-1 links are inadequate (since there can only be one occurrence of a given attribute on a property list). The value of R1 is then a list, e.g., in this case the value of R1 of x is the list "(y,z)"*. These are called type-2 links.

Occasionally descriptive information is desired in addition to a fact like "y stands in relation R1 to x", say information pertinent to that particular occurrence of the R1 relation. This can be handled by using type-3 links, where the value of an attribute is a list of items, each of which is itself in property list format; e.g., if R holds between x and y and also between x and z, and type-3 links are used, the following

---

*It is assumed that variables will always be replaced by the quantities they are bound to, even in quoted expressions.

structure would be placed on the property list of x:

"(... R1 ((PLIST NAME y)(PLIST NAME z)) ...)

"PLIST" identifies an item as a sub-property list, and "NAME" is an attribute whose type-1 value is the principal object on the list.

In the interest of generality and uniformity, type-3 links are the predominant mechanism for structuring the model.

V. Operation of the Program

A. Relations and Functions

Each part of this section will illustrate the use of a different group of relation-attributes. With minor exceptions the parts are cumulative, i.e., later parts freely use functions introduced earlier but not conversely. Each part consists of a list of relations involved; a dialogue illustrating their use; and a section describing the operation of relevant LISP functions.

Each portion of the dialogue consists of a sentence, prefixed by "****.", which is the statement presented to the program; a LISP form to be evaluated, which represents the results of the natural-language-to-LISP-function translation phase of the program*, and which normally does not appear on the output unless specifically requested; and the responses produced during the evaluation of the form.

In the "operation of functions" portions of the following, the "method" sections are meant to serve as explanations of the principles used in specifying the functions. The "procedure" sections may be thought of as rough flow-charts. English and logical notation are frequently intermixed

*The operation of this part of the program involves rather straightforward format matching and is beyond the scope of this memo.

in order to explain methods and procedures clearly and concisely.

1) Relations: SUBSET, SUPERSET

Dialogue

(***. EVERY KEYPUNCH-OPERATOR IS A GIRL)
setr[KEYPUNCH-OPERATOR;GIRL]
(I UNDERSTAND)

(***. ANY GIRL IS AN EXAMPLE OF A PERSON)
setr[GIRL;PERSON]
(I UNDERSTAND)

(***. IS A KEYPUNCH-OPERATOR A PERSON Q)
setrq[KEYPUNCH-OPERATOR;PERSON]
(YES)

(***. IS A PERSON A PERSON Q)
setrq[PERSON;PERSON]
(YES)

(***. IS A PERSON A GIRL Q)
setrq[PERSON;GIRL]
(SOMETIMES)

(***. IS A MONKEY A KEYPUNCH-OPERATOR Q)
setrq[MONKEY;KEYPUNCH-OPERATOR]
(INSUFFICIENT INFORMATION)

Operation of functions

1. setr[x;y]

purpose: To specify in the model that set x is included in set y.

method: Create a type-3 link between x and y which indicates set-inclusion.

procedure: a) Add "(PLIST NAME x)" to the value list of attribute "SUBSET" of y.

b) Add "(PLIST NAME y)" to the value list of attribute "SUPERSET" of x.

c) Respond "(I UNDERSTAND)".

2. setrq[x;y]

purpose: To reply as to whether an arbitrary element of set x is an element of set y.

method: A member of x is considered to be a member of y if the sets x and y are identical; or if there is a chain of explicit set-inclusion links proving that x is a subset of y, i.e., if there exists a (possibly empty) sequence of sets v,w,...,z such that

$$x \subset v \wedge v \subseteq w \wedge \cdots \wedge z \subset y.$$

A member of x is "sometimes" in y if there is a chain of explicit set-inclusion links proving that y is a subset of x.

procedure: a) If x = y, respond "YES".

b) If there is a path from x to y through type-3 links following the attribute "SUPERSET", respond "YES".

c) If there is a path from y to x through type-3 links following the attribute "SUPERSET", respond "SOMETIMES".

d) Otherwise, respond "(INSUFFICIENT INFORMATION)".

2) Relations: MEMBER, ELEMENTS

Dialogue

(***. MAX IS AN IBM-7094)
setrs[MAX;IBM-7094]
(I UNDERSTAND)

(***. AN IBM-7094 IS A COMPUTER)
setr[IBM-7094;COMPUTER]
(I UNDERSTAND)

(***. IS MAX A COMPUTER Q)
setrsq[MAX;COMPUTER]
(YES)

(***. THE BOY IS AN MIT-STUDENT)
setrsl[BOY;MIT-STUDENT]
(G00018 IS A BOY)
(I UNDERSTAND)

(... WHEN MIT-STUDENT IS A BRIGHT-PERSON)
setr[MIT-STUDENT;BRIGHT-PERSON]
(I UNDERSTAND)

(... IS THE BOY A BRIGHT-PERSON Q)
setrsq[BOY;BRIGHT-PERSON]
(YES)

(... JOHN IS A BOY)
setrs[JOHN;BOY]
(I UNDERSTAND)

(... IS THE BOY A BRIGHT-PERSON Q)
setrsq[BOY;BRIGHT-PERSON]
(WHICH BOY . . (G00018 JOHN))

Operation of functions

1. setrs[x;y]

purpose: To specify in the model that x is a member of the set y.

method: Create a type-3 link between x and y which indicates set-membership.

procedure: a) Add "(PLIST NAME y)" to the value list of attribute "MEMBER" of x.

b) Add "(PLIST NAME x)" to the value list of attribute "ELEMENTS" of y.

c) Respond "(I UNDERSTAND)".

2. setrsq[x;y]

purpose: To reply as to whether x is a member of set y.

method: Reply "YES" if the following is true:

$$(\exists u)[[u = x \vee [u \text{ is equivalent* to } x]] \wedge$$
$$[[\text{there is a link indicating that } u \text{ is a member of } y] \vee$$
$$[(\exists z)[[\text{there is a link indicating that } u \text{ is a member of } z] \wedge$$
$$[\text{any member of set } z \text{ is a member of set } y]]]]]$$

---

*See next section for an explanation of "equivalent".

procedure: a) Make a list of the items connected to x by a type-3 link following the attribute "MEMBER".

b) If y is on the list, respond "YES".

c) If, for any member z of the list,

setrq[z;y] = YES,

respond "YES".

d) Repeat steps (a) through (c) with x replaced by each item equivalent* to x (if any), until a "YES" response is made.

e) Otherwise respond "(INSUFFICIENT INFORMATION)".

3. setrs[x;y]

purpose: To specify in the model that the unique element (if any) of the set x is also an element of the set y.

method: Create a type-3 link from the unique element, if any, of x to y which indicates set-membership. If x has more than one element don't set up any link.

procedure: a) Compute u = specify[x]

b) If u = NIL, terminate.

c) Execute setrs[u;y].

4. specify[x]

purpose: To determine the unique element, if any, of the set x.

method: If x has one element, find its name. If x has no elements, creat one and give it a name. If x has more than one element, ask which one and indicate failure.

procedure: a) Get the value list of the attribute "ELEMENTS" of x.

---

*See next section for an explanation of "equivalent".

b) If no list, create a new symbol u, respond "(u IS A x)", execute setrs[u;x], and return u as the value of specify[x].

c) If there is just one element named on the list, or all the elements named are equivalent, return the name of the first element as the value of specify[x].

d) Otherwise respond "(WHICH x . . v)", where v is a list of names of the elements, and return "NIL" as the value of specify[x].

5. setrslq[x;y]

purpose: To reply as to whether the unique element, if any, of the set x, is a member of the set y.

method: Determine the element referred to and apply setrsq.

procedure: a) Compute u = specify[x].

b) If u = NIL, terminate.

c) Execute setrsq[u;y].

3) Relation: EQUIV

Dialogue

```
(***. THE MAN IS A PINK)
 setrsl[MAN;PINK]
(G00019 IS A MAN)
(I UNDERSTAND)

(***. JACK IS A DOPE)
 setrs[JACK;DOPE]
(I UNDERSTAND)

(***. JOHN IS JACK)
 equiv[JOHN;JACK]
(I UNDERSTAND)

(***. IS JOHN A DOPE Q)
 setrsq[JOHN;DOPE]
(YES)
```

```
(***. IS THE MAN A DOPE Q)
 setrsql[MAN;DOPE]
(INSUFFICIENT INFORMATION)

(***. JOHN IS THE MAN)
 equivl[JOHN;MAN]
(I UNDERSTAND)

(***. IS THE MAN A DOPE Q)
 setrsql[MAN;DOPE]
(YES)

(***. JIM IS A MAN)
 setrs[JIM;MAN]
(I UNDERSTAND)

(***. IS THE MAN A DOPE Q)
 setrsql[MAN;DOPE]
(WHICH MAN . . (G00019 JIM))
```

Operation of functions

1. equiv[x;y]

   purpose: To specify in the model that x and y are equivalent.

   method: Create a type-2 link between x and y which indicates equivalence.

   procedure: a) Add x to the value list of attribute "EQUIV" of y.

   b) Add y to the value list of attribute "EQUIV" of x.

   c) Respond "(I UNDERSTAND)".

2. equivl[x;y]

   purpose: To specify in the model that x is equivalent to the unique element of the set y.

   method: Determine the element referred to and apply equiv.

   procedure: a) Compute u = specify[y]

   b) If u = NIL, terminate.

   c) Execute equiv[x;u].

4) Relations: OWNED-BY-EACH, POSSESS-BY-EACH

Dialogue

```
(***. EVERY FIREMAN OWNS A PAIR-OF-RED-SUSPENDERS)
 ownr[PAIR-OF-RED-SUSPENDERS;FIREMAN]
( I UNDERSTAND)

(***. DOES A PAIR-OF-RED-SUSPENDERS OWN A PAIR-OF-RED-SUSPENDERS Q)
 ownrq[PAIR-OF-RED-SUSPENDERS;PAIR-OF-RED-SUSPENDERS]
(NO ** THEY ARE THE SAME)

(***. DOES A DOCTOR OWN A PAIR-OF-RED-SUSPENDERS Q)
 ownrq[PAIR-OF-RED-SUSPENDERS;DOCTOR]
(INSUFFICIENT INFORMATION)

(***. A FIRECHIEF IS A FIREMAN)
 setr[FIRECHIEF;FIREMAN]
(I UNDERSTAND)

(***. DOES A FIRECHIEF OWN A PAIR-OF-RED-SUSPENDERS Q)
 ownrq[PAIR-OF-RED-SUSPENDERS;FIRECHIEF]
(YES)
```

Operation of functions

1. ownr[x;y]

   purpose: To specify in the model that every member of set y owns some member of set x.

   method: Create a type-3 link between x and y which indicates the ownership-relation between their members.

   procedure: a) Add "(PLIST NAME x)" to the value list of attribute "OWNED-BY-EACH" of y.

   b) Add "(PLIST NAME y)" to the value list of attribute "OWNED-BY-EACH" of x.

   c) Respond "(I UNDERSTAND)"

2. ownrq[x;y]

   purpose: To reply as to whether an arbitrary member of set y owns some member of set x.

   method: The answer is "YES" if x ≠ y, and

$(\exists z)[y=z \vee [y \text{ is a subset of } z]] \wedge$ [there exists the appropriate ownership-relation link between x and z]]

procedure: a) If $x=y$, respond "(NO ** THEY ARE THE SAME)".

b) Create the list l containing y and all sets u for which there is a path from y to u through type-3 links following the attribute "SUPERSET".

c) If any element of l contains a type-3 link to x following the attribute "POSSESS-BY-EACH", respond "YES".

d) Otherwise respond "(INSUFFICIENT INFORMATION)".

5) Relations: OWNED, POSSESS

Dialogue

(***. ALFRED OWNS A LOG-LOG-DECITRIG)
ownsgu[LOG-LOG-DECITRIG;ALFRED]
(I UNDERSTAND)

(***. A LOG-LOG-DECITRIG IS A SLIDE-RULE)
setr[LOG-LOG-DECITRIG;SLIDE-RULE]
(I UNDERSTAND)

(***. DOES ALFRED OWN A SLIDE-RULE Q)
ownsguq[SLIDE-RULE;ALFRED]
(YES)

(***. EVERY ENGINEERING-STUDENT OWNS A SLIDE-RULE)
owns[SLIDE-RULE;ENGINEERING-STUDENT]
(I UNDERSTAND)

(***. VERNON IS A TECH-MAN)
setms[VERNON;TECH-MAN]
(I UNDERSTAND)

(***. A TECH-MAN IS AN ENGINEERING-STUDENT)
setr[TECH-MAN;ENGINEERING-STUDENT]
(I UNDERSTAND)

(***. DOES VERNON OWN A SLIDE-RULE Q)
ownsguq[SLIDE-RULE;VERNON]
(YES)

(***. ALFRED IS A TECH-MAN)
setrs[ALFRED;TECH-MAN]
(I UNDERSTAND)

(***. DOES AN ENGINEERING-STUDENT OWN THE LOG-LOG-DECITRIG Q)
ownsq[LOG-LOG-DECITRIG;ENGINEERING-STUDENT]
(G00025 IS A LOG-LOG-DECITRIG)
(YES)

Operation of functions

1. ownrgu[x;y]

purpose: To specify in the model that y owns a member of the set x.

method: Create a type-3 link between x and y which indicates the intended ownership relation.

procedure: a) Add "(PLIST NAME x)" to the value list of attribute "POSSESS" of y.

b) Add "(PLIST NAME y)" to the value list of attribute "OWNED" of x.

c) Respond "(I UNDERSTAND)".

2. ownrguq[x;y]

purpose: To reply as to whether y owns a member of set x.

method: The reply is "YES" if there is a link indicating that y owns a member of x or of some subset of x; or if

$(\exists z)[[y \text{ is a member of } z] \wedge$

$(\exists u)[[u=z \vee z \subset u] \wedge$

[there is a link indicating that every member of set u owns a member of the set x]]]

procedure: a) Consider each set z such that there is a link indicating y is an element of z.

b) For each z, construct a list l containing every set u for which setrq[z;u] = YES.

c) Compute m = the list of all sets y such that there is a type-3 link from x to y following the attribute "OWNED-BY-EACH".

d) If, for some z, the intersection of l and m is non-empty, respond "YES".

e) Otherwise, respond "(INSUFFICIENT INFORMATION)".

3. ownrsgq[x;y]

purpose: To reply as to whether the unique element, if any, of the set x, is owned by some element of the set y.

method: Determine that a unique element of x exists. Then, the reply is "YES" if

(∃z)[[there is a link indicating that a member of set x is owned by z] ∧

(∃v)[[v=z ∨ [v is equivalent to z]] ∧

(∃w)[[there is a link indicating, that y is an element of w] ∧

[there are links indicating that w is a subset of y]]]

procedure: a) Compute u = specify[x]

b) If u=NIL, terminate

c) Generate the individuals w which are linked to x as type-3 values of the attribute "OWNED".

d) For each w, generate the sets z which w, and any individual equivalent to w, is a member of.

e) If, for some z, setrq[z;y]=YES, respond"YES".

f) Otherwise respond "(INSUFFICIENT INFORMATION)".

3) Relations: SUBPART-OF-EACH, SUPERPART-OF-EACH

Dialogue

(***. A NOSE IS PART OF A PERSON)
partr[NOSE;PERSON]
(I UNDERSTAND)

(***. A NOSTRIL IS PART OF A NOSE)
partr[NOSTRIL;NOSE]
(I UNDERSTAND)

(***. A PROFESSOR IS A TEACHER)
setr[PROFESSOR;TEACHER]
(I UNDERSTAND)

(***. A TEACHER IS A PERSON)
setr[TEACHER;PERSON]
(I UNDERSTAND)

(***. IS A NOSTRIL PART OF A PROFESSOR Q)
partrq[NOSTRIL;PROFESSOR]
(YES)

(***. IS A NOSE PART OF A NOSE Q)
partrq[NOSE;NOSE]
(NO, PART MEANS PROPER SUBPART)

(***. A PERSON IS A LIVING-CREATURE)
setr[PERSON;LIVING-CREATURE]
(I UNDERSTAND)

(***. IS A NOSTRIL PART OF A LIVING-CREATURE Q)
partrq[NOSTRIL;LIVING-CREATURE]
(SOMETIMES)

(***. IS A LIVING-CREATURE PART OF A NOSE Q)
partrq[LIVING-CREATURE;NOSE]
(NO, NOSE IS SOMETIMES PART OF LIVING-CREATURE)

Operation of functions

1. partr[x;y]

purpose: To specify in the model that every element of set x is part of some element of set y.

method: Create a type-3 link between x and y which indicates the part-whole relation between their members.

procedure: a) Add "(?LIST NAME x)" to the value list of attribute "SUBPART-OF-EACH" of y.

b) Add "(PLYST NAME y)" to the value list of attribute "SUPERPART-OF-EACH" of x.

c) Respond "(I UNDERSTAND)".

2. partrq[x;y]

purpose: To reply as to whether an arbitrary member of set x is a part of some member of set y.

method: No element may be part of itself. Reply "YES" if

(w)[[there is a chain of links indicating that an arbitrary member of set x is part of some member of w] [[y=w] [[there is a chain of links indicating that y is a subset of w]]].

Reply "SOMETIMES" if

(w)[[there is a chain of links indicating that an arbitrary member of set x is part of some member of w] [there is a chain of links indicating that w is a subset of y]]

Reply "NO" if an arbitrary member of set y is always or sometimes a part of some member of set x.

procedure: a) If x=y, respond "(NO, THEY ARE THE SAME)".

b) Generate those sets w which can be reached from x through a chain of type-3 links following the attribute "SUPERPART-OF-EACH".

c) If, for some w, setrq[y;w]=YES or SOMETIMES, respond "YES" or "SOMETIMES", respectively.

d) If the response for partrq[y;x] would be YES or SOMETIMES, respond "(NO, y IS PART OF x)" or "(NO, y IS SOMETIMES PART OF x)", respectively.

e) Otherwise respond "(INSUFFICIENT INFORMATION)".

7) Relations: SUBPART, SUPERPART

Dialogue

(***. A VAN-DYKE IS PART OF FERRAN)

```
 partrguq[VAN-DYKE;FERRAN]
(I UNDERSTAND)

(***. A VAN-[illegible] IS A BEARD)
 setr[VAN-DY[illegible]EARD]
(I UNDERSTAND)

(***. IS A [illegible]D PART OF FERRAN Q)
 partrguq[BE[illegible];FERRAN]
(YES)

(***. A CRT IS PART OF THE PDP-1)
 partrgs[CRT;PDP-1]
(G00051 IS A PDP-1)
(I UNDERSTAND)

(***. A CRT IS A DISPLAY-DEVICE)
 setr[CRT;DISPLAY-DEVICE]
(I UNDERSTAND)

(***. SAM IS THE PDP-1)
 equivl[SAM;PDP-1]
(I UNDERSTAND)

(***. A SCREEN IS PART OF EVERY DISPLAY-DEVICE)
 partr[SCREEN;DISPLAY-DEVICE]
(I UNDERSTAND)

(***. IS A SCREEN PART OF SAM Q)
 partrguq[SCREEN;SAM]
(YES)

(***. A BEARD IS PART OF A BEATNIK)
 partr[BEARD;BEATNIK]
(I UNDERSTAND)

(***. EVERY COFFEE-HOUSE-CUSTOMER IS A BEATNIK)
 setr[COFFEE-HOUSE-CUSTOMER;BEATNIK]
(I UNDERSTAND)

(***. BUZZ IS A COFFEE-HOUSE-CUSTOMER)
 setrs[BUZZ;COFFEE-HOUSE-CUSTOMER]
(I UNDERSTAND)

(***. IS A BEARD PART OF BUZZ Q)
 partrguq[BEARD;BUZZ]
(YES)

(***. THE HAND IS PART OF THE ARM)
 partrgs[HAND;ARM]
(G00041 IS A HAND)
(G00042 IS A ARM)
(I UNDERSTAND)

(***. AN ARM IS PART OF A PERSON)
 partr[ARM;PERSON]
(I UNDERSTAND)
```

(A BOY IS A PERSON)
subs[BOY;PERSON]
(I UNDERSTAND)

(A HAND IS THE HAND PART OF A BOY Q)
partrsgq[HAND;BOY]
(YES)

Operation of functions

1. partrgu[x;y]

purpose: To specify in the model that some element of set x is a part of the individual y.

method: Create a type-3 link between x and y which indicates the appropriate part-whole relation.

procedure: a) Add "(PLIST NAME x)" to the value list of attribute "SUBPART" of y.

b) Add "(PLIST NAME y)" to the value list of attribute "SUPERPART" of x.

c) Respond "(I UNDERSTAND)".

2. partrgs[x;y]

purpose: To specify in the model that some element of set x is a part of the unique element, if any, of the set y.

method: a) Determine z, the unique element of y.

b) Specify that some element of x is part of z.

procedure: a) Compute z = specify[y].

b) If z=NIL, terminate.

c) Otherwise, compute partrgu[x;z]

3. partrguq[x;y]

purpose: To reply as to whether some element of set x is part of the individual y.

method: A member of x is part of y if

$(\exists u)[[u=y \vee [u \text{ is equivalent to } y]] \wedge$

$[(\exists w)[[\text{there is a link indicating that an element of } w$

is a subpart of u] $\wedge$

[[w=x $\vee$ [there are links indicating that w is a subset of x] $\vee$

($\exists z$)[[there are links indicating that every element of z has some element of x as a part] $\wedge$ [w=z $\vee$ [there are links indicating that w is a subset of z]]]] $\vee$

[($\exists z$)[[u is an element of set z] $\wedge$ [

($\exists v$)[there are links indicating that every element of v has some element of x as a part] $\wedge$ [z=v $\vee$ [there are links indicating that z is a subset of y]]]]]]]]

procedure: a) Generate those nodes w which can be reached from y, or from any node equivalent to y, by a chain of type-3 links following the attribute "SUBPART".

b) If, for any w, setrq[w;x]=YES, respond "YES".

c) Otherwise, generate those nodes z which can be reached from x by a chain of type-3 links following the attribute "SUPERPART-OF-EACH".

d) If, for any z and any w, setrq[w;z]=YES respond "YES".

e) Otherwise, compute the list l of sets for which there is a type-3 link from y, or any node equivalent to y, following the attribute "MEMBER".

f) Generate the nodes v which can be reached by a chain of type-3 links from x following the attribute, "SUPERPART-OF-EACH".

g) If, for any v and any u in l, setrq[u;v]=YES, respond "YES".

h) Otherwise, respond "(INSUFFICIENT INFORMATION)".

4. partrsc[x;y]

purpose: To specify in the model that the unique element, if any, of set x is part of the unique element, if any, of set y.

method: Identify the unique elements u and v of sets x and y, respectively. Specify that some element of set x is part of the individual v. Then create a type-2 link from the appropriate type-3 link from x to u, specifying which element of x is involved.

procedure: a) Compute v = specify[b], and u = specify[a].

b) If u or v = NIL, terminate.

c) Execute partrgu[x;v]

d) Add u to the value list of attribute "ELEMENTS" on that member of the "SUPERPART" value list of x which refers to y.

e) Respond "(I UNDERSTAND)".

5. partrsgq[x;y]

purpose: To reply as to whether the unique element of set x is part of some element of set y.

method: The answer is "YES" if there exists a unique element z of set x and if

$(\exists w)$[[there is a link indicating that some x is part of w] $\wedge$

$(\exists u)$[[u=w $\vee$ u is equivalent to w] $\wedge$

$(\exists v)$[[there is a link indicating that u is an element of v] $\wedge$

[[y=v] $\vee$ [there are links indicating that y is a subset of v] $\vee$

$(\exists q)$[[there are links indicating that every y is part of some q] $\wedge$ [[v=q] $\vee$ [there are links indicating that y is a subset of q]]]]]]]

procedure: a) Compute z = specify[x]

b) If z=NIL, terminate.

c) Generate those nodes y which can be reached from z by a type-3 link following the attribute "SUPERPART".

d) For each y, compute the list l of those sets which y, or any set equivalent to y, is a member of.

e) If y is in l, respond "YES".

f) If, for any y∈l, setrq[y;v]=YES, respond "YES".

g) Otherwise, generate those nodes g which can be reached from y by a type-3 link following the attribute "SUPERPART-OF-EACH".

h) If, for any g, setrq[v;q]=YES, respond "YES".

i) Otherwise, respond "(INSUFFICIENT INFORMATION)".

8) Relation: NUMBER

Dialogue

```
(***. A BOY IS A PERSON)
 setr[BOY;PERSON]
(I UNDERSTAND)

(***. JOHN IS A BOY)
 setrs[JOHN;BOY]
(I UNDERSTAND)

(***. A FINGER IS PART OF A HAND)
 partr[FINGER;HAND]
(I UNDERSTAND)

(***. HOW MANY FINGERS DOES JOHN HAVE Q)
 partrnuq[FINGER;JOHN]
(I DON'T KNOW WHETHER FINGER IS PART OF JOHN)

(***. THERE IS ONE HAND ON EACH ARM)
 partrn[HAND;ARM;1]
(I UNDERSTAND)
```

```
(***. THERE ARE TWO ARMS ON A PERSON)
 partrm[ARM;PERSON;2]
(I UNDERSTAND)

(***. HOW MANY FINGERS DOES JOHN HAVE Q)
 partrmuq[FINGER;JOHN]
(HOW MANY FINGER PER HAND Q)

(***. A HAND HAS FIVE FINGERS)
 partrm[HAND;FINGER;5]
(I UNDERSTAND)

(***. HOW MANY FINGERS DOES JOHN HAVE Q)
 partrmuq[FINGER;JOHN]
(THE ANSWER IS 10)

(***. THERE ARE 11 TOES ON JIM)
 partrmu[TOES;JIM;11]
(I UNDERSTAND)
```

Operation of functions

1. partrm[x;y;n]

   purpose: To specify in the model that there are n elements of set x which are parts of every element of set y.

   method: Create a type-3 link between x and y specifying that an element of set x is part of some element of set y. Create type-1 links associating the number n with that type-3 link.

   procedure: a) Execute partr[x;y].

   b) Add "(NUMBER n)" to both the list which was added to the value list of attribute "SUBPART-OF-EACH" of y, and the list which was added to the value list of attribute "SUPERPART-OF-EACH" of x.

2. partrmu[x;y;n]

   purpose: To specify in the model that there are n elements of set x which are parts of individual y.

   method: Create a type-3 link between x and y which indicates that some element of set x is part of y. Create type-1 links associating the number n with that type-3 link.

procedure: a) Execute partrgu[x;y].

b) Add "(NUMBER n)" to both the list which was added to the value list of attribute "SUBPART" of y, and the list which was added to the value list of attribute "SUPERPART" of x.

3. partrnuq[x;y]

purpose: To reply as to how many elements of set x are parts of the individual y.

method: If

$(\exists u)$[[there is a link indicating that an element of u is part of y] $\wedge$

[[u=x] $\vee$ $(\exists v)$[[there is a chain of links indicating that a y is part of every u] $\wedge$

[[x=v] $\vee$ [there is a chain of links indicating that x is a subset of y]]]] $\vee$

$(\exists u)$[[there is a link indicating that y is an element of set u] $\wedge$

$(\exists v)$[[there is a chain of links indicating that a y is a part of every u] $\wedge$

[[x=v] $\vee$ [there is a chain of links indicating that x is a subset of v]]]],

then the answer is the product of the values of the type-1 link following the attribute "NUMBER", associated with each type-3 link used in proving the required part relations. If any such "NUMBER" attribute is missing, the reply explicitly requests it. If the part-whole relation cannot be established, the reply indicates that fact.

procedure: a) Follow the procedure of partrguq[x;y] until links are found which warrant a "YES" response. Save a list $\underline{l}$ of all required links which follow a "SUBPART" or a "SUPERPART-OF-EACH" attribute.

b) If no such list can be found, respond "(I DON'T KNOW WHETHER x IS PART OF y)".

c) For each element $\underline{\alpha}$ of $\underline{l}$, where $\underline{\alpha}$ specifies a "SUPERPART-OF-EACH" link from $\underline{u}$ to $\underline{v}$, get the value of the attribute "NUMBER" of $\underline{\alpha}$. If, for some $\underline{\alpha}$, no such value exists, respond "(HOW MANY u PER v Q)".

d) Compute z = the product of the numbers obtained above. Respond "(THE ANSWER IS z)".

9) Relations: LEFT, RIGHT, JLEFT, JRIGHT

Dialogue

```
(****. THE TELEPHONE IS JUST-TO THE RIGHT OF THE BOOK)
 jright[TELEPHONE;BOOK]
(G03096 IS A TELEPHONE)
(G03097 IS A BOOK)
(I UNDERSTAND)

(****. THE TELEPHONE IS JUST TO THE LEFT OF THE PAD)
 jright[PAD;TELEPHONE]
(G03098 IS A PAD)
(I UNDERSTAND)

(****. IS THE PAD JUST TO THE RIGHT OF THE BOOK Q)
 jrightssq[PAD;BOOK]
NO

(****. IS THE BOOK TO THE LEFT OF THE PAD Q)
 rightssq[PAD;BOOK]
YES

(****. THE PAD IS TO THE RIGHT OF THE TELEPHONE)
 right[PAD;TELEPHONE]
(THE ABOVE STATEMENT IS ALREADY KNOWN)
```

(***. THE PAD IS TO THE LEFT OF THE TELEPHONE)
 right [TELEPHONE;PAD]
(THE ABOVE STATEMENT IS IMPOSSIBLE)

(***. THE ASH-TRAY IS TO THE LEFT OF THE BOOK)
 right[BOOK;ASH-TRAY]
(G03099 IS A ASH-TRAY)
(I UNDERSTAND)

(***. THE PENCIL IS TO THE LEFT OF THE PAD)
 right[PAD;PENCIL]
(G03100 IS A PENCIL)
(I UNDERSTAND)

(***. THE PAPER IS TO THE RIGHT OF THE TELEPHONE)
 right[PAPER;TELEPHONE]
(G03101 IS A PAPER)
(I UNDERSTAND)

(***. WHERE IS THE PAD Q)
 where[PAD]
(JUST TO THE RIGHT OF THE TELEPHONE)
(SOMEWHERE TO THE RIGHT OF THE FOLLOWING . . (PENCIL))

(***. WHAT IS THE POSITION OF THE PAD Q)
 locates[PAD]
(THE LEFT-TO-RIGHT ORDER IS AS FOLLOWS)
(PENCIL (BOOK TELEPHONE PAD) PAPER)
(TO FURTHER SPECIFY THE POSITIONS YOU MUST INDICATE WHERE THE ASH-TRAY
 IS WITH RESPECT TO THE PENCIL)

(***. THE BOOK IS JUST TO THE RIGHT OF THE ASH-TRAY)
 jright[BOOK;ASH-TRAY]
(I UNDERSTAND)

(***. WHAT IS THE POSITION OF THE PAD Q)
 locates[PAD]
(THE LEFT-TO-RIGHT ORDER IS AS FOLLOWS)
(PENCIL (ASH-TRAY BOOK TELEPHONE PAD) PAPER)

(***. A TELEPHONE IS AN AUDIO-TRANSDUCER)
 satr[TELEPHONE;AUDIO-TRANSDUCER]
(I UNDERSTAND)

(***. A DIAPHRAGM IS PART OF AN AUDIO-TRANSDUCER)
 partr[DIAPHRAGM;AUDIO-TRANSDUCER]
(I UNDERSTAND)

(***. WHERE IS A DIAPHRAGM Q)
 where[DIAPHRAGM]
(JUST TO THE RIGHT OF THE BOOK)
(JUST TO THE LEFT OF THE PAD)

Operation of functions

1. jright[x;y]

purpose: To specify in the model that the unique element of set x is located just to the right of the unique element of set y.

method: Check whether the statement is consistent with existing knowledge; i.e., that nothing is known to be between x and y and that y is not known to be to the right of x. If it is not consistent, complain. Otherwise, create a type-1 link indicating the positional relation.

procedure: a) If specify [x] or specify[y] = NIL, terminate.

b) If there already is a type-1 link from y to x following the attribute "JRIGHT", respond "(THE ABOVE STATEMENT IS ALREADY KNOWN)".

c) If it can be proven that y is to the right of x, i.e., if rightp[y;x]=T; or if there is any type-1 link from y following the attribute "JRIGHT"; or if there is any type-1 link from x following the attribute "JLEFT"; then respond, "(THE ABOVE STATEMENT IS IMPOSSIBLE)".

d) If rightp[x;y]=T, and there does not exist a direct type-2 link from y to x following the attribute "RIGHT", respond "(THE ABOVE STATEMENT IS IMPOSSIBLE)".

e) Otherwise, create a type-1 link from y to x following the attribute "JRIGHT"; create a type-1 link from x to y following the attribute "JLEFT"; respond "(I UNDERSTAND)".

2. rightp[x;y]

purpose: To test whether it is known that the x is located to the

right of the y.)

method: rightp[x;y] is defined recursively as follows: if there is no type-1 link from y following the attribute "JRIGHT", and no type-2 link from y following the attribute "RIGHT", the value of rightp is NIL; if either of the above links exists and links to x, the value is T. Otherwise value is the disjunction of the values of rightp[u;y] for all u which are linked to y by one of the above links.

procedure: a) Compute u, the value of the type-1 link from y following the attribute "JRIGHT".

b) If u=x, value is T; if there is no u, go to step (d).

c) If rightp[x;u]=T, value is T.

d) Compute l, the value of the type-2 link from y following the attribute "RIGHT".

e) If x is a member of list l, value is T; if there is no l, value is F.

f) If, for any $v \in l$, rightp[x;v]=T, value is T; otherwise value is F.

3. right[x;y]

purpose: To specify in the model that the unique element of set x is located to the right of the unique element of set y.

method: Check whether the statement is consistent with existing knowledge. If not, complain. Otherwise, create a type-2 link indicating the positional relation.

procedure: a) If specify[x]=NIL or specify[y]=NIL, terminate.

b) If rightp[x;y]=T, respond, "(THE ABOVE STATEMENT IS ALREADY KNOWN)".

c) If rightp[y;x]=T, respond, "(THAT ABOVE STATEMENT IS IMPOSSIBLE)".

d) Otherwise, create a type-2 link from y to x following the attribute "RIGHT"; create a type-2 link from x to y following the attribute "LEFT"; respond "(I UNDERSTAND)".

4. jrightsq[x;y]

purpose: To reply as to whether the x is located just to the right of the y.

method: Determine whether the links in the model indicated that x is just to the right of y, x cannot be just to the right of y, or neither.

procedure: a) If specify[x]=NIL or specify[y]=NIL, terminate.

b) If there is a type-1 link from y to x following the attribute "JRIGHT", respond "YES".

c) If rightp[y;x]=T, or if there is any type-1 link from y following the attribute "JRIGHT"; or if there is any type-1 link from x following the attribute "JLEFT"; then respond, "NO".

d) If rightp[x;y]=T, and there does not exist a direct type-2 link from y to x following the attribute "RIGHT", respond "NO".

e) Otherwise, respond "(INSUFFICIENT INFORMATION)".

5. rightsq[x;y]

purpose: To reply as to whether the x is located to the right of the y.

method: Determine whether the links in the model indicate that x is to the right of y, to the left of y, or neither.

procedure: a) If specify[x]=NIL or specify[y]=NIL, terminate.

b) If rightp[x;y]=T, respond "YES".

c) If rightp[y;x]=T, respond "NO".

d) Otherwise, respond "(INSUFFICIENT INFORMATION)".

6. whereis[x]

purpose: To determine the locations of those objects which have been positioned with respect to the unique element of set x.

method: Reply with the information provided by each positional link associated with x.

procedure: a) If specify[x]=NIL, terminate.

b) Compute u = the value of the type-1 link from x following the attribute "JLEFT"; v = the value of the type-1 link from x following the attribute "JRIGHT"; l = the value of the type-2 link from x following the attribute "LEFT"; m = the value of the type-2 link from x following the attribute "RIGHT".

c) If u, v, l, and m all do not exist, respond "(NO POSITION IS KNOWN)".

d) If u does not exist, go to step (f).

e) Respond, "(JUST TO THE RIGHT OF THE u)", and go to the next step.

f) If v does not exist, go to step (h).

g) Respond, "(JUST TO THE LEFT OF THE v)", and go to the next step.

h) If l does not exist, go to step(j).

i) Respond, "(SOMEWHERE TO THE RIGHT OF THE FOLLOWING . . l)",

and go to the next step.

j) If m does not exist, terminate.

k) Respond, "(SOMEWHERE TO THE LEFT OF THE FOLLOWING . . m)".

7. Locates[x]

purpose: To determine the location of the unique element of set x with respect to as many other objects as possible.

method: Construct a diagram of the left-to-right order of objects by searching through all chains of positional links starting from x and proceeding recursively. The form of the diagram is a list, with objects known to be adjacent appearing in sublists. If no positional links from x exist or if a well-ordering cannot be determined, make an appropriate comment.

procedure: a) If specify[x]=NIL, terminate.

b) Set the initial diagram g="(x)".

c) Compute u = [the value of the type-1 link from x following the attribute "JRIGHT"]. If no u exists or if u is already in g, go to step (f).

d) Insert u just to the right of x in g, i.e., insert u right after x in a sublist of g.

e) Replace g by the result of executing this procedure starting from step (c), with the current value of u replacing the argument x and the current value of g as the diagram.

f) Repeat step (c) for the attribute "JLEFT". In case of failure, go to step (i).

g) Insert u just to the left of x in g.

h) Repeat step (e).

i) Compute l = [the value of the type-2 link from x following the attribute "RIGHT"]. If no l exists, go to step (l).

j) For each m∈l: If m is already in the current g, ignore it; if there exists a v in g which is the object (or first object on a sublist) following x (or the sublist containing x), go to step (k). Otherwise insert m after x (or the sublist containing x) in g, and repeat step (e), with the current value of m replacing x. When all m∈l have been treated go to step (l).

k) If rightp[v;m]=T, insert m after x and continue with the next m in step (j). If rightp[m;v]=T, then just for this value of m replace x by v and continue as in step (j). Otherwise, respond

"(THE LEFT-TO-RIGHT ORDER IS)

g

(TO FURTHER SPECIFY THE POSITIONS YOU MUST INDICATE WHERE THE m IS WITH RESPECT TO THE v)".

l) Perform operations analogous to (i), (j), and (k) for the attribute "LEFT" of x.

m) If the current g="(x)", respond

"(NO RELATIVE POSITION IS KNOWN)".

n) Otherwise respond,

"(THE LEFT-TO-RIGHT ORDER IS) g".

3. whereg[x]

purpose: To determine the locations of those objects which have been positioned with respect to some element of set x.

method: Find an object u of which an x is an example or a part, and which has positional links. Then find the locations of those objects which have been positioned with respect to u.

procedure: a) If x has any positional links, i.e., if the attributes "JRIGHT", "JLEFT", "RIGHT", and "LEFT" of x are not all missing, execute wheres[x].

b) If

$(\exists u)$[[there is a sequence of links following the attribute "SUPERPART-OF-EACH" from x to u] $\wedge$ [u has at least one positional link]], then execute wheres[u].

c) If the hypotheses of step (b) hold for the attribute "SUBSET", execute wheres[u].

d) If

$(\exists u)$[[there is a sequence of links following the attribute "SUPERPART-OF-EACH" from x to u] $\wedge$ $(\exists w)$[[there is a sequence of links following the attribute "SUBSET" from u to w] $\wedge$ [w has at least one positional link]]], then execute wheres[w].

e) Otherwise respond "(NO RELATIVE POSITION IS KNOWN)".

D. Special Features

1) Exemption Principle

Dialogue

```
(***. THERE ARE 5 FINGERS ON EVERY HAND)
 pattrn[FINGER;HAND;5]
(I UNDERSTAND)

(***. THERE ARE TWO HANDS ON A PERSON)
 pattrn[HAND;PERSON;2]
(I UNDERSTAND)

(***, A BOY IS A PERSON)
 setx[BOY;PERSON]
(I UNDERSTAND)

(***. TOM IS A BOY)
 setrs[TOM;BOY]
(I UNDERSTAND)

(***. DICK IS A BOY)
 setrs[DICK;BOY]
(I UNDERSTAND)

(***. HARRY IS A BOY)
 setrs[HARRY;BOY]
(I UNDERSTAND)

(***. TOM HAS NINE FINGERS)
 partrns[FINGER;TOM;9]
(I UNDERSTAND)

(***. DICK HAS ONE HAND)
 partrns[HAND;DICK;1]
(I UNDERSTAND)

(***. HOW MANY FINGERS DOES TOM HAVE Q)
 partrnsq[FINGER;TOM]
(THE ANSWER IS 9)

(***. HOW MANY FINGERS DOES DICK HAVE Q)
 partrnsq[FINGER;DICK]
(THE ANSWER IS 5)

(***. HOW MANY FINGERS DOES HARRY HAVE Q)
 partrnsq[FINGER;HARRY]
(THE ANSWER IS 10)
```

Implementation

General information about "all the elements" of a set is considered to apply to particular elements only in the absence of more specific information about those elements. Thus it is not necessarily contradictory to learn that "mammals are land animals" and yet "a whale is a mammal which always lives in water". In the program, this idea is implemented by always referring for desired information to the property list of the individual concerned before looking at the descriptions of sets to which the individual belongs.

The justification for this departure from the no-exception principles of Aristotelian logic is that this precedence of specific facts over background knowledge seems to be the way people operate, and I wish the computer to communicate with people as naturally as possible.

The present program does not experience the uncomfortable feeling people frequently when they must face facts like "a whale is a mammal which lives in water although mammals as a rule live on land". However, minor programming additions to the present system could require it to identify those instances in which specific information and general information differ; the program could then express its amusement at such paradoxes.

2) Resolving Ambiguities

Dialogue

(****. JOHN IS A PERSON)
(I UNDERSTAND)

(****. DICK IS A PERSON)
(I UNDERSTAND)

(****. A BICYCLE HAS A CHAIN)
(THE ABOVE SENTENCE IS AMBIGUOUS ** PLEASE RE-PHRASE IT)

(****. A CHAIN IS PART OF A BICYCLE)
(I UNDERSTAND)

(****. THE ROWER-SAW HAS A CHAIN)
(THE ABOVE SENTENCE IS AMBIGUOUS ** BUT I ASSUME (HAS) MEANS (HAS AS PARTS))
(I UNDERSTAND)

(****. JOHN OWNS A CHAIN)
(I UNDERSTAND)

(****. DICK HAS A CHAIN)
(THE ABOVE SENTENCE IS AMBIGUOUS ** BUT I ASSUME (HAS) MEANS (OWNS))
(I UNDERSTAND)

(****. THE CUCKOO-CLOCK HAS A CHAIN)
(THE ABOVE SENTENCE IS AMBIGUOUS ** PLEASE RE-PHRASE IT)

Discussion

The criteria used by the program to decide whether "has", in the format "x has y", should be interpreted "has as parts" or "owns" are the following:

a) Let P be the proposition, "either y is known to be part of something, or y is an element of some set whose elements are known to be parts of something".

b) Let N be the proposition, "either y is known to be owned by something, or y is an element of some set whose elements are known to be owned by something".

c) If $P \wedge \sim N$, assume "has" means "has as parts".

If $\sim P \wedge N$, assume "has" means "owns".

If $\sim P \wedge \sim N$, give up and ask for re-phrasing.

d) Let P' be the proposition,

$(\exists u)[[[y \text{ is known to be part of } u] \wedge [y \text{ is an element of some set whose elements are known to be parts of the elements of } u]] \wedge (\exists w)[[u \in w \vee u \subset w] \wedge [x \in w \vee x \subset w]]]$

e) Let N' be the proposition,

(∃u)[[y is known to be owned by u] ∨ [y is an element of some set whose elements are known to be owned by the elements of u]] ∧ (∃w)[[u ∈ w ∨ u ⊂ w] ∧ [x ∈ w ∨ x ⊂ w]]

f) If P' ∧ ~N', assume "has" means "has as parts".

If ~P' ∧ N', assume "has" means "owns".

Otherwise, give up and ask for re-phrasing.

These criteria are simple, yet they are sufficient to enable the program to make quite reasonable decisions about the intended pupose in various sentences of the ambiguous word "has". Of course, the program can be fooled into making mistakes, e.g., as if the sentence, "Dick has a chain", had been presented before the sentence "John owns a chain", in the above dialogue; however, a human being exposed to a new word in a similar situation would probably make a similar error. The point here is that it is feasible to automatically resolve ambiguities in sentence meaning by referring to the descriptions of the words in the sentence-descriptions which can automatically be created through proper prior exposure to unambiguous sentences.

3) Summarizing linkages

Dialogue

(THX. JOHN IS A PERSON)
(I UNDERSTAND)

(THX. JOHN IS AN MIT-STUDENT)
(I UNDERSTAND)

(THX. JOHN IS A BOY)
(I UNDERSTAND)

```
(***. JOHN IS A STUDENT)
(I UNDERSTAND)

(***. EVERY BOY IS A PERSON)
(I UNDERSTAND)

(***. EVERY NON-STUDENT IS A PERSON)
(I UNDERSTAND)

(***. AN NON-STUDENT IS A BRIGHT-PERSON)
(I UNDERSTAND)

(***. AN NON-STUDENT IS A STUDENT)
(I UNDERSTAND)

(***. A BRIGHT-PERSON IS A PERSON)
(I UNDERSTAND)

(***. A STUDENT IS A BRIGHT-PERSON)
(I UNDERSTAND)

(***. A STUDENT IS A PERSON)
(I UNDERSTAND)

(***. SUPPOSE (JOHN))
(I FORGET THE MEMBER-ELEMENTS RELATIONS BETWEEN PERSON AND JOHN)
(I FORGET THE MEMBER-ELEMENTS RELATIONS BETWEEN STUDENT AND JOHN)
(I FORGET THE SET-INCLUSION RELATION BETWEEN PERSON AND NON-STUDENT)
(I FORGET THE SET-INCLUSION RELATION BETWEEN BRIGHT-PERSON AND NON-STUDENT)
(I FORGET THE SET-INCLUSION RELATION BETWEEN PERSON AND STUDENT)
```

Discussion

All question-answering (modal-searching) functions which involve references to set-inclusion or set-membership relations must "know" about the basic properties of those relations, i.e., those functions must have built into them the ability to apply theorems like

$x \subset y \wedge y \subset z \Rightarrow x \subset z$ and

$a \in x \wedge x \subset y \Rightarrow a \in y$;

otherwise the functions would not be able to make full use of the usually limited information available in the form of explicit links. On the other hand, since the functions involved will be "aware" of these theorems, then the set of questions which can be answered is independent of the presence

or absence of explicit links which provide the information to the right of the "$\Rightarrow$" (provided the information to the left of the "$\Rightarrow$" is available, explicitly or otherwise).

The "STREAMLINE" operation starts with the object x which is its argument, and considers all objects linked to x, directly or indirectly, through set-inclusion or set-membership. All explicit links among these objects which can also be deduced by use of the above theorems are deleted. A response of the form "(I DON'T KNOW THE SET-INCLUSION RELATION BETWEEN y AND z)" indicates that whatever links were created by some sentence of a form similar to "(EVERY z IS A y)" are being deleted, and the space they occupied being made available for other use.

In the above example, the STREAMLINE operation deleted almost half the existing links, at no reduction in the question-answering power of the system. However, the time required to obtain answers to certain questions may have been significantly increased.

VIII. Survey of Conclusions

A. Results in Question-Answering

1) The system described above illustrates approximately the largest attainable, in terms of number and complexity of relations and language formats, with the present LISP programming system on an IBM-7090 32K computer.

2) The processing time per statement was about 1 second. The maximum number of statements processed in any one experiment was about 30.

3) The system achieved its goal, which was to investigate the feasibility, possible advantages, and particular problems of the model structure used, rather than to obtain a practical question-answering system.

4) The model used does provide a feasible basis for a larger system.

5) However, the development of such a system would be particularly difficult because of the interactions between different relations.

6) A question-answering system cannot give negative replies without special information about the completeness or consistency of its data.

7) An important feature of the model was that information could be added to it, as well as extracted from it, automatically (unlike, say, the "Baseball" system).

8) Also, automatic restructuring of the model is possible and useful (e.g., as in the "streamline" operation).

9) The input phase of the program illustrates how far one may go toward "understanding" natural language with little, if any, reference to grammatical structure.

10) The system's responses demonstrate that a few fixed formats for replies are sufficient to give the impression of coherent discourse.

B. Possible Extensions of the System

1) All the relations in the present system are binary relations. The model could be extended in obvious ways to handle unary operators (e.g., adjectives), trinary relations (e.g., those involving transitive verbs), etc.

2) Criteria should be developed for choosing "principal objects" in the model, i.e., those objects which should exist as independent described entities, rather than just in the descriptions of other objects.

3) The use of a dictionary and parsing program would greatly increase the power of the input scheme and also the number and kinds of relations to be represented in the model.

4) The intended interpretation of certain class relations, e.g., "SOMEBODY-OR-OTHER", is somewhat confused. Actually different input recognition forms, and corresponding relations, are needed to distinguish between sentences like, "Every x is part of some y", and, "Some x is part of every y". If these cases were distinguished, however, it is not clear how "An x is part of a y" should be interpreted.

5) A more elaborate parsing scheme might include provision for replacing pronouns by their antecedents, and saving certain information from previous sentences to aid in decoding future sentences.

6) Eventually programs must be developed which will automatically produce the programs for handling new relations, so that the capability for handling new relations can be introduced to the system merely by describing, in a suitable English-like language, the relations and their effects.

C. Programming Conclusions

1) This entire project would have been tremendously more difficult if not for the availability of list-processing computer languages such as LISP.

2) In order to handle the interactions between relations, I had to develop some fairly general tree-tracing functions, which proved to be of invaluable use. A programming system based on an elaborate set of such functions would be a valuable aid in developing a new, larger version of this system.

3) A uniform tree linkage and search procedure, flexible enough to handle the most complicated cases, would simplify coding and facilitate the development of the tree-tracing functions described above.

4) Constant feed-back from the program describing its actions and reasons for failure, in readable form, was an essential programming aid.

D. Subjects for Future Experiments

1) The relative merits of different tree-searching procedures should be investigated. In seeking a path between two nodes one might, for example, move one step alternately from each node, rather than moving steadily from one node looking for the other. This former procedure would cut the depth of a successful search in half, at the expense of an elaborate marking or matching procedure.

2) The optimum number of explicit links needed should be investigated--i.e., the relative merits of removing links by "streamlining" procedures versus, say, adding as explicit links all question-answers which are successfully obtained.

3) Methods should be investigated for using the present ideas in a computer system involving large amounts of high-speed auxilliary storage--e.g., by gradually transfering data structures to "read-only" photo storage.

4) Psychological experiments might be developed to test this model as a model for human representations of "meaning" in language. Ideas might thus be obtained for both improving the model and understanding human cognitive processes.

5) A system similar to the one described here could be used as a basis for a study of ambiguity in language. One could investigate algorithms for resolving those kinds of ambiguities which don't seem to give people much trouble.

CS-TR Scanning Project
Document Control Form

Date : 11/30/95

Report # AIM - 59

Each of the following should be identified by a checkmark:
Originating Department:

☒ Artificial Intellegence Laboratory (AI)
☐ Laboratory for Computer Science (LCS)

Document Type:

☐ Technical Report (TR) ☒ Technical Memo (TM)
☐ Other:_____

Document Information

Number of pages: 41 (45-IMAGES)
Not to include DOD forms, printer intstructions, etc... original pages only.

Originals are:
☒ Single-sided or
☐ Double-sided

Intended to be printed as :
☒ Single-sided or
☐ Double-sided

Print type:
☐ Typewriter ☐ Offset Press ☐ Laser Print
☐ InkJet Printer ☐ Unknown ☒ Other: POOR COPY OF MIMEOGRAPH

Check each if included with document:

☐ DOD Form ☐ Funding Agent Form ☐ Cover Page
☐ Spine ☐ Printers Notes ☐ Photo negatives
☐ Other: _____

Page Data:

Blank Pages (by page number): _____

Photographs/Tonal Material (by page number): _____

Other (note description/page number):

Description :	Page Number:
IMAGE MAP : (1-41) UNH'ED TITLE, 1-40	
(42-45) SCANCONTROL, TARGET'S (?)	

Scanning Agent Signoff:
Date Received: 11/30/95 Date Scanned: 12/6/95 Date Returned: 12/7/95

Scanning Agent Signature: Michael W. Cook

Scanning Agent Identification Target

Scanning of this document was supported in part by the **Corporation for National Research Initiatives,** using funds from the **Advanced Research Projects Agency** of the **United states Government** under Grant: **MDA972-92-J1029.**

The scanning agent for this project was the **Document Services** department of the **M.I.T Libraries.** Technical support for this project was also provided by the **M.I.T. Laboratory for Computer Sciences.**