R2—a Natural Language Question-Answering System

Kenneth O. Biss
Robert T. Chien
Fred A. Stahl

University of Illinois — Urbana, Illinois

"This document has been approved for public release and sale; its distribution is unlimited."
R2 - A NATURAL LANGUAGE QUESTION-ANSWERING SYSTEM

by

Kenneth O. Biss, Robert T. Chien and Fred A. Stahl

This work was supported in part by the Office of Education under Contract C-1-7-071213-4557 and in part by the Joint Services Electronics Program (U. S. Army, U. S. Navy and U. S. Air Force) under Contract DAAB-07-67-C-0199.

The material in this report has been accepted for presentation at the 1971 Spring Joint Computer Conference to be held May 18-20, 1971 in Atlantic City, New Jersey and will appear in the Proceedings of the Conference.

Reproduction in whole or in part is permitted for any purpose of the United States Government.

This document has been approved for public release and sale; its distribution is unlimited.
R2 - A NATURAL LANGUAGE QUESTION-ANSWERING SYSTEM*

K. Biss, R. Chien and F. Stahl
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

Abstract

An intelligent natural language question-answering system called the R2 system is described which exhibits a number of advanced features not found in other question-answering systems. This system is based upon a new high-order logical calculus that permits the representation of a wide range of natural language information. A number of examples are given showing how natural language information is represented in this high-order language. An example illustrating the question-answering ability of the system is also given in which a question is transformed into the high-order representation and the deductive steps necessary for generating the answer are performed.

*This work was supported by the Office of Education under Contract No. OE C-1-7-071213-4557 and by the Joint Services Electronics Program (U.S. Army, U.S. Navy, and U.S. Air Force) under contract DAAB-07-67-C-0199.
I. Introduction

A large number of systems involving computers require a high degree of man-machine interaction. In these systems the capability of the computer to process natural language information would be extremely useful as that eliminates the need for the user to learn various formal languages for the purpose of interaction.

In view of the fact that computational costs are decreasing while programming costs are increasing rapidly, computers with natural language processing capabilities will become practical for many applications. A good way to study natural language man-machine communication is through the development of question-answering systems; for it is in these systems that a well-defined natural language discourse takes place.

Research aimed at developing intelligent question-answering systems has been carried on for over a decade. For information concerning these systems we refer the reader to Simmons [13,14]. Of the more advanced question-answering systems Green and Raphael [6,7] have developed a very powerful deductive procedure and Simmons, et al. [12,15] introduced an extremely attractive representation scheme.

Reported in this paper is an intelligent natural language question-answering system called the R2 system. The system introduces a variety of new techniques and encompasses the features found in previous systems. It accepts a wide range of input sentences in English and deduces answers to input questions based upon available information. The data base chosen to demonstrate the capabilities of the system is the Illinois Drivers Manual, Rules of the Road, which consists of about 2000 English sentences.
This particular data base was chosen because its contents are representative of the type of facts that one normally encounters. In addition, the number of facts associated with traffic laws is large enough not to be considered trivial as verified by the fact that it is larger than any other data base currently being used for natural language question-answering systems; yet it is small enough to be handled adequately by our present computer configuration.

The internal representation in the R2 system is based upon a high-order logical calculus that permits the expression of wide range of natural language information. Input to the system undergoes syntactic and semantic analysis in order to be transformed into the internal representation scheme. A recursive goal-oriented theorem-proving algorithm is used to deduce answers to questions posed to the system.

In Section II we comment on some natural language question-answering systems by pointing out some of the features found in these systems. In Section III is described the R2 system and the high-order calculus that is used for the internal representation of natural language information. Finally, an example showing the processing of a typical question is presented.
II. Comments on Some Natural Language Question-Answering Systems

Generally, natural language question-answering systems use some formal internal representation for facts and questions in order to facilitate deductive manipulations. In a number of earlier systems the representation was based upon some type of limited relational calculi, as for example Raphael's SIR [9], and Black's SQA [5].

Green and Raphael [6,7] subsequently developed a system that offered the full expressiveness of the first-order predicate calculus for the representation of natural language information. The deductive procedure of this system was based on an automatic theorem-proving algorithm that was first described by Robinson [10] and improved upon by Wos, et al. [16,17,18,19] and others [1,2,8].

The use of first-order predicate calculus as a formal language for the representation of natural language information when used in conjunction with automatic theorem-proving procedures is a significant improvement over previous schemes. However, the first-order predicate calculus can not be used to express relationships between relations, or allow variables to range over relations as well as objects.

For example, suppose it is necessary to put into the first-order language the sentences

John crossed the street after the light changed. \hspace{2cm} (1)

or

A car must always yield to a pedestrian. \hspace{2cm} (2)
In (1) we are unable to put the sentence into the first-order language because we have a relation, namely after, whose arguments are forced to be relations, namely crossed and changed, rather than some individuals. In (2) we cannot put the sentence into the first-order language because we are faced with the quantification of a variable which ranges over situations not individuals. That is, the sentence states that for all possible situations a certain condition holds (i.e., that a car must yield to a pedestrian).

Simmons, et al. [12,15] developed a system that used nested binary relations for the formal representation of natural language information. The relations they used were of the form (aRb) where a and b could in turn be nested binary relations and R could represent a complex relationship. Note that this scheme overcomes the representational problems mentioned above for the first-order predicate calculus. However, the system still lacks the capability to handle either negation or quantification.

The R2 question-answering system uses a high-order formal language for the internal representation of information and a recursive theorem-proving procedure for performing the necessary deductive operations. This system can represent relations between relations and quantification of variables ranging over rather complex structures. In addition, the goal-oriented theorem-proving procedure performs sophisticated logical deduction through the use of contextual information. In the section that follows a description of the R2 system is given.
III. The R2 Natural Language Question-Answering System

An intelligent natural language question-answering system must be capable of performing sophisticated linguistic processing of input information in order to arrive at a well-structured internal representation upon which extensive logical processing can be performed. That is, any high quality natural language question-answering system must have:

1. an internal data structure sufficiently rich to represent the subtle semantic differences found in natural language information;
2. a method of transforming natural language into that structure; and
3. a strong deduction algorithm for manipulating the information in that structure.

The R2 question-answering system has been designed to encompass these features. A parser and a semantic converter are used to transform natural language information into the high-order language. A deduction algorithm manipulates the facts represented in the high-order language to synthesize the necessary information for the answering of questions posed to the system.

The parser (or syntactic analyzer) breaks down a sentence to show the structural relationships among the parts of the sentence. The parsing indicates what words modify other words (and which words they modify); what are the subject, object, and predicate of the sentence, etc. This type of information is essential to any system that utilizes natural language in an intelligent manner. If, for example, a sentence relating to some action is given, then it is necessary to differentiate between that which is performing the action, and that upon which the action is being performed. The syntactic analysis of that sentence indicates this type of information.
The semantic converter accepts the syntactically parsed statements (i.e., facts and questions) and determines if these parsings are semantically well-formed. If the statement is well-formed, it is converted into the high-order language, and if it is not, then it is returned to the parser.

The deducer receives its information from the semantic converter expressed in the high-order language. Included in this information is a formal statement of the question that the system is attempting to answer. Through a recursive goal-oriented deductive procedure the necessary implicit information is generated. This new information is then used to answer the question.

The high-order language

The high-order language used in the R2 system contains simple objects like car, driveway, pedestrian; more complex objects like green car, or a car in a driveway; relations like 'a car yields to a pedestrian'; relations between relations like 'John crossed the street after the light changed'; and variables that range over these various entities.

Simple type objects like car, driveway, and pedestrian are represented by character strings identical to their orthographic representation. The more complex objects that are modified in some way like green car and a car in a driveway are represented as the unary relations green(car) and (in(driveway))(car), respectively, where the modifier becomes the relation symbol and the object becomes the argument of that relation. Of course, a modifier may also modify an object that is already modified. Thus, a green car in a driveway would become (in(driveway))(green(car)).

In order to appreciate this formalism think of green as a function whose value is equal to its argument with the additional property of being
green. Thus, anything that applies to the object car, without qualification, also applies to the modified object green(car).

A relation like 'a car yields to a pedestrian' is represented as an n-ary relation symbol followed by the appropriate arguments. In this case, the relation is binary and would be represented as

\[ \text{yield(car,pedestrian)}. \]

In general, an n-ary relation may take as arguments any objects, relations, or variables (so long as they make sense in the domain of discourse being considered). These nested relations may be connected with the logical symbols \( \wedge, \vee, \sim, \) and \( \rightarrow \) as in other logical languages, and quantification may occur over any variables appearing in these expressions. For a formal definition of this language see Biss, et al. [3].

Using this language we can now represent the sentences which we could not handle previously. Thus, the sentence

John crossed the street after the light changed

would become

\[ \text{after(cross(John,street),change(light))} \]

where \text{after} is, in this example, a binary relation relating \text{cross} and \text{change}. In the same manner, the sentence

A car must always yield to a pedestrian
would become

\( \forall y (y \text{-must}(\text{yield(car, pedestrian)))} \)

where \( y \) is a variable that ranges over structures such as

\( (\text{in(driveway)})(\text{car}) \)

or

\( (\text{in(crosswalk)})(\text{drunk(pedestrian)}). \)

**Question-answering in the R2 system**

To best illustrate the performance of the R2 system let us look at an example. Suppose the system receives the question:

Do cars always have to yield to pedestrians? \( (3) \)

and it has at its disposal the facts:

Pedestrians not in a crosswalk must yield to cars \( (4) \)

and

If \( x \) must yield to \( y \) then \( y \) does not have to yield to \( x \). \( (5) \)

from the data base already stored in the formal language.

After the question has entered the system it is transformed into the internal formal language and is posed as a theorem for the system to prove. If the theorem is proven, the answer to the question is \textbf{yes}; if the negation of the theorem is proven, the answer is \textbf{no}. 
The parser starts the transformation process by noting that in (3) *cars* is the subject, *yield* the verb, *pedestrian* the object, *always* an adverb modifying the main verb, *have to* is an auxiliary, etc. The grammar used for the syntactic analysis is a modified context-free immediate constituent phrase-structure grammar [10]. The form of the output from the parser is

\[ \text{PREDICATE(modifiers)(SUBJECT(modifiers),OBJECT(modifiers))} \]

Thus, the subject and the object of the sentence are a function of the predicate.

The syntactic information is given to the semantic converter which then puts the sentence into the high-order languages, as in (6). (Notice that the question word *do* is dropped).

\[ \text{always(have to(yield(car,pedestrian)))} \] \hspace{1cm} (6)

This statement is now taken to be the theorem to be proven. At this point the expression *have to* is replaced by *must* in order to normalize the text. Thus we have:

\[ \text{always(must(yield(car,pedestrian)))}. \] \hspace{1cm} (7)

The converter must now check to make sure that (7) does not conflict with what is known about the real world. This is done by using both the semantic rules for this domain:
yield(traffic, traffic) \hspace{1cm} (8)
must(n-ary relation) \hspace{1cm} (9)
always(n-ary relation) \hspace{1cm} (10)

and the tree of Figure 1 as the axioms of a system which must prove the well-formedness of (7).

The tree in Figure 1 gives the information that cars and trucks are motor vehicles, which are vehicles, which are traffic; and that pedestrians are traffic.

From (10) the converter knows that in order for (7) to be considered well-formed it must be shown that

\[ \text{must}(\text{yield(car, pedestrians)}) \] \hspace{1cm} (11)
is a well-formed n-ary relation. But, from (9) we know that (11) is a well-formed unary relation if

\[ \text{yield(car, pedestrian)} \]  

(12)

is a well-formed n-ary relation. From (8) we know that (12) is a well-formed binary relation if it can be shown that car is a type of traffic and pedestrian is also a type of traffic. But from the tree in Figure 1 it is known that both cars and pedestrians are traffic, and, therefore, the yield relation is well-formed, consequently (11) is well-formed. Thus, (7) is well-formed and the semantic converter concludes that what it was given makes sense in the real world.

The last task that the semantic converter must perform is that of replacing always in (7) with a quantifier. Thus (7) would become

\[ \forall y (y \rightarrow \text{must(yield(car, pedestrian))}) \]  

(13)

where \( y \) is a variable ranging over situations. The formula in (13) states that under all situations cars must yield to pedestrians, which is just what (7) says. But (13) implies

\[ \forall x_1 \forall x_2 (\text{must}(x_1(\text{car}), x_2(\text{pedestrian}))) \]  

(14)

where \( x_1 \) is a variable ranging over situations on car, and \( x_2 \) is a variable ranging over situations on pedestrian.
The deducer receives (14) and tries to prove that it is a theorem. The theorem is proven by contradiction using a recursive goal-oriented theorem-proving procedure. That is, an attempt will be made to show that the negation of (14) contradicts the information which the system has. In this particular case the system has (4) and (5) stored in the high-order language as:

\[
\text{must}(\text{yield}((\text{not}(\text{in}(\text{crosswalk}))) \text{(pedestrian)}, \text{car}))
\]

and

\[
\forall x \forall y [\text{must}(\text{yield}(x,y)) \rightarrow \neg \text{must}(\text{yield}(y,x))]
\]

respectively.

At first, the deducer tries to prove that (14) is true by showing that (17) contradicts the relevant axioms, namely (15) and (16) in this case.

\[
\neg \forall x_1 \forall x_2 (\text{must}(\text{yield}(x_1 \text{(car)}, x_2 \text{(pedestrian)})))
\]

The system will, of course, not be able to prove (14) since it is not true, as can be seen by looking at (15). Thus, the system eventually tries to prove that (14) is false, i.e., that (17) is a theorem. In order to prove that (17) is a theorem, we try to prove that its negation (14) contradicts the axioms (15) and (16). Now, (16) is rewritten as

\[
\neg \text{must}(\text{yield}(x,y)) \lor \neg \text{must}(\text{yield}(y,x))
\]

since \(A \rightarrow B\) is equivalent to \(\neg A \lor B\), where universal quantification has been made implicit. Then,
follows from (14) and (18) by recursively applying the high-order resolution on them. Now, (19) resolves with (15) if we let $x_2 = \text{not(in(crosswalk))}$ and $x_1 = \emptyset$ (the empty substitution), generating a contradiction. Thus, (17) which means

cars do not always have to yield to pedestrians

is true. Therefore, the answer to the question

Do cars always have to yield to pedestrians?

is no.

Each step of the deduction may, as in this example, appear as merely the matching of two identical terms or the instantiation of some variable. It should be noted that the theorem-proving procedure can handle much more complex deductions than this. Specifically, the procedure recursively attempts to perform deduction at every level of nesting. For a more detailed description of this procedure along with other examples involving more complex deductions see Biss, et al. [4].

Now that the basic procedure has been illustrated for questions of the yes/no type, we can consider the questions of the form

$\exists x \psi(x)$?

i.e., 'Is there an $x$ such that $\psi(x)$ is true?' where $\psi(x)$ is any statement
involving the variable $x$. The deduction algorithm attempts to produce a substitution instance $p$ such that $\psi(p)$ is true.

Suppose we ask

Under what conditions must a pedestrian yield to a car?

which is represented formally as

$$\exists x (x = \text{must}(\text{yield}(\text{pedestrian, car})))$$

or equivalently as

$$\exists x_1 \exists x_2 (\text{must}(\text{yield}(x_1(\text{pedestrian}), x_2(\text{car}))))$$  \hspace{1cm} (20)

The result of the deduction procedure is that the negation of (20) contradicts the axioms. Therefore, the answer to the question is the substitution instance that was used for the variables in order to arrive at a contradiction. That is, the answer is

when $x_1 = \text{not}(\text{in(crosswalk)})$

or equivalently

$$x_2 = \emptyset$$

or equivalently

when pedestrian not in the crosswalk
IV. Conclusion

The R2 system described in this paper exhibits a number of advanced features not found in existing natural language question-answering systems. It is based upon a high-order logical calculus that allows the embedding of relations, and quantification over rather complex structures thus permitting the expression of a wide range of natural language information.

Two factors, the ease of transforming natural language information into the high-order representation, and the existence of a high-powered recursive deduction algorithm, have made R2 a very powerful system indeed. The techniques developed for the system are directly applicable to many other factual English data bases without modification. These techniques may also be applied to other areas that might involve automated natural language processing, such as: computer-aided instruction, management information systems, or document retrieval systems.
REFERENCES


Distribution List as of 1 November 1970

ESD (ESTI)
L. G. Hanscom Field
Bedford, Mass. 01731 2 copies

Naval Air Systems Command
AIR 03
Washington, D. C. 20360 2 copies

Commanding General
U. S. Army Electronics Command
Fort Monmouth, New Jersey 07703
Attn: ANSEL-KL-PB (2 copies)
(Miss F. Morrise)

Naval Electronic Systems Command
ELEX 03, Room 2046, Main Navy Building
Washington, D. C. 20360 2 copies

Director
Naval Research Laboratory
Washington, D. C. 20390
Washington, D. C. 2027 6 copies

Commander
U. S. Naval Ordnance Laboratory
Attn: Librarian
White Oak, Md. 20910 2 copies

Dr. L. A. Wood, Director
Electronic and Solid State Sciences
Air Force Office of Scientific Research
1400 Wilson Boulevard
Arlington, Virginia 22209 5 copies

Director, Electronic Programs
Attn: Code 427
Office of Naval Research
800 North Quincy Street
Arlington, Virginia 22217 3 copies

Defense Documentation Center
Attn: DOD-TCA
Cameron Station
Alexandria, Virginia 22234 50 copies

Commanding General
Attn: STEGO-WL, Technical Library
White Sands Missile Range
New Mexico 88002 (2 copies)

Commander
Naval Electronics Laboratory Center
Attn: Library
San Diego, Calif. 92152 2 copies

Dr. L. M. Hollingsworth
APXEL (GOX)
L. G. Hanscom Field
Bedford, Massachusetts 01731

Division of Engineering and Applied Physics
210 Pierce Hall
Harvard University
Cambridge, Massachusetts 02138

Director
Research Laboratory of Electronics
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

Miss R. Joyce Harman
Project MAC, Room 810
545 Technology Square
Cambridge, Mass. 02139

Professor R. H. Radiker
Electrical Engineering, Prof.
Mass. Institute of Technology
Building 13-3050
Cambridge, Mass. 02139

Raytheon Company
Research Division Library
26 Seyon Street
Waltham, Massachusetts 02154

Sylvania Electronic Systems
Applied Research Laboratory
Attn: documents Librarian
40 Sylvan Road
Waltham, Mass. 02154

Commanding Officer
Army Materials & Mechanics Res. Center
Attn: Dr. H. Priest
Watertown Arsenal
Watertown, Massachusetts 02172

MIT Lincoln Laboratory
Attn: Library A-082
P.O. Box 73
Lexington, Mass. 02173

Commanding Officer
Office of Naval Research Branch Office
495 Summer Street
Boston, Massachusetts 02210

Commanding Officer (Code 2064)
U. S. Naval Underwater Sound Laboratory
Fort Trumbull
New London, Connecticut 06320

Dept. of Eng. & Applied Science
Yale University
New Haven, Conn. 06520

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-CT-A
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-CT-D
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-CT-I
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-CT-L (Dr. W. S. McAfee)
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-CT-O
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-CT-R
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-CT-S
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-CT-U
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-CT-V
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-CT-W
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-CT-X
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-SC
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-XL-A
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-XL-C
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-XL-D (Dr. H. Bennett)
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-XL-P
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-XL-V
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-XL-DT
Fort Monmouth, New Jersey 07703

Commanding General
U. S. Army Electronics Command
Attn: ANSEL-XL-DT
Fort Monmouth, New Jersey 07703
An intelligent natural language question-answering system called the R2 system is described which exhibits a number of advanced features not found in other question-answering systems. This system is based upon a new high-order logical calculus that permits the representation of a wide range of natural language information. A number of examples are given showing how natural language information is represented in this high-order language. An example illustrating the question-answering ability of the system is also given in which a question is transformed into the high-order representation and the deductive steps necessary for generating the answer are performed.
<table>
<thead>
<tr>
<th>KEY WORDS</th>
<th>LINK A</th>
<th></th>
<th>LINK B</th>
<th></th>
<th>LINK C</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ROLE</td>
<td>WT</td>
<td>ROLE</td>
<td>WT</td>
<td>ROLE</td>
<td>WT</td>
</tr>
<tr>
<td>Natural Language</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Question-Answering</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Artificial Intelligence</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Theorem Proving</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fact Retrieval</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Semantics</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Syntax</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deduction</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>