Laboratory 13

Artificial Intelligence

Objective

- Learn how semantic networks and rule-based natural language systems can simulate intelligent behavior.

References

Software needed:

1) A web browser (Internet Explorer or Netscape)
2) Applets from the CD-ROM:
   a) Semantic networks
   b) Eliza therapist

Textbook reference: Chapter 13, pp. 417–424, 433–437

Background

Artificial Intelligence is discussed in Chapter 13 of your textbook.

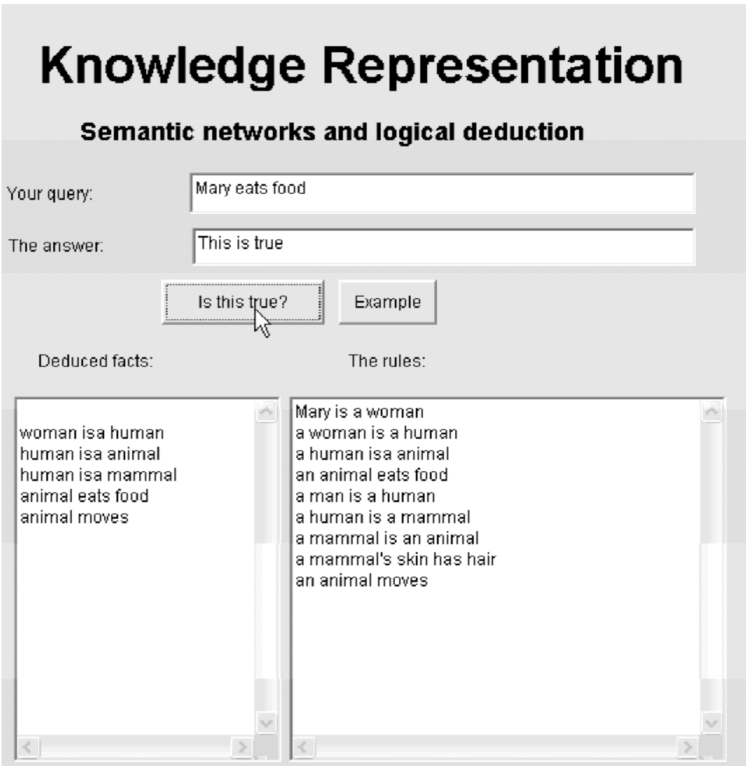
Activity

Part 1

Knowledge is an essential part of acting intelligently, as is a method for using that knowledge. Computers running AI programs use information to make decisions, filter inputs, and generate new knowledge. The structure of information inside the computer's memory is called a knowledge representation technique. Semantic networks are one kind of knowledge representation technique. They are often depicted as graphs of linked nodes (textbook p. 422). However, the graph can be represented textually by statements, and of course computers are much more comfortable with text than with pictures.

The "Semantic networks" applet allows you to enter rules in English statements and then ask a question called a *query*. Applying the rules using a process known as deduction, the applet tries to answer the query. If it can positively answer the question, the applet says, "This is true." If it can't, it says the statement is false or cannot be answered. Some logical deduction systems, such as the Prolog language, use the closed world assumption, meaning that all information that is true is included in the rule base, anything that is not in the rule base, is false. Other AI systems do not make such a grand assumption, and merely state that they cannot prove a statement is true if it's not in the rule base. The statement might be false, or the rule base might be incomplete.

Start the "Semantic networks" applet and click on the *Example* button. Then rev up the logic inference engine by clicking on *Is this true?* Here's what you will see:



Experiment with some other queries, such as `a woman eats food` or `a man is an animal`. The applet requires that you use no punctuation or capital letters.

Type in the query `a woman is a man`. What does the applet say? If we type `Mary has blood`, we also get "This is false or can't be answered." While we know this statement is true, our very small rule base doesn't cover that topic, so deduction is impossible.

Obviously, it would take a very large rule base to create a robust, "real world" logical deduction system. Just how large the rule base would have to be is a hotly debated issue in AI. Computer scientist Doug Lenat has been building a gigantic system named CYC for over 10 years. (CYC is pronounced "sike" and is short for enCYClopedia.) He believes that if about 10 million common sense facts are stored in CYC, it can begin to function like a human. Perhaps you'd like to try typing 10 million facts into the applet to see if it's able to function like a human.

As you're typing in your 10 million rules, be aware that the applet is very limited in terms of the kinds of phrases it can process (we won't use the loaded word *understand* anymore). Here are the patterns the applet processes:

```
noun isa noun
noun verb
noun verb object
noun's noun verb object
```

The only *keyword* (meaning a word that must appear exactly as it is, without substitutions) is isa. English words can be substituted for all the others. The 's that is tagged onto the end of the first noun is optional.

You might be wondering about this key word `isa`, since you can see that the rules contain regular English statements like `Mary is a woman`. The reason you can use regular English in the rules is because of the clever way the applet processes rules and queries. First, it combs through the rules and queries, replacing phrases like `is a` and `is an` with our keyword `isa`, as you can see in the *Deduced facts* column. Also, notice that articles such as `a`, `an`, and `the` are totally eliminated. This kind of pre-processing is very common in computer systems because it simplifies the programmer's task.

The logical deduction system inside this applet matches patterns. If a query matches a rule exactly, it is true. If not, the applet tries to replace the first noun with a second one that appears in an `isa` rule. For instance, we know that the statements `woman isa human` and `human isa animal` and `animal eats food` are all true. By making two replacements, we find out that `woman eats food`. Since `Mary isa woman`, we can safely conclude that `Mary eats food`. The same happens for possessive nouns; they can also be replaced by their *superclasses*. (`human` is a superclass of `woman`, and `animal` is a superclass of `human`.)

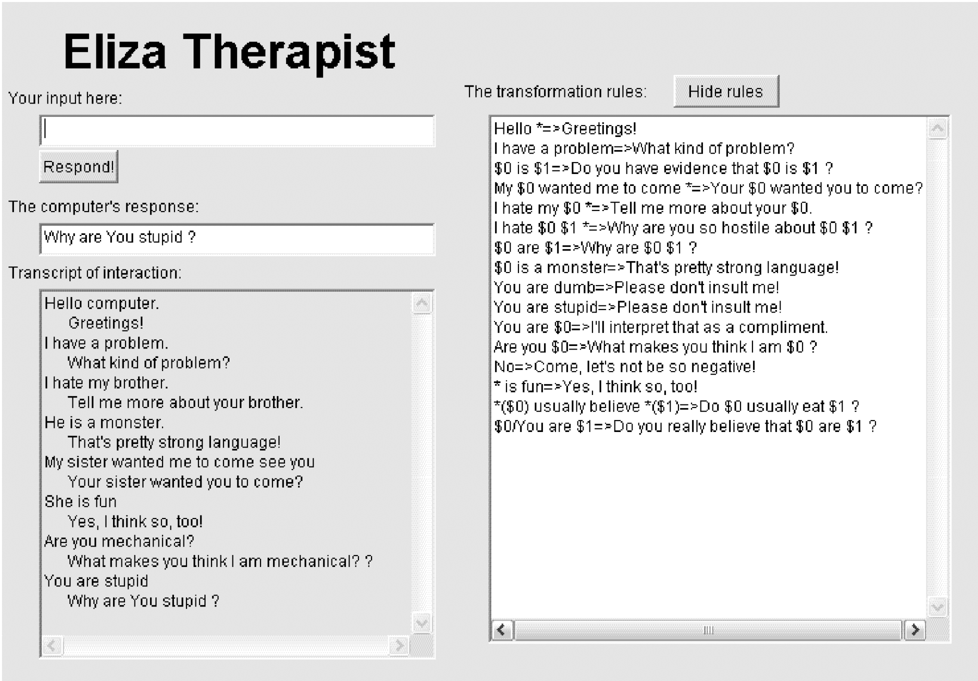
Tip

"Semantic networks" can be used as a standalone Java application. If you use it as an application (not as an applet), you can load and save your rules. To run the Java application, navigate to the folder containing the Semantic networks class files and double-click on the **run_application.bat** file.

Part 2

One of the classic, early AI programs is ELIZA, written by Joseph Weizenbaum in 1963. An example of a famous ELIZA conversation with a human is given on p. 438 of the textbook. ELIZA appears to be a therapist who responds to a patient's statements by cleverly rewording them and turning them into questions. Though ELIZA's method sounds pretty simple, and though no one could persuasively argue that the computer is really thinking when it is running ELIZA, the program's effect on people was surprisingly powerful. In fact, Dr. Weizenbaum, shocked by people's reactions and fearing the potential for misuse of artificial intelligence, turned against the field!

To begin, start the "ELIZA therapist" applet and type a question or statement into the top text area, pressing *Return* when you're done. ELIZA responds in the text area below. A running transcript of your inputs and the computer's responses is kept in the larger area below that. After you type for a while, click on the *Show rules* button.



The original Eliza used only a few simple rules. Eliza turned the client's statement into a question merely by appending a question mark to the statement and switching the pronouns, as shown below:

Eliza didn't even use the complicated English question construct "Did your mother make you come here?" This is more difficult because the verb has to be changed and a form of the verb "to do" has to be inserted.

Our version of Eliza uses a different approach. Instead of one or two simple transformation rules, the applet allows the user to enter any number of rules. Eliza then tries to match the input statements with the rules, mixing up the order by starting at a random spot in the list each time.

The rules have the form

```
pattern=>response
```

Inside both the pattern and the response, variables can appear. They are marked with a question mark and are followed by the numbers 0, 1, 2, etc. For example:

```
My $0 made me come here *=>Why did your
                           $0 make you come here?
```

In this pattern, `$0` is matched against a single word that appears between the words `My` and `made`. The asterisk at the end of the pattern means none, a few, or many words may appear here. They are ignored. The words that must match are `My`, `made`, `me`, `come`, and `here`, in that order, with a word between `My` and `made`, symbolized by $0.

If Eliza can't find a rule that matches the statement, it moves on to the next rule. If no rules match any of the statements, Eliza picks a standard response such as "Let's move on" or "Tell me more."

If you want to match one or more words, surround the variable with an asterisk and parentheses, as in the following rule:

```
*($0)=>$0 ?
```

This rule merely spits back the statement as a question, although it doesn't do pronoun switching like the original Eliza. You can see pronoun switching in some of the other rules.

Another thing that you can put in a rule is to forbid the matching of some words, as in the following rule:

```
$0/You are $1=>Do you really believe
that $0 are $1 ?
```

This rule tries to match a sentence that has "are" as the second word, but the first word *cannot* be `You`. Also, as shown in the rule above, if your rule ends with a variable followed by punctuation, leave a space between the variable and the punctuation mark.

Unfortunately, this Eliza applet has many limitations. Some of the purely mechanical ones include the inability to recognize uppercase and lowercase words as essentially the same word. It also doesn't know about English grammar rules, such as the rule that only plural nouns may precede the verb `are`. Thus, `Water are wet` would be accepted by the above rule—which is clearly wrong!

Eliza's use of key words when matching rules to client inputs can result in inept responses. A classic example is:

Client: `Necessity is the mother of invention.`
Eliza: `Tell me more about your mother.`

Eliza knows nothing about metaphors, hyperbole, or similes. In short, Eliza flunked high school English!

In addition, of course, Eliza cannot be said to *understand* the client's input. Thus, it converses without thinking, as happens far too often at cocktail parties and political rallies. There are other AI systems that actually attempt to tease out the meanings of the words and phrases, but they are still largely experimental. Natural language understanding is a *long* way off from the chatty computers on the TV series *Star Trek*!

Tip

Eliza can be used as a standalone Java application. If you use it as an application (not as an applet), you can load and save the transformation rules. To run the Java application, navigate to the folder containing the Eliza class files and double-click on the **run_application.bat** file.

Exercise 1

Name ______________________________ Date ____________________

Section ______________________________

1) Start the "Semantic networks" applet.
2) Add a new isa rule to the rule base. This rule may extend the human/animal categories or do something entirely different.
3) Add a new rule that uses a verb other than is and mentions your new category.
4) Add a new rule that gives a characteristic of your new category. This rule puts your new category in front, in the form of a possessive noun.
5) Type in one query that should evaluate to "true." Take a screenshot.
6) Type in another query that should evaluate to "false." Take a screenshot.
7) Type in

   John's skin has hair

   and press the button "is this true?" Explain the applet's answer.
8) If the applet says this is false, what rule could you add to the rule box so that it would deduce it as true?

Exercise 2

Name ________________________________ Date ____________________

Section ________________________________

1) Start the “Eliza therapist” applet.
2) Click on *Show rules*.
3) Type in a rule that turns the client’s statements:

    My _____ is ruined

   into:

    Are you sure your _____ is ruined

4) Test out your new rule by pretending to be the client.
5) Add a new rule that turns the client’s statements:

    I have a ____ for a pet

   into:

    It must be interesting to have a ____

   The first blank, however, *cannot* be the word “cat.”

6) Test this new rule. Take a screenshot, showing the rules and the above two statements, along with Eliza’s responses.
7) Type in a statement that makes Eliza come up with a grammatically incorrect response. Take in a screenshot.
8) Type in a statement that makes Eliza come up with a nonsensical response, similar to the “mother of invention” non sequitur on page 205 of this manual. Take a screen shot.

Deliverables

Turn in two screenshots from the “Semantic networks” applet, clearly showing your three new rules, your queries, and the response. Also, turn in two screenshots from Eliza.

If you are running “Semantic networks” or Eliza as standalone Java applications, save your rules files. Your instructor may want you to hand them in. Consult the instructor for details on how to do this.

Deeper Investigation

Think back to the statement `a woman is a man`, which the “Semantic networks” applet claimed was false or unanswerable. We know this statement to be false because the categories `woman` and `man` are mutually exclusive. The power of our logic machine could be vastly increased if we added some *metarules*, or rules about rules. In this case, a thing can’t belong to two mutually exclusive categories. That’s the metarule. Then we would have to tell the machine that `woman` and `man` are mutually exclusive. The great thing about this approach is that we can type in a bunch of mutually exclusive categories, and the metarule would instantiate the actual rules that make the inferences. For example, a metarule that states `woman` and `man` are mutually exclusive categories of “human” would automatically create all these rules.

```
woman isa human
man isa human
woman is not a man
man is not a woman
```

Can you think of other mutually exclusive categories? When does the line get fuzzy? Is the real world essentially nice and clean, or essentially messy? Also, can you think of other useful metarules? Finally, try to imagine how to build a system that could learn new rules on its own instead of relying on a human to type in millions of rules.