INVITED PAPER

EXPERT SYSTEMS: A QUICK TUTORIAL

Dr. Joseph Schmuller, Mgr. of Expert Systems
CDM-Federal Programs Corporation
13135 Lee Jackson Memorial Highway
Fairfax, VA 22033

ABSTRACT: An expert system is a computer program that contains stored knowledge and solves problems in a specific field in much the same way that a human expert would. The knowledge typically comes from a series of conversations between the developer of the expert system and one or more experts. The completed system applies the knowledge to problems specified by a user. This paper describes expert systems, their components, their development, and the tools used to build them.

KEYWORDS: Expert Systems, Knowledge Base, Inference Engine, Knowledge Engineer, Domain Expert, Shell.

INTRODUCTION

Within the last ten years, artificial intelligence-based computer programs called expert systems have received a great deal of attention. The reason for all the attention is that these programs have been used to solve an impressive array of problems in a variety of fields. Well-known examples include computer system design, locomotive repair, and gene cloning (1).

How do they do it? An expert system stores the knowledge of one or more human experts in a particular field. The field is called a domain. The experts are called domain experts. A user presents the expert system with the specifics of a problem within the domain. The system applies its stored knowledge to solve the problem.

EXPERT SYSTEM COMPONENTS

The part of the expert system that stores the knowledge is called the knowledge base. The part that holds the specifics of the to-be-solved problem is (somewhat misleadingly) called the global database (think of it as a kind of "scratch pad"). The part that applies the knowledge to the problem is called the inference engine.

As is the case with most contemporary computer programs, expert systems typically have friendly user interfaces. A friendly interface doesn't make the system's internals work any more smoothly, but it does enable inexperienced users to specify problems for the system to solve and to understand the system's conclusions.

BUILDING AN EXPERT SYSTEM

Expert systems are the end-products of knowledge engineers. To build an expert system that solves problems in a given domain, a knowledge engineer starts by reading domain-related literature to become familiar with the issues and the terminology. With that as a foundation, the knowledge engineer then holds extensive interviews with one or more domain experts to "acquire" their knowledge. Finally, the knowledge engineer organizes the results of these interviews and translates them into software that a computer can use.

The interviews take the most time and effort of any of these stages. They often stall system development. For this reason, developers use the term knowledge acquisition bottleneck to characterize this phase.

REPRESENTING THE KNOWLEDGE

The format that a knowledge engineer uses to capture the knowledge is called a knowledge representation. The most popular knowledge representation is the production rule (also called the if-then rule). Production rules are intended to reflect the "rules of thumb" that experts use in their day-to-day work. These rules of thumb are also referred to as heuristics. A knowledge base that consists of rules is sometimes called a rule base.

To clarify things a bit, imagine that we have set out to build an expert system that helps us with household repairs.

Toward this end, we have held lengthy interviews with an experienced plumber. Here are two if-then rules that the plumber has told us. (Bear in mind that a working expert system can contain tens of thousands of rules.)

Rule 1:

If you have a leaky faucet
And
If the faucet is a compression faucet
And
If the leak is in the handle
Then tighten the packing nut.

Rule 2:

If you've tightened the packing nut
And
If the leak persists
Then replace the packing.

(A compression faucet has two handles, one for hot water, the other for cold. The "packing" and the "packing nut" are two items that sit under a faucet handle.)

In each rule, the lines that specify the problem situation (the lines that begin with "if" and "and") are called the antecedent of the rule. The line that specifies the action to take in that situation (the line that begins with "then") is called the consequent.

WORKING WITH THE KNOWLEDGE

In the two rules in the preceding section, note that the consequent of Rule 1 appears (slightly changed) in the first line of the antecedent of Rule 2. This kind of inter-rule connection is crucial to expert system operation. An inference engine uses one of two strategies to examine interconnected rules.

In one strategy, the inference engine starts with a possible solution and tries to gather information that verifies this solution. Faced with a leaky faucet (and knowing our two rules), an inference engine that follows this strategy would try to prove that the packing should be replaced.

In order to do this, the inference engine looks first at Rule 2 (because its consequent contains the conclusion that the inference engine is trying to prove), then at Rule 1 (because its consequent matches a statement from Rule 2's antecedent). This process is called backward chaining.

Expert system work is moving in several directions. One is to use expert systems as the foundation of training devices that act like human teachers, instead of like the sophisticated page-turners that characterize traditional computer-aided instruction.

When the inference engine needs information about the problem that isn't in the knowledge base, the system questions the user (the question-answer type of interaction typifies backward-chaining systems). The user's answers become part of the problem specification in the global database. Because backward chaining starts with a goal (the solution it tries to verify), it is said to be goal-driven.

In the other strategy, the user begins by entering all the specifics of a problem into the system, which the system stores in its global database. After this, the system usually does not question the user further. The inference engine inspects the problem specifications and then looks for a sequence of rules that will help it form a conclusion.

In our leaky faucet example, the system user might specify the problem as a leaky compression faucet with a leak in the handle. The inference engine examines Rule 1 (because its antecedent matches the specifics of the problem), and then Rule 2 (because its antecedent contains a statement from Rule 1's consequent). In our example, Rule 2's consequent is the conclusion. This process is called forward chaining. Because this process starts with data (specification of a problem), it is said to be data-driven.

DEVELOPMENT TOOLS

One of the first expert system development tools was a by-product of one of the first expert systems. In the 1970s, Stanford researchers developed a system called MYCIN. MYCIN contains a multitude of rules (coded in the computer language LISP) that represent medical knowledge and enable the system to perform medical diagnoses (2).

MYCIN's developers reasoned that removing the medical diagnosis rules should not affect the workings of the system's inference engine and global database. With the medical knowledge gone (resulting in a system they named EMYCIN -- "E" for "empty"), they also reasoned that they could insert knowledge from other domains into the knowledge base and thus build a fully functioning expert system. Because they were right on both counts, the expert system shell was born.

An expert system shell contains pre-coded expert system components (including either backward chaining, forward chaining, or both). The knowledge base is empty, its framework intact. All of this makes it unnecessary to rebuild the components for each new expert system. The knowledge engineer just adds the knowledge.

CONCLUSION

Expert system work is moving in several directions. One is to use expert systems as the foundation of training devices that act like human teachers, instead of like the sophisticated page-turners that characterize traditional computer-aided instruction. The idea is to combine one expert system that provides domain knowledge with another expert system that has the know-how to present the domain knowledge in a learnable way. The system could then vary its presentation style to fit the needs of the individual learner. While this concept is not new, today's powerful PCs are starting to put such trainers, called ICAI (Intelligent Computer Assisted Instruction) systems, within everybody's reach (3).

Another important direction concerns expert system shells. Tomorrow's shells will routinely allow developers to embed inference engines into other kinds of programs.

Embeddable inference engines set up a number of fascinating potential breakthroughs. Will word processors become so intelligent that they grasp the gist of what we want to write and then write it better than we can? Will databases comprehend the information we need, help us find it, and then suggest a search for supplementary information? Will smart spreadsheets tell us the kinds of models we should use in our projections and then build them for us? The possibilities are limited only by the knowledge bases and inference engines that reside in our heads.

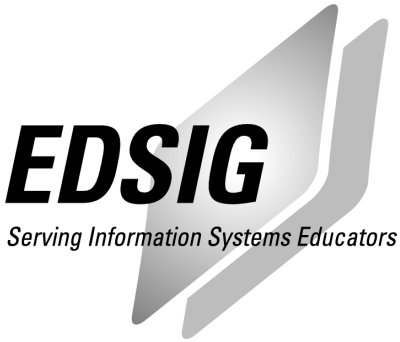
REFERENCES

1. Feigenbaum, E. and McCorduck, P. The Fifth Generation. Signet, 1984.
2. Buchanan, B. and Shortliffe, E. (Eds.) Rule-Based Expert Systems. Addison-Wesley, 1984.
3. Kearsley, G. (Ed.) Artificial Intelligence and Instruction. Addison-Wesley, 1987.

AUTHOR'S BIOGRAPHY

Joseph Schmuller, Ph.D., is Editor in Chief of PC AI (a magazine which deals with intelligent solutions for desktop computers). He has done extensive research and teaching in cognitive science, and is an experienced knowledge engineer.

Information Systems & Computing
Academic Professionals



STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 1055-3096