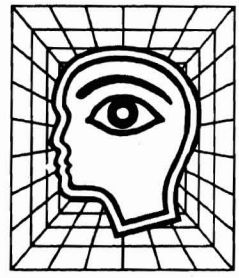
7

Expert/Knowledge-based Systems



Early workers in the field of AI hoped to construct an intelligent machine with a perception component that sensed and interpreted the world, a component that allowed the system to learn from its experiences, and a mechanism for solving general problems arising in interactions with the environment. The knowledge database would be provided initially by the human designer, and then develop autonomously as the system learned about the world.

Although research based on this paradigm is still being pursued, workers on systems that aspire to a human level of performance have considerably scaled down their original goals. Emphasis has moved from a general problem solving approach to a concentration on specialized problem domains. The basic assumption is that the main source of capability in problem solving resides in the knowledge database of the system, rather than in the power of the deductive apparatus used. There is no perceptually based autonomous learning component in these systems. Rather, all of the knowledge is entered by the human designer, or user, of the system; the emphasis is on better ways to express and use the specialized knowledge. *Expert systems* based on this philosophy have already been applied to diagnosis and design problems in many fields, and have turned out to be some of the most commercially

successful applications of artificial intelligence.

Even though the problem scope has been narrowed in going from general problem solving to nonlearning expert systems, many basic issues still remain. For example, there is the fundamental question as to whether the incremental acquisition of piecemeal knowledge can lead to global understanding of a subject. There is the issue of whether intelligent behavior can still be achieved if we partition off, and operate within, an isolated segment of human knowledge or experience. The issue of control also arises: there must be some method of determining which knowledge is pertinent to a given situation and how to guide the sequence of deductions or transformations without explicit programming. Thus, many of the problems that arise in the design of expert systems involve familiar AI issues, and especially the trade-offs between generality and effectiveness of operation.

This chapter will be primarily concerned with the "production system" approach. We will briefly describe its importance in the psychological modeling of intelligence, but we will focus particularly on its use in expert systems. We will also discuss general issues, such as the problem of knowledge acquisition, knowledge representation, and knowledge utilization. Some of the questions to be addressed are:

- What are the characteristics of a human expert?
- What is a production-rule expert system?
- What are the characteristics of a knowledge domain that make it suitable as an application for an expert system?
- How can a domain of knowledge be represented by data structures in the memory of the computer, and how can this knowledge database be used in problem solving?
- What are the basic AI issues that arise in the design of expert systems?

HUMAN EXPERTS

The dictionary defines an expert to be a person with a high degree of skill in, or knowledge of, a certain subject. The word comes from the Latin *experius*, past participle of *experiri*, to try. The word is apt, because an expert usually attains that status by investing a large amount of time in trying, in obtaining experience in the special domain.

The term *prodigy* is used when exceptional ability is shown at an early age. There have been prodigies in mathematics, chess, and music, but it is rare to find prodigies in acting, writing, drawing, and painting. It is interesting that while there are some notable exceptions, few mental prodigies meet early expectations, while most musical prodigies do go on to successful careers.

A remarkable memory ability characterizes many classes of experts; for example, chess masters are able to play many games simultaneously while blindfolded. To study visual memory in chess, subjects are typically exposed to a board position for a short time and are later asked to reproduce the board. Chess masters impressively outperform novices only when meaningful board positions are used. When the pieces are put down at random, then little difference exists. De Groot [De Groot 65], in his study of chess masters, used an introspective method where

the expert described his inner thinking processes. Chess masters reported remembering the positions of "chunks" consisting of four or five pieces rather than individual pieces. The same type of results have been obtained in analyzing the abilities of computer programming experts [McKeithen 81]. Both experts and novices are shown programs for a brief time. Experts are quite good at reconstructing these programs. However, when presented with scrambled, meaningless programs, experts perform in a manner similar to the novice.

Thus, pattern retention and recognition play an important role in expert performance. Simon [Newell 72a] estimates that masters have acquired on the order of 50,000 different chess patterns that they quickly recognize. These stored patterns and their analysis free the expert to concentrate on deeper strategies. Studies of computer programming [Soloway 80] also indicate that experts in that field have a large supply of "templates" that are applied to programming problems.

Studies of expert-vs.-novice problem solving in physics [Larkin 81] have shown that the reasoning approach used by the novice differs from the expert. The novice uses a means-ends analysis, setting up and satisfying a sequence of subgoals. This requires the retention of many intermediate results. The expert seems to "know where he is going," and uses forward reasoning. The expert has the experience to know which of the possible forward reasoning alternatives are relevant to the solution.

Representation is a crucial issue. In some fields, such as mathematics, experts often convert problem statements into the terms of their specialty, and then use their expertise in that specialty to solve the problem. Experts may have a collection of models that they employ in solving problems, e.g., a specialist in internal medicine will use a model of the digestive system, of the blood circulation, etc., in order to develop hypotheses concerning an illness.

When attempting to devise machines that are expert in a specialty, it is desirable to emulate some of the key characteristics of the human expert:

- Use of appropriate models of the world and the ability to reason using these models
- A smooth falling off of competence as the problem departs from his area of expertise, and ability to deal with departures from a standard problem
- Knowledge of the limits of the expertise
- Ability to learn from experience
- Ability to explain what is being done and why it is being done

In the remainder of this chapter, we describe some mechanisms used to achieve expert behavior in a machine.

PRODUCTION SYSTEMS

The production system (PS) is a simple concept that has been used in a wide variety of applications, ranging from investigations of human intelligence to computer-based expert systems. In 1943, Emil Post, a mathematician at the City College of New York proposed an "if-then" rule-based system that indicated how strings of symbols could be converted to other symbols [Post 43]. Post used his productions to study computability. While

he and Turing worked independently, it was later shown that the Post system is formally equivalent to the Turing machine.

In the 1960s Newell and Simon [Newell 72] embedded the generality of the if-then representation in a data-driven control structure to produce the present concept of a production rule system that views computation as the process of applying rules *in an order determined by the data*. This view is fundamentally different from the explicit sequencing of control found in conventional programs. A PS consists of three components:

1. **Working memory (WM).** The data representing the current state of the "world." The WM can be finite or "unlimited" in size. Various strategies can be used in the finite case when the WM fills up, e.g., drop the oldest data, keep only important data, etc.
2. **Production rules.** A set of rules of the form, If <condition in WM> THEN <action>, often designated using the notation A → B. This indicates that "if pattern A matches a pattern in WM, take the action described in B." In expert diagnosis systems, rules are often of the form: IF <evidence e is present in the WM> THEN <add hypothesis h to the WM>. Rule-based systems permit representation of knowledge in a highly modular manner, and are relatively easy to modify, allowing knowledge to be added incrementally.
3. **Rule interpreter.** The control module that carries out the matching operation to determine the next rule to be activated. Some strategies used when the condition portion ("left side") of more than one rule is satisfied are: rule execution, based on the ordering of rules on a rule-list; rule execution of more specific rules before general rules; and rule execution based on the length of time since a rule was previously invoked.

Thus, "Computation" in a PS is the action taken when the condition for a particular rule is satisfied by data in the WS, the repository of information about the current state of the world.

Control Structures Used in Production Systems

The strategy used by the rule interpreter is called the "control strategy." The control strategy can be employed in various ways: (1) the data-driven mode[16] that interprets rules to mean that if a specified condition is observed in the WM, then a corresponding action is taken; (2) the backward chaining mode[17] that interprets a rule to mean that if a certain action is desired, the system should try to establish the corresponding condition, and (3) a mixture of these.

Data-Driven Control. At first glance, the rules used in a production system (PS) seem very much like the well-known *if-then* conditional statements used in most high-level programming languages. However, although the form is the same, the way that the rules are used differs consid-

[16]Also called the antecedent, event driven, or forward chaining mode.
[17]Also called the consequent, goal driven, or hypothesis driven mode.

erably. Unlike a conventional system in which the program performs the sequence of programming steps specified by the programmer, activity in a PS is controlled strictly by the content of the working storage. At every computation cycle of the system, the left-hand sides of the set of rules are examined to determine which rules are satisfied by the contents of the working storage. As indicated above, if more than one rule is activated, there must be a method of determining which rule should be allowed to act. A typical "recognize-act" computation cycle is:

- **Selection.** A determination is made as to which rule modules and working storage items are allowed to be considered for use during the next cycle.
- **Matching.** The active rules examine active working storage looking for a pattern match.
- **Scheduling.** A determination is made as to which of the satisfied rules should be allowed to *fire*.
- **Execution.** The rules perform their actions in the order determined by the scheduling operation.

By choosing the rule to be executed on the basis of the contents of the working storage, a complete re-evaluation of the state of the system must be performed every computation cycle. This is one of the key features of a PS: a PS is sensitive to any change in the entire environment (any new data in memory) and can react to such changes within a single cycle. The price of this responsiveness is the computation time required for the re-evaluation. Because the working storage is accessible to every rule in the system, this database acts as a broadcast communications channel.

In a *pure* PS, the specified action (right-hand side) is a simple action and not a complex procedure. Also, one avoids rules that place private messages in the working storage (messages that enable only special rules to be activated). Practical expert systems often depart from the pure PS format. For example, it may be natural to use certain rules in various stages of problem solving, and to use different rules in other stages. The set of productions is therefore often tagged to indicate at which stage of the problem solving procedure they can be activated. The more we depart from *purity* of PS, the more we lose the advantage of modularity of the PS rules.

Goal-Driven Control. In goal-driven control, the rules are used in a *backward-chaining* mode, by examining the rules in the database to see if a desired goal can be found on the right-hand side of some rule. If such a rule is found, then the system determines the facts required to actuate the left-hand side. Further backward chaining may be required to accomplish this, and at any time several backward chains may be in progress. This type of control limits attention to the rules that can contribute to the goal. For example, suppose the goal is to establish D and we have the following rules:

$$A \rightarrow B \quad (1)$$
$$B \rightarrow C \quad (2)$$
$$C \rightarrow D \quad (3)$$

The system finds the goal, D, on the right-hand side of rule (3) and backward chains by looking at the left-hand side of (3) to find that the required subgoal is finding C in the working storage so that rule (3) can be activated. Working on this subgoal then leads to (2) which could place C in the working storage if there were a B in the working storage. B is the new subgoal, leading finally to (1) which tells the system to establish A, possibly by asking the user if A is true.

Note that if the user asks why the system wants to know A, it can delineate the backward chaining sequence as the explanation. Thus, it would state that knowing A would, by rule (1) establish B, which by rule (2) would establish C, which by rule (3) would establish our original goal of D. This type of explanation merely indicates the role that the data and the rules play in the computation and does not try to present a deeper explanation of the situation.

An example of backward chaining as used in an expert repair system is presented in Box 7-1.

Other Control Strategies. A combination of data-driven and goal-driven procedures is often used in a practical system. For example, information volunteered by

BOX 7-1 Use of Backward Chaining in an Expert Repair System

Suppose we have the set of rules given below as part of the rules in an expert system for repairing automobile air conditioners.

- RULE 002: IF CAR DOES HAVE THERMAL LIMITER, AND THERMAL LIMITER DOES NOT HAVE POWER, THEN FAULTY FUSE.
- RULE 008: IF CLUTCH DOES NOT HAVE POWER, AND CAR DOES NOT HAVE THERMAL LIMITER, THEN FAULTY FUSE.
- RULE 017: IF TEST LIGHT CONNECTED TO CLUTCH WIRE, AND TEST LIGHT IS OFF, THEN CLUTCH DOES NOT HAVE POWER.
- RULE 021: IF TEST LIGHT CONNECTED TO THERMAL LIMITER WIRE, AND TEST LIGHT OFF, THEN THERMAL LIMITER DOES NOT HAVE POWER.

Suppose the expert system is trying to establish that the air conditioning system is not working because the fuse is faulty. It establishes FAULTY FUSE as a goal. The system backward chains, looking for rules that have a THEN part FAULTY FUSE. RULE 008 and RULE 002 are such rules. Since the system has no information that can satisfy the IF part of these rules, it must ask the user a question.

System: Does the car have a thermal limiter?

User: No.

From this answer, the system finds that RULE 002 does not apply, but RULE 008 does. However, the IF part of RULE 008 requires CLUTCH DOES NOT HAVE POWER. The system has a new goal, and looking at the THEN parts of the rules, it finds RULE 017. RULE 017 requires that a test be performed by the user, and the system informs the user:

System: Connect test light to clutch wire. What is status of test light.

User: Test light is off.

The system now notes that the IF part of RULE 017 has been satisfied and that CLUTCH DOES NOT HAVE POWER. Now the IF part of RULE 008 is satisfied, and system can report to the user: "The fuse is faulty."

the user can be used in a data-driven mode to determine a goal for the goal-driven phase. The PROSPECTOR geology expert system, described in Box 7-2, works in a goal-driven mode when it seems to be making progress, but returns to the user for help in goal selection when serious trouble is encountered.

The production system formalism is useful when the knowledge can be expressed as an independent set of *recognize-act* pairs, but may be inappropriate to represent other types of knowledge, such as set/element (taxonomic) relations among objects in the domain. The PROSPECTOR system addressed this problem by using a semantic network representation in a rule-based inference system. This representation retains the desirable modularity of a rule-based approach, while permitting an explicit, structured description of the semantics of the problem domain.

PRODUCTION SYSTEMS IN PSYCHOLOGICAL MODELING

Some investigators view human behavior in terms of an information processing system, consisting of a long-term memory (LTM), a short-term memory (STM), and an ability to carry out certain processes involving symbols. They hope to gain insight into the nature of human information processing by performing psychological experiments with human subjects involving symbol manipulation and memorization, and then developing a computer model that behaves the same way. Newell and Simon [Newell 72] state their interest in production systems (PS) as the computational tool for carrying out this investigation:

> We confess to a strong premonition that the actual organization of human programs closely resembles the production system organization. . . . We cannot yet prove the correctness of this judgement, and we suspect that the ultimate verification may depend on the PS proving relatively satisfactory in many different small ways, no one of them decisive.

The features of production systems that particularly interest these investigators can be summarized as follows:

1. A production system is a completely general programming methodology; in theory, it can be used to express any desired computation.
2. The rules of the PS provide a uniform encoding of the information that instructs the PS how to behave.
3. In a PS, each production is independent of the others.
4. The PS has a strong stimulus-response flavor.
5. The productions themselves seem to represent "meaningful components" of the problem solving process.
6. The dynamic working memory for a PS corresponds to human short-term memory, and the human long-term memory may correspond to the rule-base of the PS.

Newell's production-based approach [Newell 73] was used to test theories that attempt to explain the results of certain memory scanning tasks. The subject memorizes a set of digits and responds to a digit flashed on the screen by indicating whether or not it was in the original set. The response times of the subject are

BOX 7-2 Prospector, A Geology Expert

The PROSPECTOR system emulates the reasoning process of an experienced exploration geologist in assessing a given prospect site or region for its likelihood of containing an ore deposit of a certain type. The empirical knowledge contained in PROSPECTOR consists of a number of models that encode knowledge about certain classes of ore deposits. An ore deposit model is encoded as an inference network, a network of connections or relations between field evidence and important geological hypotheses. For example, PROSPECTOR includes a sulfide model, a carbonate lead/zinc model, a copper model, a nickel sulfide model, and a sandstone uranium model. These models are intended to represent the most authoritative and up-to-date information about each class of ore deposit.

Given a rule such as *Barite overlying sulfides suggests the possible presence of a massive sulfide deposit*, a semantic network is used for the antecedent, and a separate one for the consequent. The networks are represented using links and nodes; conceptually, the network for the antecedent of the barite rule would be:

> *There is some entity, E-3A, that participates in an overlying relationship, (PHY-REL-3A) with some other entity, E-3B. Furthermore, E-3A is composed of barite, and E-3B is composed of some material, V-3A, that is a member of sulphides.*

The semantic network representation has the advantage that all parts of a consequent and a related antecendent do not have to match. Rules can be linked implicitly through set/element chains. For example, suppose that a sample composed of galena is observed. Since galena is an element of the lead sulfides which in turn is a subset of the sulfide minerals, this observation is relevant to a rule concerning sulfides, and can automatically activate such a rule.

In the interactive consultation mode, the geologist typically has promising field data and wants assistance in its evaluation. He or she provides the program with a list of names of rocks and minerals observed, and enters other observations expressed in simple English sentences. The program matches these data against its models, requests additional information of potential value for arriving at more definite conclusions, and provides a summary of the findings. The user can ask at any time for an elaboration of the intent of a question, or for the geological rationale for including a rule in the model, or for an ongoing trace of the effects of answers on PROSPECTOR's conclusions. The performance of PROSPECTOR depends on the number of models it contains, the types of deposits modeled, and the completeness of each model. Each model is encoded as a separate data structure, independent of the PROSPECTOR system *per se*. Thus PROSPECTOR is a general mechanism for using such models to deliver expert information about ore deposits to a user who can supply it with data about a prospect or region.

To deal with the uncertainty of user observations, PROSPECTOR uses an inference mechanism based primarily on subjective probability theory. A probability is associated with every statement in the knowledge base, measuring the degree to which the statement is believed to be true. When engaged in consultation about a particular propect, PROSPECTOR uses the specific geological evidence furnished by the user to update the values of its stored probabilities.

In developing the ore deposit models, a significant result has been the evolution of a methodology to acquire and encode models. This methodology involves interviewing techniques, principles for determining the overall structure of a model, tools for interactive construction, modification, and testing of models, and methods for evaluating and revising a model.

noted. The production system was refined to incorporate new hypotheses about how the symbols were brought into the subject's memory, and eventually a successful simulation was built around a small number of productions.

Some of Piaget's results have also been modeled by psychologists using the production rule representation. For example, for tasks involving the ordering of members of a set of objects based on length, weight, or size, Young [Young 76, Boden 81] shows how the behavior of any given stage in a child's development can be described by a specific production rule system.

PRODUCTION RULE-TYPE EXPERT SYSTEMS

An expert system is a program that uses large amounts of knowledge about a single domain to achieve a high level of competence in that domain. While most expert systems do not use the pure PS form, the PS framework can be found in most current applications. Some characteristics of such systems are discussed below.

It is not trivial to build up a database of rules in practical application domains, since human experts often have trouble in converting informal knowledge into formal rules. In addition, it is difficult to capture the ability of an expert to deal with uncertainty. While various Bayesian and ad hoc approaches to uncertainty have been incorporated into expert systems, experts often cannot make useful estimates of the required a priori probabilities or belief values.

As is typical of a PS, the expert systems are often highly reactive; i.e., the choice of actions to be performed next by the system depends primarily on significant features of the current situation, rather than on the type of fixed control structure that characterizes conventional software systems. Another difference from a conventional system is that theoretically the rules are modular in nature; rules can nominally be added and deleted without affecting other rules. In practice, though, there are often side effects, some of which can be quite subtle.

The expert system must be able to communicate with the user in a mode that is natural for the particular application. The systems are often interactive, using a graphics display and communicaiton via an approximation to a natural language (e.g., English) extended to include the jargon of the application domain. Furthermore, many expert systems can retrace the reasoning sequence employed and explain what was done at each step and why, often based on keeping a time-history of the rule firings or the backward chaining.

In addition to the use of production rules to represent domain knowledge, frame-based representations have been incorporated into many recent systems to provide significant help with the rule-management task by providing a means of organizing and indexing modular collections of production rules according to their intended usage [Fikes 85].

Applying Expert Systems Expert systems can only be used in applications

where knowledge can be expressed by formal rules, such as medicine, engineering, and science, to aid users in design or diagnosis tasks. For example, the Dendral and Meta-Dendral systems [Buchanan 78] for analyzing mass spectrometer data were among the first successful applied systems; MYCIN is an expert system that acts as a consultant to the physician in the field of infectious disease, [Buchanan 84, Shortliffe 76]; R1 is an expert system for designing computer configurations, [McDermott 80, McDermott 81]; and PROSPECTOR is a consultant system for geological exploration, [Duda 79].

There are no expert systems to aid in the writing of poetry or novels, or in painting a picture, since these creative arts have not been expressed in formal rules (and may never be). Another requirement for an application is the existence of consensus among experts as to what is a proper procedure or what is valid knowledge. Lack of consensus often exists among experts in art, music, and literature. The problem of consensus even arises in technical fields when a lack of understanding exists, e.g., in medicine a disease that baffles the experts may result in differing opinions as to diagnosis and remedies. One may also have schools of thought in a particular field, such as exists in the various approaches to psychiatric problems and their treatment. Finally, for a field to be suitable for expert system application, there must be a certain degree of stability over time. The effort of constantly changing rules of an expert system and validating the results would make the system unattractive.

Two examples of expert systems for relatively stable domains are PROSPECTOR in the field of geology (see Box 7-2) and MYCIN in the field of medicine (see Box 7-3).

Plausible Reasoning in Expert Systems

In some applications, such as engineering design, the rules are usually stated as certainties, e.g., "If conditions A and B exist, THEN perform some action." In diagnosis systems, such as the MYCIN medical system and the PROSPECTOR geology expert, the rules have a probabilistic flavor, "If there is evidence A and evidence B, THEN hypothesis C is true with certainty of 0.7." There also may be a measure of uncertainty attached to the evidence itself, e.g., the user may feel that the probability of evidence A being true is 0.8. Terms such as "probability" and "certainty" are not probabilities based on frequency as in the case of coin tossing or card games, but rather are subjective scores given by an expert to indicate the relevance of evidence to a hypothesis, or the believability that a certain piece of evidence is true. Such estimates are used in everyday life when we talk about the probability of a team winning the game.

If we have a database with a large number of rules containing subjective probability estimates, the expert system must be able to derive new scores for hypotheses as new evidence is presented to the system. Each rule in the database has an associated rule strength that measures the degree to which a change in the probability of the evidence changes the probability of the hypothesis. This change can be positive or negative, since evidence can be either favorable or unfavorable for a hypothesis. These rule

BOX 7-3 MYCIN, A Medical Expert System

The MYCIN system diagnoses blood and meningitis infections, and recommends drug treatment. MYCIN conducts its consultation with the user in English, and constructs lines of reasoning leading to a diagnosis and treatment plan. MYCIN's was the first major rule-based system, [Buchanan 84]. A typical MYCIN production is:

> IF: (1) The site of the culture is blood, and (2) the gram stain of the organism is gramneg, and (3) the morphology of the organism is rod, and (4) the patient is a compromised host THEN: There is suggestive evidence (0.6) that the identity of the organism is *Pseudomonas aeruginosa*.

There are also productionlike rules concerning the use of other rules:

> IF: (1) The patient is a compromised host, and (2) there are satisfied rules which mention in their premise *Pseudomonas*, and (3) there are satisfied rules which mention in their premise *Klebsiella* THEN: There is suggestive evidence (0.4) that the actions associated with the former rules (2) should be executed before the latter (3).

Each rule supplied by an expert has associated with it a *degree of certainty* representing the expert's confidence in the validity of the rule (a number from 1 to 10). An *ad hoc* procedure is used to cumulate the degrees of certainty of the rules used in an inference chain.

An interesting feature of MYCIN is that it can expose its line of reasoning by offering explanations of various kinds. "Why" and "how" questions can be asked by the user during consultation. The user can ask the system, "Why do you want to know that?" and "How did you conclude that?" For example, if MYCIN asks the user "Is the patient's illness with ORGANISM-1 a hospital acquired infection?" and the user responds with WHY?, MYCIN answers:

> MYCIN: (1.0) This will aid in determining the category of ORGANISM-1 since it has already been established that (1.1) the gram stain of ORGANISM-1 is gramneg, and (1.2) the morphology of ORGANISM-1 is rod, and (1.3) the aerobicity of ORGANISM-1 is facultative. Therefore, if (1.4) the infection with ORGANISM-1 was not acquired while the patient was hospitalized then there is strongly suggestive evidence (0.8) that the category of ORGANISM-1 is *Pseudomonas aeruginosa*.

Similarly, if the user asks HOW (1.3), the system will indicate the chain of reasoning that was used in establishing "aerobicity." Because of MYCIN's many useful features, such as interaction in English, explanatory power in answering WHY and HOW questions, and the careful construction of an expert database, it has become the model for many subsequent expert systems.

strengths are usually obtained by interviewing an authority; strengths that are expressed initially in verbal terms such as "encouraging," "very unlikely," etc., must be translated to a numerical scale. Changes in probability are then computed using some *ad hoc* or formal scheme (see the section on probabilitistic reasoning in Chapter 4 for further discussion of this topic).

An example of a formal approach to probability updating is PROSPECTOR's use of Bayes's theorem for modifying the probability of a hypothesis based on evidence E. This approach assumes that before any information has been obtained from the user, every rule S has been assigned some prior probability P(S) by the designer. As evidence is acquired from a user, an updated probability of S is

computed. If E′ denotes all the evidence accumulated up to some point in the consultation, then the probability P(S|E′) denotes the current probability of S given the evidence E′. The updating relationships and an actual PROSPECTOR computation are presented in Appendix 7-1. The simplest form of updating involves a hypothesis affected by a single piece of certain evidence. The next more complicated situation involves a hypothesis affected by a single piece of uncertain evidence, e.g., the user says, "I am 70 percent certain that evidence E is true." The most complicated case deals with updating a hypothesis using multiple rules, each with uncertain evidence. There are many subtleties that arise in the actual analysis, concerning uncertainty of evidence, independence of evidence, and the prevention of inconsistencies.

Basic AI Issues

At present, expert systems do not acquire their expertise through experience, but rather, they are given the needed information and the organization of this information by a *knowledge engineer*. An expert system can be considered as an *idiot savant* that can deal very effectively with a specialized field, but is incompetent to deal with topics not in this field. It is instructive to examine the reasons for the limitations of existing PS-based expert systems:

- **Lack of learning capability.** The designer of the system, and not the system, learns by experience as the system is used, and modifies the rule database accordingly. Because it is a nontrivial task to determine which rules need modification when the system is not performing up to expert standards, the designer must consult with the human experts to determine how the rules have to be modified or augmented. The system itself has no way of determining why the end user may not be satisfied, and no way of automatically correcting the source of the difficulty.
- **Lack of ability to generalize.** There is no reason to expect that the addition of incremental amounts of knowledge will lead to global understanding. To be able to *understand*, the system must be able to generate *higher level* concepts by comparing and generalizing problem situations or groups of knowledge elements. No such ability has been provided in existing systems. As we have indicated previously in Chapter 5, this ability to make comparisons and to determine similarity is a crucial part of generalization and learning.
- **Explanation.** The explanation approach used in most expert systems is in terms of rules that have been satisfied or will be satisfied if certain information is provided. However, the user often desires a causal explanation based on physical reasoning. This type of explanation usually requires that there be a model of the process being discussed. Several recent medical expert systems use such models for this purpose.
- **Need for representing control knowledge.** Although a pure production system is conceptually attractive, the problem of control quickly arises in any practical system. Control must be exerted when more than one rule is activated by the working storage. The

BOX 7-4 Shallow and Deep Reasoning in Expert Systems

Most current expert systems can be said to use "shallow" reasoning, since there is no mechanism in the system for "understanding" the domain of expertise. An expert system based on deep reasoning uses a model of the domain to motivate the reasoning processes. For example, a shallow electronic troubleshooting system would contain rules supplied by an expert relating failure symptoms to possible circuit problems. A "deep" system would use models of electronic components and their role in circuits to reason about the symptoms and how they imply failures of components. The difference between these approaches is well stated by De Kleer [De Kleer 84a] who has developed a program that can reason about electronic circuits, determining the effects caused by components of the circuit, and their purpose (teleology) in the circuit:

> If I were interested in building a performance program, the temptation for including. . . . extra knowledge would be overwhelming. However, that would be shortsighted. To understand what causal reasoning, or teleological reasoning is, one must study it in isolation uncorrupted by other forms of reasoning. Otherwise one has merged two types of reasoning without ever identifying either one individually. In addition, little scientific progress is made and we are not much closer to the ultimate goal. . . . To achieve robust performance, the underlying theories must be identified. This methodology stands in sharp contradistinction with the popular expert-systems methodology. Expert systems are aimed at producing what performance is possible in the short term without consideration of the longer term. Typically this is achieved by recording as many of the heuristics and rules of thumb that experts actually use in practice, as possible. This is misguided. The reasoning of experts is based on underlying theories that must be teased out. The expert systems approach can be caricatured as a stimulus-response model—good for some purposes, but ineffective in the long run.

De Kleer is correct in his assessment of what is required to attain a sophisticated expert system, but he dismisses too casually the importance of rule-based systems that use an extensive knowledge base. He is advocating a complete return to the philosophy of general reasoning in place of an extensive knowledge base. See *Artificial Intelligence*, December 1984, for an entire volume devoted to this point of view. The ideal approach is probably a system capable of both deep and shallow reasoning, together with the ability to perceive and learn. However, we should remember that it took nature 5 billion years to design such systems!

designer must provide some method for determining which rule is to change the state of the world first, and whether the other activated rules are to be allowed to fire. In addition, since the matching process is time-consuming, procedures must be preprogrammed to determine which rules should be examined, and this often depends on what phase of the problem the system is working on. This latter difficulty is related to the frame selection problem, i.e., whenever a situation is encountered where a particular set of rules is most applicable, then attempts at matching should be restricted to the set of rules in a particular frame.

- **Reasoning.** Qualitative or commonsense reasoning is not feasible since most current systems are based on

independent "chunks" of knowledge (rules), rather that on an integrated model. Thus, the system has no "overview" of its supposed field of expertise. Box 7-4 discusses some attempts to deal with this problem.

- **Fragility.** Most currrent systems are fragile. The term "fragility" is used to denote a system that suddenly loses competency when it strays somewhat from its intended domain. (This is in contrast to the performance of human experts which tends to degrade smoothly in a similar situation.) Since expert systems are designed to be narrowly focused, there may be no resolution to this problem.

DISCUSSION

In this chapter we showed how a rather simple concept, the production system, has been used in psychological modeling and in expert systems. The PS offers an interesting approach to control, one that allows the data to direct the processing. Another important feature is the modularity of the rule base, so that rules can be added and deleted without the need to modify the control structure.

Expert systems based on the PS concept have become one of the most successful and active areas of applied AI. Although often not "pure" PS, they usually retain the important features of data driven control, modularity of rules, and explanation ability. In spite of their advantages, the corresponding disadvantages of PS, described in the previous section, are severe enough to question their long-term potential. Without the ability to perceive, learn, and reason, these systems would seem to have a limited role to play in future intelligent systems.

Appendix

7-1

PROSPECTOR Procedure for Hypothesis Updating

This appendix illustrates how PROSPECTOR propagates the effects of new evidence through its inference networks (chains of evidence and hypotheses). Interactions between separate chains are not treated here.

The "odds-likelihood" form of Bayesian updating was used in PROSPECTOR because it was felt that geology experts could make their estimates best in that form. The procedure involves three quantities:

1. The prior odds for the hypothesis, O(H), where O(H)= P(H)/(1-P(H)). Note that this odds relationship corresponds to the layman's use of odds, e.g., when P=0.8, the odds are $0.8/(1-0.8)=4$, or four to one.
2. The posterior odds for the hypothesis, given that evidence E is observed to be present, $O(H|E)=p(H|E)/(1-p(H|E))$.
3. The likelihood ratio, $LS=P(E|H)/P(E|\tilde{H})$, a ratio of the probability of the appearance of evidence given that a hypothesis is true, to the probability of evidence if the hypothesis is not true.

The updating relationship involving these quantities is given by

APPENDIX 7-1

a form of Bayes's theorem:

$$O(H|E) = LS * O(H)$$

If LS is large, it means that the observation of E is encouraging for H. When LS is infinity, E establishes H.

A complementary set of equations describes the case in which E is known to be absent,i.e., when $\sim E$ is true:

$$O(H|\sim E) = LN*O(H)\text{ , where}$$
$$LN = P(\sim E|H)/P(\sim E|\sim H)$$

The quantity LN is called the necessity measure. If LN is much less than unity, the known absence of E transforms neutral prior odds on H into very small posterior odds in favor of H. If LN is large, then the absence of E is encouraging for H.

An inference rule, If E THEN (to degree LS, LN) H, states "The observed evidence E suggests (to some degree) the hypothesis H." To apply the rule, the expert must not only describe E and H, but must also supply numerical values for LS, LN, and O(H).

The updating formulas cannot be applied directly when the user is unable to state that the evidence E is either definitely present or definitely absent, but they can be extended to accommodate uncertainty in E, see [Duda 76] for details. In the example below [Duda 79], we will use a nonlinear function of the change in probability of E as a weighting factor to obtain the new odds.

The approach is to linearly interpolate between two known points on the plot of P(H|E) vs. P(E). One known point is P(H|E) when E is certain, and for the other we use the priors for both P(H) and P(E). The corresponding values in the odds updating formula at these two points is that (1) when E is known to be true, the new odds equals (LS)*(old odds), and (2) the case where we have no new evidence, for which the new odds equals the old odds (the case of the priors).

PROSPECTOR Probability Update Computations. The relationships used in the updating computation given below are:

- Probability = odds/(1+odds)
- Odds = probability/(1−probability)
- New odds = old odds * likelihood ratio

The rules that will be used are given below. (Abbreviations will be used, e.g., SMIR for "suggested morphology of igneous rocks," to shorten the exposition; LS = likelihood ratio.)

- IF INTRUSIVE BRECCIAS THEN SUGGESTIVE MORPHOLOGY OF IGNEOUS ROCKS (SMIR) WITH LS = 20.
- IF SMIR THEN HYPABYSSAL REGIONAL ENVIRONMENT (HYPE) WITH LS = 300.
- IF HYPE THEN FAVORABLE LEVEL OF EROSION (FLE) WITH LS = 200.

The system begins with the following *a priori* probabilities: SMIR = 0.03, HYPE = 0.01, and FLE = 0.005. If the user indicates certainty for INTRUSIVE BRECCIAS, the following probability updating takes place:

1. SMIR odds = 0.03/(1−0.03)
   = 0.031
   Revised SMIR
   odds = 20*(0.031)
   = 0.62
   (The LS for the INTRUSIVE BRECCIAS multiplies the SMIR odds.)

   Revised SMIR
   prob. = 0.62/(1+0.62)
   = 0.38
2. HYPE odds = 0.01/(1−0.01)
   = 0.0101
   For a weighting factor of 0.36, the revised HYPE
   odds = 0.0101*300*0.36
   = 1.09

   (The HYPE odds have been increased by the LS of 300, weighted by a function of the degree to which SMIR has increased from its prior probability.)

   Revised HYPE
   prob. = 1.09/(1+1.09)
   = 0.52
3. FLE odds = 0.005/(1−0.005)
   = 0.005
   For a weighting factor of 0.515, the revised FLE
   odds = 0.005*200*0.515
   = 0.52

   Revised FLE
   prob. = 0.52/(1+0.52)
   = 0.34

Thus, the user indicating certainty for INTRUSIVE BRECCAS has increased the probability of SMIR from 0.03 to 0.38; of HYPE from 0.01 to 0.52; and of FLE from 0.005 to 0.34. The propagation of probability updates continues in this manner throughout the network.