SOME ISSUES OF CONTROL IN EXPERT SYSTEMS*

Jeffrey A. Barnett†

USC/Information Sciences Institute
Marina del Rey, CA 90291

Abstract

Because of the nature of the problem-solving domains in which expert systems operate, there are usually choices about what to do next. Many different rules are potentially useful to extend the current solution state. The control problem is to select the rule or rules that most likely will lead to acceptable behavior. For many years, workers in Artificial Intelligence have tried to duck the control problem by compiling both control and application knowledge into the same large-grained rules. However, the desire to build systems that work in larger domains, fail softly at the boundaries of their competence, and are able to explain their behavior puts the importance of the control problem into sharp focus.

INTRODUCTION

Much of the early work in artificial intelligence concerned problems of heuristic search (often called *weak methods* [10]). These methods are a close approximation to pure control with no domain-specific knowledge. Experience showed that they fail to provide enough problem-solving power. Therefore, the Al community shifted its focus of attention to the problems of capturing and representing knowledge and has developed powerful techniques in these areas. Eventually, a strong feeling developed that with enough knowledge available, no substantive control problem exists [5].

*This article originally appeared in IEEE 1982 Proceedings of the International Conference on Cybernetics and Society, 1–5.

†Current email address: jbb@notatt.com

This proposition is examined below. First the type of knowledge encapsulated in expert systems is characterized. Then, reasons for needing powerful control mechanisms are explored, with the conclusion that the control problem for expert systems is indeed worthy of attention. Finally, we speculate on future developments relevant to the control problem.

Before proceeding, it is fair of you to ask what an expert system is. The question is more difficult to answer than may first be apparent. In fact, attempts to generate an answer have caused a change in terminology from "knowledge-based systems to "expert systems" because the term "knowledge" is so hard to define. (One interesting try is contained in [2], a taxonomy of knowledge and cognitive skills; a recent attempt that is more pertinent to Al is found in [11].) Unfortunately, the terminological change has served only to hide rather than eliminate the question. The definition of expert systems ultimately rests on a definition of knowledge, and I do not have a workable one in hand.

However, allowing for this glaring omission, an expert system is a problem-solving system that employs expert-level knowledge to solve problems in a single domain. The system must be able to operate at or near an expert level of performance to qualify as an expert system. (Note, many people cannot be trained to perform as experts in a given application, so this can be a very stiff requirement.) Further, many expert systems are interactive and provide facilities to explain their behavior and conclusions, though these features are not strictly necessary qualifications.

Now let us define the control problem. During the operation of an expert system, alternative execution sequences are available from many points in the problem-solving activity because of the diversity of knowledge in these systems. A control decision is the selection and ordering of the sequence or sequences to execute. The decision is hard to make because only imperfect models of the consequences of the decision are normally available—some but not all choices lead to problem solution. The control problem is to make control decisions that minimize the expected cost and maximize the expected benefit of using an expert system to solve problems.

The control problem is very much like a classic problem in operations research, and often its solution is amenable to the techniques developed there. However, this is not always the case. Not only do we not have good models of the consequences of control decisions, but the various costs and benefits associated with expert system behavior may be measured in units that are not combinable. For example, lower search cost versus behavior that is easier

to explain are measured respectively in units of CPU cycles and psychological satisfaction. How are these two measures combined so that the control mechanism can select for the optimal expected system behavior? This instance of the control problem seems simple, yet there is no appealing method known to handle even this case.

THE EXPERT'S KNOWLEDGE

It is useful to examine the characteristics of knowledge in expert systems before proceeding with a discussion of the control problem.

The knowledge in expert systems is expert-level knowledge that represents the"rules-of-thumb" used by human experts and includes descriptions of when that knowledge is applicable. These rules are not principles of general reasoning, such as modus ponens, and the long inference chains that develop from the use of general rules rarely occur in the operations of expert systems. Rather, the expert's knowledge is highly compiled, and inference chains are compressed into single rules; this compression gives both the expert's and the expert system's behavior the appearance of orderliness and goal-directedness. Because of this, expert systems perform well compared to the general reasoning systems found in the Al labs.

The system would appear chaotic and be so inefficient as to be unusable if the expert's rules had to be rederived from first principles of the domain (and logic) for each application. In a sense, using expert rules saves the system the chore of learning—the deductions do not need to be repeated. However, these systems perform in more limited domains and are harder to extend; for example, can any of the existing expert systems for medical diagnosis and treatment recommendation be extended, using their current knowledge base or a slight augmentation, to do preventative health care in the same area of medicine? I think not.

Compilation and compression are not unmitigated blessings because the expert's rule is usually derived from experience rather than being model-based. Thus, although these rules usually produce correct behavior, they also have the potential to produce incorrect or inconsistent results. The rules are plausible and work a high percentage of the time; this is why the expert uses them. However, when they fail, the human expert knows enough to recognize this fact and find out why. He retreats to a better-grounded model (one based upon more general principles) and determines where the compiled inference chain failed and why it is not applicable in this particular

case. Thus, another way of viewing an expert is that he can work with large knowledge chunks when it suits his purposes and revert to smaller chunks, general principles, and causal relationships when it is necessary.

Now it should be obvious that the larger the chunks of knowledge used, the less the control problem. There are two reasons for this: (1) the large knowledge chunk has compiled into it the control that would have been necessary to take the several little steps that are equivalent to it, and (2) reasoning with larger steps produces shorter inference chains and, therefore, the cost of backup from error is less, as is the cost of method selection.

Expert systems tend to perform like human experts when they use large, expert-sized knowledge chunks. The control problem has been buried. Expert systems fail abruptly when the set of expert rules fails to produce reasonable behavior because there is no way to return to more basic mechanisms.

At this point, an example is useful to clarify the concepts. For instance, an expert rule for fire control applications is

> Where there is smoke there is fire.

This rule is a summary statement that is usually true. It can be derived, in the applicable cases, from three more basic rules.

> Smoke accompanies slow oxidation of carbonaceous material.
>
> Rapid oxidation that produces light and heat is fire.
>
> If there is oxidation along with a plentiful supply of unconsumed fuel and oxygen, the rate of oxidation will generally increase.

As we know, the expert rule is correct much of the time. But not always. For example, if something is smoldering in a closed confined area, there may not be enough oxygen. On the other hand, it is evident that the rule is good enough to cause us to call the fire department if we see smoke in a dwelling.

The leverage gained by using expert rules has been so great that most expert system-building tools provide only the most rudimentary support for addressing the control problem. Two examples are OPS [6] and EMYCIN [18]; both are domain-independent systems that help the knowledge engineer construct expert systems, and both provide production rules as the knowledge representation vehicle. In OPS, there is virtually no support for specifying control regimes. The user must create "program counters" in a global data base and make the precondition portions of rules trigger on these program counters in addition to ordinary domain conditions. Thus, OPS is very much

like a machine language for constructing control plans. It is very difficult to do even primitive things though it is always possible.

EMYCIN, on the other hand, is over constrained; depth-first back-chaining is the only control mechanism available. The system builder gets three choices: live with it, trick it, or use some other system-building tool. The control dilemma facing users of production systems is reviewed in [8].

A better approach to the control problem is needed to build more flexible expert systems, i.e., those that can combine the advantages of large knowledge chunks having short reasoning chains with the ability to fall back on general principles when and if necessary.

EXPLANATIONS AND CONTROL

Providing explanations has an impact on the control problem: The possibility of having to explain one's behavior has a profound influence on that behavior, and decisions about behavior in an expert system are control decisions.

Many expert systems operate as expert consultants within a narrow speciality. MYCIN [15] is a typical example. MYCIN is (poses as) a clinical specialist in the diagnosis and treatment of blood-borne infections. The user is an MD called the attending physician. He seeks advice from the specialist about his patient, and interacts with the specialist by answering questions about and receiving recommendations for his charge. However, the attending physician retains ultimate authority over what tests and treatments are initiated.

In actual scenarios with either MYCIN or a human specialist, the attending physician is not merely a passive participant in the dialogue. He is likely to ask questions that help him better understand the issues and risks since he must make the final decisions. Typical questions are: what is your best guess to-date, on what do you base that conclusion, and why did you ask *me* that question?

Explanation facilities provide the ability for an expert system to answer such questions. The inclusion of an explanation facility is considered vital to the success and acceptance of an expert system, particularly when it is interactive and functions as a consultant. The user is not likely to consume the result proffered by the system it it engages in behavior that cannot be explained to his satisfaction. Would you hire a consultant who could not explain himself well enough to instill confidence in you?

The control mechanism has two reasonable choices and one unreasonable

choice. The reasonable choices are (1) restrict execution sequences to those that are self-explanatory or (2) select execution sequences for which post hoc explanations can usually be generated. The unreasonable, ivory-tower choice is to find the "best" solution and then give the *user* two choices: (1) accept the system's output without explanation or (2) eat cake.

A comparison of two expert systems that both do medical diagnosis may demonstrate this point. The first is MYCIN, mentioned above. It provides a reasonable but primitive explanation facility and has paid attention, in at least a rudimentary way, to the details of interaction. Though the system is not used in a clinical environment today, it has sparked considerable interest in both the computer science and medical communities. Practitioners in both fields feel that MYCIN is a prototype of the kind of expert systems that will be used in the future. However, MYCIN is a one-level-of-abstraction system and offers explanations only in terms of behavior at that level. Therefore, there is little chance for fundamental improvements unless there is a basic reorganization.

INTERNIST [14] is the other medical diagnosis system used for this comparison. Its knowledge representation resembles conditional probabilities. It is likely that INTERNIST is more accurate than MYCIN at current levels of development, and at any rate it functions in a much larger domain—internal medicine. However, few believe that INTERNIST represents a major step in the future of expert systems (though it might represent a step toward the compilation of medical knowledge). If the attending physician wants an explanation of INTERNIST's diagnosis, he can receive only a listing of the conditional probabilities involved. Since these are numbers not preceded by a dollar sign, the doctor cannot comprehend them.

There are many ways to resolve this issue. One approach is to educate the users to better understand the internal model used by the system so that they can more fully appreciate and apply the results. Unfortunately, computer science assumes the role of the tail wagging the dog; experience has shown we can have a greater impact when our systems fit into existing practices and procedures rather than forcing others to accept our ways.

A second approach is to use the superior system (say INTERNIST) to generate the problem solution then force the other, more graceful system (say MYCIN) to back into the established answer. Explanations are provided in terms of the behavior of the second system. This raises the control problem of how to force the second system to duplicate the agreed-upon solution. However, this approach does not appear totally intractable and it deserves

some investigation.

Such simple-minded strategies as mentioned above are not likely to resolve the explanation problem. Rather, the architecture of an expert system must consider providing explanation facilities as a primary goal. A discussion of the many issues involved can be found in [17]. In any event, the presence of an explanation facility in an expert system puts demands on the control mechanism.

EXPERT SYSTEMS HAVE A CONTROL PROBLEM

At current levels of performance, expert systems have virtually finessed the control problem. However, the story is just beginning to unfold, and problems are on the horizon. As mentioned previously, (1) models of future behavior based on control decisions are imperfect, (2) all rules except logic and perhaps laws of nature are imperfect, and (3) system behavior is evaluated from several viewpoints which are not necessarily combinable. Therefore, there is a control problem—it just might not be apparent in some current systems. Summarizing the present state of affairs is a good way to start speculation about the future.

Call the large-grained knowledge chunks typically found in expert systems *expert rules* and the smaller chunks of more primitive, causal knowledge *causal rules*. (Perhaps causal is a bad choice of terminology, but little else suggests itself. The intent is to capture rules of logic, physical principles, cause and effect relations, etc., in one phrase.) Table 1 compares some of the characteristics and usage features of expert and causal rules. These are extreme cases, and there is middle ground where the descriptors assume values between those shown. Expert rules appear to be better in every regard except one; they are not as accurate and, therefore, may produce anomalous behavior. More about this below. First, consider the time complexity of using expert rules versus causal rules.

If R is the number of rules that can respond to a typical situation and D rule applications are necessary to solve a problem, then the number of possible solution paths to explore is about R^D. This is an approximation because R and D are only average or expected values. If R_E and D_E are the measures for expert rules and R_C and D_C are the measures for causal rules, Table 1 says that $R_E < R_C$ and $D_E < D_C$. Therefore, the expected search cost using expert rules is very much lower than the expected cost using causal rules.

Table 1: Knowledge characterization

EXPERT RULES	CAUSAL RULES
Large chunks combining the operation of many causal rules	Small chunks
Highly plausible (but not perfect)	Very accurate
Applicability is highly selective	Each rule applies very often
Solves problems in a few steps	Solves problems in many steps

The variable, R, is called the branching factor; it is a measure of the number of alternative decisions available to the control mechanism at points along potential solution paths. Unless $R = 1$ or every alternative leads to acceptable system behavior, there is, by definition, a control problem. Note, a rule application is useless even if it produces a valid result unless that result contributes to problem solution or some other valued part of the system's behavior. The larger the fraction of bad alternatives and the deeper the reasoning chain, the more critical the control problem.

Expert systems fail sharply near the boundary of their knowledge. This is not surprising. These systems usually contain only large-grained expert rules and a primitive control mechanism. Extending such a system to work better at the limit of its knowledge so that it degrades gracefully or pushing the boundary much farther out is a difficult task. However, it is straightforward to see what improvements are necessary.

The first change is that expert systems need to contain more than expert rules—smaller-grained knowledge is necessary too. Second, knowledge that criticizes the operation of the expert rules is needed so that the other rules are used only when necessary. Third, these systems must provide adequate facilities to incorporate the more powerful control knowledge whose need is implied by the inclusion and use of small-grained knowledge chunks with their inherently longer reasoning chains.

Another issue was discussed in the previous section: explanation facilities. If the expert systems of the future extend the limits of their operations and

blend the use of expert rules with causal reasoning, the problem solutions developed by the systems may be beyond the state-of-the-art or at any rate may contain an element of surprise. Explanations become necessary not only to convince the users but also to educate them. Requiring an explanation facility implies a control problem.

In summary, an expert system must have the ability to engage in novel reasoning, using causal rules, in addition to applying the expert's "rules-of-thumb", and it must be able to decide when to make this move. Therefore, to build better expert systems, the control problem must be addressed.

THE FUTURE OF THE CONTROL PROBLEM

As mentioned previously, neither exact models of the effects of control decisions nor appropriate utility functions to evaluate system behavior are generally available. Thus, control decisions must be made in the face of uncertainty using whatever knowledge is at hand. But these are precisely the characteristics of a problem-solving task. Therefore, the control of expert systems is itself a problem domain. Further, it is a domain that I believe can and should be handled by an expert system that reasons about and makes control decisions [1]. Next, let us explore this proposition and see what mechanisms it implies.

A control decision is the selection and ordering for execution from among the set of alternative sequences available. The decision can be tactical (for example, execute next the rule most likely to achieve the system's goal) or the decision can be strategic (for example, a choice of breadth-first over depth-first search because of the nature of the solution space). On what are such control decisions as these based and how are they made?

The control problem is like any other decision problem; knowledge about the domain is applied to descriptions of the current state and the alternatives are ranked. The current control state of an expert system includes both its static and its dynamic state. The static state consists of such facts as the knowledge available in the system, descriptions of the behavior of this knowledge, opportunities and limitations imposed by the system's architecture, and the user's utility function that relates different aspects of system behavior. The content of the dynamic state may vary from system to system. However, typical components are measures of problem-solving progress, relations among activities, representation of problem statements, plans, and partial results with quality estimates.

The knowledge that makes control decisions is *control knowledge*. The control knowledge can be domain-specific, domain-independent, or some combination of these two. Consider an expert system that diagnoses patients complaining of head pains. Then, these three levels of specificity are illustrated by the following control rules.

Domain rule:	Rule out BRAIN_TUMOR before turning full attention to the hypothesis HEADACHE.
In-between:	Attend to diseases with LIFE_THREATENING potential before working on other hypotheses.
General rule:	All else being equal, prefer to work on the hypothesis representing the greater risk.

The in-between case here is represented by a control rule from the domain of medicine, which subsumes the domain of head pains. (Uses and benefits of rules like these, applied to investment counseling, are discussed in [4].)

Many types of general control knowledge exist: Other examples include the proof of formal properties of A^* [13], the analyses of the behavior of alpha-beta and other similar algorithms [9], the criterion for optimal satisticing search [16], and the Gaschnig speed-up method [7] for depth-first searching with a pair-wise constraint. The control knowledge in an expert system must know not only about these methods, but when to apply them.

If the control problem is treated as a problem-solving domain with adequate supporting mechanisms, control decisions can be made dynamically. For instance, the Gaschnig optimization is useful only if the density of solutions in the solution space is small; otherwise, it can degrade overall performance. Delaying the decision to use this optimization may result in a savings because the density can vary among search problems in the same class. This is in sharp contrast to systems like MYCIN where the control strategy is already wired in (in this case by EMYCIN, the system-building tool).

Expert systems of the future will have two layers in order to reason about control. The first layer contains mechanisms to represent application-level knowledge and the problem-solving state at this level; this is the same as in today's expert systems. The second layer is essentially new; it contains mechanisms to represent *control* knowledge and the problem-solving state for *control* decisions. The control knowledge reasons about and from the problem-solving state of both layers to decide which knowledge to apply.

Many issues are raised by the proposed architecture.

The design of a representation for control. Plans, goals, and measures of progress must be part of this representation. Further, the representation must show the relation between control and items in the application-level state.

The identification and unification of utility functions. The value of results of system activities are measured in a multi-dimensional space, for example, contribution to goal satisfaction, risk, cost, and explainability. The measures associated with alternative activities must be combined to make control decisions.

The accumulation of general control knowledge. Libraries of search strategies and the differential criterion among them must be captured and implemented, as must be other kinds of general control knowledge.

The combination of general and application-specific control knowledge. The knowledge engineer must be able to express the control knowledge idiosyncratic to his application and make it fit smoothly with the general control knowledge provided by the second layer.

The description of knowledge. Behavior of the knowledge at the application level must be described adequately so that control knowledge can reason about it without having to resort to an analysis of coding details.

The meta problem. In the same way the system must reason about control for the application level, the system must reason about the use (control) of the control knowledge itself.

Some, but not all of these problems are receiving serious attention. A group at USC/ISI is working on the representation problem [1]. At Stanford, work is in progress on the accumulation problem [12] as it is at USC/ISI. A description and goal representation mechanism, for use in a Hearsay-II environment, is evolving at Amherst [3]. However, more research and engineering are needed to make progress on the problems confronting those working on the control of expert systems.

The prognosis for the future is more attention to the control of expert systems, the control problem treated as a problem-solving task, and expert-

system-building tools that provide adequate mechanisms to reason about control.

SUMMARY

The control problem for expert systems is to make control decisions, i.e., the selection of what to execute, in a way that optimizes the expected value of system behavior. Expert systems have a control problem because (1) choices exist and (2) the consequences of choices cannot be predicted with certainty.

Further, the realization of some of our desires for better expert systems places demands on the control mechanism. One example is the desire for good explanations. As noted above, the possibility of having to explain one's behavior has a profound influence on that behavior, and decisions about behavior in an expert system are control decisions.

Another desire affecting control is to have expert systems that work in larger domains and fail softly at the boundary of their competence. As we have seen, expert systems get most of their power by using large-grained expert rules, but there is a need to incorporate smaller-grained causal rules so that difficult and unusual cases can be handled. However, small-grained rules necessarily produce longer reasoning chains, and the computational complexity grows rapidly. The issues of how to use small-grained rules only when necessary and how to manage the complexity when they are used are control problems.

Therefore, there is a control problem for expert systems even if the current batch has masked the fact. Making control decisions is a knowledg-based activity operating in the face of uncertainty, and this activity must serve many masters, for example, find a workable solution, do not take unnecessary risks, and be able to explain your behavior.

Thus, the control problem has all the aspects of an interesting problem-solving domain. Further, there is a growing supply of both general and application-specific control knowledge. Therefore, it is proposed that expert systems that work in the control domain be built and these systems be made part of the architecture of ordinary expert systems to improve problem-solving power and produce better behavior.

ACKNOWLEDGMENTS

I want to thank Lee Erman, Michael Fehling, and Bill Swartout for their helpful comments on this paper.

This research was supported by the Defense Advanced Research Projects Agency under contract No. MDA903–81–C–0335. Views and conclusions contained in this document are those of the author and should not be interpreted as representing the official opinion or policy of DARPA, the U.S. Government, or any other person or agency connected with them.

REFERENCES

1. Barnett, J. A., et al., Working papers for the Control of Expert Systems Project, 1981/82.

2. Bloom, B. S., *Taxonomy of Educational Objectives*, McKay, New York, 1956.

3. Corkill, D. D., and V. R. Lesser, *A Goal-Directed Hearsay-II Architecture: Unifying Data-Directed and Goal-Directed Control*, University of Massachusetts, Department of Computer and Information Sciences, COINS Technical Report 81–15, 1981.

4. Davis, R.,"Meta-Rules: Reasoning About Control," *Artificial Intelligence* 15, (3), December 1980, 179–222.

5. Feigenbaum, E. A., "The Art of Artificial Intelligence: Themes and Case Studies of Knowledge Engineering," in *Proc. 5th Int. Joint Conf. on Artificial Intelligence*, pp. 1014–1029, Cambridge, MA, 1977.

6. Forgy, C., and J. McDermott, "A Domain-independent Production System," in *Proc. 5th Int. Joint Cont. on Artificial Intelligence*, pp. 933–939, Cambridge, MA, 1977.

7. Gaschnig, J., "A General Backtrack Algorithm that Eliminates Most Redundant Tests," in *Proc. 5th Int. Joint Conf. on Artificial Intelligence*, pp. 457, Cambridge, MA, 1977.

8. Georgeff, M. P., "Procedural Control in Production Systems," *Artificial Intelligence* 18, 1982, 175–201.

9. Pearl, J., Selected reprints and reports on search, heuristics and complexity, 1982. Abstracts to the Members of the Mathematical and Experimental

Analysis of Search Algorithms List maintained by the Cognitive Systems Laboratory, School of Engineering, UCLA.

10. Newell, A., "Heuristic Programming: Ill-Structured Problems," in J. Aronofsky (ed.), *Progress in Operations Research* 3, pp. 360–414, Wiley, New York, 1969.

11. Newell, A., "The Knowledge Level," *Artificial Intelligence* 18, (1), January 1982, 87–127.

12. Nii, H. P., and N. Aiello, "AGE (Attempt to Generalize): A Knowledge-Based Program for Building Knowledge-Based Programs," in *Proc. 6th Int. Joint Conf. on Artificial Intelligence*, pp. 645–655, Tokyo, 1979.

13. Nilsson, N., *Problem-Solving Methods in Artificial Intelligence*, McGraw-Hill, New York, 1971.

14. Pople, H. E., "The Formation of Composite Hypotheses in Diagnostic Problem Solving: An Exercise in Synthetic Reasoning," in *Proc. 5th Int. Joint Cont. on Artificial Intelligence*, pp. 1030–1037, Cambridge, MA, 1977.

15. Shortliffe, E., *Computer-Based Medical Consultation: MYCIN*, Elsevier, New York, 1976.

16. Simon, H., and J. B. Kadane, "Optimal Problem-Solving Search: All or None Solutions," *Artificial Intelligence* 6, 1975, 235–247.

17. Swartout, W. R., "Explaining and Justifying Expert Consulting Programs," in *Proc. 7th Int. Joint Conf. on Artificial Intelligence*, pp. 815–822, Vancouver, B.C., 1981.

18. van Melle, W., "A Domain-independent Production-Rule System for Consultation Programs," in *Proc. 6th Int. Joint Conf. on Artificial Intelligence*, pp. 923–925, Tokyo, 1979.