EXPERTISE ACQUISITION AND THEORY FORMATION

Saul Amarel[1]

Department of Computer Science, Rutgers University
New Brunswick, NJ 08903

1. Expertise Acquisition and Problem Reformulations

Expertise in a given domain is commonly characterized by skillful, high performance, problem solving activity in the domain. Expert behavior requires large amounts of knowledge of various types, and highly effective ways of using this knowledge in approaching the solution of specific classes of problems in the domain.

Expert problem solving requires the conceptualization/ formulation of a given problem within the framework of one or more problem solving schemas in such a way that solutions can be attained with accuracy and with very little search. This means that available knowledge must be used in a very effective manner to instantiate parameters of the schemas in accordance with the needs of the task at hand. In other words, an expert problem solver works within a highly appropriate problem formulation - where appropriateness refers mainly to computational efficiency and it is task specific.

The problem of how to choose an appropriate problem formulation, and how to change it to fit the characteristics of a task, is at the heart of the problem of representation in problem solving. This is a problem of fundamental importance in AI[13,4,9];it has also proven to be a very difficult problem on which progress has been relatively slow in the last decade or so.

Experience with the development of expert systems is showing us that the processes of system specification and evolution are extremely demanding in time and computational resources, and they need to be better understood. The bulk of effort in expert system

[1]This research was supported in part by the Division of Research Resources, National Institute of Health, Public Health Service, Department of Health, Education, and Welfare, under Grant 5 P41 RR00643 to the Special Research Resource: Computers in Biomedicine at Rutgers University, New Brunswick, New Jersey 08903

construction is directed to the task of problem formulation in a given domain. It involves the specification of relevant bodies of knowledge in forms that are appropriate for use in the solution of problems in a given class in accordance with a given method of reasoning (problem solving schema). Clearly, progress in basic studies of problem formulation is directly relevant to the task of expert system specification.

At present, expert systems acquire their expertise by 'being told' how to behave as experts. Knowledge Engineering is providing the tools for fashioning expert systems in various domains. This involves mainly the acquiring, organizing and representing of knowledge in forms that produce expert behavior. Also, since system building is evolutionary, these tools make it possible for a designer to easily change parts of the expert system in the course of its development. Improvements in performance are obtained mainly via changes in the bodies of knowledge that the system uses. The following types of concerns are encountered here: under what conditions is incremental growth of a body of knowledge sufficient for performance enhancement; at what point is knowledge to be restructured and used in a different way during reasoning. These are important questions in processes of expertise acquisition; and they are closely related to the problem of finding 'appropriate' shifts in problem formulation. At present, problems of expertise acquisition in expert systems must be solved by the system designers. As the complexity of expert systems grows, it is becoming increasingly desirable to use AI methods for mechanizing (parts of) the expertise acquisition process. Recent psychological work on acquisition of problem solving skills in humans is providing valuable insights in this area[8]. Clearly, this work is also relevant to the basic problem of problem representation in problem solving.

There is considerable evidence that human experts have a set of powerful methods for handling in a highly efficient way problems in certain restricted classes of a domain (these are the expert's areas of special competence or expertise), and ways for recognizing that a given problem belongs to one of these classes and can be treated therefore by the appropriate special method. In addition, the human expert possesses a sufficient amount of basic knowledge in the domain to handle - by using weaker methods, and perhaps at a reduced level of efficiency - problems that fall outside the special classes. Thus, human expertise is characterized not only by high performance in certain limited problem areas, but also by robustness in handling a spectrum of problems surrounding the areas of special competence. Furthermore, a human expert is able to acquire problem solving skills and to improve his expertise by learning from his problem solving experience. This includes the ability to create specialized methods for certain restricted problem classes. Thus, human expert behavior is characterized by high performance, as well as by robustness and by skill acquisition/developmental capabilities.

The emphasis of work on machine expert systems to date has been on building high-performance agents in specific classes of problems. Commonly, relevant knowledge is 'hand-crafted' by designers in forms that are directly usable by the system. Characteristics of robustness and breadth are still difficult to mechanize. Work on processes of expertise acquisition and on mechanisms of performance improvement by learning is still in early stages of exploration. While much remains to be done in the area of 'conventional' expert systems - i.e., on the development of methodologies and tools for building high-performance systems in relatively narrow domains - it is also important to start directing more attention to the problem of building 'intelligent experts', i.e. expert systems that have characteristics of robustness and self-improvement of the type that human experts have - in addition to their ability to perform very well in a narrow domain. This has theoretical as well as pragmatic significance. The theoretical significance is clear. The pragmatic significance may be less clear; but it is based on the following considerations. For an expert system to be <u>useful</u> in a real life environment it needs <u>breadth</u>; it will be accepted by potential users if it can be relied upon to provide high performance in specific parts of a domain <u>and</u> also if it can provide (at least) some preliminary analysis and advice in parts of the domain that are outside the main areas of the system's expertise. Also, a realistic expert system is likely to be very complex. To construct such a system within acceptable periods of time (years, not decades) it would be essential to provide it with (at least) some of the capabilities of expertise acquisition and self-improvement that are characteristic of human experts.

From the previous discussion it can be seen that there are several points of contact between fundamental issues of problem representation on the one hand and the problem of building 'intelligent experts' on the other. I believe that exploring problems of problem representation in the context of expertise acquisition in problem solving will contribute to both of these AI areas.

In previous work, I analyzed specific problem solving tasks in several domains in an attempt to clarify the nature of problem representations and of shifts in problem representations[6,5,3]. The focus of most of this work has been on relationships between choices of alternative problem formulations and problem solving efficiency.

More recently, I have been exploring representational shifts in Tower of Hanoi problems within a conceptual framework which I find useful for handling problem formulations and their changes. The grammatical specification of solutions for a problem class plays an important role in this framework. Work in this area - with emphasis on the conceptual framework and its use in a detailed description of developmental processes of expertise acquisition in the Tower of Hanoi domain - is presented in[2]. Studies of expertise acquisition in this domain suggest that an 'intelligent expert' system should include the following features:

a) Several bodies of knowledge in both declarative and procedural forms; the contents would range from domain-independent items, to domain-specific and problem-class-specific items.

b) The ability to work with several problem solving schemas simultaneously; and to maintain (and coordinate the use of) several instantiations of these schemas for different formulations of a problem.

c) Theory formation capabilities for discovering 'interesting' regularities in the body of problem solving experience; and program synthesis capabilities for exploiting these regularities in the development of specialized high-performance procedures.

2. Theory Formation

In my analysis of expertise acquisition in the Tower of Hanoi domain[2], I find that a wide range of theory formation capabilities are needed in order to extract knowledge from problem solving experience so that it can be used to obtain problem formulations of increased power. It is characteristic of theory formation processes in this area that they involve the finding of a restricted class of problems in a domain for which certain properties/regularities can be obtained that lead to a good, specialized, solution method. These are difficult processes as they are aimed at the formation/discovery of two mutually interdependent concepts: the concept of an 'interesting subclass of problems', which depends on the existence of a 'good specialized, solution method' for it, and the concept of a 'good solution method' which depends on the ability to focus on an 'interesting subclass' and to experiment with it.

In view of the centrality of theory formation problems in problems of problem reformulation and expertise acquisition, it is important that more effort be directed to the study of these problems and of methods for solving them. Recently, I have renewed my work in this area (this work goes back to the early sixties[7]). I have been exploring processes of theory formation in the context of a program synthesis task. More specifically, I have been considering situations where functional properties of a computer program are specified in the form of explicit input-output associations, and the problem is to synthesize a program, in a given programming language, that satisfies the given associations. The program that constitutes a solution to the problem can be viewed as a theory that summarizes/explains the input-output data.

Theory formation problems in a given domain face representation problems themselves as questions of improving their performance arise. To improve theory formation processes, shifts in representation may be required; and to effect these shifts, theory formation processes may be required in turn, etc. I believe that it is important to recognize and to clarify these relationships.

In general, theory formation involves both the <u>choice of a domain of phenomena</u> that are to be expressed within some conceptual framework, and the <u>finding of an expression within the framework</u> for explaining/defining the domain. The interplay between domain choice and the formulation of a theory for the domain is an interesting and complex process which is difficult to capture at present. It is clearly related to the problem of identifying 'interesting' problem class restrictions in expertise acquisition processes. In recent work I have been exploring this process in the context of the program formation task. My approach is to assume that the initial statement of the theory formation problem includes a fundamental <u>supposition</u> about the domain of the theory. The supposition can be regarded as an exploratory one - to be confirmed or denied. The effort to find a coherent theory for the given domain can be seen as an attempt to validate the supposition. Only in the course of this attempt, it may be possible to find out whether changes should be made in the supposition, and in what way. Preliminary work in this area is discussed in[1].

As in other areas of AI, progress in theory formation requires the detailed exploration of representations and of methods for solving such problems, as well as of processes of improvement via reformulation. The program synthesis problem has provided a good vehicle for such explorations

The problem that we face here is how to search in the given language of programs for a program that satisfies the constraints imposed by the given set of data associations. We can regard the constraints as presented in a <u>data space</u>, and the possible solutions presented in a <u>program space</u>; and furthermore, the initial body of available knowledge is not in a form that permits useful links to be established between the two spaces. Thus, we are facing a characteristic problem of formation[4]. In problems of this type, the solution process cannot proceed by reasoning from the problem constraints to specific parts of possible solutions. The bulk of search is being done in program space, where candidate solutions are generated, and the data is mainly used to test the solution candidates. To increase the power of the process, it is essential to increase the a priori control that problem constraints, i.e., the data associations, can have on the generation of candidate programs. This requires the development of strong links between data space and program space, and a strategy for good coordination between the search/reasoning activities in the two spaces. The significance of the two-space model for studying problems of theory formation is discussed in detail by Simon and Lea[14].

In my work I have considered several formulations of the program formation problem. The movement to formulations of higher effectiveness involves structuring the program space in a way that facilitates links with data space, and increasing the amount of reasoning that takes place in data space. The role of models (algebraic, geometric) is extremely important in the stronger problem formulations. I have explored three main approaches to the problem.

The first approach which I developed about 20 years ago[7] is based on heuristic hill climbing in program space. The second approach, which I developed in the early seventies[3], is more goal-oriented, and it involves searching the program space under the guidance of an algebraic model. Work on the third approach is more recent, and it is based on detailed reasoning in data space. Two methods have been developed for using the results of the detailed analysis of individual data associations in the construction of a program/solution for the entire domain. The first is a combination method, where it is crucial to have effective techniques of program modification in order to proceed to a successful assembly of a solution. The second is a method of elimination which can be seen as a variant of the "version space" approach to learning[12]. By introducing a geometric representation of data space, it is possible to move to a very strong formulation of the formation problem where a small set of macro-moves and of well chosen intermediate concepts provide the basis for highly effective ways of constructing solutions. The present state of research in this area is described in[1].

Work on theory formation problems has brought out forcefully the fundamental importance of the degree of dependence between problem conditions on the choice of problem solving method. The problems of combining several 'partially correct' solutions into a desired solution, and of modifying 'almost correct' solutions into a correct one, are closely related to the problem of constructing plans for satisfying several interacting goals. This is a basic problem in AI where much more research is needed.

Also, it is becoming evident that the notion of problem formulation in theory formation is somehow fluid and open-ended. The interplay between domain choice and the formation of a theory for the domain is an interesting and difficult problem that requires more work. It is closely related to concept discovery problems of the type studied by Lenat[10,11].

Another area of significance for theory formation tasks is the discovery and use of appropriate models in support of the search/reasoning processes involved. While we appreciate the significance of these processes at present, we are still far from being able to mechanize them.

REFERENCES

1. Amarel, S. (1983), 'Program Synthesis as a Theory Formation Task - Problem Representations and Solution Methods', Technical Report, CBM-TR-135, LCSR, Rutgers University, June 1983; to appear in 'Machine Learning: An Artificial Intelligence Approach, Vol. II, Michalski, Carbonell, Mitchell (eds), Morgan Kaufmann, Los Altos, CA.

2. Amarel, S. (1981) 'Problems of Representation in Heuristic Problem Solving; Related Issues in the Development of Expert Systems', in Methods of Heuristics, Groner, Groner and Bischof (eds), Lawrence Erlbaum, 1983; also available as Technical-Report CBM-TR-118, LCSR, Rutgers University, 1981.

3. Amarel, S. (1971) 'Representations and Modeling in Problems of Program Formation', in Machine Intelligence 6, Meltzer and Michie (eds), U. of Edinburgh Press, 1971.

4. Amarel, S. (1970), 'On the Representation of Problems, and Goal-Directed Procedures for Computers', in 'Theoretical Approaches to Non-Numerical Problem Solving', Banerji and Mesarovic (eds), Springer-Verlag, Heidelberg, 1970.

5. Amarel, S. (1968) 'On Representation of Problem of Reasoning About Actions', in Machine Intelligence 3, Michie (ed), U. of Edinburgh Press, 1968.

6. Amarel, S. (1967) 'An Approach to Heuristic Problem Solving and Theorem Proving in the Propositional Calculus', in Systems and Computer Science, Hart and Takasu (eds), U. of Toronto Press, 1967.

7. Amarel, S. (1962) 'On the Automatic Formation of a Computer Program Which Represents A Theory'. In Yovits, Jacobi and Goldstein (editors), *Self-Organizing Systems-1962.* Spartan Books, 1962.

8. Anzai, Y. and Simon, H. A. (1979) 'The Theory of Learning by Doing', Psychological Review, 86, 1979.

9. Korf, R. E. (1980), 'Toward a Model of Representation Changes', AI Journal, Vol. 14, No. 1, Aug. 1980.

10. Lenat, D. (1983a) 'EURISCO: A Program that Learns New Heuristics and Domain Concepts. The Nature of Heuristics III: Program Design and Results'. *AI Journal* 21(2), March, 1983.

11. Lenat, D. (1983b) 'Theory Formation by Heuristic Search. The Nature of Heuristics II: Background and Examples'. *AI Journal* 21(1), March, 1983.

12. Mitchell, T.M. (1978) *Version Spaces: An Approach to Concept Learning.* PhD thesis, Stanford University, December, 1978. Also, Stanford CS Report STAN-CS-78-711, HPP-799-2.

13. Newell, A. (1969), 'Heuristic Programming: ILL Structured Problems', in Progress in

Operations Research, Aronofsky (ed), Vol. 3, Wiley, New York, 1969

14. Simon, H.A., Lea, G. (1974) 'Problem Solving and Rule Induction: A Unified View'. In Gregg (editor), *Knowledge and Cognition.* Lawrence Erlbaum, 1974.