WHAT KIND OF COMPUTATION IS INTELLIGENCE?
A FRAMEWORK FOR INTEGRATING DIFFERENT KINDS OF EXPERTISE

B. Chandrasekaran

Laboratory for AI Research
Department of Computer & Information Science
The Ohio State University
Columbus, OH 43210

Abstract

The view that the deliberative aspect of intelligent behavior is distinct type of algorithm, in particular a goal-seeking exploratory process using qualitative representations of knowledge and inference, is elaborated. There are other kinds of algorithms that also embody expertise in domains. We discuss the different types of expertise and how they can and should be integrated to give full account of expert behavior.

Extended Summary

1. Intelligence as Computation

The idea that intelligence as a process is algorithmic in character is at the foundation of Artificial Intelligence (AI) as both a science and a technology. With the exception of a new movement called *connectionism*, almost everyone in AI subscribes to this view, especially as intelligence relates to cognitive activities such a problem solving. I have elsewhere[1] discussed connectionism and AI, and noted that connectionist-style systems may have some advantages to offer in modeling perceptual phenomena such as speech or visual recognition, and some cognitive phenomena such as memory retrieval. However, for complex problem solving activities that take place in the *deliberative* mode, the view of the process of intelligence as largely algorithmic, i.e., manipulation of discrete symbol structures, is the most common paradigm. This is mainly because of the fact that "thought", at least as people are aware of it, has a large propositional and hence symbolic content. Discrete symbol structures are generally used to represent the underlying structure of propositions, and hence thoughts. In what follows, I will restrict my remarks to AI and problem solving, in particular expert systems, and I am going to take the algorithmic view for granted for the purposes of this distinction.

Now of course the modern engineer is very familiar with a number of powerful algorithms for various problems. She has cut her teeth on computers, programming

and optimization theories, and has been exposed to linear programming algorithms, finite element analysis, various kinds of numerical simulation programs, etc., etc., each giving precise and useful answers to a number of decision-making problems. Thus a natural question for such a person is whether intelligence is characterized by a special kind of algorithm, different in its power and limitations from the more traditional algorithms she is familiar with. She has probably heard that AI programs are "heuristic" in nature. Additionally, the engineer will surely want to know the extent to which the current AI technology is actually useful in creating intelligent programs. Or is it simply a new programming technology? Even as a programming technology what does it offer in terms of power?

In this talk I plan to offer a view of what characterizes intelligence as a problem solving process, what AI, especially the set of ideas that have come to be called, variously, knowledge-based systems or expert systems, is about and how to understand and characterize the relationship of the algorithmic processes of intelligence to the more traditional types of algorithms that engineers are familiar with.

2. Different points of view about AI

AI is not a unified field of scientific activity; there are significant differences within the field about the nature of intelligence, how to approach its study, and about the goals of AI as a science and technology.

1. For some people AI means understanding the principles behind intelligence as we know it in humans and seeing how that can be computationally supported and exploited for technology. For example, theories that the appropriate architecture for modeling human intelligence is a rule-architecture, that human concepts are stored as a linked network of frames, that problem solving comes in a variety of generic strategies, or that much human problem solving is based on storage, retrieval and modification of large number of cases are theories which attempt to explain some aspects of human intelligence. These theories have become the basis of AI programming or AI-type approaches to solving some problems.
2. There is what one might call a pragmatic view that AI means any programming technique or algorithm that helps us solve complex problems by any combination of techniques, but most commonly by use of so-called "heuristic" search techniques. If the heuristic technique is based on some theory in 1 above, then this view is really close to the view of AI in 1. However, often there is no particular concern that the heuristics reproduce human-like performance, but rather that they reduce the complexity of computation in any way. In fact, human beings aren't particularly good problem solvers in a number of problems such as the

traveling salesman problem for which heuristic algorithms have been invented.

3. Yet others go along with the just mentioned view of AI simply complex algorithms, but restrict the types of problems to be considered as those that humans do particularly well, e.g., problems such as diagnosis, planning, design, and theory formation, which are normally considered tasks that humans excel in. However, the connection to humans is often only at the level of choice of the problem. Once a problem such as diagnosis is chosen, that problem is often treated as an abstract problem for which "normative" algorithms based on some notion of rigor are to be found. There is no particular commitment to use methods of or theories about human-like intelligence. This approach results in a theory of the task-- diagnosis, planning, induction, or whatever -- rather than a theory of intelligence as a process. In this view, people often solve the problems more or less approximately, but the normative algorithm is offered as the ideal algorithm for the problem. A good analogy here is multiplication of two numbers. True some people are good at solving some versions of the multiplication problem in their heads, but studying multiplication as an abstract problem and discovering good algorithms for it has turned out to be very effective. An attempt to do for diagnosis or planning what arithmetic has done for multiplication is often the aim in this approach. If this can be done effectively, and systems can be built based on such algorithms, that would be very welcome from a technological perspective, since one would have algorithms for problems of importance and which until now only humans did well.

   Typically, however, the algorithms obtained by an abstract study of the task of this type tend to be computationally intractable in the general case (unlike the multiplication example). Examples of this are several approaches to diagnosis where the problem is elegantly formulated in some formalism such as logic or statistics, but the algorithms proposed for diagnosis based on the formalism are in general very complex, and heuristic approximations need to be made in particular problems. But these approximations need not, and in general do not, match the power or behavior of human intelligence in these problems, since the original formulation was not based on a theory of intelligence.

I will take the position that intelligence is not an arbitrary collection of complex algorithms as in (2) above, nor is a theory of tasks *per se* a theory of intelligence as in (3). Just because people perform multiplications in their head doesn't mean the theory of multiplication is an AI theory. Just changing multiplication to diagnosis doesn't change the logical character of the situation. That is, while it is certainly technologically useful to obtain good algorithms for diagnosis, independent of a theory of intelligence, we have to be careful about characterizing it as AI. Whenever we have a problem for which we can generate a tractable algorithm about whose behavior we can have some confidence, we should of course use it for our applications. However, where AI is

important is in problems where the normative or formal algorithms have one or both of the following properties: they are intractable, i.e, they are of high computational complexity, or they require information in a form that is not generally available in the domain. In these cases, if there is evidence that human experts solve it, then we can look for AI-type approaches instead of traditional algorithms. (If human experts solve this problem, it is not because they can magically overcome the theoretical limitations of computational complexity. The power of human intelligence in these problems arises from domain-specific knowledge and powerful information processing strategies that characterize intelligence. Together they enable the human to solve particular and important *subsets* of the original problem.) Of course for a number of problems of importance that humans solve, we neither have any kind of algorithms, nor do we have any real idea of how humans solve them. After all, AI as a science is only in its infancy.

3. Intelligence as exploratory process

I would like to propose a view of intelligence as a particular kind of computational process. I hope my description will help in seeing how these processes can be integrated with other kinds of algorithms that also embody human expertise.

Consider an engineer working on a design problem. Let us say she has some paper and pencil and maybe a computer terminal in front of her. Part of the design activity takes place in her head, part of it is recorded in the paper, and part of the needed information processing takes place in the computer by means of execution of some algorithms either she wrote or she invoked. Typically she uses her thought or mental problem solving processes to decompose the problem, think of possible design approaches, previous successful designs which have some similarity to the current problem, visualize the design, do some spatial reasoning or qualitative simulation of the artifact under design, etc., etc. Partial designs are probably set down on the paper, which thus provides an enlarged short term memory capacity. Note that the algorithm executed by the computer is such that it would not be particularly appropriate or possible or efficient for her to do it in her head. She might herself have designed the algorithm, but there is a qualitative difference between the operation of that algorithm and her processes of intelligence.

One way in which to clarify some of the issues is to make a distinction between computations which are *being intelligent* versus those which use the result of earlier intelligent behavior. One might look at an algorithm for the greatest common divisor, and exclaim, "What a clever algorithm!". In reality, the creator of the algorithm was the one who was being clever during its construction, but the algorithm itself, during its running, is not engaging any of the processes that intelligence is composed of, such as exploration of possibilities, hypothesis-making, etc., using general methods for such behavior.

We can focus the discussion by considering what it means to be intelligent in problem solving. Mycin, R1, finite element methods, linear programming algorithms, multiplication algorithms, etc., are all computational methods which provide solutions to some problems. Let us now consider a subclass of methods which are "intelligent" in the following sense: they explore a *problem space*, implicitly defined by a *problem representation*, using general search strategies which exploit typically qualitative heuristic knowledge about the problem domain. A working hypothesis in AI is that humans, unassisted by other computational techniques or paper and pencil, engage in this kind of problem solving. The subarea of AI concerned with problem solving takes as its subject matter the phenomena that surround this kind of knowledge-based and general strategy-directed exploration of problem spaces. The power of these phenomena come from the effective way in which they explore very large problem spaces to make plausible and interesting hypotheses, which can then be verified by other means if necessary. Also, if information that can only be obtained by other kinds of computations are necessary during this kind of exploratory problem solving activity, then these other methods can be invoked, much as an engineer flits between, on one hand, "thinking" about a problem and making intermediate hypotheses, and, on the other hand, writing down some equations to solve before further he or she engages in further exploration.

For lack of a better term, let us call computational methods that are characterized by such knowledge-based and strategy-directed exploration of qualitative problem spaces *problem space exploratory* techniques. Other kinds of computational techniques, let us call *solution algorithms*. These terms are unsatisfactory, but with proper qualifications they will do.

Note that our distinction is some what different from the fairly classical "heuristic" vs "algorithmic" distinction. For example, the algorithm for the Traveling Salesman problem is complex, so a number of programs which approximate the solution by making various assumptions and approximations have come to be called heuristic solutions to the problem. But, these are still solution algorithms according to our definition, albeit without the properties of provable correctness of the solutions given by them, since these algorithms do not, at run time, engage in exploration of the underlying problem space by use of explicit knowledge and general exploration strategies.

There are many problems for which solution algorithms which are not computationally complex are known. Computer science in general and a number other disciplines take as their subject matter the production of solution algorithms for a number of problems of a general or domain-specific nature. Sorting, multiplication, well-defined optimization problems for which linear programming is applicable are of these types. When problems of this type are identified in any domain, there is no reason to engage in problem-space exploratory techniques. Adopting AI-type solutions to these problems

will merely produce solutions which are qualitative and approximate, and, in comparison with the corresponding solution algorithms, the AI methods are likely to be computationally expensive as well. If during diagnostic reasoning one needs to know the exact value of pressure in the reactor chamber, if one has an equation that can be evaluated for it, and if one has all the information needed to evaluate the equation, then it is silly to use problem-space exploratory methods. On the other hand, for a number of problems such as diagnosis and design in the general case, the underlying spaces can be very large, and solution algorithms of limited complexity are not available, except perhaps for some subcases. This is when consideration of AI methods is appropriate.

It is important to emphasize that expert knowledge consists of both kinds of computations. Thus expert systems should use both kinds of techniques, deploying each kind for subproblems where their power is needed. But, since these solution algorithms are domain-specific, or use methods, such as linear programming, that are not the subject matter of AI per se, it is most useful to confine the discussion in AI to problem space exploratory techniques, especially those inspired by human intelligence. For example, I have described elsewhere a study of human-like problem space exploration for the task of design[2].

This view of intelligence as problem space exploration is close to Newell's *problem space hypothesis*[3]. However, in my view an additional characterization of intelligence emerges from a consideration of the nature of exploratory control strategies. In our research we have identified a number of general strategies that we call *generic tasks* to set up and explore problem spaces. I have described them in a number of papers[4,5]. These strategies have the property that they bring an element of computational tractability by using knowledge expressed and organized in specific forms.

4. Concluding Remarks

So far my goal has been to help engineers interested in AI for problem solving and the construction of expert systems to understand what makes AI as a distinct type algorithm by pointing out that a part of human problem solving expertise comes in the form the abilities to explore and search in problem spaces. In this view, the relationship between the thoughts of a problem solver is that they stand for descriptions of the states of a problem space as the problem solver is exploring alternatives in pursuit of a goal. However within AI there is another paradigm about the relationship between thoughts. In this view thoughts are connected by their logical relations and thus "reasoning" is the basic metaphor of artificial intelligence as it applies to deliberative behavior. Different kinds of logics are proposed to capture this relationship and the term "inference" is used to describe the process.

We have used the term "deliberative" several times to make sure that we have been referring to "thinking" as the basic activity of intelligence. However, even within symbolic or algorithmic AI, there have been researchers who emphasize the importance of non-deliberative aspects of intelligence, in particular about phenomena of memory. Minsky and Schank have been noted for theories about organization of knowledge and events in memory structures. The phenomena of memory organization in this research paradigm do not have much to do with reasoning in the sense of logic, nor with problem solving in the sense of search through alternatives that are generated at run-time and examined. Instead they emphasize memory organization and indexing for recognition and retrieval. Sometimes a solution to a problem can be obtained by couching it as a problem for which recognition or retrieval can produce a solution, such as in case-based reasoning.

Thus given a problem to be solved, there are a number of ways in which it may be solvable:

- A "closed form" algorithm may exist for that class of problems, e.g., algorithms for multiplication, sorting, or finite element analysis. If the domain is such that numerical quantities are central to it, then this algorithm will be quantitative in character.
- The solution to the problem may be obtained by generating alternatives in a problem space. This is the *problem solving* view and knowledge, often in a qualitative form, is used to select alternatives that are likely to lie in the path towards a solution.
- The solution may be obtained by logical reasoning from assumptions about the domain, ie., the problem is thought of as a reasoning problem. Some form of theorem proving may be used, and knowledge about the domain is stated in the form of statements in some logical language, most commonly in predicate calculus.
- The problem may be solved by retrieving solutions from memory or transformations of solutions of analogical problems stored in memory. This version of the solution may be implemented in some theories in symbolic forms and in others in a connectionist framework.

Given these possibilities, expertise may come in a form appropriate for any one of the above approaches.

In what follows, I will discuss applications of AI to building problem solving systems, and see what implication the above analysis has to how expertise can be integrated. For simplicity, I will restrict myself to only one type of AI process, namely deliberative goal-seeking, problem space exploration. My comments on integration can be extended to

other types of AI processes as well.

5. Acknowledgments

I acknowledge the support of Defense Advanced Research Projects Agency, contract RADC F30602-85-C-0010, and of Air Force Office of Scientific Research, grant 87-0090, in developing this view of intelligence as a computational process.

6. References

1. B. Chandrasekaran, "What kind of information processing is intelligence? A perspective on AI paradigms and a proposal," to appear in *Foundations of Artificial Intelligence: A Source Book*, D. Partridge and Y. Wilks, editors, Cambridge University Press, 1988.

2. B. Chandrasekaran, "Design: An information processing-level analysis," to appear as Chapter 2 of *Design Problem Solving: Knowledge Structures and Control Strategies*, D. C. Brown and B. Chandrasekaran, forthcoming.

3. A. Newell, "Reasoning, problem solving and decision processes: The problem space as a fundamental category," in R. Nickerson, ed., *Attention & Performance VIII*, Erlbaum, Hillsdale, NJ, 1980.

4. B. Chandrasekaran, "Generic tasks in knowledge-based reasoning: high-level building blocks for expert system design," *IEEE Expert*, Fall 1986, pp. 23- 30.

5. B. Chandrasekaran, "Towards a functional architecture for intelligence based on generic information processing strategies," *Proc. International Joint Conference on AI*, Milan, Italy, August 1987, pp. 1183-1192.