#557

Search and Reasoning in Problem Solving*

Herbert A. Simon
Department of Computer Science, Carnegie-Mellon University, Pittsburgh. PA 15213, U.S.A.

0. Introduction

In the course of the extensive research that has been done on problem solving in A1, several rather distinct ways have emerged for representing and thinking about problem solving tasks. One way to view problem solving is as search. We postulate some kind of space in which treasures are hidden. We build symbol structures (nodes) that model this space, and 'move' operators that alter these symbol structures, taking us from one node to another. In this metaphor, solving a problem consists in searching the model of the space (selectively), moving from one node to another along links that connect them until a treasure is encountered.

A second way of viewing problem solving is as reasoning. We postulate a system of logic that allows us to deduce new statements from axioms and previously deduced statements. We represent a problem by a set of axioms in the formal language of our logic. In this metaphor, solving the problem consists in accumulating more and more information (more and more statements) by inference until the answer to the problem has been found.

A third way of viewing problem solving is as constraint satisfaction. We postulate a set of objects and various subsets defined by the constraints they satisfy. In this metaphor, solving a problem consists in narrowing down the original set to a subset or unique object that satisfies all of the constraints.

In no sense are the metaphors mutually exclusive; metaphors seldom are. The same problem-solving algorithm can be viewed, now as search, now as reasoning, now as constraint satisfaction. Consider, for example, a simple theorem-proving program that works forward from a set of axioms, applying its rules of inference to these to obtain new expressions that can be added to the

*This research was supported by Research Grants MH-07722 from the National Institute of Mental Health, and N0014-79-C-0661 from the Office of Naval Research.

axiom set. When it finishes tracing a path to a desired theorem, it has succeeded. Clearly it is a search algorithm.

At the same time, the theorem prover is adding, at each step of its search, new propositions that follow logically from its axioms. It is gradually accumulating a larger and larger collection of deduced propositions. Clearly it is reasoning.

Moreover, the theorem prover does not search in the space of all possible well-formed expressions. Each of the new expressions it creates is guaranteed, by the manner of its creation, to be deducible from the axioms. Expressions that do not satisfy this deducibility constraint are simply never generated. Clearly the theorem prover is also a constraint-satisfying system.

Although all three metaphors may be applied to the same algorithm, they do not say the same things about it or call attention to the same aspects of the problem-solving process. The search and constraint metaphors focus upon the process of finding the problem solution, while the reasoning metaphor focuses upon the logical validity of the linkage between initial problem state and solution. Search is centrally concerned with discovery, reasoning with proof.

The search algorithm is continually analyzing the situation from new viewpoints, migrating from one state to another. Assertions that are true in one situation, or state, in the search space may not be true in other situations. Consider, for example, a search algorithm for choosing chess moves. At any given moment the algorithm focuses upon a particular chess position in which certain specified relations hold among the pieces. When the algorithm considers a new move, the situation changes. Relations that held in the original position may no longer hold, and new ones may have been established. We cannot rely on the proposition, from previous knowledge, that the Queen is still pinned, or that the Rook is not attacked. Information is situation-specific and must be continually updated.

The reasoning algorithm is viewed as accumulating information. When the working-forward theorem prover takes a step, deducing a new theorem, all the theorems that were proved previously remain valid. There is no large question of revising or updating information. (We shall see presently, however, that not all reasoning systems are truth-maintaining or cumulative.)

The constraint-satisfying algorithm is viewed as taking giant steps. It does not create new objects, painstakingly, one at a time. Instead, it starts with the entire space of objects and eliminates whole classes of them at a step by applying successive constraints, until it has narrowed the set down to objects that satisfy all of the constraints.

It is the purpose of this paper to throw some light on the relations between the search and reasoning approaches to problem solving, and in particular to explore some of the issues that have been raised about the adequacy and efficiency of each of these approaches for solving problems in particular kinds of problem domains. It will not aim at comprehensiveness, but will focus on a small number of questions that seem fundamental. In particular, it will omit any discussion of the constraint-satisfaction approach.

The discussion will be organized in five main sections, the first devoted mainly to reasoning systems, the second to search systems, the third to comparing and contrasting the two types of systems when they are employed for problem representations that have real-world denotations, the fourth to the generation of representations for real-world problems, and the fifth to the continuing readjustment of representations to reality.

1. Problem Solving as Reasoning

The reasoning metaphor—the idea that problem solving consists in applying reason to problem situations—has appealed to many researchers in artificial intelligence.[1] Some but not all of them have interpreted 'reasoning' to mean the use of a formal system of logic, the standard predicate calculus[2] or some other, to represent and operate upon the problem situation.

1.1. Reasoning as logic

Landmark publications developing the idea that reasoning is to be equated with the employment of formal logic include McCarthy's "Programs with common sense" [4] and the McCarthy and Hayes paper, "Some philosophical problems from the standpoint of artificial intelligence" [6]. These papers are especially concerned with augmenting standard logic with supplementary inference processes to handle the modal relations of possibility and causality that arise in reasoning about actions.

The domain of resolution theorem proving has always equated problem solving with deductive logic—or to put the matter otherwise, has conceived of formal theorem proving systems as the fundamental engines for problem solving. Research in this domain has generally stayed within the boundaries of the standard predicate calculus, except that it was soon recognized that, in order to reduce search to manageable proportions, there had to be some replacement of axiom sets by additional inference rules to allow reasoning to move foreward in macro-steps.

Finally, in recent years, there has been considerable interest in systems for drawing inferences from information stored in large data bases, often in the form of semantic nets. The problems encountered in this domain have led to a burgeoning of non-standard, and usually non-monotonic, logics to deal with the need for terminating inference processes before all paths are exhausted, the need for providing default values to supply missing explicit information, and

[1]Note, for example, the emphasis upon predicate calculus representations in a standard textbook like Nilsson [7].

[2]I will use the phrase 'standard predicate calculus' as a shorthand for 'first-order or higher-order predicate calculus'.

the need for resolving contradictions arising from the application of non-standard inference procedures [1].

What is shared in common among all of these approaches is the idea that reasoning should somehow be equated with the use of deductive logic. The near-synonymity of 'reasonable' and 'logical' in ordinary language lends seductive plausibility to this view. It is sometimes even demanded, especially in theorem proving, that the systems employed pass tests of completeness. Although there seems to be a general preference for the standard predicate calculus over other formalisms, task demands have gradually forced departures from it. I have mentioned three kinds of deviations—or augmentations—above: addition of modal operators, augmentation of inference rules to simplify the axiom systems, and departures that admit non-monotonicity.

1.2. Reasoning in mathematics

Now the reasoning we ordinarily encounter in mathematics, and especially in applied mathematics, does not restrict itself to the explicit axioms and inference rules of the predicate calculus, but uses much more powerful inference procedures. A linear equation, for example, can be rearranged to express any one of its variables as a linear function of the others. Each step of the rearrangement does not have to be justified by referring it back to the axioms of arithmetic and algebra—although it presumably *could* be so justified. We simply define convenient sets of operators ('macros') to make such rearrangements and use them as needed when we solve equations.

It is, of course, possible to introduce such operators formally through a higher-level axiomatization, or to embed them in a reasoning system by a device like Weyhrauch's [13] 'semantic attachment', which allows one to construct programs of arbitrary complexity in order to model syntactic structures of the formal system semantically. But in approaches of this hybrid kind, the semantic components have the properties of search systems rather than reasoning systems, and when they are relied upon to do all, or most, of the work of the logical system, one wonders what is gained by the formal syntactic embedding. The same comment applies to Nilsson's [7] proposal for introducing 'procedural attachments' into reasoning systems. Procedural attachments are operators of arbitrary complexity embedded in formal systems.

It will be useful to look at a simple example of problem solving (see [9] for additional discussion) that employs mathematical reasoning without making explicit the formal logic on which it rests. This particular example embodies both the metaphors of search and of reasoning, but especially the latter. Viewed as search, it employs a process of working forward, from initially known premises to conclusion. Viewed as reasoning, it employs a process of accumulating knowledge, until the values of all desired quantities have become known. A salient characteristic of systems that solve problems by accumulating knowledge is that once a statement is known to be true, it remains true. At no later step in the process can it lose its validity, once that has been established.

Consider now the problem:

> A partially filled beaker contains one liter of a 90% alcohol-water mixture. How much water must be added to dilute the mixture to 80% alcohol?

What is known initially about the problem is that certain equations hold (e.g., the initial amount of alcohol equals 0.9 times the total initial quantity of liquid), and that certain quantities are given (e.g., the initial quantity of liquid is one liter). Other quantities are initially unknown (e.g., the amount of water to be added).

But the information given explicitly in the problem statement is by no means adequate for solving it. In addition to the equations that can be extracted from the statement, a number of others must be supplied by the problem solver. He is assumed to 'know' (irrespective of whether it is true empirically) that quantities of liquids are additive; that, for example, the final volume of the liquid is one liter plus the volume of the water added.

In this extremely simple problem situation (which is not simple, by the way, for most students of algebra), at least ten equations can be written down directly: (1) $A_1 + \mathrm{d}A = A_2$, (2) $W_1 + \mathrm{d}W = W_2$, (3) $T_1 + \mathrm{d}T = T_2$, (4) $A_1 + W_1 = T_1$, (5) $\mathrm{d}A + \mathrm{d}W = \mathrm{d}T$, (6) $A_2 + W_2 = T_2$, where A, W, and T stand for alcohol, water, and liquid, respectively, the subscripts, 1 and 2, for quantities before and after water is added, and d for the amounts added. In addition, we have (7) $A_1 = 0.9T_1$, and (8) $A_2 = 0.8T_2$. Finally, since no alcohol is added. (9) $\mathrm{d}A = 0$, and (10) we are told that $T_1 = 1$.

A simple production system could solve each of these equations for the remaining unknown as soon as values were found for all but one of its variables. This system would rapidly arrive at the desired value for $\mathrm{d}W$. For example, since T_1 is known from (10), it could solve (7) for A_1, and substitute that value and $\mathrm{d}A = 0$ from (9) in (1), then solve the latter for A_2. That value substituted in (8) yields T_2, which substituted in (3) with the known value from (10) for T_1, gives $\mathrm{d}T$. Since $\mathrm{d}A$ and $\mathrm{d}T$ are now both known, (5) can at last be solved for $\mathrm{d}W$.

The particular solution path just given is not unique; in fact there are 21 different derivation sequences, each leading to the answer in five to seven steps. Different members of this set of sequences could be produced by slight variants of the production system. Notice that in this problem environment, there is no danger of exponential explosion from uncontrolled search. Initially, there are ten equations, not all independent. Each time the value of a variable is found, it can be stricken from the 'wanted list' and its value substituted in all the equations in which it appears. This sequence of actions needs be performed only once for each variable, and only a subset of the equations needs to be solved. As values are found for successive unknowns, the system quickly runs out of things to do and comes to a halt with the desired answer. The secret of its convergence is that it does not create new variables beyond those introduced by the initial representation.

This is, of course, a trivial example[3], although an examination of textbook problems in physics (kinematics and thermodynamics) reveals that it is not atypical of problems encountered in science courses. Experts typically solve these problems by working-forward methods that require little or no initial planning and that are very similar to the production system that I have sketched above.

In the algebra example, the explicit problem statement mentions only three of the ten equations that are implicit in the situation (equations (7), (8), and (10)). All the rest, which mainly take the form of conservation laws, are supplied by the problem solver, who generates, in effect, the bounded problem space in which he then conducts his search[4]. Thus the problem statement does not contain all of the information that is required to solve the problem. It relies on the 'common sense'—as John McCarthy and others call it—of the solver to provide a great many essential premises. In fact, in this particular case (and in many other textbook problems), it requires the solver to provide an assumption that is empirically false, for the volume of a mixture of water and alcohol is not exactly equal to the sum of the volumes of the water and the alcohol. Similarly, in textbook problems about falling bodies, the solver is generally expected to proceed as if the fall were taking place in a vacuum, unless the contrary is explicitly mentioned.

It cannot be emphasized too strongly that the additional premises that must be supplied by the problem solver are empirical, and represent knowledge that he has about the world. Hence, it would be unreasonable to try to provide these premises by strengthening the logic used in the problem solving[5]. Doing so would only lead to error on other problems where different empirical assumptions were to be made. There is no substitute for the problem solver knowing what can validly be assumed about the empirical situation in the context of any given problem. I shall have more to say later about how this knowledge might enter into the problem solving process.

[3]It can be trivialized further by making some of the explicit assumptions implicit. For example, if we make $A_1 = A_2 = A$, suppressing the subscripts since the amount of alcohol is not changed, and if we rewrite equation (3) as $T_1 + \mathrm{d}W = T_2$, then we need only equations (3), (7), (8), and (10) which we solve successively for T_1, A_2, T_2, and $\mathrm{d}W$.

[4]In the simplified representation of footnote 3 equation (3) is one conservation assumption, the identification of A_1 with A_2 from equations (1) and (9) is the other.

[5]Of course, this particular consideration argues only against embedding these premises in the axioms of the logic, not against expressing them in the predicate calculus or some other language of logic.

1.3. Implications of the example

Several lessons, then, can be drawn from the example of the mixture problem. The first is that nothing is to be gained in a case like this from translating the mathematical representation into a more formal representation in terms of the predicate calculus. In fact a great deal would be lost: specifically, the power of the inference rules of ordinary mathematical reasoning which reduce the problem space to trivial size and allow a solution to be found with the aid of a very simple search control structure. Of course, this power may be reintroduced into the formal system by the 'back door' of the semantic or procedural attachments mentioned in Section 1.2. But, as I have pointed out, using this back door is equivalent to replacing formal reasoning by search and modeling. We must beware, then, of equating 'reasoning' with the use of the predicate calculus.

The second important lesson is that problems like this one, as ordinarily stated, leave essential assumptions implicit, and that these include not only logical assumptions (i.e., what rules of inference are permitted), but also factual premises. A system for carrying out common sense reasoning must have capabilities for supplying appropriate factual assumptions.

1.4. Nondeterministic character of logic

The remark was made earlier that search is concerned with discovery, reasoning with proof. The rules of logic are permissive: they determine what inferences *may* be drawn directly from a set of premises, not what inferences *must* be drawn, or in what order they must be drawn. A logic, taken by itself, may be viewed as a non-deterministic algorithm for finding (say, by breadth-first search) all of the consequences of a set of premises.

When it is important to be selective in drawing inferences, because the ones we are interested in constitute only a few elements in a large space of logical consequences, then the logic must be supplemented by some kind of control structure. A search strategy must be superimposed upon it. Thus, a system like PROLOG consists both of an underlying logic and a working-backward search algorithm. In this sense, reasoning processes are a subset of search processes; the subset that uses rules of inference as the sole operators.

The nondeterministic character of logic was obscured, in the example we have just been considering, by the small size of the total space of consequences and the power of the algebraic operators in taking sizable steps in transforming expressions. We could be indifferent to the order in which inferences were drawn because we would not be seriously delayed or inconvenienced if irrelevant ones were produced along the way to the consequence in which we were interested. As soon as we tackle problems where the search space is large, we must be concerned with finding an efficient search algorithm for controlling the order of application of operators.

2. Problem Solving as Search

Although the mixture problem is perhaps most naturally viewed as a problem of reasoning, we have just seen that it can also be viewed in terms of search. The search space is the space of possible values of the variables that are unknowns after the problem has been represented and the conservation laws added. Each time the value of one variable has been found, a difference between present situation and desired goal has been reduced. The problem has been solved when no such differences remain.

The problem is especially well adapted to means-ends analysis because eliminating a difference (solving for a variable) does not reintroduce differences previously eliminated. Nor does the exact order of elimination matter; as soon as an operator is available for solving for a particular variable (i.e., as soon as that variable is the only remaining unknown in an equation) it can then immediately be applied.

2.1. A more familiar search problem

Nevertheless, the mixture problem is a rather atypical search problem. The venerable Missionaries and Cannibals problem will be more suitable for illustrating search processes. The problem is usually represented in terms of a set of states, each corresponding to a particular distribution of the missionaries, cannibals, and boat between the two banks of a river. Move operators act on these state representations to change the distribution—i.e., to move from one state to another.

In the state-space representation, truths are only contingent. We can say, "in state *A*, there are two cannibals on the left bank", but not without qualification, "there are two canibals on the left bank". This contingency of assertions causes no difficulty when the state representation is used; at any given stage in the solution process a particular state *S* is instantiated. Since S incorporates, explicitly or implicitly, what is true of that state, test or inference processes can be constructed that will derive propositions that are *true in S*.

The representation can be viewed as a *basis* for all of these valid propositions. "There are two cannibals on the left bank" is more or less explicitly represented, while "there are more cannibals than missionaries on the left bank" is more or less implicitly represented. That is to say, a process to test the validity of the former proposition will be simpler and more direct than a process to test the validity of the latter. 'Explicit' and 'implicit' are relative terms, depending both on the representation and upon the primitive tests that can extract information from it. Notice that no special logic is needed with this representation. The inferences involve, at most, ordinary mathematics. The contingency of statements is absorbed by incorporating the value of the current state as an argument in all of them.

2.2. State-space representations

The state-space representation is borrowed from the classical representations of physics and other domains of applied mathematics. In these domains, a set of basis variables is selected (position and velocity in the case of classical dynamics), and each space-time point is characterized by a vector of the values of these variables. The laws of the system, typically in the form of differential equations, are the 'move operators'. Ordinary mathematical reasoning permits inferences to be drawn about the values of variables other than the basis variables.

In the state-space representation the existence of a relatively small and parsimonious set of basis variables is of the greatest importance, for it permits relatively simple move operators to accomplish the shift from one state to another. To instantiate a state it is not necessary to derive all the propositions that are true in that state, nor to cancel all the propositions true in the previous state that are no longer true. Attention need be given only to the values of the basis variables.

There is no requirement, of course, that the representation be minimal in this sense. In some circumstances it may be convenient to carry other, redundant, variables along as well, updating the values of these at the same time as the values of the basis variables are updated. The important point is that such redundant updating may be convenient, but it is not necessary. A great deal of the power of classical mathematics to represent quite complex systems hinges on this fact.

2.3. STRIPS-like systems

This economy of representation is exploited effectively in systems like STRIPS [2]. In order to avoid the necessity of updating every proposition that is true in a particular state, the representation is usually limited to a set of basis variables, which are updated by application of move operators that add and delete elements (propositions) from the representation. Properties of a state that are not included in the basis are derived from the basis by ordinary reasoning from premises that refer to a single state. Since truth is maintained within each state, no special non-monotonic logics are needed. When it is desired to keep around information about states other than the current state, this information can simply be labeled with the name of the state in which it holds.

STRIPS-like systems often provide another important kind of processing efficiency. In many systems, the move operators alter only a few characteristics of the situations to which they are applied. Properties of the state that are unchanged in the succeeding state are simply retained in the representation without any need for processing. They now are interpreted as holding in the successor state rather than the predecessor state. The differential and difference equation systems of physics and economics seldom have this convenient property, since in general the values of all state variables change at each 'instant'. But many of the problems we deal with in AI are much kinder in this respect, the move operators affecting only a few components at a time.

In the AI literature, STRIPS-like systems have often been treated as reasoning systems rather than search systems. (The initial STRIPS publication [2] is subtitled, "A new approach to the application of theorem proving to problem solving".) The analysis here suggests that, on the contrary, their power lies in their ability to model the movement of a system through state space without updating all the facts about successor states by explicit deduction. This ability, in turn, stems from the fact that the move operators serve as powerful inference rules, dispensing with the need for repeated reference to an underlying formal deductive logic.

2.4. Reasoning about actions

The systems of classical physics with which I have been comparing AI state-space search systems are, of course, systems for predicting system behavior, not systems for choosing actions. Their move operators define what *must* happen, not what *may* happen. Their logics are deterministic, not non-deterministic. Perhaps different methods are required when the task changes from prediction to planning or choice. Perhaps the case for modal or other non-monotonic logics rests on the need for considering alternative possible worlds in addition to an actual world path.

That this is not the case can be seen by examining any of the many normative formal systems that have been constructed in economics and operations research for choosing optimizing or satisficing policies. Techniques like dynamic programming, for example, employ state-space representations with parsimonious bases, and qualify all assertions about the values of variables by the states for which they hold. The visible evidences of this are the time subscripts that are prominent in every equation. The methods of reasoning employed are the methods of ordinary mathematics; no special logics are required or used. In the contemporary logical literature, this approach goes under the rubric of 'possible world semantics'. I have discussed the logical foundations for these kinds of normative and predictive models more fully in [10, Chapters 3.1 and 3.2].

In the literature of non-monotonic and modal logics, I do not believe that it has been observed that the physical sciences and economics have, for many generations, been using the state-space representation and standard mathematical reasoning to analyze the paths of systems through time, and that no particular problems of truth maintenance arise. The temporal (or spatial) contingency in the values of variables is dealt with economically and rigorously by the use of subscript notation.

In accord with what we discovered about reasoning systems in the previous section a consideration of systems that solve problems by search indicates that the power of such systems depends in considerable measure upon the use of efficient move operators which serve as macro inference rules. Provided that such operators are defined appropriately reasoning carried out with their aid is rigorous, in the same sense that applied mathematics is rigorous, and dispenses with the need for a more formal logical apparatus.

3. Model and Reality

The earliest discussions of modal logics for AI focussed mainly on the problems of reasoning about actions. McCarthy and Hayes [6] sought to construct a causal calculus containing predicates like CANCAUSE(A, C), where A is an action and C a consequence. Modal logics of this kind with appropriate inference rules—neither too weak nor too strong—have proved to be very elusive [11].

3.1. Logics of causality and possibility

The basic difficulty in constructing a logic that will handle satisfactorily the modalities of possibility and causality lies in the rules of composition. We are tempted, in analogy with the predicate calculus, to introduce a rule that would allow us, given "A enables B" and "A enables C", to infer "A enables B and C". But it is certainly not the case that "\$15000 in the bank enables us to buy that car" and "\$15000 in the bank enables us to buy that yacht" allow us to infer "\$15000 in the bank enables us to buy that car and that yacht".

Many other paradoxes of this kind can easily be constructed [11]. They generally stem from some sort of non-independence of events. When events are in fact interdependent, composition laws of ordinary strength produce paradoxes. But in the absence of such laws, the logic is too weak to make the inferences we need for solving problems by reasoning about causality and possibility.

This difficulty can be avoided by employing the modal concepts as heuristics instead of embedding them in a system of logical reasoning. For example, GPS-like heuristics for means–ends reasoning perform the basic functions of a calculus of causality and possibility. The search procedure of GPS is built on the implicit premise that if the present situation differs from the goal situation by features A, B, C, . . . , then the goal situation can be attained by removing the differences A, B, C, . . . , in some order. Of course this premise is false unless the matrix of connections between differences and operators can be triangularized. This matrix can be triangularized just under those conditions when an appropriate composition axiom would be valid in the modal logic; that is, just when there is independence among the actions.

The advantage of proceeding heuristically is that we then do not *assert* statements that turn out to be invalid, we merely consider possibilities, which may or may not be realized. Of course there are other search heuristics besides means–ends analysis. A preference for heuristics over modal logics does not commit us to this particular one.

3.2. Non-monotonic reasoning

Most recent discussions of non-monotonic logics have emphasized rather different issues from those just discussed, and particularly the issue of what might be called 'inference from insufficient evidence'. The gist of such inference is the assertion of certain propositions without direct proof on the basis of their consistency with the remaining propositions, so that there is no reason for them not to be true. An example would be, "If there is no known reason why you cannot take a bus to get to *X*, you can take a bus."

Why it would be convenient to have inference rules of this kind will become apparent a little later. Using them calls for several pieces of machinery. First, there must be a procedure for deciding what is sufficient evidence for asserting a proposition, or sufficient evidence that it does not contradict other propositions. Second, such systems leave open the possibility that what is at one time non-contradictory, hence assertable, later becomes contradictory. Hence, there must be a procedure for resolving contradiction when it is discovered. Obviously, many different inference systems can be constructed by ringing the changes on these two procedures.

John McCarthy [5] has recently used the Missionaries and Cannibals puzzle to illustrate why he believes there is a need for modal and non-monotonic reasoning. He points out that the puzzle is usually represented formally by a state description much like the one I sketched earlier. He then goes on to object [5, pp. 29–30]:

> We are not presently concerned with the heuristics of the problem but rather with the correctness of the reasoning that goes from the English statement of the problem to ... [the formal] ... state space representation. A generally intelligent computer program should be able to carry out this reasoning. Of course, there are the well known difficulties in making computers understand English, but suppose the English sentences describing the problem have already been rather directly translated into first order logic. The correctness of ... [the formal] ... representation is not an ordinary logical consequence of these sentences for two further reasons.
>
> First, nothing has been stated about the properties of boats or even of the fact that rowing across the river doesn't change the numbers of missionaries or cannibals or the capacity of the boat. Indeed it hasn't been stated that situations change as a result of action. These facts follow from common sense knowledge, so let us imagine that common sense knowledge, or at least the relevant part of it, is also expressed in first order logic.
>
> The second reason we can't *deduce* the propriety of ... [the formal] ... representation is deeper. Imagine giving someone the problem, and after he puzzles for a while, he suggests going upstream half a mile and crossing on a

bridge. "What bridge", you say. "No bridge is mentioned in the statement of the problem." And this dunce replies. "Well, they don't say there isn't any bridge." ... So you modify the problem to exclude bridges, and pose it again, and the dunce proposes a helicopter...

In spite of our irritation with the dunce, it would be cheating to put into the statement of the problem that there is no other way to cross the river than using the boat and that nothing can go wrong with the boat. A human doesn't need such an ad hoc narrowing of the problem, and indeed the only watertight way to do it might amount to specifying the [formal] representation in English. Rather we want to avoid the excessive qualification and get the [formal] representation by common sense reasoning as humans ordinarily do.

If we return for a moment to our earlier water–alcohol example, we see that the very same difficulties that McCarthy points to in the Missionaries and Cannibals puzzle were present in that algebra problem. Equations crucial for solving the problem were absent from the problem statement, and no explicit assurances were given that the equations given and inferred constituted *all* the relations constraining the situation. The "fact that rowing across the river doesn't change the numbers of missionaries or cannibals or the capacity of the boat" is precisely the same kind of conservation assumption as the assumption that the total volume of liquid will be the sum of the original volume and the volume of water added.

3.3. Common sense knowledge

McCarthy is surely right, then, in observing that the problem solver must supply common sense knowledge that is not explicit in the problem statement. Is he right, however, in supposing that this common sense knowledge must be supplied, as in the algebra case, in the form of explicit propositions? The knowledge is needed independently of whether we take a search viewpoint or a reasoning viewpoint. How is such knowledge represented in a state-space representation of the Missionaries and Cannibals puzzle, and where does the knowledge come from?

A typical representation of the puzzle in a list-processing language might consist of a property list for each bank of the river. The value of 'missionaries' could be either the number of missionaries on that bank, or a list of them; and the cannibals and boat could be represented correspondingly by two similarly encoded properties. The move operator would alter the values of each of these properties for both banks of the river, but in such a way as to conserve the total number of missionaries, cannibals, and boats. Tests to insure the legality of the move could be incorporated in the move operator, or could be separate routines.

By this legerdemain, the assumptions of conservation of missionaries, cannibals, and boats can be embedded in the representation without propositionalizing them. The system does not *reason* that these quantities must be conserved, nor does it know any propositions that could be used as premises in such reasoning. All this knowledge is built into the structure of the move operator, which *does* conserve them.

As we have already seen, we can restore the reasoning viewpoint by a procedure that we also use in formal logic. In logic, we generally have a choice between adding new axioms to a system or adding new rules of inference. If we regard the move operators in a state-space representation as truth-preserving rules of inference, then what we have done is to postulate such a rule rather than augmenting the axioms.

The analogy, while formally defensible, must be interpreted with caution. The conservation assumptions embedded in what we are now calling a rule of inference, are, as has already been emphasized, empirical, not logical, assumptions. Any claims for their legitimacy must be based on factual knowledge and not on purely logical arguments. Lest we be beguiled by the 'obviousness' of conservation principles, we should remind ourselves of several things we know about them.

First, as Piaget has shown, children are not born ready to assume that objects are conserved when they disappear momentarily from sight. This assumption is acquired by learning and/or maturation. There is a large psychological literature in the Piagetian tradition on the acquisition of conservation principles. Second, the human species as a whole has been a long time arriving at the more sophisticated conservation principles, like conservation of mass and conservation of energy. There apparently was nothing self-evident about these principles. Their discovery cost scientists much sweat and tears, if not blood. Third, quantities are not in fact always conserved. We have seen that even the simple assumptions of conservation of volume in mixture experiments are not always empirically correct.

We return again to the conclusions of our earlier discussion. The common sense knowledge that has to be provided for problem representations is empirical knowledge, specific to the domain of the problem under consideration and subject to revision as empirical knowledge changes. But this knowledge need not be provided in the form of explicit propositions; alternatively, it can be embedded implicitly in move operators that simulate the effects of actions or events in the world that is being modeled. It is stretching matters to interpret these move operators as 'rules of inference', for empirical as well as logical assumptions are built into them implicitly.

Taking the reasoning viewpoint toward problem solving can encourage the attitude that all problem assumptions should be explicit and propositionalized. Experience with problem solvers using the state-space representation suggests that it may be far more expendient to embed much of the knowledge (empirical as well as logical) in operators.

3.4. Common sense reasoning

We turn next to the second problem posed by McCarthy's dunce. How does the problem solver know that there are not alternative operators in addition to those of which he is aware? What rules of reasoning allow him to assume, in the Missionaries and Cannibals puzzle, that there is not a bridge half a mile upstream?

Despite the fact that McCarthy here refers to 'reasoning' rather than 'knowledge', this difficulty is not really different from the previous one. Whether a bridge does or doesn't exist is not a matter of logic, but a matter of fact. If there is a convenient bridge, and it is not included in the formal problem representation, then the latter is a factually incorrect description of the problem.

In order to produce a correct problem representation, it is certainly not necessary to propositionalize everything that is absent—that there is not a bridge, that there is no helicopter, and so on. Mother Hubbard did not have to enumerate all the items that were not in her cupboard; she simply had to observe that she could see its back wall, hence that it was bare. Whether we adopt the reasoning viewpoint or the search viewpoint, it remains true that we can reason correctly about problem situations on the basis of a representation that only specifies explicitly what *is* true of the situation, and omits mention of what *is not* true. And what is or isn't true is a matter of fact, not of logic.

3.5. Bounded rationality

Many of the difficulties that have been raised with respect to common sense reasoning disappear when we recognize that we are not concerned with a world in which problems can always be solved, and moreover in some optimal way. We are concerned with the kinds of bounded rationality of which human beings are capable.

The 'reasoning' that is called for in problem solving is generally inductive rather than deductive. We wish to find a sequence of actions that will transform the initial situation into a situation satisfying the solution conditions. There is usually no requirement that the solution be unique or, for that matter, that it satisfy any conditions of optimality. As posed, the Missionaries and Cannibals puzzle does not require us to get across the river as speedily as possible, or with the least labor on the part of missionaries. The solution can, of course, be regarded as a proof that taking certain actions will realize a certain goal, but there exists no deduction that will prove that these actions *must* be taken to attain this goal.

Now this observation takes away much of our concern for the possibility that there may be bridges upstream. Our ignorance of additional alternatives does not make it impossible for us to solve the problem. Mankind had many effective transportation systems before it learned how to build and fly airplanes. Our formal representation simply includes those alternatives of which we are aware or have access to. Failure to include other possibilities in the representation should not be interpreted as implying, logically or empirically, that no other possibilities exist.

It is fortunate that this is so; otherwise, we would be able to solve hardly any real-world problems. It has long been noted [12] that human beings generally satisfice—look for adequate rather than optimal solutions to their problems. One of the major reasons they do so is that they can seldom be sure that their problem representations contain all the alternatives. Another reason they do so is that, even within a given problem representation, it may be immensely less difficult to find *some* solution to the problem than to find a unique best solution.

4. Generating the Representation

But I have really only redefined the problem posed by McCarthy; I have not solved it. The difficulty he points to is a real one, but its nature is not revealed by applying the rubric of 'common sense reasoning'. The problem is to specify how we can go from an informal specification of a task to the formal specification—whether the latter be in terms of the first-order predicate calculus or a state-space representation.

Problems may be posed in natural language (e.g., the problems at the end of a physics textbook chapter), or they may be posed by stimuli coming directly from the real sensory world. McCarthy makes a useful distinction between real-world problems and puzzles. In puzzles, there is an implicit guarantee that no unmentioned alternatives of action exist (there are no bridges up the river). Nevertheless, puzzles may still require empirical knowledge, like conservation laws, for their solution. In the case of real-world problems, we are provided with no assurances that we possess all the relevant information. Hence, by acquiring more real-world information, either in the process of solving the problem or as a deliberate prelude to attempting it, we may drastically alter the problem formulation, that is, pose a new and different problem to be solved.

Regardless of whether we are faced with a puzzle or a real-world problem, at some point we use the information available to us to create a formal problem representation in which we can conduct our search or reasoning. It is *after* that representation has been created that the problem of logical reasoning arises. Up to that point, no formal situation on which a system of logic could operate has been presented to us.

When we use the reasoning metaphor, constructing a representation 'simply' requires providing a set of assertions about what is and isn't true of the problem world, but a set complete enough to serve as the axiomatic base for the inference engine. When we use the search metaphor, it requires providing the objects and relations that define the state space and the operators for changing states. If a reasoning system is augmented by macro-operators—procedural and semantic attachments—to facilitate search, then these must be supplied in exactly the same way.

When can we say about the process of creating a search representation? There exist at present several AI programs that cast some light on this question. I will discuss two of them: the UNDERSTAND program built by J.R. Hayes and myself [3], and the ISAAC program built by Gordon Novak [8]. The crucial characteristic of both of them, for present purposes, is that they translate problems posed in natural language into formal state-space representations.

4.1. The UNDERSTAND program

The UNDERSTAND program [3] was designed to handle puzzle-like problems. That is to say, it does not have empirical knowledge of any real-world domain, and would fail if such knowledge were required for a correct problem formulation. Though lacking specific empirical knowledge, it is prepared to make empirical assumptions of several sorts in the process of generating problem representations. The operators it builds embody implicit conservation principles, and it creates representations of only those things, properties, and actions that can reasonably be inferred from the problem statement.

Given natural-language instructions for a problem, UNDERSTAND undertakes to determine from them what *objects* and classes of objects the problem situation contains, what *relations* hold among objects, and what *operators* can be applied to alter situations (taking account of the legality conditions for the application of operators). Its seeks also to find a characterization of the starting situation and the goal situation. It chooses a representation for objects and their relations, and builds a data structure that represents the starting situation. Finally it builds programs that operate on the representation in the manner defined by the legal move operators of the problem statement. If it succeeds, UNDERSTAND builds a data structure and set of operators that could be turned over to a GPS-like problem-solving system, and that would provide to that system the information it would need to undertake attempts at finding the solution.[6]

For example, confronted with a natural-language description of the Missionaries and Cannibals problem, UNDERSTAND would proceed somewhat as follows. It would recognize the problem statement as mentioning four kinds of objects: river banks, missionaries, cannibals, and boats. It would detect the numbers of each of these. It would also recognize that certain sentences in the problem instructions describe a 'move' operator, and, associated with it, conditions of legality.

Using this information, UNDERSTAND would transform the natural language representation into a state-space representation with an associated operator for legal moves. The representation of the initial situation might consist of a data structure for each river bank. The list of missionaries, cannibals, and boat on that bank could be the value of an attribute of this structure. The move

[6]The UNDERSTAND program, as implemented, did not include a process for defining differences or for building the matrix of connections between differences and operators. From the information it provides, however, a GPS system with learning capabilities could generate these.

operator would remove a subset of these items from the list for one river bank and place them on the list for the other bank, after checking that all the conditions for a legal move were satisfied (i.e., checking the lists for the presence of the boat on the FROM side, and for the differences between numbers of missionaries and cannibals after the move was made)[7].

In order to construct a problem representation, UNDERSTAND does not have to understand what missionaries, cannibals, boats, or river banks are. Syntactic cues will generally suffice to identify these as objects and the missionaries, cannibals, and boat as associated, at any time, with a river bank. The program needs a little more knowledge to handle the move operator correctly. It might be given, for example, a definition of 'row' as: "to *row* a X across a Y is to *move* the X and all objects associated with the X from the Y bank where it was located to the other Y bank". This information could be provided in the problem instructions, or it could have been made available to UNDERSTAND previously.

It is not quite correct to say that UNDERSTAND knows *nothing* about the world. It operates on the basis of minimal information, and uses syntactic cues wherever possible to reduce its reliance on real-world knowledge. It creates a representation that is no more complex than is needed to represent the information presented to it, and does not create entities spontaneously. Because of the way in which it goes about constructing representations and operators, it makes a number of conservation assumptions—for example, the move operator conserves objects.

A large part of UNDERSTAND's knowledge is incorporated in the set of basic operators it has available, operators like 'move', 'change', and 'copy'. Each of these operators embodies different conservation assumptions. Other knowledge is used to build the legality tests. For example, predicates like 'less' and 'more' are available, and can be applied to various sets of objects. This knowledge is so scanty that UNDERSTAND can deal only with puzzles that deliberately abstract from real-world information. Even within the domain of puzzles, there are no guarantees, of course, that UNDERSTAND will represent a problem correctly (i.e., as the person who created it intended). Since correctness is an empirical rather than a logical matter, there can be no such guarantees.

4.2. The ISAAC program

Novak's [8] ISAAC program shows how we might begin to move from puzzle domains to problems denoting situations in the real world, still restricting ourselves, however, to problems stated in natural-language prose.

[7]All of the detail here has to be stated conditionally, since the precise representation that UNDERSTAND would construct, and the nature of the corresponding operators, would both depend on the language used to describe the problem. For examples of some of the alternative possibilities, see [3].

ISAAC is designed to deal with the kinds of problems that are found in physics textbooks—specifically problems in statics. The general strategy of ISAAC is similar to that of UNDERSTAND, with one important difference. As we would expect from the nature of its task, ISAAC depends far more than UNDERSTAND upon pre-stored empirical knowledge about the domain of its problems. In its memory are stored a number of object schemas, in the form of property list structures, that describe the kinds of objects that appear in statics problems—levers, for example, masses, pivots, and so on. It also knows such facts as that a 'man' can be treated (in statics problems!) as either a mass or a pivot.

When ISAAC recognizes a term (e.g., 'lever') in a problem statement, it maps the referent of the term onto the appropriate object schema in memory, using the information about the referent provided in the problem statement to instantiate a copy of the schema. Thus, a representation is created of a particular lever, with particular dimensions, point of attachment, and so on. ISAAC assembles these instantiated object schemas to form a representation of the whole problem. This latter representation, or problem schema, can then be used by other components of ISAAC that incorporate laws of mechanics to generate the appropriate equations and solve them.

Clearly, an ISAAC capable of operating over a wide range of problem domains would have to have a large store of appropriate object schemas—even, one might say, as you and I. A human expert in any domain might possess tens or possibly hundreds of thousands of such schemas. (Natural language vocabularies of college graduates are of the order of 50000 words.) But our present concern is not with how large the body of knowledge must be for a system to exhibit common sense in some domain. Our interest is in how that knowledge might be organized. The experience with ISAAC suggests that propositionalizing the knowledge, in any usual sense and within the format of a standard logic, is not the only way to proceed, and may well not be the best way.

5. The Reasoning of Robots

UNDERSTAND and ISAAC handle problems presented in natural language, but not real-world problems where new information can continue to be acquired while the solution is being executed. To see what is involved here, we must consider some additional objections raised by McCarthy's dunce. I take up again McCarthy's narrative [5, p. 30]:

> You now see that while a dunce, he is an inventive dunce. Despairing of getting him to accept the problem in the proper puzzler's spirit, you tell him the solution. To your further annoyance, he attacks your solution on the grounds that the boat might have a leak or lack oars...

The dunce's new objections are not as easily disposed of as those we considered earlier. Now it is more than a matter of not finding the best solution because we are ignorant of the existence of a bridge. Now it is objected that, because of our ignorance, the solution may not be a solution at all, for we may not have included in our representation of the problem all of the operator conditions. Ignorance of a necessary condition for workability of a solution may be much more damaging than ignorance of additional possible solution paths.

A proper answer to this objection is "just as in the real world". Every day, probably, someone does paint himself into a corner, and someone does generate and attempt to execute a plan that is infeasible. There is no way in which 'logic' or reasoning can assure us that our premises form a complete set, or, indeed, that some of them may not be empirically false.

The formal state-space representations of problems, which we use in searching or reasoning about them, are at best simplified and approximate models of the real world. We operate on the representations as *though* they were the real world, and sometimes the lacunae and inaccuracies are not so damaging as to prevent us from forming reasonably realizable plans. If we wish to formalize the rule of insufficient reason we are following in building and using such models, it is: "The things we have left out don't matter to a damaging extent, and the things that are in the model are as accurate as they need to be." Assuring ourselves that a model has these properties in relation to particular decision tasks is a difficult job, but mainly an empirical job, and not an exercise in logic.

The fact that models represent the real world only very imperfectly and incompletely does not make them useless. An important mitigating circumstance is that plans do not usually have to be executed without periodic feedback from the external world. When, in the process of plan execution, actual outcomes fail to match expected outcomes, the model of the situation and the solution can be revised to fit the reality. The unrealism of planning models need not always be remedied by making them more realistic, hence more complex. In many situations, it may be advantageous to carry out problem solving in the context of the over-simple model, and to depend on feedback to provide the second-order approximation.

The relation between the state-space representation and the external world is not unlike the relation between an abstract planning space and the original problem space that has been abstracted. In the more abstract space, proposed problem solutions are generated which must then be tested against the additional constraints of the external world or the problem space, as the case may be. In Gelernter and Rochester's early geometry theorem prover, the semantic space of diagrams performed the same function of enabling hypotheses to be made quickly and cheaply, subject to subsequent test of their validity.

What distinguishes robotics, then, from other areas of AI (except for this analogy with planning) is that the robot is embedded in an external environment that it can sense and act upon. This has two consequences. First, it imposes a severe performance test: the robot's performance is evaluated by its behavior in its external environment, not by its self-imagined behavior in a toy problem space. Second, the robot can adapt the complexity and other properties of its internal problem space to its own computational capabilities, depending upon feedback to eliminate the discrepancies between expectations and reality [10, Chapter 5.3].

For this reason, robotics appears to be the most promising area in AI in which to study everyday problem solving. The peculiar flavor of robotics research is that it continually reminds the researcher that the representation is not the reality, and that only by constant readaptation of the former to the latter can actions be produced that are likely to be reasonably effective in the external world. Producing models at an adequate level of accuracy and complexity, and feedback mechanisms sufficiently sensitive to keep them in adjustment is not a matter of logic. It is mainly a matter of having good, empirically valid, scientific theories of how the outside world works. The critical component of human bounded rationality is empirical knowledge.

6. Conclusion

It has been the purpose of this paper to consider some of the relative advantages and disadvantages of viewing problem solving as a process of search or a process of reasoning, respectively. John McCarthy has pointed out that in assessing problem solving procedures we must consider not only their performance in *solving* problems once those problems have been represented, but also their adequacy in *representing* problems when those problems are presented initially in natural-language prose, or when the problems are to be regarded not as toy laboratory tasks but as denoting real situations in the real world outside. In this paper, I have discussed the role of reasoning both in problem solution and in problem representation.

Our discussion has made clear that in its initial presentation, whether by verbal means or by direct confrontation with the external world, problem information is usually radically incomplete, and must be supplemented by information supplied by the problem solver. Most of this information does not derive from principles of logic, but represents empirical assumptions about the problem domain. Whatever the form of problem representation, this knowledge must somehow be available to the problem solver and cannot be supplied by the problem statement. In search-oriented problem solving systems, like UNDERSTAND or ISAAC, this information is supplied by schemas stored in memory, and matched to cue information provided by the problem statement.

There are some fundamental and nearly universal difficulties that have to be overcome in problem representation. There are the difficulties of interpretation and translation of natural language prose or information gained through the senses into internal representations. There are difficulties of supplying the alternative actions that are not mentioned in the problem statement. There are difficulties of supplying additional information about the objects in the problem representation. There are difficulties of supplying all of the side conditions that must be met if those actions are to be feasible.

There have been numerous suggestions for building non-standard logics that would, in essence, supply this additional information required by the problem-solving system. The flaw in these suggestions is that almost all the information required is empirical information, which will vary from one problem situation to another. It is not information about permissible forms of logical inference. Hence, it seems unproductive to look for standard, domain-independent rules that will take the place of specific empirical assumptions, varying from one problem to the next.

The alternative procedure is to store in the memory of the problem solver schemas that embody such information, and that can be evoked by appropriate cues in the problem statement and by appropriate sensory cues. That is procedure employed by problem solving systems like UNDERSTAND and ISAAC that use state-space representations, and by SCRIPT schemes.

But even these systems lack an important component that is essential for successful response to real-world situations. That component is a feedback mechanism which receives sensory information from the external world that enables it continually to adjust its expectations to the unfolding reality. Only as we begin to build systems with such capabilities will we be able to talk about 'common sense'. It is perhaps more than an etymological accident that the second word in that idiom refers explicitly to this feedback tie with the outside world.

Finally, reasoning, in the sense of accumulating information by inference, is only one of the several thinking processes that are useful for solving problems. And formal logic is only one of the instruments for reasoning. I have tried to make the case here for employing in AI the whole range of available information processes and for not equating thinking with formal logic.

ACKNOWLEDGMENT

I am deeply indebted to John McCarthy and Nils J. Nilsson for thoughtful comments on an earlier draft of this paper. The fact that their viewpoints on this topic are fundamentally opposed to mine made their criticisms doubly useful. It will not surprise readers that my draft was not sufficiently eloquent to reverse their beliefs, and that I have, therefore, more than the usual reasons for acquitting them of the responsibility for errors in this revised version, while thanking them warmly for forcing me to clarify my position. I am also grateful to the members of CMU's Cognitive Science Seminar for their valuable comments on an oral presentation of this paper.

REFERENCES

1. Bobrow, D.G., (Ed.,) Special Issue on Non-Monotonic Logic, *Artificial Intelligence* **13** (1, 2) (1980).
2. Fikes, R.E. and Nilsson, N.J., STRIPS: A new approach to the application of theorem proving to problem solving, *Artificial Intelligence* **2** (1971) 189–208.
3. Hayes, J.R. and Simon, H.A., Understanding written problem instructions (1974) in: H.A. Simon, *Models of Thought* (Yale University Press, New Haven, CT, 1979).

4. McCarthy, J., Programs with common sense, in: *Proceedings of the Teddington Conference on the Mechanization of Thought Processes* (H.M. Stationery Office, London, 1960).
5. McCarthy, J., Circumscription—a form of non-monotonic reasoning, *Artificial Intelligence* **13** (1980) 27–40.
6. McCarthy, J. and Hayes, P., Some philosophical problems from the standpoint of artificial intelligence, in: D. Michie (Ed.), *Machine Intelligence* **4** (American Elsevier, New York, 1969).
7. Nilsson, N.J., *Principles of Artificial Intelligence* (Tioga Press, Palo Alto, CA, 1980).
8. Novak, G.S., Representations of knowledge in a program for solving physics problems, *Proceedings of the Fifth International Joint Conference on Artificial Intelligence* **1** (1977) 286–291.
9. Simon, H.A., The theory of problem solving, in: *Information Processing* 71 (North-Holland, Amsterdam, 1972) 261–272.
10. Simon, H.A., *Models of Discovery* (Reidel, Dordrecht, 1977).
11. Simon, H.A., On reasoning about actions, in: H.A. Simon and L. Siklóssy (Eds.), *Representation and Meaning* (Prentice-Hall, Englewood Cliffs, NJ, 1972) 414–430.
12. Simon, H.A., *Administrative Behaviour* (Macmillan, New York, 1947).
13. Weyhrauch, R.W., Prolegomena to a theory of mechanized formal reasoning, *Artificial Intelligence* **13** (1980) 133–170.

Received August 1982; revised version received September 1982