The Structure of Ill Structured Problems*

Herbert A. Simon
Carnegie-Mellon University, Pittsburgh, Pa.

Recommended by Saul Amarel

ABSTRACT

The boundary between well structured and ill structured problems is vague, fluid and not susceptible to formalization. Any problem solving process will appear ill structured if the problem solver is a serial machine that has access to a very large long-term memory of potentially relevant information, and/or access to a very large external memory that provides information about the actual real-world consequences of problem-solving actions. There is no reason to suppose that new and hitherto unknown concepts or techniques are needed to enable artificial intelligence systems to operate successfully in domains that have these characteristics.

1. Introduction

Certain concepts are defined mainly as residuals—in terms of what they are not. Thus a UFO is an aerial phenomenon not explainable in terms of known laws and objects; and ESP is communication between persons, without evidence of the transmission of signals of any kind.

In just the same way, "ill structured problem" (ISP) is a residual concept. An ISP is usually defined as a problem whose structure lacks definition in some respect. A problem is an ISP if it is not a WSP (well structured problem).

Residual categories are tenacious: it is extremely difficult, or even impossible, to prove that they are empty. The scope of a residual category can be narrowed progressively by explaining previously unexplained phenomena; it cannot be extinguished as long as a single phenomenon remains unexplained.

In this paper I wish to discuss the relation between ISPs and WSPs with the aim of asking whether problems regarded as ill structured are inaccessible to the problem solving systems of artificial intelligence in ways that those

* This research was supported in part by Research Grant MH-07722 from the National Institute of Mental Health and in part by the Advanced Research Projects Agency of the Office of the Secretary of Defense (F44620-70-C-0107) which is monitored by the Air Force Office of Scientific Research. I am grateful to Dr. Aaron Sloman and Prof. Saul Amarel for helpful comments on an earlier draft of this paper.

Artificial Intelligence 4 (1973), 181–201

regarded as well structured are not. My aim will not be to restrict the class of problems we regard as ISPs—in fact I shall argue that many kinds of problems often treated as well structured are better regarded as ill structured. Instead, I will try to show that there is no real boundary between WSPs and ISPs, and no reason to think that new and hitherto unknown types of problem solving processes are needed to enable artificial intelligence systems to solve problems that are ill structured.

Some years ago, Walter Reitman provided the first extensive discussion of ISPs (which he called "ill defined problems"; see [10, 11]). More recently, the topic has been developed in a somewhat different vein by Allen Newell [6], who emphasized the relation between problem structure and problem-solving methods. Newell characterized the domain of ISPs as the domain in which only weak problem-solving methods were available.

In this account, I continue the discussion begun by Reitman (and on which I earlier made some preliminary remarks in [14, Section 5, pp. 274–76]). I shall try to give a positive characterization of some problem domains that have usually been regarded as ill structured, rescuing them from their residual status; and then I shall ask whether the methods used in contemporary artificial intelligence systems are adequate for attacking problems in these domains. I shall not prejudge whether the methods applicable to these problems are weak or strong, but shall leave that to be decided after the fact.

The first section sets forth a set of strong requirements that it is sometimes asserted a task must meet in order to qualify as a WSP. Each of these requirements is examined, in order to characterize the kinds of ISPs that fail to satisfy it. The meaning of the requirements, and their relation to the power of the available problem solving systems, is then explored further by considering some specific examples of WSPs and of ISPs. Finally, this exploration provides the basis for a description of problem solving systems that are adapted to attacking problems in the domains usually regarded as ill structured.

2. Well Structured Problems

For reasons that will become clear as we proceed, it is impossible to construct a formal definition of "well structured problem". Instead, we must be content simply to set forth a list of requirements that have been proposed at one time or another as criteria a problem must satisfy in order to be regarded as well structured. A further element of indefiniteness and relativity arises from the fact that the criteria are not absolute, but generally express a relation between characteristics of a problem domain, on the one hand, and the characteristics and power of an implicit or explicit problem solving mechanism, on the other.

With these caveats, we will say that a problem may be regarded as well structured to the extent that it has some or all of the following characteristics:

1. There is a definite criterion for testing any proposed solution, and a mechanizable process for applying the criterion.

2. There is at least one problem space in which can be represented the initial problem state, the goal state, and all other states that may be reached, *or considered,* in the course of attempting a solution of the problem.

3. Attainable state changes (legal moves) can be represented in a problem space, as transitions from given states to the states directly attainable from them. But considerable moves, whether legal or not, can also be represented—that is, all transitions from one considerable state to another.

4. Any knowledge that the problem solver can acquire about the problem can be represented in one or more problem spaces.

5. If the actual problem involves acting upon the external world, then the definition of state changes and of the effects upon the state of applying any operator reflect with complete accuracy in one or more problem spaces the laws (laws of nature) that govern the external world.

6. All of these conditions hold in the strong sense that the basic processes postulated require only practicable amounts of computation, and the information postulated is effectively available to the processes—i.e., available with the help of only practicable amounts of search.

As I have warned, these criteria are not entirely definite. Moreover, phrases like "practicable amounts of computation" are defined only relatively to the computational power (and patience) of a problem solving system. But this vagueness and relativity simply reflect, as I shall try to show, the continuum of degrees of definiteness between the well structured and ill structured ends of the problem spectrum, and the dependence of definiteness upon the power of the problem solving techniques that are available.

2.1. The General Problem Solver

If a problem has been formulated in such a way that it can be given to a program like the General Problem Solver (GPS), can we say that it is a WSP? Before GPS can go to work on a problem, it requires:

(1) a description of the solution state, or a test to determine if that state has been reached;

(2) a set of terms for describing and characterizing the initial state, goal state and intermediate states;

(3) a set of operators to change one state into another, together with conditions for the applicability of these operators;

(4) a set of differences, and tests to detect the presence of these differences between pairs of states;

(5) a table of connections associating with each difference one or more operators that is relevant to reducing or removing that difference.

The first three requirements for putting a problem in a form suitable for GPS correspond closely to the first three characteristics of a WSP. The fourth and fifth requirements for putting a problem in a form suitable for GPS correspond closely to the fourth characteristic of a WSP. Since GPS operates on the formally presented problem, and not on an external real world, the fifth requirement for a WSP appears irrelevant to GPS.

In our description of the conditions for GPS's applicability, it is implicit that the sixth requirement for WSPs is also satisfied, for the operators and tests mentioned above can all presumably be executed with reasonable amounts of computation. This does *not* imply, of course, that any problem presented to GPS within the defined domain can be solved with only reasonable amounts of computation. Many problems may not be solvable at all. Of those that are solvable in principle, many may require immense numbers of applications of operators and tests for their solution, so that the total amount of computation required may be impractical.

Thus, it would appear at first blush that all problems that can be put in proper form for GPS can be regarded as WPSs. But what problem domains satisfy these, or similar, requirements? Let us examine a couple of possible examples.

2.2. Is theorem proving a WSP?

Consider what would appear to be an extreme example of a WSP: discovering the proof of a theorem in formal logic. Condition 1 for a WSP will be satisfied if we have a mechanical proof checker. Condition 2 might be regarded as satisfied by identifying the problem space with the space of objects that can be described in terms of wffs. However, we should note that limiting the problem solver in this way excludes it from even considering expressions that are not wffs.

The same reservation must be made with respect to Condition 3: definitions of the axioms, the rules of inference, and the processes for applying the latter determine the legal moves and attainable state changes; but the problem solver may wish to consider inferences without determining in advance that they meet all the conditions of "legality"—e.g., working backwards from unproved wffs. Hence the set of considerable moves is not determined uniquely by the set of legal moves.

Satisfying Condition 4 is even more problematic. There is no difficulty as long as we restrict ourselves to the object language of the logic under consideration. But we have no reason to exclude metalinguistic knowledge, knowledge expressed in a model space, or even analogical or metaphorical knowledge. A human theorem prover, using a metalanguage, may prove a

theorem that is not provable in the object language; or may use a truth table as a model for solving a problem in the propositional calculus; or may use the proof of one theorem as an analogical guide to the proof of another that seems, in some respect, to be similar to the first.

Of course there is nothing magical here in the problem solver being human. Mechanical systems can be, and have been, given the same kinds of capabilities. (For an example of the use of metalinguistic techniques in theorem proving, see [9]; for the use of analogies, see [3]). What some notions of well-structuredness require, however, is that these capabilities be defined in advance, and that we do not allow the problem solver to introduce new resources that "occur" to him in the course of his solution efforts. If this condition is imposed, a problem that admits restructuring through the introduction of such new resources would be an ill structured problem.

A problem that is not solvable with reasonable amounts of computation when all knowledge must be expressed in terms of the original problem space may be easily solvable if the solver is allowed to (or has the wits to) use knowledge in another space. It follows that, under a literal interpretation of Condition 4, problems of discovering proofs in formal logic are not, for all problem solvers, well structured.[1]

Condition 5 is always satisfied in theorem proving since there is no external "real world" to be concerned about. Condition 6 is usually satisfied, as far as the basic processes are concerned. Nevertheless, because of the enormous spaces to be searched, contemporary mechanical theorem provers, confronted with difficult theorems, usually fail to find proofs. It is sometimes said that they could only succeed if endowed with ingenuity; but ingenuity, whatever it is, generally requires violating Condition 4—moving out into the broader world of ISPs.

2.3. Is chess playing a WSP?

Next to theorem proving, the world of games would appear to offer the best examples of well-structuredness. All of the reservations, however, that applied to the well-structuredness of theorem proving apply to game playing as well. In addition, new reservations arise with respect to Condition 1—the solution criterion—and Condition 5—the correspondence between the inner world of thought and the outer world of action on real chess boards. Let us consider these two matters in more detail.

In both cases the difficulty stems from the immense gap between computability *in principle* and practical computability in problem spaces as large

[1] Notice that we are not appealing here to formal undecidability or incompleteness. Our concern throughout is with effective or practicable solvability using reasonable amounts of computation. Problems may be (and often are) unsolvable in this practical sense even in domains that are logically complete and decidable.

as those of games like chess. In principle, the concept of "best move" is well defined; but in practice, this concept has to be replaced by, say, maximizing some approximate evaluation function. When a chess playing program has found (if it does) the move that maximizes this function, it can still be far from finding the move that will win the game of chess—as the modest ability of the best contemporary programs testifies.

In terms of Condition 5, it is not hard to define the WSP of playing an approximate kind of "chess", where "winning" means maximizing the postulated evaluation function. But the values of moves as calculated by the approximate evaluation function are simply a means for predicting the actual consequences of the moves in the real game "outside." Feedback in terms of the expected or unexpected moves of the opponent and the expected or unexpected board situations arising from those moves calls for new calculations by the problem solver to make use of the new information that emerges.

The ill-structuredness, by the usual criteria, of chess playing becomes fully evident when we consider the play of an entire game, and do not confine our view to just a single move. The move in the real game is distinguished from moves in dynamic analysis by its irrevocability—it has real consequences that cannot be undone, and that are frequently different from the consequences that were anticipated. Playing a game of chess—viewing this activity as solving a single problem—involves continually redefining what the problem is. Even if we regard chess playing as a WSP in the small (i.e., during the course of considering a single move), by most criteria it must be regarded as an ISP in the large i.e., over the course of the game).

2.4. Summary: The elusiveness of structure

As our two examples show, definiteness of problem structure is largely an illusion that arises when we systematically confound the idealized problem that is presented to an idealized (and unlimitedly powerful) problem solver with the actual problem that is to be attacked by a problem solver with limited (even if large) computational capacities. If formal completeness and decidability are rare properties in the world of complex formal systems, effective definability is equally rare in the real world of large problems.

In general, the problems presented to problem solvers by the world are best regarded as ISPs. They become WSPs only in the process of being prepared for the problem solvers. It is not exaggerating much to say that there are no WSPs, only ISPs that have been formalized for problem solvers.

A standard posture in artificial intelligence work, and in theorizing in this field, has been to consider only the idealized problems, and to leave the quality of the approximation, and the processes for formulating that approximation to informal discussion outside the scopes both of the theory and of the problem solving programs. This is a defensible strategy, common to many

fields of intellectual inquiry; but it encourages allegations that the "real" problem solving activity occurs while providing a problem with structure, and not after the problem has been formulated as a WSP. As Newell and I have observed elsewhere [7, p. 850, footnote 20] these allegations are refuted simply by observing that "if [they] were correct, and tasks from the same environment were presented sequentially to a subject, only the first of them would present him with a problem, since he would not need to determine a new problem space and program for the subsequent tasks". Nevertheless, there is merit to the claim that much problem solving effort is directed at structuring problems, and only a fraction of it at solving problems once they are structured.

3. Ill Structured Problems

Perhaps something is to be learned by turning the question around. We have generally asked how problems can be provided with sufficient structure so that problem solvers like GPS can go to work on them. We may ask instead how problem solvers of familiar kinds can go to work even on problems that are, in important respects, ill structured. Since the problem domains that have been most explored with mechanical techniques fail in several ways to satisfy the requirements for WSPs, perhaps we have exaggerated the essentiality of definite structure for the applicability and efficacy of these techniques. Perhaps the tricks that have worked in relatively well structured domains can be extended to other domains that lie far over toward the ISP end of the spectrum.

To explore this possibility, we will again examine several examples. Each example will illustrate some specific facet (or several facets) of ill-structuredness. Analysis of these facets will provide us with a positive characterization of ISPs, rescuing them from the status of a residual category. With this positive characterization in hand, we will be in a better position to set forth the capabilities a problem solving system must have in order to be able to attack problems that are initially ill structured in one or more ways.

3.1. Designing a house

It will generally be agreed that the work of an architect—in designing a house, say—presents tasks that lie well toward the ill structured end of the problem continuum. Of course this is only true if the architect is trying to be "creative"—if he does not begin the task by taking off his shelf one of a set of standard house designs that he keeps there.

The design task (with this proviso) is ill structured in a number of respects. There is initially no definite criterion to test a proposed solution, much less a mechanizeable process to apply the criterion. The problem space is not defined in any meaningful way, for a definition would have to encompass

all kinds of structures the architect might at some point consider (e.g., a geodesic dome, a truss roof, arches, an A-frame, cantilevers, and so on and on), all considerable materials (wood, metal, plexiglass, ice—before you object, I must remind you it's been done—reinforced concrete, camel's hides, field stone, Vermont marble, New Hampshire granite, synthetic rubber, . . .), all design processes and organizations of design processes (start with floor plans, start with list of functional needs, start with facade, . . .).

The hopelessness of even trying to sketch the congeries of elements that might have to be included in the specification of a problem space proves the greater hopelessness of defining in reasonable compass a problem space that could not, at any time during the problem solving process, find its boundaries breached by the intrusion of new alternatives. The second, third, and fourth characteristics of a WSP appear, therefore, to be absent from the house design problem.

The fifth characteristic is also lacking. One thing an architect often does is to make renderings or models of the projected structure. He does this partly because these productions predict, more accurately than other means, properties that the real-world structure will possess if it is actually built. Viewing a model, the architect can detect relations among components of the design that were not available to him directly from his plans. Of course, even the renderings and models fall far short of predicting the actual characteristics of the real building, or the way in which the laws of nature will operate upon it and affect it. Hence, while Frank Lloyd Wright was not greatly disturbed by a leaking roof, it can hardly be supposed that he designed his roofs to leak, which they often did. Nor was the action of New York's atmosphere on the surface of the Seagram Building, and its consequent change of color, predicted. Doors stick, foundations settle, partitions transmit noise, and sometimes even happy accidents (examples of these are harder to come by) conspire to make the building as it actually exists and is used something different from the building of the plans.

Finally, even if we were to argue that the problem space can really be defined—since anything the architect thinks of must somehow be generated from, or dredged from, his resources of memory or his reference library—some of this information only shows up in late stages of the design process after large amounts of search; and some of it shows up, when it does, almost accidentally. Hence, the problem is even less well defined when considered from the standpoint of what is actually known at any point in time than when considered from the standpoint of what is knowable, eventually and in principle.

All of this would seem to make designing a house a very different matter from using GPS to solve the missionaries and cannibals puzzle, or from discovering the proof of a theorem in the predicate calculus. It surely is

different, but I shall try in the next paragraphs to show that understanding what the architect does goes far toward bridging the gulf between these problem domains. In this I shall be following quite closely the path first blazed by Walter Reitman [11, ch. 8] in his analysis of the thinking-aloud protocol of a composer writing a fugue. It should not, of course, appear surprising that a house-designing process would have much in common with a process of musical composition. Showing that they do have much in common is critical to the attempt here to provide a positive characterization of the processes for solving ISPs.

3.2. The architect's processes

Reitman uses the term "constraints" quite broadly to refer to any or all of the elements that enter into a definition of a problem. He observes [11, p. 169]:

> "One of the interesting features of many of the problem instances . . . is that even though they generally would be considered complex, they include very few constraints as given. Composing a fugue is a good example. Here the main initial constraint, and it is an open constraint at that [i.e., one that is incompletely specified], is that the end product be a fugue. All other constraints are in a sense supplementary, generated from one transformation of the problem to the next."

Similarly, the architect begins with the sole problem of designing a house. The client has presumably told him something of his needs, in terms of family size or number of rooms, and his budget (which the architect will multiply by 1.5 or 2 before accepting it as a constraint). Additional specification will be obtained from the dialogue between architect and client, but the totality of that dialogue will still leave the design goals quite incompletely specified. The more distinguished the architect, the less expectation that the client should provide the constraints.

3.2.1. *Evaluating the specifications*

We can imagine a design process that proceeds according to the following general scheme. Taking the initial goals and constraints, the architect begins to derive some global specifications from them—perhaps the square footage or cubic footage of the house among them. But the task itself, "designing a house", evokes from his long-term memory a list of other attributes that will have to be specified at an early stage of the design: characteristics of the lot on which the house is to be built, its general style, whether it is to be on a single level or multi-storied, type of frame, types of structural materials and of sheathing materials, and so on. The task will also evoke from memory some over-all organization, or executive program, for the design process itself.

Neither the guiding organization nor the attributes evoked from memory need at any time during the process provide a complete procedure nor complete information for designing a house. As a matter of fact, the entire procedure could conceivably be organized as a system of productions, in which the elements already evoked from memory and the aspects of the design already arrived at up to any given point, would serve as the stimuli to evoke the next set of elements.

Whether organized as a system of productions or as a system of subroutine calls, the evocation of relevant information and subgoals from long-term memory can be sequential. As Reitman says of the fugue [11, p. 169]:

> "Just as 'sentence' transforms to 'subject plus predicate', and 'subject' may transform to 'article plus noun phrase' . . ., so 'fugue' may be thought of as transforming to 'exposition plus development plus conclusion', 'exposition' to 'thematic material plus countermaterial', and 'thematic material' to 'motive plus development of motive'."

Applying the same linguistic metaphor to house design, "house" might transform to "general floor plan plus structure", "structure" to "support plus roofing plus sheathing plus utilities", "utilities" to "plumbing plus heating system plus electrical system", and so on.

The requirements that any of these components should meet can also be evoked at appropriate times in the design process, and need not be specified in advance. Consideration of the heating system can evoke from the architect's long-term memory (or the appropriate reference handbooks) that the system should be designed to maintain a temperature of 70°, that the minimum outside temperature to be expected is −5°, that the heat transmission coefficient of the proposed sheathing is kBTU per hour per square foot per degree of temperature differential, and so on.

Design alternatives can also be evoked in component-by-component fashion. The subgoal of designing the heating system may lead the architect to consider various fuels and various distribution systems. Again, the source of these generators of alternatives is to be found in his long-term memory and reference facilities (including his access to specialists for helping design some of the component systems).

The whole design, then, begins to acquire structure by being decomposed into various problems of component design, and by evoking, as the design progresses, all kinds of requirements to be applied in testing the design of its components. During any given short period of time, the architect will find himself working on a problem which, perhaps beginning in an ill structured state, soon converts itself through evocation from memory into a well structured problem. We can make here the same comment we made about playing a chess game: the problem is well structured in the small, but ill structured in the large.

3.2.2. *Coordination of the design*

Now some obvious difficulties can arise from solving problems in this manner. Interrelations among the various well structured subproblems are likely to be neglected or underemphasized. Solutions to particular subproblems are apt to be disturbed or undone at a later stage when new aspects are attended to, and the considerations leading to the original solutions forgotten or not noticed. In fact, such unwanted side effects accompany all design processes that are as complex as the architectural one we are considering. As a result, while the final product may satisfy all the requirements that are evoked when that final product is tested, it may violate some of the requirements that were imposed (and temporarily satisfied) at an earlier stage of the design. The architect may or may not be aware of the violation. Some other appropriate design criteria may simply remain dormant, never having been evoked during the design process.

The danger of inconsistencies and lacunae of these kinds is mitigated to some extent by that part of the architect's skill that is imbedded in the over-all organization of his program for design. Part of his professional training and subsequent learning is directed to organizing the process in such a way that the major interactions among components will be taken care of. Certain ways of dividing the whole task into parts will do less violence to those interactions than other ways of dividing it—a good procedure will divide the task into components that are as nearly "self-contained" as possible [1]. Early stages of the design can also establish global parameters which then become constraints operating on each of the components to which they are relevant. Thus general decisions about "style" can impose constraints on the stylistic decisions about particular portions of the house.

Much of the coordination of the various well structured design subtasks is implicit—built into the organization of the whole process. To the extent that this is so, the local design activities are guaranteed to mesh into a reasonable over-all structure. This means that the final product may be very much influenced by the order in which the design steps are taken up. As a result, differences in style between different designs can result as readily from the organization of the design process as from explicit decisions of the architect to specify one style or another. If the process calls for designing the facade before the floor plan, different kinds of designs will emerge than if the process calls for specifying the room arrangements before the facade [13].

3.2.3. *The over-all design process*

The design process sketched above can be composed from a combination of a GPS, which at any given moment finds itself working on some well structured subproblem, with a retrieval system, which continually modifies

the problem space by evoking from long-term memory new constraints, new subgoals, and new generators for design alternatives. We can also view this retrieval system as a recognition system that attends to features in the current problem space and in external memory (e.g., models and drawings), and, recognizing features as familiar, evokes relevant information from memory which it adds to the problem space (or substitutes for other information currently in the problem space). The retrieval system, then, is capable of interrupting the ongoing processes of the problem solving system. A schematic flow diagram for a system with these characteristics is shown in Fig. 1.

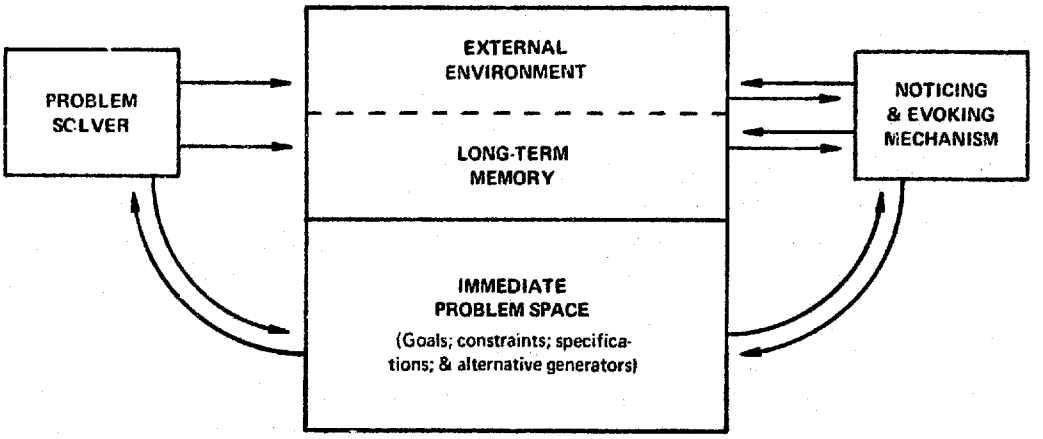

FIG. 1. Schematic diagram of a system for ill structured problems. It shows the alternation between a problem solver working on a well structured problem, and a recognition system continually modifying the problem space.

One might ask why the system is organized in this fashion—with alternation between problem solving in a (locally) well structured problem space and modification of the problem space through retrieval of new information from long-term memory. The answer revolves around the basically serial character of the problem solving system. The problem solving processes are capable, typically, of taking a few arguments as inputs and producing a small number of symbol structures as outputs. There is no way in which a large amount of information can be brought to bear upon these processes locally—that is, over a short period of processing. If a large long-term memory is associated with a serial processor of this kind, then most of the contents of long-term memory will be irrelevant during any brief interval of processing.

Consider the alternative, of bringing all of the potentially relevant information in long-term memory together once and for all at the outset, to provide a well structured problem space that does not change during the course of the problem solving effort. One must ask "bringing it together where?" Presumably it could all be assembled in some designated part of long-term

memory. But to what purpose? Retrieval processes would still be necessary for the serial problem solving processes to discover the inputs they needed at the time they needed them. To the outside observer, the continuing shift in attention from one part of the assembled task information to another would still look like a series of transformations of the problem space.

In the organization described in Fig. 1, there is no need for this initial definition of the problem space and task structure. All of the necessary definitory information is potentially available, but distributed through long-term memory. It is retrieved through two mechanisms: first, the normal subroutine structure, which enables processes to call subprocesses and to pass input and output information from one to another; second, the evoking mechanism, which recognizes when certain information has become relevant, and proceeds to retrieve that information from long-term memory.

3.3. Design as an organizational process

If this description of how an ill structured problem of design can be handled seems at all fanciful, its realism can be supported by comparing it with the description of complex design processes that take place in organizations. Let me quote at length the process, described by Sir Oswyn Murray some fifty years ago, of designing and producing a new battleship [5, pp. 216–217):

> "We start with the First Sea Lord and his Assistant Chief of Naval Staff laying down in general terms the features that they desire to see embodied in the new design—the speed, the radius of action, the offensive qualities, the armour protection. Thereupon the Directory of Naval Construction, acting under and in consultation with the Controller, formulates provisional schemes outlining the kind of ship desired together with forecasts of the size and cost involved by the different arrangements. To do this he and his officers must have a good general knowledge—in itself only attainable by close relations with those in charge of these matters—of the latest developments and ideas in regard to a great range of subjects—gunnery, torpedo, engineering, armour, fire-control, navigation, signalling, accommodation, and so on—in order to be reasonably sure that the provision included in his schemes is likely to satisfy the experts in all these subjects when the time for active cooperation arrives.
>
> With these alternative schemes before them, the Sea Lords agree on the general lines of the new ship, which done, the actual preparation of the actual designs begins. The dimensions and shape of the ship are drawn out approximately by the naval constructors. Then the Engineer-in-Chief and his Department are called in to agree upon the arrangement of the propelling machinery, the positions of shafts, propellers, bunkers, funnels, etc., and at the same time the cooperation of the Director of Naval Ordnance is required to settle the positions of the guns with their barbettes, and magazines and shell rooms and the means of supplying ammunition to the guns in action.
>
> An understanding between these three main departments enables further progress to be made. The cooperation of the Director of Torpedoes and the Director of Electrical Engineering is now called for to settle the arrangements for torpedo armament, electric generating machinery, electric lighting, etc. So the design progresses and is elaborated from the lower portions upwards, and presently the Director of Naval

> Construction is able to consult the Director of Naval Equipment as to the proposed arrangements in regard to the sizes and towage of the motor boats, steamboats, rowing and sailing boats to be carried, as well as of the anchors and cables; the Director of the Signal Department as to the wireless telegraphy arrangements; the Director of Navigation as to the arrangements for navigating the ship, and so on. In this way the scheme goes on growing in a tentative manner, its progress always being dependent on the efficiency of different parts, until ultimately a more or less complete whole is arrived at in the shape of drawings and specifications provisionally embodying all the agreements. This really is the most difficult and interesting stage, for generally it becomes apparent at this point that requirements overlap, and that the best possible cannot be achieved in regard to numbers of points within the limit set to the contractors. These difficulties are cleared up by discussion at round-table conferences, where the compromises which will least impair the value of the ship are agreed upon, and the completed design is then finally submitted for the Board's approval. Some fourteen departments are concerned in the settlement of the final detailed arrangements."

The main particular in which this account differs from our description of the architectural process is in the more elaborate provision, in the ship design process, for coordinating the numerous criteria of design that are evoked during the process, and preventing criteria, once evoked, from being overlooked in the latter stages of the process. In the ship design process too the "evoking" takes on an organizational form—it involves consulting the relevant specialists. The long-term memory here is literally a distributed memory, divided among the various groups of experts who are involved at one or another stage of the design process.

What is striking about Sir Oswyn's account is how well structured each part of the design process appears. We can visualize each group of experts, provided with the overall specifications, sitting down with their specific subproblem of designing a particular system, and finding that subproblem to be a relatively well structured task. Of course the more complex subproblems may themselves be in numerous ways ill structured until further subdivided into components (cf. the descriptions of the role of the Engineer-in-Chief and of the Director of Naval Ordnance).

Wherever complex designs are produced by organizations, and where, as a consequence, the design process must be partially externalized and formalized, we find descriptions of that process not unlike this description of ship design. An initial stage of laying down general (and tentative) specifications is followed by stages in which experts are called up ("evoked") to introduce new design criteria and component designs to satisfy them. At a later stage, there is attention to inconsistencies of the component designs, and a search for modifications that will continue to meet most of the criteria, or decisions to sacrifice certain criteria in favor of others. Each small phase of the activity appears to be quite well structured, but the overall process meets none of the criteria we set down for WSPs.

3.4. An intelligent robot

A different aspect of structure comes to the forefront when we consider the design of an intelligent robot capable of locomoting and solving problems in a real external environment. The robot's planning and problem solving must be carried out in terms of some internal representation of the external environment. But this internal representation will be inexact for at least two reasons: First, it must abstract from much (or most) of the detail of the actual physical environment. (It surely cannot represent the individual molecules and their interactions, and it must almost always ignore details that are much grosser and more important than those at the molecular level.) Second, the internal representation includes a representation of the changes that will be produced in the external environment by various actions upon it. But for this prediction of the effects of operators to be exact would require an exact knowledge of the laws of nature that govern the effects of real actions upon a real environment.

The robot, therefore, will continually be confronted with new information from the environment: features of the environment which have become relevant to its behavior but are omitted from, or distorted in, its internal representation of that environment; and changes in the environment as a consequence of its behavior that are different from the changes that were predicted by the planning and problem solving processes.

But this external information can be used by the robot in exactly the way that information evoked from long-term memory was used by the architect. The problem representation can be revised continually to take account of the information—of the real situation—so that the problem solver is faced at each moment with a well structured problem, but one that changes from moment to moment.[2]

If the continuing alteration of the problem representation is short-term and reversible, we generally call it "adaptation" or "feedback", if the alteration is more or less permanent (e.g., revising the "laws of nature"), we refer to it as "learning". Thus the robot modifies its problem representation temporarily by attending in turn to selected features of the environment; it modifies it more permanently by changing its conceptions of the structure of the external environment or the laws that govern it.

[2] "Robots" that operate on synthesized worlds represented inside the computer, like the well-known system of T. Winograd, are not robots in the sense in which I am using the term here; for they do not face the issue that is critical to a robot when dealing with a real external environment—the issue of continually revising its internal representation of the problem situation to conform to the facts of the world. Thus the information given PLANNER, the problem solving component of Winograd's system, is a complete and accurate characterization of the toy world of blocks that the system manipulates. The accuracy of the information guarantees in turn that any theorem proved by PLANNER will be true of the block world.

3.5. Chess playing as an ISP

We can now return to the task environment of chess, and reinterpret our earlier comments—that in some respects chess appears to resemble an ISP rather than a WSP. To make the point still sharper, we consider a (oversimplified) chess program that has three principal components:

(1) a set of productions,
(2) an evaluator,
(3) an updating process.

The set of productions serves as a move generator. Each production consists of a condition part and an action part; the condition part tests for the presence of some configuration of pieces or other features in a chess position; the action part evokes a move that should be considered when the condition with which it is associated is present. Such a set of productions can also be viewed as an indexed memory, or discrimination and recognition net. When the presence of a condition is recognized by the net, the corresponding action is accessed in long-term memory. By the operation of this system of productions, each chess position evokes a set of moves that should be considered in that position.

The second component of the chess program is an evaluator that takes a move and a position as input and (recursively) makes a dynamic evaluation of that move. We will assume—without specifying exact mechanisms—that the dynamic evaluation converges. The evaluator produces a search tree that constitutes its prediction of the consequences that might follow on any given move.

The third component of the chess program is the updating routine. After a move has been made and the opponent has replied, the updater brings the position on the board up to date by recording the moves, prunes the analysis tree, and turns control over again to the discrimination net.

Consider now a problem space consisting of the set of features recognizable by the productions, together with the moves associated with those features. Consider the subspace consisting of the features actually evoked by a given position, together with the moves associated with this smaller set of features. If we regard the latter, and smaller, space as the effective problem space for the program (since its processing during a limited period of time will be governed only by the productions actually evoked during that time), then the effective problem space will undergo continuing change throughout the course of the game, moving from one subspace to another of the large space defined by the entire contents of long-term memory. The problem faced by the chess program will appear just as ill structured as the architect's problem or the robot's problem—and for exactly the same reasons.

3.6. Serial processors and ISPs

Our analysis has led us to see that any problem solving process will appear ill structured if the problem solver is a machine that has access to a very large long-term memory (an effectively infinite memory) of potentially relevant information, and/or access to a very large external memory that provides information about the actual real-world consequences of problem-solving actions. "Large" is defined relative to the amount of information that can direct or affect the behavior of the processor over any short period of time; while "potentially relevant" means that any small part of this information may be evoked at some time during the problem solving process by recognition of some feature in the current problem state (the information available *directly* to the processor).

If we refer back to the original definition of WSP, we see that the present view of the boundary between WSPs and ISPs derives from our insistence that notions of computability in principle be replaced by notions of practicable computability in the definition of WSP. But this shift in boundary has highly important consequences. It implies that, from the standpoint of the problem solver, any problem with a large base of potentially relevant knowledge may appear to be an ill structured problem; and that the problem solver can be effective in dealing with it only if it has capabilities for dealing with ISPs. Conversely, it suggests that there may be nothing other than the size of the knowledge base to distinguish ISPs from WSPs, and that general problem solving mechanisms that have shown themselves to be efficacious for handling large, albeit *apparently* well structured domains should be extendable to ill structured domains without any need for introducing qualitatively new components.

However well structured the problem space in which a problem solver operates, if it is to be capable of modifying that space as problem solving progresses, it must possess means for assimilating the information it acquires from long-term memory, from problem instructions, and from the external environment. The next section discusses briefly the nature of the capabilities of these kinds that are required.

4. Assimilating New Information

When the problem space remains unchanged throughout the problem solving process, the assimilation of new information by the problem solving system offers no particular difficulties. Any information that can be used belongs to one of the forms of information that are specified in the definition of the problem space. In fact, the only new information that can be acquired is information descriptive of new states that are reached in the course of the problem solving search.

When the problem space is subject to modification during problem

solving, provision must be made for accepting and assimilating information from one or more of three sources: information evoked from long-term memory, information contained in problem instructions or additions and modifications to instructions, and information obtained through sensory channels from the external world.

4.1. Information from long-term memory

Information evoked by the production–recognition system from long-term memory should not create particular difficulties of assimilation. The forms in which such information is stored, the processes for retrieving it and incorporating it in the redefined problem space are all part of the problem solving system, broadly construed.

This does not mean that we have had much actual experience in constructing and using such semantic information stores. Perhaps, when we come to have such experience, difficulties will emerge that cannot now be anticipated. Nevertheless, the designer of the problem solving system controls the format in which information is to be stored in long-term memory. The storage scheme can put that information into a relatively simple, general, and homogeneous format (e.g., a network of description list structures—commonly referred to as a colored directed graph); and the system can be provided with relatively simple, general, and homogeneous processes for searching for information and retrieving it from the store. Interfacing with an environment whose design is under our control (memory search) is always several orders of magnitude easier than interfacing with a given external environment (perception).

4.2. Information from instructions

Tasks presented to a problem solver through natural language instructions pose the difficult initial problem of understanding the instructions, that is, of generating from them a well structured (or ill structured) statement of the problem. A general discussion of understanding natural language would take us far afield from our main concerns here, and must be omitted from the present paper. The reader is referred to Simon [14], Siklóssy and Simon [12], chapters by Bobrow and Raphael in [4], and Winograd [15].

Hayes and Simon [2] have recently constructed a system that reads problem instructions in natural language, and constructs from them a problem representation in the form of input appropriate for a problem solving system like GPS. Their program carries out the major steps in translating relatively simple ISPs into WSPs.

4.3. Information about the external world

An ability to assimilate information about the external world—either information about the effects of the problem solver's actions, or information

about autonomous changes in that world, or both—is the earmark of those problem solvers we have called "robots". Here we are concerned with the problem solver's perceptual or pattern recognizing capabilities—another topic that falls outside the scope of the present paper. The outer limits on acquisition of such information are defined by the primitive sensory discriminations of which the problem solver is capable, but in the short run, the higher level concepts already developed and stored may pose the most severe problems of assimilating new information.

In the cases both of information from instructions and information about the external world, the acquisition process involves a continual interaction between the incoming raw data and programs and data already stored in the problem solver. This interaction is both a vital aid to, and a limitation upon, the understanding process. It is an aid because it fits the new information to formats and structures that are already available, and adapts it to processes for manipulating those structures. It is a limitation, because it tends to mold all new information to the paradigms that are already available. The problem solver never perceives the *Ding an sich*, but only the external stimulus filtered through its own preconceptions. Hence the acquisition process exercises a strong influence in the direction of conserving existing problem formulations. The world as perceived is better structured than the raw world outside.

5. Implications for Artificial Intelligence

Molière's hero discovered that he had been speaking prose all his life without knowing it. Our analysis here implies that throughout the history of artificial intelligence, computer problem solving programs have, also unknowingly, been handling many aspects of problem solving that are usually regarded as ill structured. Several examples of programs now fifteen years old can be cited.

The early NSS chess program [8] contained a number of independent move generators, each associated with a particular subgoal (development, center control, King safety, etc.). A move generator was activated by the presence in the chess position of features relevant to the goal in question. When evoked, the generator proposed one or more moves for advancing the goal in question. These moves were evaluated by dynamic analysis which again was sensitive to features noticed in new positions as they arose during the analysis. Hence the over-all organization of the program was quite close to that of the hypothetical program we described earlier.

The proposals for using a planning method in the General Problem Solver can also be interpreted as a way of handling problems that are not completely well structured. Planning was done by abstracting from the detail of a problem space, and carrying out preliminary problem solving in the abstracted (and consequently simpler) space. But the plan then had to be tested by trying to

carry it out in the original problem space. The detail of the plan was thereby elaborated and its feasibility tested. The relation between the abstract planning space and the original problem space was quite analogous to the relation we have discussed between an internal problem space of a robot and the external real world in which the robot performs.

What has been absent from (or at least not prominent in) schemes like these is a very large long-term memory of potentially evocable information that can be used to bring about repeated modifications in the problem space. The recent growth of interest in semantics, in the design and construction of semantic nets, in the interpretation of instructions, and in interfacing robots with the external world are all movements in the direction of enriching our arsenal of artificial intelligence methods along the dimensions that will become increasingly important as we move toward more comprehensive schemes for handling ill structured problem solving. Our analysis gives us reason to be optimistic that progress will not require us to introduce mechanisms that are qualitatively different from the ones already introduced in artificial intelligence schemes—mechanisms with which we have already had some limited experience.

The boundary between well structured and ill structured problem solving is indeed a vague and fluid boundary. There appears to be no reason to suppose that concepts as yet uninvented and unknown stand between us and the fuller exploration of those problem domains that are most obviously and visibly ill structured.

REFERENCES

1. Alexander, C. *Notes on the Synthesis of Form.* Harvard Univ. Press, Cambridge, Mass. 1964.
2. Hayes, J. R. and Simon, H. A. Understanding written problem instructions. *Proc. 9th Carnegie Symp. on Cognition*, L. W. Gregg (ed.), forthcoming 1974.
3. Kling, R. D. A paradigm for reasoning by analogy. *Proc. 2nd Intern. Joint Conf. on Artificial Intelligence*, British Computer Society, London, 1971, pp. 568–585.
4. Minsky, M. (ed.) *Semantic Information Processing.* M.I.T. Press, Cambridge, Mass., 1969.
5. Murray, Sir Oswyn A. R. The administration of a fighting service. *J. Public Admin.* 1 (July 1923), 216–217.
6. Newell, A. Heuristic programming: ill structured problems. *Progress in Operations Research*, Vol. 3, J. Aronofsky (ed.), Wiley, New York, 1969, pp. 360–414.
7. Newell, A. and Simon, H. A. *Human Problem Solving.* Prentice-Hall, Englewood Cliffs, N.J., 1972.
8. Newell, A., Shaw, J. C. and Simon, H. A., Chess-playing programs and the problem of complexity. *IBM J. Res. Develop.* 2 (1958), 320–335.
9. Pitrat, J. Réalisation des programmes de démonstration des théorèmes utilisant des méthodes heuristiques. Doctoral Thesis, Faculty of Sci., Univ. of Paris, 1966.
10. Reitman, W. R. Heuristic decision procedures, open constraints, and the structure of

ill-defined problems. *Human Judgments and Optimality*, M. W. Shelley and G. L. Bryan (eds.), Wiley, New York, 1964, pp. 282–315.
11. Reitman, W. R. *Cognition and Thought*. Wiley, New York, 1965.
12. Siklóssy, L. and Simon, H. A. Some semantic methods for language processing. *Representation and Meaning*, H. A. Simon and L. Siklóssy (eds.), Prentice-Hall, Englewood Cliffs, N.J., 1972.
13. Simon, H. A. Style in design. *Proc. 2nd Ann. Environ. Design Res. Assoc. Conf.*, 1–10, October 1970, J. Archea and C. Eastman (eds.), Carnegie-Mellon Univ., Pittsburgh, Pa., 1971.
14. Simon, H. A. The heuristic compiler. *Representation and Meaning*, H. A. Simon and L. Siklóssy (eds.), Prentice-Hall, Englewood Cliffs, N.J., 1972.
15. Winograd, T. Understanding natural language. *Cognitive Psychol.* 3 (1) (1973), 1–191.

Received 18 July 1973; revised version received 29 October 1973