BOLT BERANEK AND NEWMAN INC

CONSULTING • DEVELOPMENT • RESEARCH

Job No. 11597

"ROBOT" COMPUTER PROBLEM SOLVING SYSTEM

Quarterly Progress Report No. 3

For the period 23 February 1972 to 22 May 1972

Contract No. NASW-2236

Joseph D. Becker
Principal Investigator

15 June 1972

Job No. 11597

"ROBOT" COMPUTER PROBLEM SOLVING SYSTEM

Quarterly Progress Report No. 3

For the period 23 February 1972 to 22 May 1972

Contract No. NASW-2236

Joseph D. Becker
Principal Investigator

15 June 1972

TABLE OF CONTENTS

I. INTRODUCTION

The following is a report on progress made during the third quarter of work on NASA Contract No. NASW-2236, whose subject is the development of a "robot" computer problem solving system. Our entire effort in this period was devoted to a single theoretical investigation, concerning the general properties of behavioral systems. The motivation and results of this investigation will be described below. To summarize, we did not obtain the sort of results that we might have hoped for in the best case, but we did gain a great deal of insight into the problems that are confronting us, and we would certainly regard our theoretical probing as worthwhile at this early stage of the project.

A. Motivation

Our immediate motivation arose out of direct experience in programming the robot simulation. As we mentioned in the previous Progress Report, we often found it possible to organize a behavioral routine in several quite different ways, and we had no theoretical basis on which to decide such choices.

We were further motivated by the fact that certain organizational properties recur with remarkable consistency in large behavioral systems (computer programs), even in different systems with very diverse purposes. For example, feedback control of an external condition is found in many programs, as are means for coping with the attendant problems of waiting for the condition to become satisfactory, of interrupting some higher process if the condition goes out of bounds, and so on.

Other examples include hierarchical organization of processes, predictive decision-making in the face of uncertainty, and priority scheduling in cases of competition for limited resources.

These considerations led us to undertake the investigation of complex behavioral systems in general. Our grandiose ideal hope was that we might uncover some comprehensive theory, along with an attendant notation, which would allow us to describe and design behavioral systems at will, much as the calculus allows us to describe and design various physical systems. Certainly we did not evolve such a general theory, and indeed we now see some deep reasons why it may not be possible to create one at all. Still, we have learned a great deal about ways of organizing behavioral systems, and about the circumstances under which a given organization is appropriate.

B. The Relativity of Behavioral Description

Traditional mathematics arose from the attempt to describe the physical world. In the last two centuries, man has learned that this descriptive tool can be turned around and used to *design* new physical systems to suit particular needs. In the case of computer programming languages, the story is the opposite: These languages were created in order to enable the design of computational algorithms, and only later have they been applied to the description of natural systems. This is certainly a case in which the solution (computer languages) supplied the problem (describing behavioral systems). Unfortunately, this order of events has led to the common assumption that computer languages do in fact solve the problem,

that is that they are an adequate mathematics for describing behavioral systems. There may be some difficulties with this assumption.

Traditional mathematics will allow an engineer to analyze a design for a new electronic circuit (say), but it usually will not lead him to a *good* design. For this he must rely on experience, inspiration, or at best on a much more complex sort of mathematical computation. Thus, in the traditional case, design is basically an *optimizing* process, while description is not. We are beginning to believe that in the case of behavioral systems, this is not true; rather the problem of describing a behavioral system is also inherently one of finding a *good* description, so that describing behavioral systems is just as much an optimizing process as designing them.

To see how this might be true, consider a desk calculator, of the sort that is 100% mechanical. A set of blueprints for such a machine might tell us all that we could possibly know about its structure; they might even allow us to build a working copy of the device ourselves, so that in some sense they constitute a complete description of the calculator. But certainly the blueprints fail in some fundamental way to tell us *how* the machine works, and what it *does*, for they do not describe any of the *operations* that it performs. Here is the apparent paradox: Even a *complete* description of the structure of a system may fail to be a description of its behavior. But what is there left to describe?

What is left is a set of criteria which exist in the mind of the (human) *recipient* of the description. Suppose we undertake to describe how the calculator divides. To one man we must say: "You punch in A, press the Divide button, punch in B, and then press the Equals button." To another man we must say that

it divides by repeated subtraction. To a third man we must give a long rigmarole about which ratchets turn which shafts. In short, there is no answer, no description of "how the machine divides", except in terms of the questions that the description is intended to answer. It is in this sense that a description is not a description unless it is a "good" one.

Now, no doubt this sort of relativity holds trivially for any descriptive system, but the mere existence of traditional mathematics proves the existence of broadly agreed-upon, and therefore implicit, description criteria for physical systems. Apparently such implicit agreement is lacking when it comes to behavioral systems, with the consequence that there is no canonical description of such a system, and hence no simple "mathematics" in the traditional sense of a symbolic system which can be applied descriptively in a straightforward way.

We might remark that the fact that there is no single "correct" or "true" description of the behavior of a complex system does not, of course, mean that there is no true substrate to the behavior. The desk calculator clanks away unconcernedly, leaving *us* to puzzle out behavioral notions such as "the process of division".

We have belabored this section because we feel that it dominates the rest of our discussion. Indeed, if the intuitions expressed here are correct, then it may never be possible to find the sort of calculus of behavioral organization that we set out in search of. Still, we believe that there are fruitful ways of proceeding, as we will describe in the next section.

C. A Piecemeal Approach

Failing in our grander motivation, we should retreat to our secondary motivation, namely the observed commonality of organizational devices in widely differing behavioral computer systems. Concepts like "interrupt", "backtracking", "executive", and so on are known to be important, and they will not disappear on us like the notion of a general calculus of behavioral organization. Therefore, as a first step we have set out to examine such concepts piecemeal - that is, without any attempt at synthesis. By concentrating on these concepts, we can gain useful insights into important behavioral mechanisms, and at the same time we can slowly flush out the underlying relationships among various aspects of behavioral organization.

II. ASPECTS OF THE ORGANIZATION OF BEHAVIOR

A. Hierarchical Organization of Processes

Any behavior that we observe must unfold linearly with time; why then should we describe or design a behavioral system in terms of a hierarchy of processes? Why do we not represent every system simply as a linear sequence of actions? The reason, evidently, is that we are able to see significant recurring patterns in a linear sequence of events, and we attribute the appearance of similar sub-sequences to the presence of a single "sub-process". That is, we form the concepts of individual sub-processes, such as "squaring a number" or "grasping an object", by *induction over time*, in precisely the same manner that we form object-concepts such as "dog" or "sunset". The nesting of process-concepts gives us the same sort of hierarchy that we have in the case of object-concepts, where "collie" is a sub-concept of "dog", which is a sub-concept of "mammal".

Because of our experience with hierarchically *structured* systems (e.g. computer programs and human management structures), we tend to think of hierarchical *behavioral* organization as being similarly "real", i.e. part of the mechanism that actually generates the behavior. This need not necessarily be the case. For example, we can take any activity, such as "grasping an object", and break it down into further ones, such as "opening the hand", "orienting the hand", "moving the hand to the object", etc.; but this analysis does not mean that grasping *actually* proceeds in phases. The activity *could* be entirely preprogrammed and integrated, or it could be organized in some very different way. (Recall the example of the desk calculator: Its "behavior" is not the same thing as its "mechanism".)

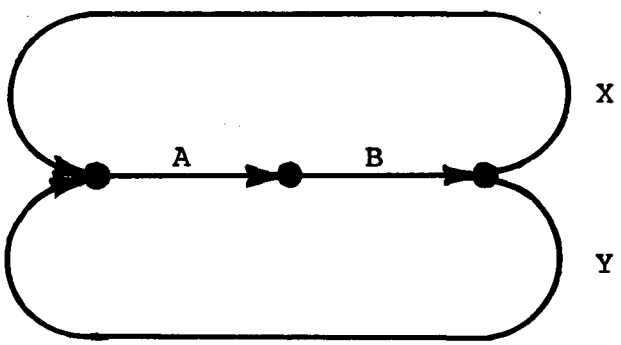
Thus, the act of temporal induction, and hence the description in terms of a hierarchy, come from *us*, and not necessarily from the system that we are describing. This relativity implies that the level of detail and the descriptive particulars in a hierarchical representation depend on the needs of the person performing the induction, and not on absolute properties of the behavior in question. This fact is familiar to any programmer, who must continually decide whether or not a sequence of actions is worth encapsulating as a closed subroutine.

B. Branch Points and Information

One of the major problems in induction is what to do with event sub-sequences which are similar but not identical. An important solution is the use of branch points to allow some elements of a sequence to be collapsed while others remain distinct. For example, suppose that a system has been observed to emit activities A, B, X, and Y, in the following sequence:

...A B X A B Y A B Y A B X A B Y A B X A B X A B X A B Y ...

We might well represent this system by a finite-state device:

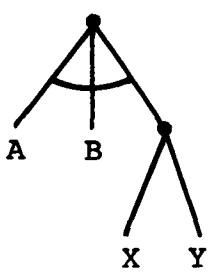


Here, the state after the emission of B constitutes a branch point, where the system "decides" whether to emit an X or a Y. This use of the word "decides" is critically important. It is a prime example of a behavioral imputation that need not correspond to any mechanism actually used by the system that we are describing. In other words, when our inductive analysis leads us to postulate a branch point, we also postulate a decision process.

Furthermore, we inevitably go on to ask *on what basis* the decision was made. We ask what *information* goes into determining the choice at the branch point. For example, our finite-state machine above becomes understandable if we assert that after emitting a B, it reads a symbol off of a tape; if the symbol is 1, it emits an X, if it is 0, the machine emits a Y. Thus, we identify the influence of *information* with (apparent) *choice*. This is, of course, a fundamental intuition of formal information theory; we see here that it is just as fundamental in understanding the organization of behavior.

Sometimes it is the apparent seeking of information that leads us to postulate a branch point, rather than the other way around. For example, when a cat carefully scans a ledge before jumping onto it, we assume that he is deciding precisely how he can execute the jump, if at all.

Although the postulation of branch points does not force a hierarchical organization (as the finite-state representation demonstrates), the two are very importantly related. One simple way of seeing this is to think of a behavioral "parsing tree" such as the following, for the sequence on the previous page:



(The arc indicates an "AND" node; the lower node is an "OR" node.) It is extremely convenient to imagine that there is some entity, some decision process, associated at each branch point, and that this entity "supervises" the activities that are found below it. In the case of an "OR" branching, the supervisor of course makes the decision of which branch should be taken. In the case of an "AND" branching, the supervisor at least decides when one phase should end and the next commence (which is sometimes a non-trivial problem in complex systems like our robot).

We suspect that such postulated decision processes or supervisors are the essence of behavioral representation. Certainly our remaining sections will all revolve about this concept.

C. Spheres of Influence

Once we have postulated a hierarchy of supervisors, it is natural to think of them in terms of the managerial structure of a human organization. While there are a number of inadequacies to this metaphor, it can be quite instructive. We think of a human supervisor as having certain "sphere of influence." This includes the agents "below" him whose work he controls, and the administrators above him who specify and evaluate his own work. It is important to note that the supervisor's world, that is, his sources of information, are *local*, being restricted to the nearby realms above and below him. Of course, there is no precise definition of "local"; what is important is that some information is harder for the supervisor to come by than other information.

To give an important example, let us consider the case of a man sitting in his living room watching t.v. who suddenly

desires a can of beer. At some peripherally conscious level, he realizes that he must get up, go into the kitchen, and open the refrigerator in order to get a can of beer. In order to get up, he calls upon a skilled activity involving placing both feet on the floor, bending forward at the waist, placing his hands on the arms of the chair, etc. In order to place a foot on the floor, perhaps specific neural circuits are used, containing internal feedback loops to ensure smooth control of the muscles. Now, what interests us is that the near-conscious supervisor has not the slightest idea of how the muscles are moved, while the muscular circuits have not the slightest idea of the desirability of beer. (By a valid analogy, a corporation president and a laborer for the corporation have no idea of each other's tasks.) Putting this in terms of information and decisions, we can say that the near-conscious planner is not capable of making any decisions on the basis of signals from individual muscles, and the muscular control circuits are not capable of making any choices based on needs or knowledge involving beer.

Many of the hardest problems in designing the robot control system arise from precisely such disjoint spheres of influence. At one level the robot decides to look at a particular building, but the eye was already being moved in the other direction for a different reason, and besides the building in question is too far behind the robot to be seen any more. Such problems of coordination are basic to any behavioral system which is sufficiently ramified to contain supervisors with non-intersecting spheres of influence. We will return to the matter of coordination after examining one more fundamental notion.

D. Goals

Perhaps the most tenuous concept involved in the description of behavior is that of "goal". Even more than the other notions that we have discussed, the idea of a "goal" is clearly a descriptive artifact. The desk calculator clunks along perfectly well with no goals driving any of its gears or pinions. We have found no single answer to the question of the proper role for the concept of goals, but we are beginning to have some ideas as to where it fits into the scheme of things.

If we consider our hierarchy of supervisors or executives, we realize that the *administrative* tasks performed by these entities (tasks such as keeping track of which subordinates are doing what) are distinct from the overall task of the system. That is, the manager of a steel mill pushes papers, but his ultimate responsibility is to produce steel. We may suggest that the notion of "goal" arises precisely when we have such a separation between an ultimate responsibility and the administrative work required to meet that responsibility. In straightforward behaving systems, where there is no such separation, we do not need to postulate goals. For example, the engine of an automobile drives the wheels, period -- we do not need to say that it has the goal of driving the wheels.

Of course, the designer of the automobile had the goal of making the wheels go around, which is why he supplied the car with an engine. For this reason, it does not sound nonsensical to say that the goal of the engine is to drive the wheels, but in saying so we are merely including the *human* into the system that we are describing. This would be made clearer by a careful linguistic distinction: We should say that the *purpose* of

the engine is to drive the wheels; of itself, the engine has no goals.

To take an example at the opposite end of the spectrum, suppose that a man decides to discover a cure for cancer by next February. Here we have the ultimate separation between the end product of a system and the procedure for obtaining it, namely there is no known procedure for obtaining it. In this case, the only useful description of the man's behavior is in terms of a goal.

We see, then, that the notion of goal is a function of the way in which a behavior is described. We should be very careful about postulating goals as a *mechanism* of the behavior itself. This comment applies specifically to the new goal-oriented programming languages, and to some of our own programming on the robot simulation.

It is common to talk of goals in terms of *states*. Even in terms of the cancer example, such a notion seems artificial: the man's goal is to *do something*, namely discover a cure, not to be in the state of having discovered a cure. Also, we may think of organisms whose behavior is commonly described in terms of tropisms: the worm's goal is to move toward water, away from light. Here we may salvage the notion of state by speaking in terms of gradients, but we should be aware that we are embalming time- or space- derivatives in what is supposedly a static description. Thus, it is unduly restrictive to think of goals only in terms of states.

Goals, too, are things that are desirable. What does this mean? Perhaps it means that what a system wants, or what it wants to do, defines its goals? A certain amount of programming experience or philosophical reflection will show that such an analysis is tautological. We must admit that what a system *does*

is *identical with* our (often *post hoc*) imputation of goals. However, this identity does not render the concept of "desirability" meaningless. We suspect that this concept can be usefully related to that of *expenditure of resources*. Suppose that on a Sunday a man has to choose between going fishing or mowing the lawn. We observe him to be packing up his fishing gear. We therefore say that he has selected the goal of doing some fishing, this being *(ipso facto)* the more desirable alternative. If it had been possible for the man to do both activities at the same time, the description in terms of goals would have been much less useful. Therefore, ultimately the notion of goal brings us right back to the notion of branching, of decision.

E. Resource Conflicts

As the foregoing discussion indicated, there is a close connection between decisions and limitations of resources. If a system had unlimited resources of all sorts, it would still have to make decisions involving coordination (see the next two sections), but many of its organizational problems would disappear. This is strikingly clear in the case of our robot simulation, where much of the subtlety arises from the fact that the robot is capable of entertaining many simultaneous hypotheses about the world, but it must check them out serially because of the focal nature of visual attention. (This is not to imply that focal attention is informationally inefficient; on the contrary, it is rich in informational benefits, but these come at a high organizational cost to the system that employs focal mechanisms.)

The resource limitation which is most familiar to computer programmers is that of "processing power", i.e. the enforced serial nature of most of our machines. When a process has "AND-ed" subprocesses, we tend to think of them as sequential steps; when

a process has "OR-ed" subprocesses, we worry about the order in which they should be tried until one of them succeeds. It is important to note that these primary concerns of the programmer are in fact artifacts of serial processing in our computers (and perhaps of serial analysis in our conscious thought). In human managerial systems, and in biological nervous systems, there is ample opportunity for simultaneous activity among processes at the same level. In such cases, the notions of "AND-ed" or "OR'ed" subprocesses merge into each other, and we must find new bases for describing the activity of the supervisor. The next two sections will suggest some principles that may be useful.

An important fact about resource conflict is that it may cut across the sphere-of-influence boundaries of individual local supervisors. For example, in our robot simulation, no matter what is the hierarchical relationship of various processes that may wish to move the eye, there is only one eye, and all must compete for it. It follows that the entity which allocates such a resource cannot have *its* sphere of influence confined to any sub-locality; therefore, it must become a *global* decision-maker. This seems to us to be an extraordinarily powerful conclusion. It seems to mean that a system, no matter how homogeneous its elements (e.g. a nervous system), cannot have a homogeneous *behavioral* structure if contains conflict over resources. There must be some mechanism which allows the attainment and enforcement of a global decision as to the allocation of the resource. We might even suggest that, according to this argument, the appearance of a unitary "mind" is unavoidable (albeit at the level of behavioral description) in any system with a high ratio of potential behaviors to bodily resources.

behavioral description) in any system with a high ratio of potential behaviors to bodily resources.

F. Condition Conflicts

More general than resource conflicts are the problems that arise when two supervisors of independent subprocesses have incompatible requirements as to the state of the world. To air condition your house, the windows must be closed; to ventilate it, they must be open; therefore you cannot do both at once.

In an algorithmically-behaving system, especially a sequential one, the initial design of the system assures in advance that the preconditions for a given subprocess will be met at the time that the process is called for. The more potent, the more adaptive a system becomes, the more its organization must explicitly cope with the meeting of preconditions before a subprocess can be unleashed. Perhaps the ultimate of such organization is a collection of independent "demons", which are subprocesses that themselves actively "monitor" their preconditions, and autonomously commence their activity as soon as their conditions are met. This "pandemonium" organization is powerful because of its inherent parallelism, but in most cases it must be combined with some sort of executive mechanism which will provide the requisite administrative (global) control. In order to see how such hybrid organizations function, we must gain an understanding of some of the more basic elements of the condition conflict problem.

We often think of "conditions" in terms of predicates which are either true or false. There are a number of reasons why this conception is inadequate. Many conditions (such as spatial position) take on a range of values, which may well be continuous. In many cases it is worthwhile to consider both the value of some measurable quantity (e.g. intensity of a stimulus) and its

time-derivative; this complicates the specification of a "condition" involving such a quantity. Often in real systems, the value of a condition can be obtained only to some degree of certainty less than 1.0 ; in such cases there must be a balance between the overhead of ascertaining the condition and the chance of making an erroneous decision. Even worse is the problem of the possible variation in a condition over time. That is, the system cannot afford to monitor all conditions at all times, but conditions may have changed in the interval since they were last observed (with some conditions being more likely to change than others). This latter has come to be known as the "frame problem"; clearly it must be solved in terms of "expected truth values", rather than with binary true-false predicates.

These kinds of problems are compounded whenever the system takes any overt action, because then it produces some not-wholly-predictable change in the world. In general, the possibility that any one subprocess will change the preconditions for any other (either favorably or unfavorably) can be computed only in terms of expected probability, since a system has only a partial knowledge of the world, and only limited time to spend predicting the consequences of its actions. Of course, it is precisely this sort of uncertainty which underlies the importance of *sensory feedback*. If you want to know whether or not your elbow is resting in your coffee cup, don't figure it out -- take a look. Or, even better, have "passive" sensors which can *interrupt* an action if it results in the placement of your elbow in the coffee.

The notion of interrupt relates back to the idea of a "demon" silently watching until a certain condition is met, but it further implies the power of one subprocess to halt or at least influence another. Once this vital concept is allowed, our intuitive ability to comprehend the control organization of a complex behavioral system goes from poor to abysmal. This is

just the point at which we would like to have a workable mathematical representation, but at the moment we must be content with an informal examination of the concepts that such a mathematics must represent.

G. Temporal Organization

The problems of condition conflict can be looked at from a temporal as well as from a logical point of view. In a sequential system, each subprocess is invoked only when the previous one is complete, at the behest of the administrating superprocess. In a pandemonium system, the temporal interaction is more complex, with the demons "waiting" in some kind of limbo status until they get an opportunity to perform, perhaps interrupting some other demons in the process. In all of this there is still one element lacking: What sets the pace, what determines the global temporal organization of events? This can be made into a fairly deep question.

In many computer programs, the question of pace is totally irrelevant. For example, suppose we are given the mathematical relation $X = (Y + 2*Z)^2$. This relation is inherently atemporal. Now consider a sequential program for computing X in terms of Y and Z:

```
(1)  X ← Z
(2)  X ← 2*X
(3)  X ← Y + X
(4)  X ← X*X
```

It does not matter how fast this program is run. All that matters is that the steps be performed in order; this is what determines the equivalence between the program and the formula.

The situation is of course entirely different for a system that must interact with the real world. If any one subprocess has a temporal extension, then the others must be placed in some temporal relationship to it. We can think of several ways of

achieving such temporal coordination, each with its advantages and disadvantages. It is possible to define a global time-scale, "clock time", against which all activities are mapped out. It is possible to specify events in *relative* time; e.g., B happens five seconds after A, but C is temporally independent. It is possible to control a process in terms of the *rate* at which it proceeds. And it is possible to regard time as one of the preconditions to the commencement or branching of a process: e.g. one subprocess could take a certain branch if another subprocess had run for such-and-such a period, or if the clock time were such-and-such. No one of these devices is adequate for all purposes, and certainly all are used in effecting the time-coordination of human affairs.

We feel that time is less understood relative to its importance than any other aspect of behavioral organization. This is especially true in regard to *simultaneous* processes, which are just beginning to receive formal study. For example, the notion of *monitoring*, and of the *supervision* of one process by another are most clearly exemplified when the supervisor and the supervisee are functioning at the same time. Clearly this and similar concepts are crucial to the organization of process control.

H. Executive Bookkeeping

One of the functions of a supervisor is to keep track of what is going on among its subordinate processes. In current programming systems, subprocesses are usually run sequentially, they terminate of their own accord, and their success or failure is evaluated only after they terminate. Even in so straightforward a case, the supervisor may require considerable bookkeeping in order to keep track of what has and what has not been done. The problem grows very complicated if the supervisor is

to gain anything from attempts which fail. There is the problem of computing which portion of the acquired hard-knock experience was a function of the particular approach that was tried, and which experience is relevant to any further approach that might be tried. Ultimately, this is a form of the frame problem, solvable only by estimation.

The notions of "success" and "failure" should be treated gingerly, since we would like to distinguish between goals which are explicit to the supervisory process, versus those which are implicit in the organization of the system (e.g., in our little program for computing $(Y + 2*Z)^2$, all goals are implicit, and the supervisor has only to make sure that the steps are executed one after the other, since they automatically "succeed" and "terminate"). Of course, it is even harder to define when a process is "succeeding" or "failing", in terms of a measure of *progress*, yet this must be done in any system where processes cannot be expected to terminate themselves automatically (e.g. the search for an item in a large memory store).

The notion of "backtracking" in case of a failure is subject to complexities, even in case failure is well-defined, and even disregarding the problem of learning something from the failure. It presupposes that there is in fact a record somewhere of what was being tried (this is not automatically the case in a pandemonium-like system). Also, the problem of diagnosing where to place responsibility for the failure may be effectively insoluable in cases where the chain of command passes through several disjoint spheres of influence. For example, if our beer-seeking t.v. watcher finds that he cannot move his foot (perhaps it is asleep), the analysis of the situation and corrective action must be made at a very much higher level than that at which the failure actually occurred. Thus, recovering from a failure may be a challenging exercise both in bookkeeping and decision-making finesse.

I. Executive Decision-Making

Given that the supervisor can keep track of what its subordinates are doing, in most systems it must allocate "processing power" or some other resource to them on a merit basis. Presumably the most meritorious course of action is that which will produce the best or most results with the lowest expenditure of resource (including time). Of course, the question is how the supervisor is to know ahead of time, in a non-algorithm-like system, how to estimate the effectiveness and expense of the various alternatives that are presented to it. It is tricky to define how a supervisor can predict or estimate the behavior of a subprocess without of course carrying out the actual execution of the subprocess.

We should also mention that the very generation of alternative subprocesses may be a task that consumes non-negligible resources. For example, if the robot (or an animal) is confronted with a visual scene, it must match that scene with long-term memory in order to draw out hypotheses by which it may recognize parts of the scene. This match-search of memory is a major part of the recognition process, and the system obviously cannot afford to draw all possible hypotheses out of memory before testing any of them. Thus, part of the executive responsibility is to generate new potential subprocesses in a manner which is efficient, as well as efficiently managing the subprocesses which have already been proposed.

This sort of executive decision-making is perhaps the crux of efficient behavior. At the same time, it is relatively simple conceptually (if stated as a choice among alternatives), and relatively well-studied by traditional means (e.g. statistical decision theory). Therefore we will go no further into the mechanisms of decision-making here, since our object is to consider the structure of the behavioral system as a whole. And while it

might be relatively simple to enumerate the criteria for any individual decision, it is usually not so simple to specify how such decisions should interact, how supervisors should coordinate and decide priority among themselves, what spheres of influence should be open to each supervisor, and so forth.

J. Deciding the Overall Organization: Statistical Information

The question of overall organization, as we asserted in the introduction, is one of *optimizing* with respect to certain goals, whether one is describing a given behavioral system or designing a new one. Thus, in many cases the finding of a *good* formalization is substantially a different problem from the finding of any one that will work at all. The optimization must take into account the system's behavior over a large class of similar inputs; that is, it is essentially an inductive process. For example, suppose that you are introduced to a person, and that he reacts moodily to your attempts to talk about football. From this one event, you have no idea whether the problem is that he hates football, or that he hates introductions, or that he was having a bad day, or that he is generally a surly person. These possibilities can be distinguished only by observing him in a number of similar situations. It is easy to see that the same sort of procedure is necessary for arriving at the proper description of any complex behavioral system.

We would like to emphasize that the information gathered in such experimentation with a behavioral system is *statistical* in nature, and that therefore the selection of an optimum model of a behavioral system is closely connected to the statistics of its responses to typical inputs. This fact has been implicit in everything we have said about alternative organizations of systems. For example, if subprocess B is *always* both desirable and possible after the execution of subprocess A, then the best

organization is to make them sequential steps under some larger process. If the applicability of B depends on some particular set of conditions, it might be best to provide a test on those conditions, with the execution of B being dependent on this test. If B is only rarely applicable, or if the circumstances of its applicability are not readily predictable from tests, it might be best to establish B as a "demon" which independently waits and watches for its opportunity to proceed.

Thus, the proper organization of a particular behavior is entirely dependent on the particular statistical peculiarities of the task at hand. This is true of the global organization, and of the details of control throughout the system. Furthermore, there are some problems, such as the handling of the "frame problem" mentioned in Section II.F, which have solutions *only* in terms of a statistical conformity of the system to its informational environment. Perhaps this ubiquitous influence of the statistical properties of the task is the most important general principle that can be stated about the organization of behavioral systems.

III. FUTURE DIRECTIONS

The discussions that we have set forth in the previous section of course fall far short of a general theory of behavioral organization, but we believe that they do shed light on some of the important issues which such a theory should encompass. While the results of our research are not yet unified, they still serve to deepen our understanding of many problems that arise in the design of our robot simulation system. They also allow us to relate the robot system intelligently to progress made in the creation of other large systems, such as speech understanding, air traffic control, biomedical simulation, and computer network systems. It was the potential for constructive interaction among these projects that provided much of the initial motivation for our theoretical investigations.

At present, we feel that we have pushed theory as far as it can go without further practice. Therefore, in the next period we will return to the goals of establishing particular behaviors in our robot simulation, as described in the previous Progress Reports. In particular, we will be directing our attention to the problem of a proper internal representation for the robot's experience, such that the robot can use this representation in order to recognize where it is by reference to the visual scenery.