REASONING, PROBLEM SOLVING AND DECISION PROCESSES: THE PROBLEM SPACE AS A FUNDAMENTAL CATEGORY

by

Allen Newell*

DRC-15-06-80

*Department of Computer Science
Carnegie-Mellon University
Pittsburgh, PA 15213

To be published in R. Nickerson (ed.) Attention and Performance VIII, Erlbaum (forthcoming).

This research was sponsored in part by the Palo Alto Research Center of Xerox and in part by the Defense Advanced Research Projects Agency (DOD) ARPA Order No. 3597, monitored by the Air Force Avionics Laboratory under Contract F33615-78-C-1551.

The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the US Government.

ABSTRACT

The notion of a problem space is well known in the area of problem solving research, both in cognitive psychology and artificial intelligence. The *Problem Space Hypothesis* is enunciated that the scope of problem spaces is to be extended to all symbolic cognitive activity. The paper is devoted to explaining the nature of this hypothesis and describing some of its potential implications, with no attempt at a critical marshalling of the evidince pro and con. Two examples are used, one a typical problem solving activity (the Tower of Hanoi) and the other syllogistic reasoning. The latter is an example where the search behavior typical of problem spaces is not clearly in evidence, so it provides a useful area to explore the extension of the concept. A focal issue used in the paper is the origin of the numerous flow diagrams that serve as theories of how subjects behave in tasks in the psychological laboratory. On the Problem Space Hypothesis these flow diagrams derive from the interaction of the task environment and the problem space.

Table of Contents

REASONING, PROBLEM SOLVING AND DECISION PROCESSES: THE PROBLEM SPACE AS A FUNDAMENTAL CATEGORY[1]

1. INTRODUCTION

I am concerned with human goal-oriented cognition: what humans do when they bring to bear what they know to attain some end. 1 take my title from the session in which I have been invited to give an opening presentation, because it exhibits a particular feature of cognitive psychology's current state I can use as a starting point.

Substantial areas of psychological study exist in *reasoning* (Falmagne, 1975, Revlin *&* Mayer, 1978, Wason *&* Johnson-Laird, 1972); *problem solving* (Greeno, 1977, Newell *&* Simon, 1972); and *decision processes* (Slovic, Fischhoff, *&* Lichtenstein, 1977). Yet, in looking at any of these fields it is hard to detect the existence of the others. These three areas must in most ways be the same: problematic situations presented verbally to be dealt with by cognition and decision. Indeed, we often use the same term in all three areas. We "decide" which of two probabilistic bets to take (much studied in the area of decision processes); "decide" what chess move to make (much studied in the area of problem solving); and "decide" which conclusion follows from a syllogism (much studied in the area of reasoning). Similar semantic games could be played with the other terms. However, studies of each of these areas rarely give more than lip service to the results and theories from the others. What should be a unified scientific endeavor seems fragmented.

Multiple diagnoses for this fragmentation are possible. Deny the symptom. Argue, along with Voltaire's Dr. Pangloss, that it reflects a perfectly appropriate state for a developing science. Note that the areas have distinct intellectual roots and mutter that history explains (and excuses) all. Grasp the nettle and assert the three psychological domains to be genuinely distinct. However, I do not wish to argue the case in detail here. I do wish to assume that enough others share my unease to let me use this issue of fragmentation as a point of entry to what fundamental category of cognition might help resolve it.

Consider the tasks used by psychologists to study cognition. Traveling through them yields a distinct impression of just one damn task after another: Maier's two string problem, Edward's book-bag and poker chip task, R. Sternberg's analogies task, Restle's Tangled Tale sentence, Wertheimer's parallelograms, Revlin and company's classical syllogisms, Newell *&* Simon's cryptarithmetic. The world of all tasks seems like a zoo formed from a medieval bestiary — far from random, lots of odd structure, perhaps bias (too many toy tasks?), but without essential orderliness.

Repeat that trip, stopping at tasks where some theorizing has occurred. For much

I wish fo acknowledge Stu Card and Tom Moran who helped immensely in the development of (his theory, as we struggled together to find a way of constructing a useful taxonomy of tasks. To Kerb Simon, as always, must go credit for many of the underlying ideas.

of the best, we find a *flow diagram* presented, such as the one in Figure 1 (Revlin & Leirer, 1973), with control flowing between boxes that carry labels such as *encode, compare,* etc. These flow diagrams constitute the framework of the theory of each task.

Figure 1 is taken from the area of reasoning. Analogous diagrams t:ould be found in abundance in all three areas. I assume such diagrams are familiar enough not to require further illustration. It is not uncommon to criticize flow diagrams, questioning their usefulness as a theoretical language; I have been kr,»:wn to do so myself (Newell, 1973). However, my question today is different. I am willing to accept that these flow diagrams acquire additional rules and data associated with the boxes, even equations and simulation programs. My question is: *Where* did those flow diagrams originally come from? They are different for every task, though they all surely bear a family resemblance. Theorists seem able simply to write dov/n a different theory for each task. Details get filled in experimentally, but the frameworks, ie, the flow diagrams, are just written down.

The diversity of these flow diagrams is connected to the fragmentation of cognitive studies. Diversity per se does not cause the trouble, for the tasks themselves are indeed diverse and the theories must reflect that. The difficulty lies in the emergence of each of the microtheories full blown from the theorist's pen. There is no way to relate them and thus they help ensure the division of the study of human cognition into qualitatively isolated areas.

The difficulties in finding communality do not rest just in giving common technical content to the terms in the boxes, such as *encode* and *compare%* though these surely pose problems. They emerge with each flow diagram, and equally without essential discipline. They seem weak vessels to try to shore up, though some are trying (Sternberg, 1979). But beyond the terms themselves the structure of the flow diagrams also embodies the theory, and must be dealt with as well.

The diversity of these flow diagrams arises in large part because these diagram-theories incorporate the detailed structure of each task within the very fiber of the theory. They are a version of the magician's trick — by the time the theory emerges, the scientific magic has already taken place. Though the theorist does not have a theory of how the subject would do a task, he himself can do what the subject does, ie, analyze the task; hence, he can create each theory separately out of his own subjective analysis.

1 have no quarrel with the theorist's direct analysis as a source of theoretical ideas. The difficulty stems from the volume of separate analyses that produce a corresponding volume of distinct flow diagrams. The prescription I take from this is the need to understand where these flow diagrams come from — not to understand where theorists get them, but to understand where subjects get them. Given that humans are cognitively integrated, how does this organization occur. Can we understand how the task gives rise to the flow diagram?

I propose a solution to this in terms of *problem spaces,* a concept already familiar in the study of search in human problem solving (Erickson & Jones, 1978) and applied widely there (Simon & Lea, 1974). It was introduced formally in Newell & Simon (1972), but derives from extensive work in artificial intelligence, where all programs characterized as *heuristic search* provide prototypic examples of problem spaces. The central proposition of this paper is to extend the scope of this concept:

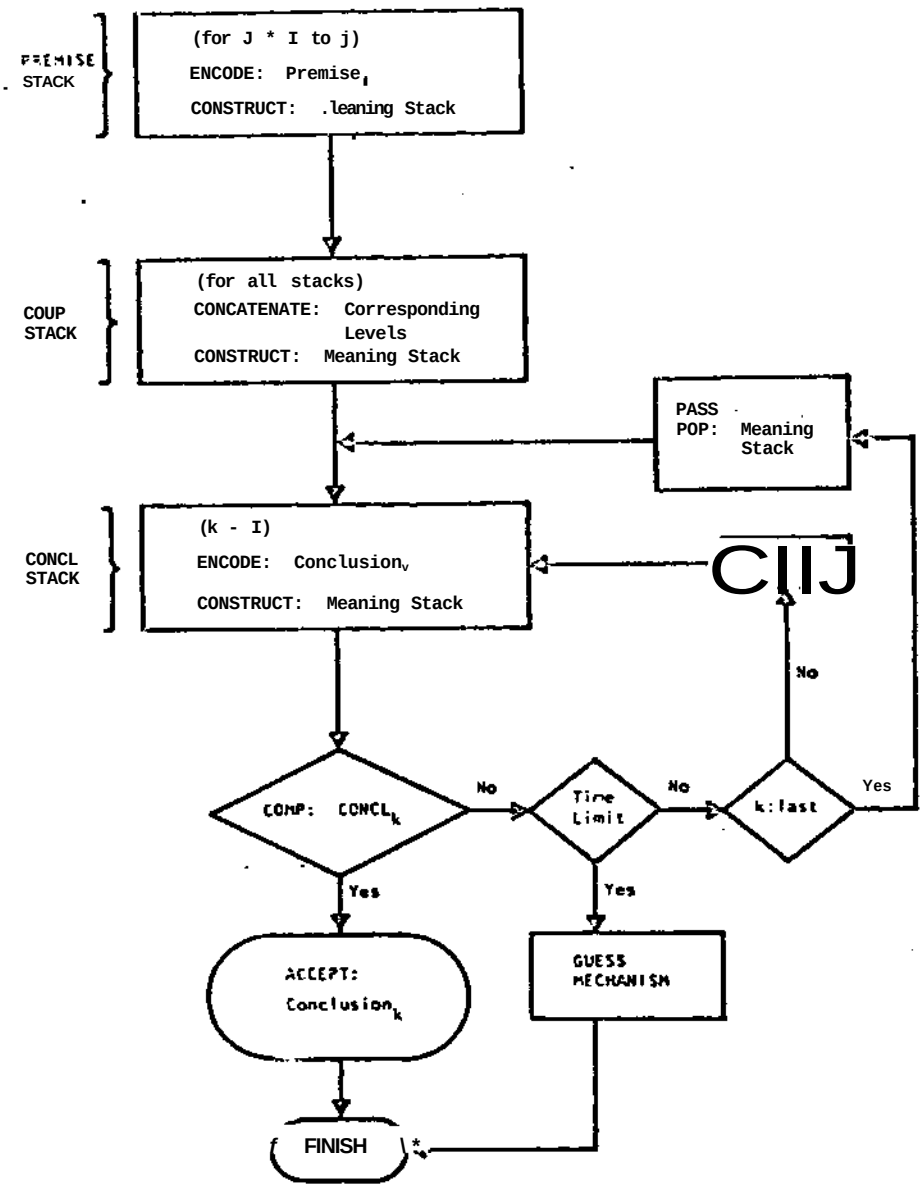


Figure 1: Typical flow diagram:

Conversion model of formal reasoning (Revlin & Leirer, 1978).

Problem Space Hypothesis: **The fundamental organizational unit of all human goal-oriented symbolic activity is the *problem space.***

In general terms this proposition is clear enough. There are things called problem spaces, which humans have or develop when they engage in goal-oriented activity. To understand such activity is to discover what problem spaces a human is using. From these flow the descriptions and predictions of interest, especially those concerned with how behavior is organized to accomplish tasks. The proposition is inclusive, claiming coverage of all symbolic goal-oriented activity; but hedges on whether all cognitive activity is symbolic. (The notion of *symbolic* (Newell & Simon, 1976) is ultimately central to the hypothesis, but doesn't enter explicitly into this paper.) It is an empirical hypothesis about the nature of human behavior. It is clearly of.more general import than just where flow diagrams come from; that issue is just an entry point. In the words of the title, the hypothesis claims the problem space to be a fundamental category of cognition.

This paper attempts to make this proposition intelligible. It does not present the case for it critically. There is no space for that. We will avoid formalism as much as possible to concentrate on the central notions. We lay out quickly and over-simply the notion of a problem space, using the Tower of Hanoi task as an example, where the notion has already been applied. Then we pick another example, syllogistic reasoning, to develop what the hypothesis means in areas other than problem solving. Again, space permits only one such example, though the hypothesis is intended much more broadly. However, this will provide enough of an illustration to discuss some general issues.

2. PROBLEM SPACES AND PROBLEMS

The Tov/er of Hanoi puzzle provides a convenient initial example to make the notion of a problem space concrete. It is normally considered to be a problem solving task (Nilsson, 1971, Simon, 1975) and has been analyzed in essentially problem space terms.

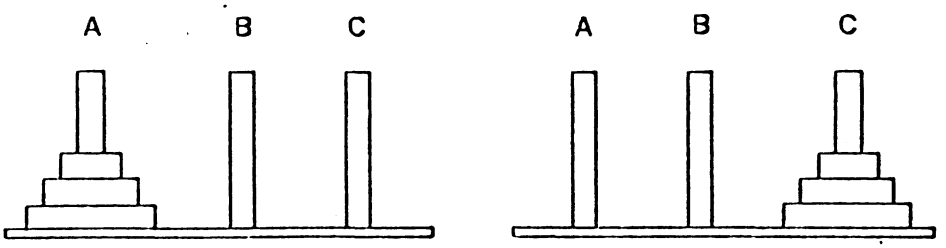
We start with informal definitions:

> *Problem Space:* A problem space consists of a set of symbolic structures (the *states* of the space) and a set of *operators* over the space. Each operator takes a state as input and produces a state as output, although there may be other inputs and outputs as well. The operators may be partial, ie, not defined for alt states. Sequences of operators define *paths* that thread thetr way through sequences of states.
>
> *Problem:* A problem in a problem space consists of a set of *Initial* states, a set of *goal* states, and a set of *path constraints.* The problem is to find a path through the space that starts at any initial stale, passes only along paths that satisfy the path constraints, and ends at any goal state.

A problem space is a set of symbolic structures within which to move around, an arena wherein many specific problems can be posed and attempted. A problem space and problem are mental constructs, ie, mental operators and states, though they, may lead to external actions. A subject *has* a problem space if he can mentally represent the states of the space and carry out the operations. He *has* a problem in a problem space if: (1) he has the space; (2) he can obtain representations of the initial states, recognize paths that satisfy the path constraints, and recognize the goal states; and (3) his behavior is controlled so as to attempt the problem in the space.

The task and one possible problem space for the Tower of Hanoi are given in Figure 2. The states are all the configurations of a fixed set of disks on three pegs. There are two operators. One takes an arbitrary disk and peg as input and moves the disk to the peg (a new state). The other produces a symbolic expression asserting that a given configuration fits a given pattern. Possible patterns include concrete configurations, such as the tv/o in the figure, and also patterns such as "peg P is empty" and "the disks are not in order on peg P^H. The problem statement specifically asserts this space to be the arena in which action takes place. It gives both initial and goal states explicitly. It imposes a path constraint; this is necessary because the space itself consists of *all* configurations of disks. It is not difficult to see that the Tower of Hanoi puzzle can be expressed in problem space form. Normal adults have the ability to create the space of configurations of Figure 2 and perform the moves and recognitions. Thus, they can *have* the space of Figure 2 in the manner called for — to be able to select and apply operators, test for results, etc.

Other problem spaces are possible for the Tower of Hanoi problem. Indeed, any problem space that contains this one works fine. For instance, the basic operators might



Task and Problem Statement

A board has three pegs, A, B and C (as shown at left). On Peg A are N disks of different sizes (3 in the diagram), in order with the largest at the bottom. The task is to get all the disks on Peg C in the same order (as shown at right). A disk may be moved from any peg to another, providing (1) that it is the top disk on its peg and (2) that it cannot be put on top of a disk smaller than itself.

Problem Space

States: Arbitrary configurations of the N disks on the three pegs.

Operators:

Move a disk by removing it from a peg and putting it on another peg.

Recognize a configuration as an instance of a pattern.

Problem

Initial state: The configuration shown in the diagram at left.

Goal state: The configuration shown in the diagram at right.

Path constraint: No disk may be placed on a smaller disk.

Figure 2: Tower of Hanoi: Problem Space

be pick-up a disk, move the hand, and deposit the disk. Then the move operator described as unitary in Figure 2 becomes composite. Additional path constraints become necessary: a disk can be set down only on a peg; and only one disk can be moved at a time. Without these, one hand could hold a disk in mid-flight while another is moved.

Smaller spaces may also be possible. The states could be limited to ordered stacks of blocks and the move operator could apply only where the disk on the receiving peg was larger than the disk being moved. This space incorporates the path constraint as part of the operator. Whether this problem space is possible for a given subject depends on whether the subject's behavior can become organized so that the test for legality of a move is reliably incorporated within the move itself. If a move must actually be considered and the result viewed to see if it is legal, then the subject can have the space of Figure 2 but cannot have this smaller space.

Each problem space provides a possible way to represent a task so as possibly to obtain a solution. It describes an ability of the subject to confine behavior to the problem space. It describes the units of behavior (ie, the operators) the subject v/ill use in working on a task.

Resource and capacity limits exist on processing. In problem spaces these take the form of two principles.

> *Serial Action:* At most one problem space operator can be performed at a time.

Thus, at any point in time the subject is located at a single *current state* to which a *current operator* is being applied to yield a *current result,* ie, a new state. Seriality specifically refers to this action. More than one problem or problem space may be active. Eg, to apply an operalor in one problem space may require passing into another problem space (concerned with the mechanics of the operator), thus working-in two problem spaces at once, though of course doing only one operator.

> *Finite Stock:* The subject has a limited set of states (the stock) available to become the current state.

This is a memory and access limitation. However, the stock is not identical with short term memory; some states may exist in long term memory or be available perceptually as external memory. The actual size of the stack is variable in the same way as the size of human short memory, depending on the complexity of the state, the external memories used, etc.

In the Tower of Hanoi problem these two principles mean that a subject can consider only one move at a time (in thought, not just in the external world) and, after having considered a move, there will be only a few new states possible: the new state resulting from the move; the state from which the move was just considered; the state that is set up in the external world (if that is different); the original initial state; possibly another remembered one.

Given a problem in a problem space, the only way a subject can solve the problem is by *searching* in the space: working out from the current state by applying operators, adding new states to the stock to be used at new points of search, evaluating whether the results help, etc. To accomplish this search requires performing repeatedly a fixed

set of functions:

Search Control: The following functions determine the behavior in a problem space while working on a given problem.

Decide to quit the problem.

Decide if a goal state has been produced.

Select a state from the stock to be the current state.

Select an operator to be the current operator.

Decide to save the new state just produced by an operator.

These functions operate within a cycle consisting of repeating the following three steps:

1. Select a state; Select an operator.

2. Apply operator to state, producing new state.

3. Decide if a goal state; Decide to quit; Decide to save the new state.

The subject has a mechanism (the *architecture)* for carrying out this control cycle. It is a fixed mechanism that works for all problem spaces: bringing the selected operator and state in contact, so the new state can be produced; bringing the decision processes in contact with this new state, so goal attainment can be determined; and so on. Its exact properties are important, including memory management for the limited stock and basic error detection, but we will not discuss the architecture further.'

More than one function can be performed at some steps: the two selections in step 1; and the three decisions in step 3. Within a step no process takes priority over any other. In what order the selection of an operator and of a state takes place depends on the content of the selection processes. Likewise, depending on the particular situation, the decisions could be made in any order.

Control over the search depends on the knowledge of these functions that the subject has *immediately available.* This restriction to immediate availability arises as follows. All knowledge about a task is obtained by taking steps in a problem space; that is what it means to be working in a problem space. Hence, choosing what step to take cannot itself be an extended deliberation that considers and reasons about the problem — that would involve taking steps in the problem space, contra assumption. What the control processes of selection and decision can involve is applying stored knowledge available in the subject's long term memory. But there is a limit to how much can be done without thinking a new thought about the task its^f, ie, without moving in the problem space.

The restriction implies that subjects cannot normally take too long to make a step, though no strict temporal limit exists. Subjects take steps in a problem space every few

seconds (Nev/ell *Si* Simon, 1972). Where the time per state is fairly long (10 - 15 sees.), cither the date is relatively complex, taking time to assimilate, or the operators take time to apply. Unlike the search control, the application of an operator need not be immediate, but can involve going into a different problem space. A typical example is applying an unfamiliar complex formal rule, as in algebra.

The restriction that search control involves only bringing to bear immediately available knowledge is important, since it limits the complexity of the search control process. It lets us think of the search control primarily in terms of the knowledge it embodies, rather Ihan the processes for making that knowledge effective.

A subject can attempt to solve a problem in a problem space with *any* body of search control knowledge: from none at all, yielding undirected search; to knov/ledge that completely specifies all choices correctly, yielding the solution directly. Thus, to have a problem space is already to have relevant v/ays of behaving in a task situation, though the larger the space and the less effective the control knowledge, the smaller the chance of success. For instance, if a subject has only the knowledge represented in Figure 2, ie, no special search control knowledge at all, the problem can still be attempted. A combinatorial search problem then occurs, which can be analyzed in terms of the branching factor of the search (three) and the depth to solution (2^{N} for N disks) (Nilsson, 1971).

Subjects of course are never this ignorant; they have some effective knowledge for controlling the search. Figure 3 gives some examples. In accordance with the remarks above, it is sufficient to express the search control just by the knowledge involved, without explicit characterization of the processing. KJ to K3 simply provide minimal knowledge for their respective control functions. K4 says to move forward with each step, it induces a depth-first strategy. K5 is the most obvious means-ends principle. The Tower of Hanoi is problematic just because K5 does not suffice, ie, disks cannot simply be moved to the desired peg. It is a puzzle just because K6, the obvious means-ends knowledge to avoid difficulties, is ineffective with more than two disks.

K7 - K9 reflect a general avoidance of duplicate and redundant activity. K7 cuts down the branching factor from three to two. K8 cuts it to one on every other move, giving an effective branching factor of 1A Finally, K9 is a slightly more penetrating formulation to avoid looping, reducing the branching to 1 (except at {he initial position). However, if K9 is to be implemented, the subject must remember the peg from which the smallest disk came, which is a minor augmentation of the problem space state. Thus, adding K7 - K9 completely determines the course of problem solving. It leaves open only the (fateful) choice on the very first move. Subjects do exist who proceed in exactly this

Decide to quit the problem

K1. Quit if succeed.

K2. Quit when told by experimenter.

Decide if %oal state has been produced

K3. A state is a goal state if it is the exact pattern desired.

Select a state from the stock to be the current state

K4. Make new state the current state.

Select an operator to be the current operator

K5. Move a disk to the peg specified by a goal state.

K6. Move an obstructing disk to another peg.

K7. Do not move back to just prior position.

K8. Do not move a disk twice in a row.

K9. Do not move smallest disk back to its just prior peg.

Decide to save a new state just produced by an operator

K10. Add newly produced state to the stock.

Figure 3: Tower of Hanoi: Search Control Knowledge

Various, means exist for organizing search control knowledge. One is to create new problems (ie, cubgoals), attempt them, and incorporate the results in the further search. For oimplicity we cac>t the basic functions of search control to be those required, taking the problem to be fixed. But the subject also selects problems and problem spaces. Though not requiring additional capabilities, except for memory managment, much additional *power* and complexity accrues thereby, namely, the goal hierarchy. For instance, organizations of subgoals and search control knowledge can be fashioned into *methods,* which coordinate the selection and actions in various useful ways. A number of these methods, which have shown up repeatedly in Artificial Intelligence investigations of problem solving and elsewhere, have acquired familiar names:

> *Generate and Test:* Generate in any way possible (eg, systematically or haphazardly) a sequence of candidate states, testing each for whether it is the desired state.
>
> *Heuristic Search:* Apply heuristics to reject possible operators from the current state and to reject newly produced states; remember the states with untried operators in some systematic way (different schemes yield search strategies, such as *depth first, breadth first, progressive deepening;* and *best first).*
>
> *Hill Climbing:* Generate and apply operators from the current state; select one that produces a state with an improved evaluation and move to it.
>
> *Means-ends Analysis:* Compare the current state with the desired state to detect any difference; use that difference to select an operator that reduces or eliminates it; otherwise proceed as in heuristic search.
>
> *Operator Subgoaling:* If an operator cannot be applied to a current state, set up a subgoal to find a state in which the operator can be applied; otherwise proceed as in heuristic search or means-ends analysis.
>
> *Flanning:* Abstract from the present state (by processing only selected information throughout) and proceed to solve the simplified problem; use what is remembered of the path as a guide to solving the unabstracted problem.

These are often called the *weak* methods, because they can be relevant when little is known about the task, though, when so evoked, they are not very powerful. However, they often suffice to solve a problem, if the space is not large or if extensive search is undertaken. These methods are simply organizations of search control knowledge, though it is beyond the scope of this paper to lay this out. Functions, such as evaluate (hill climbing) and detect differences (means-ends analysis) can be performed within search control if sufficiently simple, ie, sufficiently like a recognition. Complex versions would require search control to evoke subgoals to accomplish them in some problem space. The methods also are *open,* in that they do not specify completely all the functions and hence admit the addition of more search control knowledge, eg, about differences in means-ends analysis.

The first two methods, generate-and-test and heuristic search, occur automatically in n problem space (indeed, heuristic search gave rise to the notion of the problem sp^ce). They *MC* identified as distinct methods, because they are normally realized within control structures (standard programming languages) v/here they must be constructed and deliberately evoked, just like any other method. The other methods all require the task to have some additional characteristics (the weak conditions) which are known to the subject: hill climbing requires evaluation of states; mean-ends analysis requires differences; operator subgoaling requires the conditions of operator applicability to be symbolizable; etc.

Let us add a single item of control knowledge for the Tower of Hanoi:

K11. Get an obstructing disk on the other-peg.

K11 expresses operator subgoaling, since the construct of obstruction derives from inapplicable operators. This differs from K6 only in setting a subgoal (get) rather than selecting an actual operator *(move).* With K11 some problems can be solved without stringent knowledge about avoiding duplicate paths (K8 or K9). The architecture is assumed to manage the resulting stack of subgoals that builds up as the knowledge is used repeatedly and rescursively. The limits to subgoal management set limits to the effectiveness of solving Tower of Hanoi problems with just K11.

Subjects not only show problem solving behavior in the Tower of Hanoi, they ultimately show skilled behavior, proceeding to solve each puzzle in a direct way that takes time but does not exhibit search. Simon (1975) has provided an analysis of four distinct, complete strategies for skilled behavior in the Tower of Hanoi. Three of the strategies arise by adding more search control knowledge to what we have assembled so far. The *Goo l-recursion strategy* simply expands K11 to apply to pyramids of disks, rather than just disks; its key notion being the invention of the concept of pyramid by the subject. The *Simple perceptual strategy* expands K11, not by the pyramid, but by the less powerful notion of the largest obstructing disk (on the source-peg). The *Sophisticated perceptual strategy* adds to this latter an item that extends the notion of obstructions to the target-peg as well as the source-peg. (The fourth strategy, *move-pattern,* raises issues about mimicking external behavior sequences that lie outside our illustration.)

The problem space hypothesis asserts that skilled, routine behavior is organized within the problem space by the accumulation of search control knowledge. This is just what we see in these three statcgies. Though how the accumulation occurs is not given, the final result consists simply of the addition of search control knowledge. There is no fundamental difference between problem solving and routine behavior in control organization, except in completeness and adequacy of search corttrol knowledge.

Simon (1975) embodies the various strategies in a *production system^* a species of pattern-directed rule-oriented programming system (Newell & Simon, 1972). This can be seen as an operational realization of the search control knowledge, and in fact such systems are active candidates for the underlying architecture of human cognition (Newell, 1973; in press).

3. THE CATEGORICAL SYLLOGISM: A REASONING TASK

There is a long history of research into human performance on classical Aristotelian syllogisms. Its fascination arises from the basic clash in our civilization between rationality «wd irrationality, and its projection to the clash between logic and psychology. Should humans behave according to the dictates of formal logic? Do they have their own "psychologic"? Are they simply "irrational"? Formal syllogisms provide ample evidence that people in fact cannot reason very well. They make lots of mistakes in answering (say) "All A are B; some C are not B; are some A not C necessarily?" The tone for research was set many years ago by Woodworth *&* Sells (1935), whose hypothesis of an *atmosphere* effect describes one form of illogic (or at least superficiality) whereby subjects respond according to the affirmative or negative tenor of .the syllogism.

Current research is constructing information processing theories of how people perform syllogisms, decomposing the task into various stages (eg, encoding, comparing, responding) and finding experimental demonstrations of which stages seem responsible for various aspects of performance (Revlin &• Mayer, 1978, Falmagne, 1975). It forms a useful example for us, because, though research is active and current, there is little contact between its processing models and those in problem solving and decision making, as a glance at the two recent books just cited will show. Thus, we can see what analysis in terms of problem spaces might add.

Standard syllogistic reasoning tasks are formed from three terms (eg, A, B, C), combined into two assertions (the major and minor premise) involving pairs of terms (A and £3; B *and* C), from which some assertions (the conclusion) about the third pair (A and C) may or may not follow. Four logical assertions are possible from the two quantifiers (a*ll*, *some),* negation *(no/not)* and the copula of implication (arc):

Major Premise:	All A are B	
Minor Premise:	Some B are C	
----------	----------	
Conclusion:	All A are C	Which of the list follow?
	No A are C	
	Some A are C	
	Some A are not C	
	Nothing follows	

Both abstract syllogisms (above) and concrete ones (No Senators belong to the Harem Club; some sheiks belong to the Harem Club; therefore no sheiks are Senators) are used.

Current theory can be typified by the flow diagram shown earlier (Figure 1) from Revlin *&* Leircr (1978). Its stages permit focussing on whether the difficulty lies in encoding (how sentences are interpreted) or in inferencing; or whether the difficulty lies with the givens or the conclusion. Typical of recent work is the hypothesis (Revlin's conversion model) that subjects apply conversion operations that take "All A are B" into "All B are A", thus producing errors in the encoding stage; and the hypothesis (Erickson, 1978) that subjects solve the problem by constructing internal Venn diagrams, but fail to construct all possibilities.

The problem space hypothesis implies that subjects solve syllogistic reasoning tasks by working in a problem space. They have a representation for the possible states of knowledge about the syllogism plus operators for generating and manipulating that knowledge. If their search control knowledge is sufficient they will go directly to an answer; when faced with difficulties, they will search in this space, ie, try different manipulations in attempting to solve the problem. They do not have simply an algorithm or method, as in Figure 1. If such a flow diagram describes their behavior, then it is the resultant of the problem space and the task environment.

Let's look at one possible problem space for syllogistic reasoning.

Abstract Object Problem Space

Figure 4 shows a problem space based on positing objects that have the attributes, A, B, C mentioned in a syllogism. These objects are abstract: they can be either *necessary* or only *possible*; and they can have only some of the attributes mentioned in the syllogism. Thus, they represent various situations of partial knowledge, which the subject is to flesh out by moving in the space.

This space is a natural one. It corresponds to attempting to imagine the things that are being talked about in the syllogism, as in the following monologue about the syllogism: all A are B; no C are B; therefore necessarily some A are not C.

> "Ok, there are some things that have A and also have B.
> Though some things that don't have A could also exist, and they could have B or not.
>
> But things that have C cannot also have B.
> Though things that don't have C could have B or not.
>
> Now, does that force some things that have A to not have C?
>
> Well, those things that have A, have B ...
> and having B they can't have C.
>
> And, ok, there isn't any thing that has B that has C.
>
> So there are some things that surely have A and haven't C."

An example of an object in a state is [Possible A+ C-], which has attribute A, does not have attribute C, and is only known to possibly exist. That the attribute B does not occur means that no knowledge is yet available on it. Relevant partial knowledge may be acquired about whether a possible object is in fact necessary, namely, that an object with some of its attributes is already necessary. Those attributes for which it is still unknown are called *open*.

Consider the encoding operator E1, which produces a new state in the space by taking as input (along with the state) an externally given assertion and adding the indicated objects. From "All X are Y" is known: (1) some objects with attribute X necessarily exist and they also have Y (hence [Necessary X+ Y+]); (2) some objects possibly exist without attribute X and either with or without attribute Y (hence [Possible

States: Sot of abstract objects with various properties

[Possible A+ E3-] = Possible object with attribute A and without B
Non-occurring attributes are not yet examined
[Necessary C-] = Object without attribute C necessarily exists
[Possible A+ opcn-B-] = B- is an open attribute for the object
All possible objects are instances of objects already in the state

Encoding Operators

E1. All X are Y => [Necessary X+ Y+], [Possible X- Y+], [Possible X- Y-]

E2. No X are Y => [Necessary X+ Y-], [Possible X- Y+], [Possible X- Y-]

E3. Some X are Y => [Necessary X+ Y+], [Possible X+Y-],
[Possible X- Y+], [Possible X- Y-]

E4. Some X are not Y => [Necessary X+ Y-], [Possible X+ Y+],
[Possible X- Y+], [Possible X- Y-]

Production Operators

Q1. [Any X Y], [<\ny X Z] => [Possible X Y Z] if attributes agree, else nothing

02. [Necessary X Y], [Possible X Y Z] => [Possible X Y open-Z]

Q3. [Possible X Y open-Z], no [Any ok-X ok-Y op-Z] *> [Necessary X Y Z]
[Possible X Y Z] if find

Recognition Operators

FU. [Any X Y ...] is *an* instance of [Any X Y] if occurring attributes agree

Figure 4: Abstract Object Problem Space for Syllogisms

X- V+] and [Possible X- Y-]). El adds all three of these objects to produce the new state. The objrets in a *r,\rtc* represent *nil* the objects than can exist; hence, El expresses that [X+ Y-] is not possible simply by not adding it to the state. The encoding operators embody assumptions about language. For instance, subjects see [X+ Y+] as necessary, rather than just possible, because they do not make the logician's interpretation of quantification, which admits vacuous cases.

Operator Ql produces abstract objects with the combined attributes of two other objects (which are therefore instances of the input objects). Suppose one was [Any X+ Y+] and the other was [Any X+ Z-] (where *any* means either *necessary* or *possible);* Ql produces [Possible X+ Y+ Z-J. In words: If some things that have X have Y, and some things that have X don't have Z, then it's possible that some things that have X both have Y *and* don't have Z. Ql can be applied only if all the common attributes agree in sign. Eg, [X+ Z*-] and [X- W+] can't possibly refer to the same object, since it couldn't both have X and not have X. The result of Ql can never be known for sure to exist, so it is always only *possible:* Eg, in the above example with [X+ Y+ Z-], all the [X+ Y+]s might actually be [X+ Y+ Z+] and the [X+ Z+]s might be [X+ Z+ Y-], so no common object would in fact exist. The role of Ql is to generate candidate inferences.

Q2 is part of showing that a possible object is necessary. The grounds for such an inference comes from: If [X+ Y+] is necessary then for any additional attribute, Z, either [X+ Y+ Z+], [X+ Y+ Z-] or both must exist. Hence, if [X+ Y+ Z-] is not possible, then [X+ Y+ Z+] must be necessary. Q2 records the existence of a necessary object and identifies the attribute lo be used to make the inference. This is the *open* attribute — where it is not known whether objects with the complementary sense of the attribute can exist. There may be more than one way to identify an open attribute in a possible object, hence more than one way to infer that it is actually necessary.

Q3 finishes the job of inferring that a possible object is necessary. It searches the state for the complementary object, ie the one with the opposite sense of the open-attribute. This object must also be compatible with the other attributes of the object. In the figure we express this as ok-X: either the attribute agrees in sense with the one in the input object, or it is missing entirely (in which case it could be either sense). If such an object cannot be found, then the inference to necessity can be made; if the object can be found, then the the object remains only possible. For example, suppose Q2 had produced [Possible X+ Y- open-Z+], because [Necessary X+ Y-] was in the state. If *an* object such as [Possible X+ Y- Z-] occurred, no inference could be made that [Po[r]>f>ible X+ Y- open-Z+] was necessary; if no such object was around, then Q3 could produce [Necessary X+ Y- Z+]. [Possible Y- Z-] would also be sufficient to deny the inference, so there need not be an identity match. Note that Q3 depends on an object *not* existing in the state. By representing non-possibility by absence, this particular problem space is vulnerable to errors of omission.

Finally, besides operators that produce new objects, there must also be a recognition operator, RI, that sees that one object is an instance of another, eg, that [Possible X+ Y+] is an instance of [Possible X+].

Figure 5 shows how reasoning would go in this space. Starting with a state with no objects, the first three steps translate the problem into an internal state. Step 4 applies Ql to [Necessary A+ B+] and [Necessary B+ C-] to produce a possible inference, [Possible A+ B+ C-J. In step 5, Q2 uses [Necessary A+ 8+] to identify C- as open. This permits Q3 to make the inference to [Necessary A+ B+ C-], since nothing of the form [Necessary

ok- A^ ok-B* C*] exists. All the objects with C+ have B-, so don't apply to an object with B-K The final step is by RI, which detects that [Necessary A* B+ C-] is an instance of [Necessary A+ C-] in Ihe goal state.

Figure 5 gives a direct route to the solution. How could it be found? Though by no menus I,HPO, the space itself contains a number of paths, eg, QI can be applied to any pair of the objects (nine cases) and will be successful in four cases, each of which can be the target for upgrading to necessary. Elementary search control is in fact sufficient. Means-ends analysis ᶜayo to apply operators that produce necessary objects with A+ and C-. Thus QI i» selected with [Necessary A+ B+] or [Necessary O B-] for one of the inputs, say [Necessary A+ B+] to be concrete. Operator subgoaling obtains the other input, via B+, getting [Necessary C- B*]. (That is, the lack of an input is a failure of the operator to apply, which leads to a subgoal of getting the input.) Once [Possible A+ B+ C-] is in hand, moans-ends analysis selects Q2 to make it necessary. There is again a need for a second input, and operator subgoaling selects [Necessary A+ B+J. Q3 follows directly by the attempt to make the output necessary.

The point is not that this is terribly difficult problem solving — precisely the opposite. Whereas completely random behavior is not satisfactory (one can rattle around even in a space of less than ten elements), just a little elementary search control is enough to dictate reasonably focusscd behavior. No extra apparatus is required to create appropriate taok-dircctcd behavior. The elementary search control does not always completely dictate the path. In the present case, the initial path could have followed QI to [Necessary B+ C-] instead of [Necessary A+ B+]; it would have generated the alternative legitimate inference of no C are A, but would have lead to backtracking to make contact with some A *sro* not C.

What Problem Spaces say about Syllogistic Reasoning

We now can see what problem spaces suggest about the nature of syllogistic reasoning and how it should be studied.

Where do flow diagrams come from? Flow diagrams are essentially the trace of the operators under search control regimes. The path of Figure 5 is *cncode*$_t$ then *compare* (it lacks a *respond,* because we didn't bother to define a task output). Such a path, of course, is only a single path through the flow diagram. However, if we merge several such paths over a set of tasks, we will get the full blown flow diagram. The decision boxes represent the places where operator or state selection goes one way or another, for different tasks. Whether decision boxes and separate paths exist in the flow diagram depends on how much is aggregated under the terms *encode^ compare^* etc. In our example, *compare* comprises QI, Q2 and Q3.

Obvious flow diagrams (ie, obvious to us in their logic) are ones where, given the problem space, it requires only *obvious* control (eg, means-ends analysis) to determine the sequence of operators to solve the problem.

As the subject acquires skill in this task, there will occur both development of the problem space itself and growth of search control knowledge. Special features of the task will be recognized and incorporated, well beyond the elementary organizational schemes we have focused on here. They will have the character of unique domain-dependent knowledge, not general methods. What the hypothesis then prescribes is the general form that the theory will take, ie, an accumulation of search control

P1. All A arc B
P2. No C are B

C1. Some A are not C

1.	EKP1)	=>	II:	[Necessary A+ B+], [Possible A- B+]$_f$ [Possible A- B-]
2.	E2(P2)	=>	12:	[Necessary O B-], [Possible C- B+], [Possible C- B-], [Necessary A+ B+], [Necessary A- B+], [Possible A- B-]
3.	E4(C1)	=>	G:	[Necessary A+ C-]
4.	QKI2)	=>	13:	[Possible A-^ B+ C-], [Necessary C+ B-], _
5.	Q2(I3)	=>	14:	[Possible A+ B+ open-C-1 [Necessary C+ B-], _
6.	Q3(I4)	=>	15:	[Necessary A+ B+ C-], [Necessary C+ B-], .^
7.	R1(I4,G)	=>	16:	(This state is an instance of G), [Necessary A+ B+ C-]$_f$ _

Figure 5: Example of Solving in Abstract Object Space

knc vlrrlgc nivl predominantly a further specification of the more general methods.

Avoidance of the fixed-method fallacy. Much current work in syllogistic reasoning proceeds by positing a particular representation and/or method, the flow diagram of Revlin and Lcircr again being a good example, but not the only one. Erickson (1973) posits that all subjects encode into (multiple) Venn diagrams. These studies commit the *fired method fallacy,* which attributes to all subjects (on all occasions) the same method without any attempt to ascertain either that all subjects indeed follow the specified method or that the consequences derived are invariant over different possible methods. They take as indicative of the model's success that it generates results in reasonable agreement with &roup data on percent correct.

Many of these results, however, are not unique to the method, but flow from disparate methods defined for different problem spaces. To pick a central issue, whether errors occur during encoding or inference, the conversion hypothesis posits an encoding error that transforms "All A is B" to an internal code for "All B is A", proceeding afterwards with flawless logic. The model of Revlin and Leirer embodies this hypothesis and from its success these authors conclude for the "human's are logical" position. But in other problem spaces what is encoding conversion in the Revlin and Leirer model is either short term memory error and/or inference error. For instance, consider conversion in the Abstract Object problem space:

"All A are B" - [Necessary A+ B+], [Possible A- B-], [Possible A- B+]

"All B are A^H = [Necessary B+ A+], [Possible B- A-], [Possible B- A+]

The difference lies not in the focal object (where the knowledge is strongest) but in a secondary object. This latter could certainly be misconstructed by the encoding operator. However, it could also be transformed by short term memory errors during reasoning. Further, the role of these objects in reasoning is either to inhibit Q3 from concluding necessity, by being the complementary expression; or to be the input to Ql in producing an alternative that reduces an "all" inference to a "some" inference. In both, retrieval failure masquerades for encoding failure, since the result is that Ql or Q3 respectively doesn't obtain an input that does the job.

The point is not that the example problem space is right and Revlin & Letrer wrong. The point is that problem spaces imply that ranges of possible behaviors are there to be exploreci. They make clear that the same subject on repeated occasions will exhibit a *range* of behavior, even though working in the same space. They offer suggestions (which have not been exploited yet) of how to characterize ranges of variability by variation in search control knowledge. Positing a. single method or flow diagram provides a much more constrained basis to consider individual variations, and continually seduces into the fixed-method fallacy.

Extension and task variation. A problem space provides an arena for a class of problems (one of its essential notions). Whether the problem is one of making an inference, validating an inference, or choosing inferences by multiple choice, the tasks all occur in the same problem space by a change in goals, though they may evoke different search control knowledge. Extending the task, while keeping within the problem space, should still elicit appropriate behavior, with much of the original search control knowledge

still .· pplicable (though perhaps less effective). Eg, more entities can occur: "All A are B; No A are C; Some B arc D; All C are D; is it true necessarily that some A are not D?" Flow diagrams, by being an amalgam of task and control, have difficulty dealing with families of related lar.ks. Information processing models built to handle just the classical two-prcmisc syllogism require serious augmentation and modification for such extensions.

Multiple problem spaces. Several qualitatively distinct problem spaces are possible for reasoning about categorical syllogisms. We can get spaces corresponding to the first order predicate calculus, various concrete models of the syllogisms-(eg, *Venn* diagrams), relations between terms, and even natural language — all in addition to the abstract or prototypic object space used here for illustration. There are variations within a given type of space. For instance, a variant of the Abstract Object space is to admit explicit objects that are *not-possible,* rather than relying on the absence of objects in the state.

The general grounds of these different spaces are not necessarily foreign to existing work. Venn diagrams are used as the explicit basis for some models (Erickson, 1973). Johnson-Laird (1975, Johnson-Laird *&* Steedman, 1978) has developed a model based on a representation with similarities to the abstract objects; he creates multiple individuals corresponding to each abstract object. However, the point here is not that one problem space is right. Humans can reason about syllogisms in many different ways; and the same individual can use different problem spaces on different occasions, possibly even within the same experiment. The point is that reasoning is the (more or less) knowledgeable exploration of an area, not just the following of a given procedure.

4. DISCUSSION

We have used a single task domain to illustrate and make concrete v/hat the problem <:.pace hypothesis moons outside its original area of problem solving search. Lot us now state more generally some consequences that flow from the hypothesis.

Predicting the control structure for a task. The hypothesis implies that the problem space structure is common ground for all symbolic cognition. Analysis of a cognitive task involves first specifying the problem space and then specifying the search control knowledge used within that problem space. Elementary search control knowledge is also common, namely, subgoals, weak methods, obvious loop avoidance, obvious waste motion avoidance, etc. Thus the problem space, with its standard complement of control knowledge, provides the starting point in looking for specific theories for a task.

Under some conditions the problem space and elementary search control knowledge can be sufficient to obtain a basic theory of the subject's behavior in the task, ie, the structure corresponding to the flow diagrams. We saw this in the syllogistic reasoning task. This is what satisfies the initial goal of this paper to explain where the idiosyncratic, yet obvious, flow diagrams come from that grace so much of our current cognitive theorizing. They come from the problem space. They are obvious because the only knowledge used is elementary search control knowledge, which is "obvious" to all of us, as participating human beings. They are idiosyncratic to each task, because the structure of the tasks is idiosyncratic — a reflection of the oft-quoted parable of the ant whose complex path results from the contours of the sand grains, not from complex internal mechanisms (Simon, 1969).

Tv/o conditions at least permit the structure of behavior for a task to be obtained from just the general problem space structure. In one, the task is simple and transparent, relative to the cognitive ability of the subject. Then elementary search control knowledge suffices. This is the situation in the syllogistic reasoning tasks. In the other, the situation is novel, so that the subject does not have much prior knowledge and does not have time to extract much new specific knowledge. Then the subject must rely on elementary knowledge. This condition produces novice problem solving behavior.

Experience with a task leads to the growth of search control knowledge. Behavior increasingly reflects this knowledge, so that the mechanisms implied by the elementary search control knowledge no longer suffice for determining behavior. However, this additional knowledge slill serves the basic functions of control and the problem space remains the control framework within which behavior is generated. Furthermore, if the situation shifts outside the reach of the subject's specific knowledge, then again behavior is generated by more elementary search control knowledge — which is all the subject knows that is relevant.

Experience leads to the growth of new problem spaces, and not just to the growth of search control knowledge within a problem space. Dramatic change of space occurs, as in seeing that a verbal problem can be reformulated as an algebraic equation. But evolution of the space also occurs, with new data structures being added to the state and new operators to the repertoire. Such changes can occur concurrently with changes in search control, since they are not necessarily incompatible.

Is the hypothesis disprovable? The hypothesis as stated is empirical, asserting how humans process information and behave thereby. However, the flexibility of the concept (pick any states, any operators, any search control knowledge) suggests that for any behavior there might be some problem space (or several) that yield it. If so, the usual uncomfortableness at having gotten a tautology in tow might follow. There is a genuine issue here, though not one that is insurmountable.

The set of problem spaces is universal relative to its architecture. That is, there exists a problem space that produces any behavior stream compatible with the architecture. It is important that this be so, to meet the human's need to shape behavior in whatever way is necessary to meet the demands of an unpredictable environment. Moreover, this is important for whatever control structure might exist to guide human behavior, whether a problem space or some other kind. Conceivably, no universal control might be possible -- any control scheme would have some deficiency (ie, some way in which it restricts the possible behaviors of the subject), with different schemes having characteristic deficiencies. Then it would be possible, in principle, to determine subject's control schemes by observing these deficiencies. In fact, we know that universal computational schemes are possible -- that is what the theory of universal Turing machines is all about. Whatever one universal control scheme can do, another can imitate. Thus, if the question is posed right, in a technical sense one can never disprove the basic problem space hypothesis from performance (Anderson, 1976) -- nor disprove its competitors.

The issue is not insurmountable, because tests of control structure are not forced to consider only the qualitative (in the sense of time-independent) aspects of the behavior stream, which is where the universality is guaranteed. Even putting to one side physiological and evolutionary data as too remote, there is the timing of behavior, the acquisition and modification of the control structure, and errors -- to name just the more obvious sources of data that can penetrate the masquerade of one universal control system by another. However, the ability for all control schemes to produce flexible behavior to the point of mutual imitation still makes the issues of testing indirect, complex and difficult, to say the least.

Let us note briefly some consequences of the problem space hypothesis that make it potentially vulnerable to disproof.

1. In novel situations problem spaces must be created by the subject. Since the subject has no special knowledge, the space must be constructed in terms of the surface features of the external environment. Thus, there are places where it is possible to predict what the problem space will be. Work on novel tasks (with underlying structure isomorphic to the Tower of Hanoi) gives some evidence for this already (Hayes & Simon, 1976).

2. Many events can throw the subject off a straight-and-narrow path, wherein the search control knowledge was adequate, eg, external interruptions, memory errors, etc. Such events leave the subject engaged in a wider search in the problem space, using more elementary search control knowledge. These error and recovery behaviors should be *systematic*, in reflecting the

same space. That is, the subject's behavior when errors occur r.hould appear rule governed in ways appropriate to being in a particular problem space.

3. The subject's movement toward skilled behavior should be continuous, in that change occurs by accretion of search control Knowledge within a fixed space. Unfortunately, other forms of procedural learning may be sufficiently hard to distinguish so that testing the hypothesis in this fashion is especially difficult, eg, learning by successive invention of new methods (which surely occurs) and learning by compiling the experience obtained in the problem space into some other procedural language.

4. The problem space has strong implications for the transfer of skill. Indeed, the assertion about the universal availability of the elementary search control, eg, the weak methods, is an assertion about transfer. If a subject maps a task into an existing problem space, then the transfer of this knowledge to the new task is implied (as transformed by the encoding of the task, of course).

5. Last, but not least for this paper, is the use of the problem space hypothesis to predict flow diagrams. Failure to predict the actual control organizations, as derived in the usual way by informal analysis, would count against the hypothesis.

Banishment of (half) the homuncuLis. A major item on the agenda of cognitive psychology is to banish the homunculus, ie, the assumption of an intelligent agent (little man) residing cloewhere in the system, usually off stage, who does all the marvelous things that need to be done to actually generate the total behavior of the subject. It is the homunculus that actually performs the control processes in Atkinson & Shiffrin's (1963) famous memory model; who still does all the controlled processing, including determining the strategies, in the more recent proposal of Shiffrin & Schneider (1977); who makes all the confidence judgments; who analyzes all the payoff matrices and adjusts the behavior appropriately; who is renamed the "executive" in many models (clearly a promotion); who decides on and builds all those flow diagrams.

The universal organization in terms of problem spp.ces solves naif of the problem of the homunculus. It provides the top level executive organization that is used for all cognitive behrvior. At the top, it claims, is a search effort in terms of the operators of the current space. No intelligent agent is required to run this search, since it runs in any event, even v'ith no search control knowledge, and it runs automatically with whatever search control knowledge it happens to have.

The claim on the simplicity of the top-level executive is double barreled. For it will not do to tanish the homunculus from performance, only to have him show up arranging for the intelligent creation of problem spaces. But much of problem space creation is a symbolic cognitive tack, hence must be performed by the subject by means of a problem space (or so the hypothesis claims). But then the elementary character of the highest level space applies to this aspect of behavior as well.

In any event, the problem space only solves half the problem. The other half of

banishing Ihc homunculus resides with the architecture, which closes the gap betv/een the problem space and complete mechanism. The claim that the architecture is also nonintclligcnt, ie, not a homunculus, is also plausible, given that the architecture primarily carries out housekeeping functions; but discussion is outside the bounds of this paper.

5. CONCLUSION

We have laid out in this paper a proposal for a fundamental category of all symbolic cognitive activity — the problem space, wherein the human always is embedded in an *ensemble* of possible behaviors, to be realized by selective search. We have claimed that it provides grounds for understanding where all the diverse flow diagrams come from. Further, that it has the potential for helping to make cognition whole again. But, though important, these two claims simply provided an orientation for the presentation. The basic point is the hypothesis that this is the way human cognition is.

This paper has lived within many constraints. It has been limited to an exposition of the hypothesis, neither assembling the evidence pro and con, nor contrasting it with alternative control structures. It has been limited to exploring a single example that bears on extending the problem space from the domain of its initial formulation (problem solving) to other areas of cognition. The title contains both the terms *reasoning* and *decision processes.* Though taken from the session title, they are intended to indicate the breadth of the hypothesis, namely to all of cognition. There is indeed nothing special about the area of reasoning, nor about the categorical syllogism. Thus, it seems important to leave *decision processes* in the title, though it remains purely a promissory note.

Finally, the pnper has been limited to the central issue of performance in a problem space, and has not dealt with the acquisition of problem spaces or their dynamic modification. This perhaps laid too much stress on the role played by the search control knowledge, in contradistinction to the operators or the representational structure of the states. This seems justified because "of the unfamiliarity. of the control structure and the need to exhibit how it welds the problem space into a functioning unit. But the tilt should be made explicit.

Still, with all the limitations stated, enough has been said, I hope, to convey the potential of the hypothesis and to recommend it for your consideration. The presentation obviously forms an implicit argument for the hypothesis. But there should be little difficulty separating the general and illustrative sorts of evidence presented here from the detailed demonstrations that are still necessary.

6. REFERENCES

Anderson, J. R. Language, *Memory, and Thought.* Hillsdale, N.J.: Erlbaum 1976.

Anzai, Y. and Simon, K A. The theory of learning by doing. *Psychological Review,* 1979, 86, 124-140.

Atkinson, R. C. and Shiffrin, R. M. Human memory: A proposed system and its control processes. In *The Psychology of Learning and Motivation,* New York: Academic Press, 1968.

Erickson, J. R. and Jones, M. R. Thinking. *Annual Review of Psychology•*, 1973, 29, 61-90.

Erickson, J. R. Models of formal reasoning. In *Human Reasoning,* Washington, D.C.: Winston, 1978.

Falmagne, R. J. (Ed.). *Reasoning: Representation and Process.* Hillsdale, New Jersey: Erlbaum 1975.

Greeno, J. Nature of problem solving abilities. In Estes, W. K. (Ed.), *Handbook of Learning and Cognition,* New York: Wiley, 1977.

Hayes, J. R. and Simon, H.A. Understanding complex task instruction. In Klahr, D. (Ed.), *Cognition and Instruction,* Hillsdale, New Jersey: Erlbaum, 1976.

Johnson-Laird, P. N. & Steedman, M. The psychology of syllogisms. *Cognitive Psychology^* 1978, 10, 64-99.

Johnson-Laird, P. N. Models of deduction. In Falmagne, R. J. (Ed.), *Reasoning: Representation and Process,* Hillsdale, New Jersey: Erlbaum, 1975.

Newell, A. and Simon, H. A. *Human Problem Solving.* Englewood Cliffs: Prentice-Hall 1972.

Newell, A. and Simon, K A. Computer science as empirical inquiry: Symbols and search. *Communications of the ACM,* 1976, 19, 113-126.

Newell, A. Production systems: Models of control structures. In Chase, W. (Ed.), *Visual Information Processing,* New York: Academic, 1973.

Newell, A. Harpy, production systems and human cognition. In Cole, R. (Ed.), *Perception and Production of Fluent Speech,* Hillsdale, New Jersey: Erlbaum, 1979. (in press).

Nilsson, N. *Prv'olem-Solving Methods in Artificial Intelligence.* New York: McGraw-Hill 1971.

Revlin, R. and Leirer, V. 0. The effect of personal biases on syllogistic reasoning: Rational decisions from personalized representations. In *Human Reasoning,* Washington, D.C.: Winston, 1978.

Revlin, R. and Mayer, R. E. (Eds.). *Human Reasoning.* Washington, D.C.: Winston 1978.

Revlir., R. Two models of syllogistic reasoning: Feature selection and conversion. *Journal of Verbal Learning and Verbal Behavior,* 1975, 14, 130-195.

Shiffrin, R. M. and Schneider, W. Controlled and automatic human information processing: II. Perceptual learning, automatic attending and a general theory. *Psychological Review,* 1977,34, 127-190.

Simon, H. A. and Lea, G. Problem solving and rule induction: A unified view. In Gregg, L (Ed.), *Knovyledge and Cognition,* Hillsdale, New Jersey: Erlbaum, 1974.

Simon, H. A. *The Artificial Sciences.* Cambridge: MIT Press 1969.

Simon, H. A. The functional equivalence of problem solving skills. *Cognitive Psychology^* 1975,7,253-288.

Slovic, P., Fischhoff, B. and Uchtenstein, S. Behavioral decision theory, /lamia/ *Review of Psychology,* 1977, 28, 1-39.

Sternberg, R. The nature of mental abilities. *American Psychologist,* 1979, 34, 214-230.

VVason, P. C. and Johnson-Laird, P. N. *Psychology of Reasoning: Structure and Content.* Cambridge: Harvard University Press 1972.

Woodworth, R. S. and Sells, S. B. An atornsphere effect in formal syllogistic reasoning. *J. Experimental Psychology,* 1935, 18, 451-460.