DOCUMENT RESUME

ED 113 206	95	SE 019 816

AUTHOR Simon, Herbert A.
TITLE Learning with Understanding. Mathematics Education Report.
INSTITUTION ERIC Information Analysis Center for Science, Mathematics, and Environmental Education, Columbus, Ohio.
SPONS AGENCY National Inst. of Education (DHEW), Washington, D.C.; National Inst. of Mental Health (DHEW), Rockville, Md.
PUB DATE Jun 75
NOTE 30p.; Paper presented at the Annual Meeting of the American Educational Research Association (Washington, D.C., March 30-April 3, 1975) at the invitation of the Special Interest Group for Research on Mathematics Education
AVAILABLE FROM Information Reference Center (ERIC/SMEAC), Center for Science and Mathematics Education, The Ohio State University, 244 Arps Hall, Columbus, Ohio 43210 ($1.25)

EDRS PRICE MF-$0.76 HC-$1.95 Plus Postage
DESCRIPTORS Artificial Intelligence; Computers; *Information Processing; *Learning; *Mathematics Education; Memory; *Problem Solving; *Research; Thought Processes

ABSTRACT

In this paper, Simon describes contemporary information processing approaches to the study of learning and thinking, and discusses the relevance of these studies to the distinction between rote and meaningful learning. Before defining the basic terminology of information processing research, he provides a brief literature review, describing some of the results in the study of capacity and search times for short-term and long-term memory. He then describes the basic methods by which the computer simulates mental activity, and the "thinking-aloud" methods by which human problem solving and computer problem solving are compared in the laboratory. Illustrating his theses with discussion of six types of problems (Katona's matchstick problem, towers of Hanoi, geometry proofs, algebra word problems, understanding instructions, and chemical thermodynamics), Simon describes various approaches to problem solving, and weighs the relative merits of each in terms of memory load, generalizability, and transfer. He distinguishes between the notions of problem solutions as a sequence of states or a sequence of operations. Other issues discussed include the importance of the semantic content of a problem or class or problems and of the schemata available to the problem solver. (SD)

ED113206

MATHEMATICS EDUCATION REPORT

Learning With Understanding

by

Herbert A. Simon
Carnegie-Mellon University

SE 019 816

ERIC Information Analysis Center for
Science, Mathematics, and Environmental Education
The Ohio State University
1800 Cannon Drive
400 Lincoln Tower
Columbus, Ohio 43210

June 1975

2/3

Mathematics Education Reports

Mathematics Education Reports are being developed to disseminate information concerning mathematics education documents analyzed at the ERIC Information Analysis Center for Science, Mathematics, and Environmental Education. These reports fall into three broad categories. Research reviews summarize and analyze recent research in specific areas of mathematics education. Resource guides identify and analyze materials and references for use by mathematics teachers at all levels. Special bibliographies announce the availability of documents and review the literature in selected interest areas of mathematics education. Reports in each of these categories may also be targeted for specific sub-populations of the mathematics education community. Priorities for the development of future Mathematics Education Reports are established by the advisory board of the Center, in cooperation with the National Council of Teachers of Mathematics, the Special Interest Group for Research in Mathematics Education, the Conference Board of the Mathematical Sciences, and other professional groups in mathematics education. Individual comments on part Reports and suggestions for future Reports are always welcomed by the editor.

In 1975, as in previous years, the Special Interest Group for Research in Mathematics Education sponsored a presentation at the annual meeting of the American Educational Research Association. This publication is based on the presentation made by Professor Simon on 31 March 1975 at the AERA meeting in Washington, D.C.

Professor Simon describes how contemporary information processing approaches to thinking and learning are beginning to illuminate the rote-meaningful distinction in the way in which students learn. He presents some concrete examples of research that illustrate important characteristics and significant findings of the information processing approach. He also provides some background information on human information processing and on computer simulation that should help readers to understand the scope of the field better.

ERIC/SMEAC is pleased to make this publication available.

Marilyn N. Suydam
Editor

This publication was prepared pursuant to a contract with the National Institute of Education, U. S. Department of Health, Education and Welfare. Contractors undertaking such projects under Government sponsorship are encouraged to express freely their judgment in professional and technical matters. Points of view or opinions do not, therefore, necessarily represent official National Institute of Education position or policy.

LEARNING WITH UNDERSTANDING

Herbert A. Simon
Carnegie-Mellon University

Everyone who teaches becomes aware through his classroom experience that there are important differences between the student who has learned by rote and the student who has learned with understanding, or "meaningfully." There are differences in what these two kinds of students have learned, and there are consequent differences in what they can do with what they have learned. The teacher becomes alert to signals that indicate which kind of learning a student is achieving, and tries to develop techniques for transforming rote learning into meaningful learning.

While the distinction between rote and meaningful learning is part of the common-sense equipment of every teacher, it is an intuitive rather than a formal notion that has never been provided with a solid foundation in the form of a satisfactory psychological theory. Although Gestalt psychology paid considerable attention to the distinction, it succeeded only in labeling it and describing some of its manifestations, without providing a set of mechanisms and processes that could account for it in operational terms. Psychology in the S-R tradition, on the other hand, tended to ignore the distinction (sometimes even denying the usefulness of asking "what is learned?"), and hardly undertook to construct an explanation for a phenomenon it did not recognize.

From many other fields of human endeavor (medicine and engineering, for example), we have learned that practical knowledge without theory can carry us only a certain distance, even in practical affairs. Theories that explain underlying mechanisms give us powerful new tools and methods for use in our work. But then, I need hardly labor that point in a meeting of the American Educational Research Association. Belief in the relevance of theory to improved practice is one of the foundation stones on which this organization stands.

In my remarks today, I should like to describe how contemporary information-processing approaches to thinking and learning are beginning to illuminate the rote-meaningful distinction, enabling us to determine with some precision what the student has stored in memory as a result of a learning experience, what the consequences are of storing one thing rather than another, and how particular kinds of learning can be encouraged and others discouraged. Our new knowledge and precision in these matters is closely linked with our growing ability to write computer programs that describe and simulate in detail the processes that humans use to carry out complex cognitive tasks, and that describe how the knowledge and information used in these tasks is stored in human memory.

My discussion will center around some half dozen concrete examples of research, some carried out in our own laboratory and some done elsewhere, that illustrate important characteristics and significant findings of the information processing approach to these questions. These examples will show how a combination of standard experimental approaches, analysis of thinking-aloud protocols of ongoing thought processes, and computer simulation are beginning to give us a clear and detailed picture of what goes on in problem solving, and the variety of methods--rote and meaningful--that can be employed in a single, relatively simple, task.

Some of the examples that will be discussed involve the kinds of puzzles and trick problems that psychologists like to use as laboratory tasks, while others involve important school subjects. In particular, I have tried to select examples that would cast light on one of the central skills we try to teach in the mathematics-science parts of our curricula--the use of mathematics to model physical and other real-world phenomena, and through modeling, to understand them and predict their behavior.

The belief that mathematics should be learned with understanding rather than by rote was one of the main motivations leading to the development and introduction of the "new math" programs into primary and secondary schools. In the design of these new programs, it was frequently assumed that "understanding" mathematics was closely associated with proceeding rigorously and defining underlying concepts carefully. While considerable attention was devoted to the proper treatment of mathematical syntax, semantic considerations tended to enter mainly at the very basic levels (e.g., the definition of cardinal number). It is possible that the conception of "understanding" that has entered into the construction of these programs captures only part of the meaning of that term. In particular, it may be one thing for a professional mathematician, concerned mainly with the discovery and demonstration of new mathematical truths, to "understand" mathematics, and another thing for a scientist or applied mathematician, concerned mainly with using mathematical tools to discover and derive generalizations about empirical phenomena, to "understand" it. Some of the research I shall report at least suggests, without demonstrating conclusively, that such a dualistic conception of mathematical understanding has a genuine psychological basis.

Before beginning an analysis of specific examples, it will be useful to say a little more about what is involved in an information processing approach to these matters, and how a computer can be used to simulate cognitive processes. The first sections of this paper constitute an introduction to these topics.

Human Information Processing

When human beings are observed working on well-structured problems that are difficult for them, their behaviors reveal certain broad characteristics of the underlying information processing system that supports the problem-solving processes; but at the same time, the behaviors conceal almost all the detail of that system. As a result, we can describe the system, for our purposes, in rather broad terms.

The basic characteristics of the human information processing system (IPS) that shape its problem solving efforts are these: The system operates mostly serially, one process at a time, not in parallel fashion. (A partial exception must be made for recognition processes, which are probably carried out in parallel.) Its elementary processes take tens or hundreds of milliseconds. The inputs and outputs of these processes are held in a small short-term memory with a capacity of only a few symbols. The system has access to a practically infinite long-term memory, but the time required to store a symbol in that memory is of the order of seconds or tens of seconds. Access to long-term memory is obtained by recognizing stimuli, the recognizer serving as a sort of "index," or by associating from one item in memory to another.

The evidence that the human system has the properties just listed comes partly from research on complex cognitive tasks. No problem-solving behavior has been observed in the laboratory that seems interpretable in terms of simultaneous rapid search of disjoint parts of the solver's problem space. On the contrary, the solver always appears to search sequentially, adding small successive accretions to his store of information about the problem and its solution.

Additional evidence for the basic properties of the IPS comes from the simpler standard laboratory tasks. The evidence for the 5 or 10 seconds required to store a symbol in long-term memory comes mainly from rote memory experiments; evidence for the seven-symbol capacity of short-term memory, from immediate recall experiments; evidence for the 200 milliseconds needed to transfer symbols into and out of short-term memory, from experiments requiring searches down lists or simple arithmetic computations. (Some of this evidence is reviewed in Newell and Simon, 1972.)

The detail of the human IPS is elusive because the system is adaptive. For a system to be adaptive means that it is capable of grappling with whatever task environment confronts it. Hence, to the extent that a system is adaptive, its behavior is determined by the demands of that task environment rather than by its own internal characteristics. Only when the environment stresses its capacities along some dimension--presses its performance to the limit--do we discover what those capacities and limits are, and are we able to measure some of their parameters (Simon, 1969, Ch. 1 and 2).

Because of the adaptivity of the human IPS, any explanation of its behavior in the face of a particular task must take into account the strategy or "program" it employs for that task. The examples to be discussed in this paper will have a great deal to say about the nature of these strategies and their relation to learning and understanding.

If we wish a slightly more formal description of the human IPS, it can be constructed along these lines:

1. There is a set of elements, called symbols, which are capable of denoting, or pointing to, objects.

2. There are symbol structures, consisting of organizations of symbols connected by a set of relations.

3. A memory is a component of an IPS capable of storing and retaining symbol structures.

4. An information process is a process that takes symbol structures as its inputs or outputs.

5. A processor is a component of an IPS consisting of: (a) a set of elementary information processes; (b) a short-term memory that holds the input and output structures of the information processes; and (c) an interpreter that selects the order in which the information processes will be executed.

6. The symbol structures that determine for the interpreter the order in which it will execute processes are its program.

Everything we know about the human information processing system indicates that it meets these specifications in addition to the more specific ones mentioned earlier.

In discussing human problem-solving behavior, it is often convenient to distinguish the "objective" problem situation as the experimenter would describe it from the situation as the problem solver represents it to himself in attacking the problem. We will call the former, objective, description of the problem situation the task environment, and the latter, subjective, representation the problem space. For many purposes, problem solving can be viewed as a search through the problem space, which may be thought of as a tree-like network or maze. If the problem space is inappropriate to the objective task environment, it may be difficult or impossible to solve the problem. On the other hand, an especially happy choice of problem space may greatly facilitate finding a solution. These possibilities will be illustrated by the examples.

Computer Simulation

The modern electronic digital computer has proved to be a powerful aid to research on human higher mental processes, and our knowledge about these processes has advanced greatly since this new tool became available about twenty years ago. The computer has made its contribution in at least three ways. First, as our description of the human IPS suggests, it has served as a valuable metaphor. It was the computer that first led psychologists to think of human cognition in information processing terms, a far more useful metaphor than the earlier picture of the central nervous system as a "switchboard." The switchboard is a passive, the IPS a highly active, system.

Second, the computer provides us with programing languages that can be used to construct formal descriptions of the human IPS's strategies or programs. In fact, the strategy, in such a formalization, literally becomes a computer program described in some computer language. Programing languages have been developed (list-processing and string-processing languages) that are well adapted to representing the memory contents and processes of the human IPS. Thus, computer programing languages replace conventional mathematical notation as tools for formalizing theories of cognition. In describing specific examples in this monograph, we will not actually use these formal languages, but all of the programs to be discussed have been written in one or another of them.

Third, the computer program that describes our theories of human strategies can actually be run on computers that have been given the same problems given to the human subjects. The output or <u>trace</u> of the computer program then simulates the sequence of problem-solving efforts displayed by the human subject. Discrepancies between the predicted and actual behavior can be observed, and can become the basis for new efforts to improve the accuracy of the simulation.

It cannot be emphasized too strongly that in this application of the computer it is not being used as a super-fast "number cruncher," nor is it competing in speed or accuracy with the human subject. The computer is being used, as its very general capabilities enable it to be, as a general purpose information processor. It is programed to imitate as closely as possible the actual processes used by humans, including their foibles, and it avoids entirely taking advantage of its powerful arithmetic capabilities, which are patently unlike those of a human. If the computer programed for simulation solves a problem either more skillfully or less skillfully than do the human subjects, then the program is a poor simulation--a poor theory of the human processes. The same may be said if it makes more errors or fewer errors than the human subjects, or searches quite different parts of the problem space, or searches with greater or less selectivity than the human subjects. A computer simulation makes extremely detailed predictions about the problem-solving behavior. Hence, it is highly desirable to be able to match these predictions against a dense stream of observations of the human behavior. The standard experimental paradigms in which subjects respond only at intervals of several seconds are not very powerful for testing these kind of theories. For many problem-solving tasks it is possible to induce subjects to "think aloud" (<u>not</u> to introspect or retrospect) about what they are doing while they are solving the problem. Thinking-aloud protocols, sometimes supplemented by eye-movement recordings, provide us with data of the highest temporal density that we are usually able to obtain. Such data have been invaluable in discovering and testing information processing theories of complex human cognitive behavior. Some progress has now been made in objectifying and automating the difficult task of analysing thinking-aloud protocol data.

An important development of the past five years or so has been the achievement of sufficient understanding of natural language to permit computer programs to be written that can process and understand, in several senses of that word, natural language text. Programs for processing natural language are useful not only for analyzing problem-solving protocols, but also for simulating human language processing, e.g., understanding written problem instructions. We shall examine an example of such an application later. Since language is a fundamental component of human cognition, a theory that ignored it would be very incomplete. It is no longer necessary to abstract from natural language in simulation programs.

As a simple example of how problem-solving tasks may be represented in computer programs, consider the Tower of Hanoi task, which appears later as one of our examples. The task involves three pegs, and some disks of various sizes that can be impaled on the pegs. Symbols would be stored in memory to represent the three pegs: PEGA, PEGB, and PEGC, say. Another set of symbols would represent the disks: DISK1, DISK2, DISK3, DISK4, and so on. A relation, symbolized as ON, would be defined to connect a peg with the disks impaled on it, e.g., PEGA: ON, DISK2, DISK4 (read: "On PEGA are DISK2 and DISK4"). Each disk would have associated with it a symbol indicating its size: e.g., DISKA: SIZE, 2. A process would be defined for moving disks from one peg to another: MOVE(DISKx,PEGy,PEGz). The program for executing the move would remove DISKx from the On-list of PEGy, and add it to the On-list for PEGz. Thus, the symbol manipulating processes of the IPS would mirror exactly the problem situation outside.

A strategy would be a program for determining a sequence of moves. It might include component instructions like: Detect the largest disk that is not yet on the goal peg and store its name in short-term memory; store in short-term memory the goal of moving DISKx to PEGy; detect the largest disk that is blocking the move of DISKx and store its name in short-term memory. Instructions like these are readily written in appropriate programing languages.

With these introductory explanations out of the way, we can now proceed to the examples.

Match-Stick Problems

My first example, which predates computer simulation techniques, refers to the important work, in the Gestalt tradition, of George Katona (1940). Here is one form of the task he used: Sixteen matches are laid out in five squares, as shown in Figure 1. By moving exactly three matches, reduce the number of squares from five to four. All the matches must be used, and all the squares must be of the same size.

Katona taught three different solutions, or solution hints, to three different groups of subjects. To one group of subjects, he taught a "rote" solution: move Match 4 to V, 9 to W, and 2 to AA.

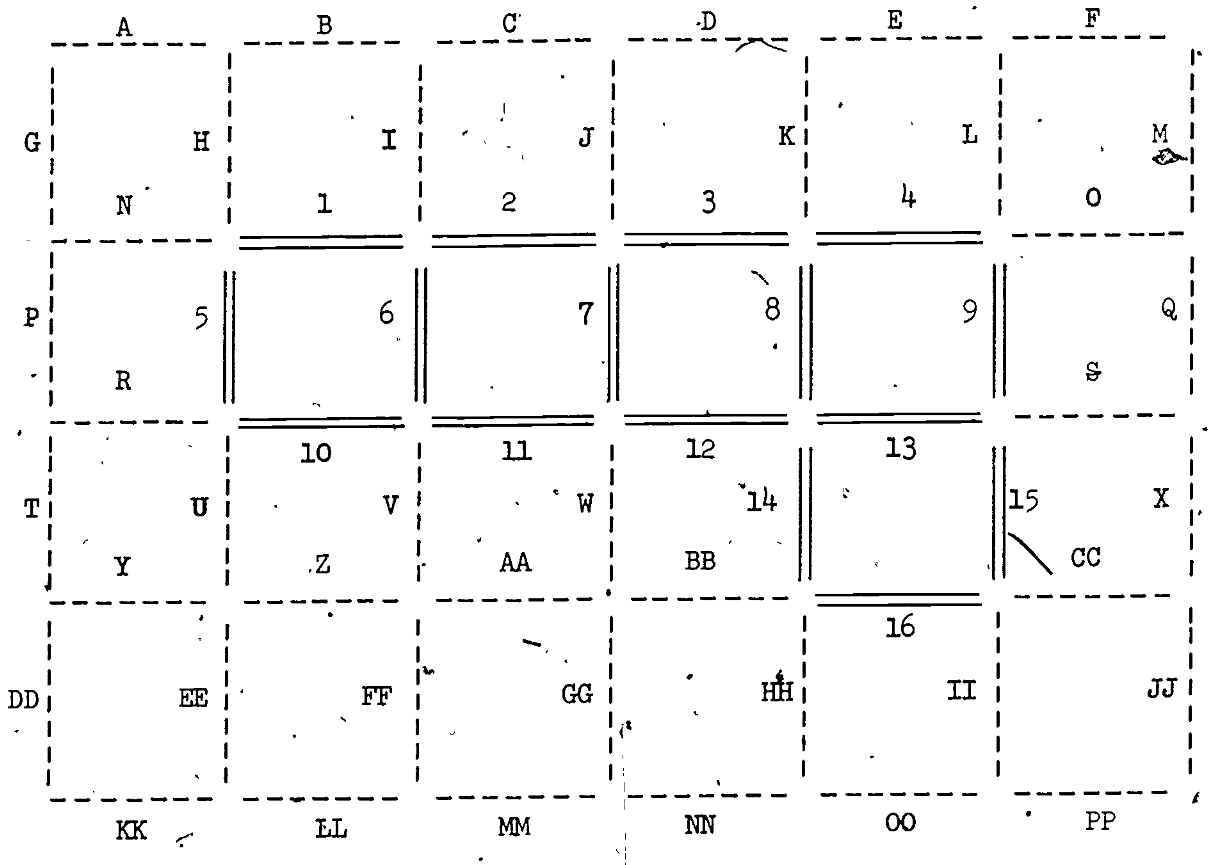

Figure 1

Katona's Matchstick Problem

To a second group, he taught a "logical" solution: since there are 16 matches, if there are to be four squares, each match must belong to only one square. In the starting situation, four matches (6, 7, 8, 13) perform a double function, for each belongs to two squares. To solve the problem, you must get rid of the double-function matches.

To a third group, he taught an "intuitive" solution: the figure is very compact, with the squares all jammed together. To solve the problem, you must open the figure up.

Katona measured (1) the time it required subjects to learn the solution, (2) how well they retained it if they were asked to perform the same task again some weeks later, and (3) how well they were able to transfer their skill to other problems of the same kind, but with different initial configurations of the matches. In general, he found that the intuitive and logical solutions were learned more quickly than the rote solution, but especially, that they were retained better and produced a greater amount of transfer. Further, the intuitive solution was a little better in these respects than the logical solution.

Katona himself did not provide an information-processing explanation of these findings. Clearly the solution I have called "intuitive" is the one that accords most closely with Gestalt principles, but it would be hard to give a formal demonstration of that fact, or a formal explication of just what it means. Let me hazard an explanation along information-processing lines.

First, the rote solution is the only explicit one, but it requires six facts to be learned (which three matches are to be moved, and the three target locations for them). If any one of these facts is lost, the solution cannot be carried out by rote. The solution will not work at all for different configurations of the matches. Thus, there is much to be learned by this method; a little forgetting will cause failure; and there is no basis for transfer to variant tasks.

The logical solution is, of course, incomplete. It merely states one necessary condition that must be satisfied by any solution. However, it requires only a single "idea" to be learned, and it is applicable to all forms of the matchstick problem. But notice that the hint is stated in terms of a characteristic of the problem solution, and not in terms of actions needed to bring the solution about. Moreover, the hint calls attention to the double-function matches, which are not the matches that need to be moved in order to achieve the solution.

The intuitive solution is also incomplete, and vague as well. It has a perceptual, rather than a cognitive, flavor. Again, only one idea must be learned, and it is applicable to all forms of the problem. This hint, unlike the hint for the logical solution, is stated in terms of an action--something to do. It could easily be made more complete and explicit by being rephrased: Open the figure up by breaking up two squares, then make one new one.

In summary, the Katona matchstick experiment illustrates several fundamental points. As a result of different explanations, students can be induced to learn quite different things about the same task. These different kinds of learning are superficially similar, in that they all allow the task to be performed successfully. They are quite different, however, with respect to the important criteria of efficiency in learning, retention over time, and transfer to other tasks. Most of us would say that the students who learned the logical or intuitive solutions "understood" the problem at some level, while those who only learned the rote solution did not. But the experiment warns us also that "understanding" is not necessarily a unitary thing. It may have a cognitive flavor or a perceptual flavor; it may largely involve explaining why something works, or it may involve explaining how to make it work.

The distinction between what I have been calling the logical and intuitive solutions can be looked at in a slightly different way. The former refers to a state space--a space of matchstick configurations. The latter refers to a space of operations--of actions to change the configuration. Since problem solving involves finding sequences of operations that bring about a desired change in state, problem solutions imply relations between these two spaces: mappings of operation sequences upon state-space differences. The difference between understanding properties of the solution in the state space and understanding properties of the operator sequences, is analogous to the difference between proving the existence of something by reductio ad absurdum and proving it constructively by means of an algorithm. I would conjecture that the applied mathematician most often requires the actual algorithm for his understanding. Understanding of the other kind does not meet his needs (to know "what to do next"), and he may even be indifferent to it.

The Tower of Hanoi

The Tower of Hanoi puzzle will be familiar to many of you as a wooden toy. There are three vertical pegs, call them A, B, and C. On Peg A is impaled a pyramid of disks (say 5 of them) of graduated sizes. The task is to move the pyramid to Peg C, under the conditions that only one disk may be moved at a time and that a larger disk may never be placed on top of a smaller disk. With n disks, the solution requires a minimum of 2^n-1 moves (e.g., 31 moves for 5 disks).

During the past year, we have been constructing in our laboratory a taxonomy of solutions of the Tower of Hanoi problem preparatory to doing experiments on learning, retention, and transfer (Simon, 1975). Each solution is embodied in a computer program (written in the SNOBOL string-processing language) that is capable of using the method it embodies to solve the problem. Thus each program constitutes a theory of the knowledge and skills possessed by any human being who can apply that method to the problem. By comparing the programs, we can see what differences they imply in the demands made upon memory or processing capabilities. By small "parameter" changes in the programs, we can construct innumerable variants of the basic methods, as well as hybrids that combine elements of several of them.

We have found four types of solutions of the Tower of Hanoi problem, each with quite distinct characteristics in the demands they make upon long-term memory, short-term memory, and perception. The computer programs express in a mathematical language our theories of the processes being used to perform the tasks. Formalizing the process theories in this way guarantees their completeness in a certain sense: computer programs lacking essential components don't run properly. The formal expression of each program also allows us to make exact estimates of the memory and perceptual requirements of the method that the program describes. Finally, the programs can be used with actual Tower of Hanoi problems to simulate human performance in the task, so that the simulations can be compared with human data.

The first of the four methods we have programed is a rote method. The Tower of Hanoi problem for five disks can be solved by making, in proper sequence, the 31 moves that have been learned by rote. As in the case of the rote solution of the matchstick problems, this solution is burdensome to learn, is easily forgotten, and offers no help to solving the problem with an arbitrary number of disks, or the problem with a change in starting situation (e.g., two smallest disks on Peg C, largest on Peg B, and remainder on Peg A).

The second method, which we call "goal recursion," has the greatest mathematical elegance. To move a pyramid of n disks from Peg X to Peg Y, first move the pyramid of n-1 smallest disks from Peg X to Peg Z; next move the largest disk from X to Y; and finally, move the pyramid of n-1 smallest from Z to Y. Of course moving a pyramid is not a legal move, but it constitutes a Tower of Hanoi problem with one less disk than the original problem. Hence, if we know how to solve the one-disk problem, then, by mathematical induction, we can solve the n-disk problem for arbitrary n.

Acquiring the goal recursion method involves less learning than the rote method, provided that the learner understands the concept of recursion or of mathematical induction. Again, we would expect good retention, and the solution transfers to problems with any numbers of disks, but not to arbitrary starting configurations unless the algorithm is amplified. The method illustrates how understanding may be relatively easy if certain other understandings (in this case induction) have been acquired previously, but may be relatively difficult otherwise.

The goal recursion method illustrates another important point also. To execute it requires holding each goal in short-term memory while executing the goal at the next lower recursive level, and holding the latter goal, in turn, while the next lower is executed. The stack of pending goals in memory reaches a maximum of n-1 in an n-disk problem. Hence, we would expect a human subject to encounter real difficulty in executing the method, even if he understands it, when the recursion depth exceeds short-term memory limits (four of five disks). There is a variant of the method, however, that requires regenerating the goals instead of remembering them. It should be slower of execution than the "pure" method, and requires perception of the current situation to choose the next goal.

This brings us to the third method, which may be called "perceptual." The goal recursion method could be applied without sight of the actual Tower of Hanoi apparatus, for all the information used in applying it is stored in memory. In the perceptual method, each move is chosen by looking at the characteristics of the current problem situation, which requires looking at the actual apparatus (or retaining a visual image of it). In its most sophisticated form, it works like this: Find the largest disk not yet on the target peg; if it can be moved to that peg, move it; if not, find the largest disk (either above it or on its target peg) that is blocking its move, and set up the goal of moving that disk; repeat the process until a disk is found that can be moved, and move it. This process is also recursive, using the recursion to find the largest removable barrier to a desired move.

The perceptual method is relatively easy to learn (providing the perceptual recursion is understood), requiring mainly the acquisition of the perceptual predicate "largest blocking disk." It can be transferred to problems of all sizes, and it works for any starting situation, whether on the optimal solution path or not. What this solution teaches us is that understanding a problem may involve learning to see the right perceptual chunks in the stimulus display--in this case learning to perceive the largest blocking disk for any given disk.

Finally, the Tower of Hanoi problem can be solved by executing a sequential pattern of moves. Number the disks from smallest to largest. Then, they should be moved in the following sequence (for the four-disk problem): 1-2-1-3-1-2-1-4-1-2-1-3-1-2-1. It is easy to see how this pattern can be extended to five or more disks. Since the smallest disk can always move to either of two pegs, an additional simple rule is needed to specify the direction of movement of that disk.

The pattern solution can be simplified even further, because we know that the smallest disk is only moved on the odd-numbered moves and there always exists only a single move on the even-numbered moves. The sequential pattern solution is easier to learn than the rote solution; it is more-or-less transferable to problems of any number of disks; it is usable only along the optimal solution path, and it makes few demands on either perception or memory (other than keeping track of the parity of the move).

Clearly, a person who learns a particular one of these four solutions understands quite different things about the Tower of Hanoi problem than does a person who learns a different solution. We would probably say that a person who learns only the rote solution doesn't understand the problem at all. But what about the sequential pattern solution?

It is concise, even elegant, and certainly rhythmic. The person who knows it does not thereby understand why it works. Is this a kind of understanding that it is useful for our students to acquire? Is it important for them to look for pattern in nature as well as mechanism in nature?

I can point to at least three very important discoveries in the natural sciences that involved detection of pattern without knowledge of mechanism: Kepler's discoveries of the elliptical orbits of the plants, of the Law of Areas, and of the relation of period to radius of orbit; Mendeleev's discovery of the Periodic Table of the elements; and Balmer's discovery of the formula for the Balmer lines of the hydrogen spectrum (Simon, 1968; Banet, 1966, 1970). These discoveries provided parsimonious descriptions, not explanations, of their respective phenomena. It can be shown that some common basic pattern-detecting processes were implicated in all of them. Whether or not we wish to call such pattern detection a form of "understanding," we may well want to help our students to acquire this skill if it is teachable.

The goal-recursive and perceptual solutions for the Tower of Hanoi problem come closer than does the sequential solution to our common notions of "understanding," yet they are quite different from each other. If by understanding we mean being able to prove that something works, then the goal-recursive solution is superior, for it is the easiest one on which to construct such a proof for the puzzle. A proof can be constructed for the perceptual solution, but it is a little more complicated. The perceptual solution, on the other hand, goes to the heart of the question: What feature in this situation tells me what to do next? Thus, the solution works for any situation, whether on an optimal solution path or not, while the goal recursive solution only works along such a path.

Let me leave the Tower of Hanoi with this demonstration of a plurality of understandings, and move on to another example.

Geometry Proofs

My third example is based upon the work of Scandura in teaching students to discover proofs for theorems in geometry, and upon the artificial intelligence research of Gelernter and Rochester, who built a computer program capable of finding such proofs.

Scandura has been especially concerned with geometry problems that involve constructions (Scandura, Durnin and Wulfeck, 1974). Discovering constructions is perhaps the most difficult skill that geometry students have to acquire. Scandura and his colleagues have developed procedures for teaching students schemata (called, in their papers, "higher-order rules") that are helpful in searching for appropriate constructions to solve geometry problems.

An example of the approach is provided by the two-locus problem: Given a line and a point not on the line, and a radius R, construct a circle of radius R which is tangent to the given line and which passes through the given point. This problem can be solved by drawing a circle of radius R about the given point; finding the line at distance R from the given line, and parallel to it; finding an intersection of these two loci; and constructing a circle of radius R with that intersection as center. By the first construction, the new circle will pass through the given point, and by the second construction, it will be tangent to the given line.

Scandura et al. observe that the solution fits the general schema: To find a locus satisfying two conditions, find the locus satisfying the first, then the locus satisfying the second, then the intersection of these two loci. This schema is, of course, a special case of the more general schema of means-ends analysis used in problem-solving programs like the General Problem Solver (GPS) (Ernst & Newell, 1969). Means-ends analysis works roughly as follows:

Given a starting situation and a goal situation, detect the differences between them. For each such difference, find an operator in memory that has the function of removing differences of that kind and apply the operator. Continue until all differences have been removed. To apply the means-ends schema, or the two-locus schema, there must, of course, be available in memory the operators to be applied to carry out the individual steps. In the geometry case, this means a set of basic one-locus constructions, accessible from knowledge of the conditions they satisfy.

It is easy to show that the effectiveness of a program for discovering proofs for theorems in geometry or solving other kinds of problems by search processes depends heavily upon the selectiveness of its search. The number of alternative paths through the forest is usually far too large to permit crude trial-and-error search. Hence, efficient search rests on the availability of processes, like the schemata described above, that choose promising search paths and avoid others that are not likely to lead in the direction of the goal.

It follows that "understanding" a subject like geometry requires not only acquiring an adequate store of theorems, to be used as the basic operators in carrying out proofs, but also acquiring sufficiently powerful schemata to guide the search for solutions. Whereas geometry books are always fully explicit about the theorems, they generally make only sparing reference to schemata and their use.

A similar lesson is taught by the Gelernter-Rochester geometry program (Gelernter, 1963). That computer program (which was not designed as a simulation of human problem-solving processes) makes use of a number of heuristic devices to guide its search for proofs. Among these, it uses a diagram of the problem situation. Suppose that, working backwards, it determines that the theorem, T, could be proved if the antecedents, T1, T2, etc., could be proved. Before it undertakes to prove these antecedents, it checks the diagram to see if they are empirically true--that is, true within the margin of error of the diagram. If they are not, it abandons that particular line of search. The usefulness of this procedure depends, of course, on the fact that it takes far less processing time to detect, for example, that a triangle in a diagram is not right-angled than it does to exhaust the possibilities for proving that it is.

Thus, geometry, like the tasks examined previously, teaches us that understanding is a pluralistic concept. In particular, it teaches us that we do not understand a mathematical subject when we simply understand the axioms, theorems, and rules of inference. Understanding requires, in addition, the acquisition of a whole host of heuristic problem solving capabilities, some of which are peculiar to the given subject, but others of which have a wider range of application.

Algebra Word Problems

Among the least-formalized aspects of mathematical learning are the skills of expressing in mathematical language physical or other empirical situations described in natural language. At the core of formal arithmetic and algebra are symbolic expressions and their manipulation. Word problems, or story problems, extend beyond these formal boundaries in two directions: in their use of natural language, and in their reference to the semantic denotations of the language and corresponding equations. It should not be surprising, then, if ability to handle word problems were relatively independent of skill in symbolic manipulation. I have no systematic data on this point, but my friends who teach mathematics seem to see little relation between the two skills. Nor, and this is a little more surprising, does a high level of verbal skill appear to be sufficient for proficiency in handling word problems. On the contrary (and again anecdotally) persons with good verbal skills but without other mathematical aptitudes appear to be relatively more disadvantaged in doing word problems than in manipulating uninterpreted mathematical expressions.

We are beginning to understand from an information-processing standpoint what is involved in understanding and performing word-problem tasks. Let me begin with an account of a computer program that was not intended to simulate in detail how people solve word problems, but was constructed as a study in artificial intelligence--in how to program computers to do clever things (Bobrow, 1968). Since the program has been described several times in the literature, there is no need to repeat that description here. The important thing about the program for our purposes is that it is primarily syntactic rather than semantic in its methods. Given the text of an algebra word problem, it undertakes to translate that text into a set of algebraic equations, and then to solve the equations. The task is approached as a problem in automatic translation. The program (called STUDENT) has some syntactical capabilities that enable it to parse simple English sentences of the sort found in word problems. In general, the system does not need to know the meanings of the words in the sentences, except those words that perform grammatical functions or have specific mathematical meanings (e.g., "times," "45," "half"). Hence, for this program, understanding a problem means being able to extract enough of the structure of the input sentences to translate them into equations having the same structure.

To make this more concrete, consider the following example: "If the number of customers Tom gets is twice the square of two-tenths times the number of advertisements he runs, and the number of advertisements he runs is 45, what is the number of customers Tom gets?" Here, "the number of customers Tom gets" and "the number of advertisements he runs" need only be recognized as noun phrases, to be treated as "unknowns" and provided with algebraic names like x and y in the translation. On the other hand, "is" must receive its appropriate semantic translation as "=," and "twice," "square," "two-tenths," and "times" must also be interpreted semantically. Nothing, obviously, need be known about the world of customers and advertisements.

Some time after STUDENT had appeared, and had demonstrated its ability to solve high-school level problems, it occurred to us to ask whether the processes it used bore any resemblance to the processes used by students in algebra courses (Paige and Simon, 1966). We constructed some problems and tested them with human subjects, asking the subjects to think aloud as they worked the problems. We then analysed their tape-recorded protocols to identify the sequence of processes they had used, and the way in which they represented the problems at various stages during the translation. Again, let me give you one of the problems we used: "A man has 7 times as many quarters as he has dimes. The value of the dimes exceeds the value of the quarters by $2.50. How many has he of each coin?"

A number of our subjects proceeded just as STUDENT would, parsing the input sentences and mapping them over to an algebraic equation like: 10D = 25(7D) + $2.50. You can verify that this is an accurate translation into algebra of the English sentences. Other subjects, however, wrote down a similar equation, but with the "plus" replaced by "minus." A third, and smaller, group of subjects read the problem statement and said to the experimenter: "Isn't there a contradiction here?"

Of course there is no contradiction in the problem statement. There is a contradiction only if we add to the statement some semantic knowledge that an American student might be expected to have stored in his long-term memory: a quarter is worth more than a dime, and the numbers of both quarters and dimes must be non-negative integers. The difference in the processes of the three groups of students now becomes rather clear. The students in the first group proceeded purely syntactically (except for recovering from semantic memory the fact that a dime equals ten cents and a quarter, twenty-five cents). The students in the second group used their semantic knowledge to infer that the total value of the quarters must be greater than the total value of the dimes, and that therefore the $2.50 must be added to the latter or subtracted from the former. Evidently, they never checked this inference against the syntactical detail of the sentence (after all, something was to be added to something), but used the semantic knowledge to construct the "correct" equation. The students of the third group processed the input sentences both syntactically and semantically, thereby discovering the "contradiction."

This simple example illustrates some of the alternative ways in which the same problem may be processed. Of course the alternatives become more numerous as the semantic context of the problem becomes richer. For many problems of applied mathematics, "physical intuition" (which we can now translate as "semantic information") may go a long way toward reducing the need for careful, detailed syntactic processing. If we are training a student in applied mathematics, we may take the position that he does not understand what he is doing unless he is able to evoke from memory, when he is confronted with a problem, the rich set of semantic information relevant to the problem which he has stored (or should have stored). On the other hand, there comes a point in training a pure mathematician where we may have to discourage him actively from employing semantic cues instead of holding carefully to purely syntactic processes. Unless we succeed in teaching him the distinction, he will never know what mathematical rigor is. There is no reason, of course, why a student should not learn that there are at least these two different kinds of understanding of mathematical problems, each appropriate to certain times and circumstances.

Understanding Problem Instructions

In handling algebra word problems, the student is essentially given the problem representation: the output of his translation is to be a set of algebraic equations. In other kinds of problem situations, choice of problem representation becomes a key part of the solution process. Let me cite an extreme example, which may be familiar to some of you: the "mutilated checkerboard" problem.

Given an ordinary 8 x 8 checkerboard, with each square one inch on a side, and 32 dominoes, each 1 x 2 inches in size, the board can be exactly covered by the dominoes, with no dominoes left over. Suppose that the upper-left-hand square and lower-right-hand square of the board are now cut off. Can the mutilated board be covered exactly with 31 dominoes?

We can try to solve the problem by testing all coverings of the board. Since there are only a finite number of possibilities, we will sooner or later find a solution or prove there is none. Of course, when we calculate the number of alternatives, we realize that we will find the answer later, not sooner. Is there a better way?

If we recall that the squares of a checkboard are alternately black and red, then the better way becomes evident. We do not need, in our representation, to keep track of which squares we have covered, but only of the number of black squares and the number of red squares covered. It is easy to verify that both squares cut from the board had the same color (red, say); hence the mutilated board has only 30 red, but 32 black squares. But each domino covers exactly one black and one red square, hence there is no way they can be arranged to cover more of one color than of the other.

Consider now another puzzle-like problem that is considerably less subtle than the checkerboard problem: "Three five-handed extra-terrestrial monsters were holding three crystal globes. Because of the quantum-mechanical peculiarities of their neighborhood, both monsters and globes come in exactly three sizes with no others permitted: small, medium, and large. The medium-sized monster was holding the small globe; the small monster was holding the large globe; and the largest monster was holding the medium-sized globe. Since this situation offended their keenly developed sense of symmetry, they proceeded to teleport globes from one to another so that each monster would have a globe proportionate to his own size. Monster etiquette complicated the solution of the problem since it requires: (1) that only one globe can be transmitted at a time; (2) that if a monster is holding two globes, he may transmit only the larger of the two; and (3) that a globe must not be transmitted to a monster who is holding a larger globe. By what sequence of teleportations could the monsters have solved this problem?

Before he can begin working on this problem, a person must find some way of organizing the facts about the situation and the permissible operations on it. This resembles the translation stage of the algebra word problems, but it is substantially more difficult. One reason it is difficult is that there are alternative representations to be considered. For example, he can associate with each monster the set of globes it is holding at a given moment; or he can associate with each globe the monster who is holding it. Which representation is selected has consequences for the ease with which moves can be made and their legality tested. With the former representation, for example, moves are made by deleting the name of a globe from one monster's list, and adding it to the list of another. With the latter representation, moves are made by changing the name of the monster holding a particular globe

The difference between these two representations is not trivial. Subjects who adopt the former representation are able to solve the problem in about one-half the time that is required by subjects who adopt the latter representation. By a combination of laboratory experiments, analyses of thinking-aloud protocols, and computer simulation of the understanding process, we are beginning to get clues as to why this is so, and clues also as to why subjects adopt one representation or the other.

The computer program, UNDERSTAND, which we have built as our research vehicle for this task, has a gross structure not unlike the STUDENT program. Its first task is to parse the input sentences, making use mainly of syntactic knowledge to get at their surface structure. Next, it makes some judgments about what is relevant in the problem statement, primarily by identifying the sets, lists, and relations that are discussed. Then it is ready to synthesize a representation for the problem--a way of storing the problem information in list structures (essentially, directed colored graphs) in memory. At the same time, it interprets the problem statements that describe legal moves, and adapts the move processes appropriately to the representation that has been chosen. It is now ready to begin its attempts to solve the problem. (The reader who is interested in the content of the program will find a fuller description in Hayes and Simon, 1975.)

The UNDERSTAND program does not in a literal sense "choose" its representation. That is to say, it does not explicitly consider a number of different representations and select one of them. Instead, it is led to synthesize a particular representation by the form of the problem statement, and a different problem statement may cause it to synthesize a different representation. For example, in the form in which the problem is stated above, the system would in fact assign lists of globes to monsters. If the problem statement said something like: "Globes are owned by monsters, and the owners are changed until each globe has an owner whose size corresponds to its own size," then it is likely that UNDERSTAND would select the representation that associates with each globe the monster holding it. Thus representation is highly sensitive to problem statement, and the system has no capability for seeking a "best" representation that will facilitate solution of the problem. It is as helpless in this respect as the typical person confronted with the mutilated checkerboard problem.

Our reason for constructing the UNDERSTAND program in this relatively "unintelligent" fashion is that it appears to reflect just what people do when confronted with instructions like those for the monster problems. That is, the deliberate search for a good problem representation, or even a capability for generating and considering alternative representations does not seem to be a common part of the human problem-solving repertory. Our subjects appear to be as readily trapped as is UNDERSTAND into inefficient problem representations whenever these are the representations that follow most directly and transparently from the wording of the problem text. If we think, therefore, that our students should possess skills of generating and modifying problem representations, we will probably have to give explicit attention to those skills in their training.

There is very little more I can say at this time about how this desirable result is to be attained. We are still in the early stages of our research on these phenomena, and far from the point where we are ready to prescribe learning processes that will enhance the capabilities of problem solvers on this dimension. Practice does not always have to wait for science, however. I expect that if we gave more explicit attention to the processes whereby representations are generated in our teaching of applied mathematics, we could likely help our students acquire better skills of choosing representations.

Chemical Thermodynamics

The environment of monster problems is still rather impoverished from a semantic standpoint. Knowledge of quantum mechanics will not help a person formulate or solve these problems. In fact, several of the subjects we have tested bogged themselves down hopelessly, precisely by failing to abstract out these irrelevancies at the very outset. The problem domains we have discussed in previous sections are also not very rich semantically: algebra word problems, the Tower of Hanoi, and matchstick problems. Geometry, of course, has a considerable content, but it is not semantic, strictly speaking, unless we make use of diagrams or other instantiated models.

For my last example, I should like to use an area of science that is probably not atypical for applied mathematics in its intermingling of syntactic and semantic elements. This is the area of chemical thermodynamics at the level of an upper-division undergraduate course. The choice of chemical thermodynamics rather than electrical circuit theory or engineering mechanics is a matter of chance. For various reasons that are irrelevant to the present discussion, we happen to have chosen thermodynamics as the setting for studying the scope and organization of students' semantic knowledge in a scientific subject.

What I can now say on this topic is even more provisional than what I have said about the UNDERSTAND process. What I should like to share with you is not so much our conclusions as our current plans for exploration.

The vehicle we have chosen for an initial exploration is a computer program that generates problems in chemical thermodynamics, and that is capable also of solving the problems and of offering successive hints to students who are having difficulty solving them. The problems are of the sort that you will find in standard textbooks on the subject. To take a simple example: a pump takes in water at such and such a temperature and pressure, and outputs it at some other specified temperature and pressure; what is the horsepower of the pump if the water flow is 100 gallons per minute?

The program that generates these problems is somewhat different from the programs generally used today in computer-aided instruction. The usual problem-generation method is to have a considerable number of templates of different problem types, to choose one of the templates, and then to fill it in with appropriate numbers. Our program contains, instead, what amounts to a theory and a body of factual information about chemical thermodynamics. It uses its theoretical and factual knowledge to generate problems which it can then solve with the same resources. Since, when solving the problem, it must make decisions about the order in which to solve for the unknowns, it can record these decisions and use them to offer advice to students about the solution process.

Let me describe the program (which I am constructing in collaboration with R. Bhaskar) a little more concretely. The program has access to equations that express the laws of conservation of mass and of energy, and others that represent the equations of state for the working substances that are used in thermodynamic devices. (The gas laws for ideal gasses are a special case of these.) It has a list of devices (e.g., pumps, compressors, pipes, etc.), with each of which is associated information about the working substances it can employ, and the usual conditions of its operation. It has a list of working substances, with each of which is associated the equation of state and information about reasonable upper and lower bounds for its temperature, pressure and other properties.

The program is able to examine the incidence matrix of the equations it is using in order to choose subsets of variables to be the independent and dependent variables, selecting these so that they will be consistent. It can then assign values to the independent variables and ask questions about the dependent ones. Using the same incidence matrix, it can also select efficient solution paths that minimize the need to solve simultaneous equations.

This brief description will provide a feel for the character of the system. It was designed initially, as I have said, to generate problems for students. However, as it has developed, we are more and more persuaded that it also provides a starting point for describing the organization of knowledge in the memory of a student who has completed a thermodynamics course. Accordingly, we are now beginning experimental work to see if we can obtain direct evidence about how such semantic knowledge is stored, by observing students as they solve thermodynamics problems. We proceed on the assumption that one of the main components of "understanding" in this kind of task is to be able to evoke elements from a considerable body of semantic information, as and when that information becomes relevant to the problem under attack. We wish now to see whether the organization of knowledge that enables a person to handle such problems effectively bears any resemblance to the organization we have imposed upon our program in order to enable it both to generate and to solve the same problems.

Conclusion

In this entire account of "learning to understand," ranging over a half dozen rather dissimilar problem domains, no attempt has been made to define either of the key terms "learn" or "understand." The omission is deliberate. Neither "learning" nor "understanding" denotes a single, simple set of human cognitive processes. Whenever a change takes place in the cognitive system that enables it to perform a task better than it could previously, we say that learning has taken place. It is extremely doubtful that all learning involves one kind of change, or change in one particular component of the system. Learning, then, is simply a portmanteau term that denotes any semi-permanent improvement in performance.

In a similar way, we say that a system exhibits "understanding" of a domain when it demonstrates that it possesses relevant knowledge of that domain and is able to marshal that knowledge in the performance of various tasks. Understanding can be of various kinds and degrees, may support the performance of a variety of different kinds of tasks, and may reside as much in the organization of knowledge, and in the processes capable of operating upon it, as in its content.

All of this sounds very indefinite. The way to make it definite, and ultimately applicable to problems of instruction, is to explore in detail how knowledge and skill are stored in the human mind and brain in specific task domains of various sorts. And the machinery for carrying out those explorations includes both the standard armamentarium of psychological experimentation and the powerful new tools of computer simulation of cognitive processes.

In this paper I have tried to illustrate how this exploration can be carried out, citing six examples, more or less relevant to the enterprise of applied mathematics, of the present state of the art. We have already reached a point where the research begins to give us new conceptions of the nature of knowledge, skill, and understanding in applied mathematics; and where we can begin to draw some common-sense lessons from it that are applicable to our pedagogy.

Let me conclude by listing some of the suggestions for practice that are implicit in the examples of research surveyed in this paper.

1. The kinds and degree of understanding that the student achieves in a task domain can have important consequences for his retention of skill and knowledge, his ability to transfer that skill and knowledge to similar tasks, and the speed and efficiency with which he can acquire additional knowledge.

2. Understanding has many facets. It may require the acquisition of new cognitive concepts (e.g., recursion in the Tower of Hanoi problem), and ability to recognize new percepts (e.g., the "largest blocking disk" in the same problem). It may be expressed in terms of properties of problem situations, or properties of operator sequences, that is, in terms of state or process.

3. An important component of problem-solving skill lies in being able to recognize salient problem features rapidly, and to associate with those features promising solution steps. Much current instruction probably gives inadequate attention to explicit training of these perceptual skills, and the kind of understanding that is associated with them.

4. Limits of short-term memory may prevent application of a problem-solving method that is understood. Sometimes alternative methods exist that permit a tradeoff of conceptual recognition for short-term memory of goals.

5. Understanding generally requires not only storage of adequate semantic information, but also availability of problem-solving schemata, both those specific to the subject matter (e.g., the two-locus heuristic for geometry), and those that are more general in application (e.g., means-ends analysis). These schemata deserve an explicit and prominent role in instruction.

6. It is often possible to substitute syntactic for semantic processing, and vice versa. Awareness of these alternatives, and skill in employing both of them can enhance accuracy of understanding by exploiting redundancies in the problem information (e.g., the redundancies in the "contradictory" algebra problems).

7. Understanding processes include the processes of constructing representations of problem situations. Most problems are capable of being represented in a variety of ways, and problem difficulty may be greatly affected by the representation chosen. The skills of searching for effective problem representations are probably learnable and teachable skills, but they are not now generally taught in a systematic way.

8. Finally, it is becoming increasingly possible to determine in detail what is involved in understanding any specific subject matter area to the point of writing computer programs that specify what a person who understands knows, what processes he has available for solving problems and acquiring new knowledge in that domain, and how his knowledge is organized in memory.

REFERENCES

Banet, Leo. Balmer's manuscripts and the construction of his series. American Journal of Physics, 1970, 38, 821-828.

Banet, Leo. Evolution of the Balmer Series. American Journal of Physics, 1966, 34, 496-503.

Bobrow, D. C. Natural language input for a computer problem-solving system. In M. Minsky (ed.), Semantic Information Processing. Cambridge, Massachusetts: M.I.T. Press, 1968.

Ernst, G. and Newell, A. GPS: A Case Study in Generality and Problem Solving. New York: Academic Press, 1969.

Gelernter, H. Realization of a geometry theorem proving machine. Proceedings of 1959 International Conference on Information Processing, 273-282. Paris: UNESCO. (In E. Feigenbaum and J. Feldman, Computers and Thought. New York: McGraw Hill, 1963).

Hayes, J. R. and Simon, H. A. Understanding written problem instructions. In L. W. Gregg (ed.), Knowledge and Cognition. Potomac, Maryland: Lawrence Erlbaum Associates, 1974.

Katona, George. Organizing and Memorizing. New York: Columbia University Press, 1940.

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

Paige, J. M. and Simon, H A. Cognitive processes in solving algebra word problems. In B. Kleinmuntz (ed.), Problem Solving. New York: John Wiley & Sons, 1966.

Scandura, J. M., Durnin, J. H., and Wulfeck, W. H., II. Higher order rule characterization of heuristics for compass and straightedge constructions in geometry. Artificial Intelligence, 1974, 5, 149-183.

Simon, H. A. The functional equivalence of problem solving skills. Cognitive Psychology, 1975, 7, 268-288.

Simon, H. A. and Hayes, J. R. Understanding Complex Task Instructions. In David Klahr (ed.), Cognition and Instruction. Potomac, Maryland: Lawrence Erlbaum Associates, 1975.

Simon, H. A. The Sciences of The Artificial. Cambridge, Mass.: M.I.T. Press, 1969.

Simon, H. A. On judging the plausibility of theories. In B. van Rootselaar and J. F. Staal (eds.), _Logic, Methodology and Philosophy of Sciences III_. Amsterdam: North-Holland Publishing Co., 1968.

ACKNOWLEDGEMENT

This research was supported by Research Grant MH-07722 from the National Institute of Mental Health.