#498

In D.T. Tuma, & F. Reif, (Eds.), PROBLEM SOLVING AND EDUCATION: ISSUES IN TEACHING AND RESEARCH. Hillsdale, NJ: Lawrence Erlbaum Associates, 1980.

6 Problem Solving and Education

Herbert A. Simon
Carnegie-Mellon University

A central design issue, when we are planning learning experiences for our students (otherwise known as curriculum and course planning), is how much and what kinds of transfer of knowledge we can expect from the specific content of textbooks, lectures, and homework problems to the tasks that students will be expected to handle in subsequent courses and in professional life. If we think that basic, broadly transferable knowledge and skills are learnable and teachable, then we will aim our courses at such knowledge and skills, without too much concern for "covering" any specific information, topics, or techniques.

On the other hand, if we believe that the human capability for transfering knowledge and skill from specific situations to analogical but not identical situations is very limited, then we will be much more concerned, in planning courses and curricula, with predicting the exact kinds of problems our students will have to deal with, and with making sure that we "cover" these topics in our course materials.

In view of the enormous change in the world's knowledge that can take place in a student's professional lifetime, and of the constantly growing range of topics for which coverage can plausibly be demanded, it is hard to be optimistic about predicting students' specific future needs for knowledge or skill, or providing adequate coverage in already crowded curricula for a host of specific topics. Our teaching responsibilities urge us to a faith in transfer of training.

EVIDENCE ON TRANSFER

The empirical evidence for the transferability of knowledge and skills to new task situations is very mixed. The belief that students can be taught to "think logically" by offering them courses in Latin or logic was punctured by the celebrated studies of Thorndike in the 1920s (Thorndike, 1924). But Thorndike's research, and other studies in the same genre, did not prove transfer impossible (nor did Thorndike make such claims). They simply showed that certain specific kinds of instruction don't produce transfer. In the past half-century, we have learned a great deal about the conditions that could make it possible.

1. Transfer from Task A to Task B requires that some of the processes or knowledge used in Task B be essentially identical with some of the processes or knowledge that have been learned while acquiring skill in Task A. To take a trivially obvious example, in a college physics course, a student who has previously learned the calculus well will have an easier time solving problems that require the calculus than a student whose calculus skills are shaky. The calculus skills have been transfered to the physics performance because they are used directly in that performance. (Even here, of course, the transfer is not necessarily painless and automatic. The student may possess calculus skills without recognizing that they are applicable to a particular physics problem, or without knowing exactly how to apply them.)
2. To secure substantial transfer of skills acquired in the environment of one task, learners need to be made explicitly aware of these skills, abstracted from their specific task content. (Some of the evidence is reviewed in Woodworth and Schlosberg, 1954, pp.825-830.)

It might, indeed, be possible to teach logical thinking in Latin courses, provided that the students were given tasks in which such thinking was a major component, and provided that the processes of logical thinking that the students were acquiring were made explicit. Whether it would be *efficient* to teach logical thinking in this way is another question.

KNOWLEDGE AND SKILL

Another thing that research on cognitive skills has taught us in recent years is that there is no such thing as expertness without knowledge—extensive and accessible knowledge. No one, no matter how intelligent, skilled in problem solving, or talented, becomes a chess grandmaster without 10 years of intense exposure to the task environment of chess. Hence, in the training of professionals, the problem of coverage cannot be avoided. We cannot produce physicists without teaching physics, or psychologists without teaching psychology. I doubt whether anyone will want to dispute these propositions, but I mention them to qualify any later impression I may create to the effect that I believe that training in problem solving can compensate (at least compensate very far) for ignorance of subject matter.

We are just beginning to get a picture, through research in artificial intelligence and computer simulation of human thinking, of the amounts of knowledge the expert has available—of the size of his or her data bank. For example, the *INTERNIST* system now performs medical diagnoses, over the whole range of internal medicine, at a highly competent professional level (Pople, 1977). *MYCIN* has a similar capability for bacteriological diseases (Shortliffe, 1976). The knowledge stored in the data structures associated with these programs cannot be too dissimilar to the knowledge on these same topics possessed by a good human diagnostician. By constructing artificial-intelligence programs for professional-level tasks, we have acquired similar insights into the knowledge possessed by bank investment officers, by chemists who interpret mass spectrograms, and by professional chess players.

We do not yet have enough information about most of these data bases or the way they are represented in human memory, to provide a numerical measure of their size, in psychologically meaningful units like chunks. However, in one domain, chess, we do actually have a rough quantitative measure. A chess master has to have approximately 50,000 chunks stored in memory just to be able to recognize various configurations of pieces that are encountered repeatedly in games (Simon & Gilmartin, 1973). To explain what that means, I will have to say a little about what a "chunk" is and about the role of recognition memory within the total data structure (Simon, 1974, 1976).

A "chunk" is any perceptual configuration (visual, auditory, or what not) that is familiar and recognizable. For those of us who know the English language, spoken and printed English words are chunks. Even some hackneyed phrases may be stored as chunks: "In union there is strength" is a chunk for most of us. For a person educated in Japanese schools, any one of several thousand Chinese ideograms is a single chunk (and not just a complex collection of lines and squiggles), and even many pairs of such ideograms constitute single chunks. For an experienced chess player, a "fianchettoed castled Black King's position" is a chunk, describing the respective locations of six or seven of the Black pieces.

As a rule of thumb, human short-term memory can hold about four chunks at a time, but long-term memory an unlimited number. However, adding one new chunk to the long-term memory store requires about 8 seconds of attention. Adding a complex chunk that is itself a composite of a number of more elementary chunks may require several minutes, if the elementary chunks have to be acquired at the same time. Hence, even if memorizing activities were perfectly efficient, which they aren't, and if a person spent all his time at them, which he doesn't, acquiring 50,000 chunks would involve a substantial number of hours' application. When we remember that these recognizable patterns are only a fraction—and probably a small fraction—of the total body of knowledge the expert must have available, we see that we can account for a significant part of the 10 years' learning time he needs for acquiring his expertness.

However, if we carry out the arithmetic, it does seem that there are a good many more working hours in the year than are accounted for by this learning, and that we should be able to compress the learning processes by a factor of 10 or more. I shall return to this point later to express some skepticism as to the recoverability of much of this learning time. To anticipate, the 8-second parameter derives mainly from experiments on rote memorization, and memorization may be only a part of what is involved in skill acquisition. Meanwhile, the chunk parameter does provide us with an upper bound on learning rates.

Data on the rates at which schoolchildren are expected to acquire knowledge are consistent with estimates of the sizes of expert data bases. Japanese school children learn about 300 new ideograms per year (for words already known to them in the oral language), together with a substantial number of ideogram pairs. Learning each ideogram also involves learning several more elementary chunks. Reading books for American school children in the elementary grades typically introduce them to about 500 words each year. We made some informal estimates of the number of essential new "facts" that are introduced by each chapter in a college physics textbook and arrived at comparable estimates for a year's course—hundreds of such "facts," each fact involving a constellation of new chunks.

What is the role in the performance of complex tasks of this memory for familiar patterns? We may think of long-term memory as a large indexed and cross-referenced encyclopedia in which the articles are arranged irregularly, so that all information in the text must be accessed directly through the index or indirectly by means of cross-references (Simon, 1976). The familiar patterns are the index entries that enable the user, upon recognition of critical features of the problem situation, to evoke relevant knowledge from the body of the text. A skilled physician recognizes symptoms, which evoke possible courses of treatment. A chessplayer recognizes configurations of pieces, which evoke possible effective moves. An important part of professional skill appears to be embedded in such pairs of recognizable conditions linked to appropriate actions.

In recent years, a number of powerful applied artificial-intelligence programs have been constructed in which the knowledge is almost entirely embedded in such condition-action pairs. The technical AI name for a system using this architecture is *production system* (Newell & Simon, 1972). One of the main advantages claimed, and to some extent demonstrated, for production systems is their extensibility. New knowledge is added to such a system simply by inserting new productions—new text and new index entries to access it. Moreover, the homogeneity of the system facilitates the designing of completely general processes to operate on it. All is not as simple as this sounds; nevertheless, the purported advantages have proved not to be illusory. Production systems today provide both the most promising formalism for applied AI systems that require large data bases, and the most plausible model for human learning of professional subjects.

GENERAL PROBLEM-SOLVING SKILLS

If professional knowledge is largely incorporated in large production systems—in an indexed encyclopedia—one might well wonder what place there is in professional skill for general problem-solving methods, and whether there are any such general methods to be learned or transfered.

Indeed, some rather extravagant statements have appeared in the AI literature that could be interpreted to assert that general methods are unimportant. For example, Goldstein and Papert (1977) have said: "Today there has been a shift in paradigm. The fundamental problem of understanding intelligence is not the identification of a few powerful techniques, but rather the question of how to represent large amounts of knowledge in a fashion that permits their effective use and interactions.

The error in this claim lies in its either-or stance. Two-bladed scissors are still the most effective kind. In addition to the large body of knowledge that is represented in semantically rich systems, there have to be processes for operating on that knowledge to solve problems and answer questions. When the knowledge is incorporated in condition-action pairs, as it is in production systems, these pairs themselves represent processes: When the conditions of such a pair are satisfied, the actions are executed. In a pure production system, then, the general methods are embedded in the productions themselves, in the form taken by the conditions and the actions. If we examine large systems like DENDRAL and MYCIN, we find incorporated in the productions and their organization the standard general-purpose problem-solving techniques of artificial intelligence: for example, the hypothesize-and-test method, means-ends analysis, and best-first search (Feigenbaum, 1977).

Bare facts, however they are stored in memory, do not solve problems. However intimately data and process are intermingled in them, the components of production systems, the individual productions, are processes, not simply pieces of data. And these processes will be effective precisely to the extent that they embody the "few powerful techniques" that Goldstein and Papert dismiss as unimportant.

In addition, most of the large production systems that have been built so far (like the two mentioned previously) are specialized to a particular class of problem-solving tasks, tasks we may describe as taxonomic, using data primarily for purposes of identification. DENDRAL uses mass spectrogram and nuclear magnetic resonance data to identify the molecular species that produced the data. MYCIN uses symptomatic manifestations to identify bacterial diseases. In both cases, the system does what the index of our hypothetical encyclopedia does: It recognizes familiar patterns. And this is just what productions do: recognize when their conditions are satisfied. In systems designed to do other kinds of problem solving, the general methods used, in addition to the production-system architecture itself, are likely to be more conspicuous.

I have commented on this issue at some length because it has important implications for education, in general, and for the topic of my chapter, in particular. If it were true that the storage of properly represented and organized bodies of knowledge were the whole story, then we could return peacefully to our preoccupation with subject-matter content as the main concern in curriculum-building and teaching, and we could give up our efforts to teach general techniques for problem solving that would be transferable from one domain to another.

The evidence from close examination of AI programs that perform professional-level tasks, and the psychological evidence from human transfer experiments, indicate both that powerful general methods do exist and that they can be taught in such a way that they can be used in new domains where they are relevant. We reassert our earlier conclusion that the scissors does indeed have two blades and that effective professional education calls for attention to both subject-matter knowledge and general skills.

LEARNING PROCESSES: ADAPTIVE PRODUCTION SYSTEMS

Artificial intelligence research on the *performance* of problem-solving tasks is far ahead of research on how problem-solving skills are *acquired.* In spite of the enormous amount of research that has been done in psychology on the topic of learning (mainly within a behaviorist tradition), the amount of knowledge we have about the information processes involved in solving problems is large compared with the very modest amount we have about the information processes involved in learning. Any proposals to teach general problem-solving skills must take account not only of the role such skills play in problem-solving performance but also of what we know about how they can be acquired.

If a problem solver is organized as a production system, a discrete set of condition-action pairs, then learning can occur by the addition of new productions, by the deletion of productions, or by rearrangement of the order in which the conditions of products are tested (Waterman, 1970). Systems that have the capability of modifying production systems automatically in one or more of these ways are called *adaptive production systems* (Rychener et al., 1977). Systems that permit human users to insert, delete, or modify productions interactively are called *instructible production systems.* Adaptive and instructible production systems provide two models of how learning can take place: a self-instruction and a teaching model.

Students learn both by being taught and by self-instruction—in varying mixes for different students. A priori, it might appear that it would obviously be easier to learn with the help of a teacher than by self-instruction. The teacher could simply "provide" the student with the appropriate program—whether organized as a production system or otherwise—which the student could then internalize. This notion rests on what might be called "the fallacy of rote memorization." there is no direct way in which the words pronounced by a teacher can be stored directly as productions available to the student. There must be a conversion of the external language into the internal representation of the student's production system, and neither he nor the teacher know explicitly what that representatin is. Rote memorization, as we know all too well, produces the ability to repeat back the memorized material but not the ability to use it in solving problems.

In the instructible production systems of artificial intelligence, this difficulty can be by-passed, since the way in which information is stored internally is known to the user or the programmer (Feigenbaum, 1977). A simple "front end" can be provided (e.g., the TEIRESIAS front end for MYCIN) that converts productions stated in the user's language (e.g., a subset of English) into productions represented in the programming language (e.g., LISP in the case of MYCIN). In human learning, this conversion program must be provided by the learner.

If we admit this much, then we can see that self-instruction need be no more difficult, and no different in kind, from learning with the help of a teacher. We simply replace the teacher's oral words with the written words of the textbook. Of course I am oversimplifying. The textbook cannot monitor the performance of the student, observe his difficulties, and modify the instruction accordingly.[1] Nevertheless, the processes of converting input language into usable skills must be basically the same in the two cases, and

[1]Programmed texts, at their present state of development, can do a little of this but only a trivial amount. A few automated tutorial programs, of which SOPHIE is an example, take several more steps toward simulating the teacher's capabilities.

what we learn about doing the one should be useful in understanding how the other is done.

Our proposed model of a human learner, then, becomes an adaptive production system that can take a textbook as its input and acquire the skills that the textbook is undertaking to teach. At least in scientific and technical subjects, two of the ways in which students learn from textbooks are: (1) examining closely the worked-out examples that the textbook provides; and (2) working the problems at the end of the chapter. Textbooks contain much other material besides worked-out examples and problems, but it is my experience, both as teacher and learner, that these two classes of items occupy a large fraction of the productive learning time.

LEARNING FROM EXAMPLES

Consider the chapter in an algebra textbook that explains how to solve a single linear algebraic equation in one unknown. The first part of the chapter will set out the basic operations that may be performed on an equation without changing its solutions: adding, subtracting, multiplying, or dividing, with the same number on both sides of the equation. Each of these operations will be illustrated by one or more examples. The remainder of the chapter will be devoted to the algorithm for solving a linear equation, using these basic operations. This will mostly be done by example, starting with easy cases and gradually generalizing to more difficult ones.

A clever student, after studying no more than a couple of the examples, will have acquired an algorithm for solving linear equations that will probably be with him for the rest of his life. He may confirm and practice the skill by working problems at the end of the chapter, but the whole process can be over remarkably quickly. Less talented students may follow the same process successfully but less quickly and more painfully (i.e., with more false steps along the way). What can we say about the processes that are going on here?

David Neves (1978) has constructed an adaptive production system that simulates this learning process. Let us suppose that the basic operations for manipulation of equations have already been mastered, and that the system is provided with the following worked-out example:

1. $5X + 2 = 3X + 8$
2. $5X = 3X + 8 - 2$
3. $5X = 3X + 6$
4. $5X - 3X = 6$
5. $2X = 6$
6. $X = 3$

The textbook will usually also supply the information that the second expression was obtained from the first by subtracting 2 from both sides of the equation; the third from the second by combining the numerical terms on the right-hand side; and so on. The system begins by comparing the first two expressions and by noting the differences between them. It notices that the left-hand side has lost a numerical term and that the right-hand side has gained a term. It notes also that the operation performed was to subtract that numerical term from both sides. From these facts, it infers and constructs a new production:

> If there is a numerical term, *N*, on the left-hand side, then subtract *N* from both sides.

The same process is followed with the second and third expressions. The production derived here might take the following form:

> If there are two distinct numerical terms on the right-hand side, *N* and *M*, say, replace them by their algebraic sum.

The system repeats the process for each successive pair of expressions, producing, in all, five different productions. This set of productions, in their order of discovery, constitute a fairly general system for solving a linear algebraic equation in one unknown, as the reader can verify by hand-simulating its performance with other equations. Hence, having built this system of productions, the system can now solve most of the problems at the end of the textbook chapter.

There are no end of troublesome details I have skipped over in my description of this learning procedure, which makes its execution less trivial than may appear from the example, but I have illustrated the main principles. One "detail," of course, is that the system has to select the appropriate level of generalization in inducing the general rule from the particular numbers and letters that appear in the example.

Let us see what general ideas about learning can be extracted from this example. First, it is clear that the procedure rests on quite general principles that are not restricted in their application to this particular case. The logic of the learning system might be paraphrased as follows:

The solution of an equation is an expression consisting of an *X* followed by an equals sign followed by a real number. If we are given two expressions, one of which lies closer to this goal than the other along a solution path, find the differences between the two expressions, find the operator that converted the first into the second, and find in the first expression any parameter or parameters used to instantiate the operator. Now create a new production.

The left-hand side should consist of a test for one of the differences that distinguished the first expression from the second, and the difference tested for should contain the parameter mentioned previously. The right-hand side of the production should consist of the operator that converted the first expression into the second, appropriately instantiated. Now change the constants that appear in condition and action of the production into variables of the appropriate types.

LEARNING BY DOING

Anyone familiar with problem-solving systems like the General Problem Solver (GPS) will recognize the logic of the system as an application of means-ends analysis (Newell & Simon, 1972). We have a certain situation (the first expression of a pair); we want a different situation (the second expression). We detect a difference between the two situations, and we search memory for an operator that is relevant for reducing differences of this kind. We apply the operator and examine the resulting expression to see if the difference has been removed. (In the case of the worked-out example, a successful outcome is guaranteed.) This is a particularly simple case of means-ends analysis, because only a single application of a single operator is needed to produce the result; hence, there is no danger that the system will embroil itself in a large search for the answer. This is precisely the point in exhibiting all the intermediate steps in the example. Of course, this can be done at various levels of detail. For example, in the instance we have used, it may be assumed that the student will always simplify expressions "automatically" (i.e., that each arithmetic operation in an equation will include the subsequent simplification step) so that expressions (2) and (4) need not be exhibited.

Omitting the low-level details suggests the next step of generalization. Even if only expressions (1) and (6) were exhibited, the logic of the program would still apply. In fact, we would not need the exact detail of expression (6), but only its form: an expression consisting of *X* followed by equals followed by a real number. Then a GPS-like adaptive production system could still construct a set of productions for solving algebraic equations by successively removing a whole set of differences between (1) and (6), and not just a single difference. This might require some search and some guidance of that search; hence, the adaptive GPS would have to have a few additional capabilities in addition to those possessed by the system that always had a worked-out example available.

We may view the adaptive GPS in a slightly different way (Anzai & Simon, 1979). Suppose that we have a quite flexible and general problem-solving system, which depends mainly on selective search and means-ends analysis to solve problems. It probably will not be very powerful in any particular problem domain, relying as it does on weak general methods. However, it may be able to work its way, by brute force and awkwardness, through problems in some domains that are new to it. Having found at least one solution in this way, it can ignore the various false trails it followed in its search and can focus its attention on the successful solution path. But this solution path is precisely a worked-out example, and the same procedures that were used to build a production system from such an example can be applied to it.

Finally, there are situations in which the adaptive GPS can succeed even though it cannot initially find a problem solution. It may be able to detect when it has made some progress toward a solution or, alternatively, when it is searching ineffectively (e.g., when it has gone around a loop). Then, it can create new productions that exploit its progress or avoid its inefficient behaviors. These new productions may increase the efficiency of its search until it can solve the problem—at which point we are reduced to the case of the worked-out example. Anzai and Simon have constructed a system that is capable of learning to solve the Tower of Hanoi problem in an efficient way, starting only with general problem-solving procedures of the kind just indicated, together with adaptive means for profiting from its progress and its errors.

LESSONS FOR INSTRUCTION

What lessons may we plausibly draw from these experiments? First, the adaptive production systems I have described provide an explanation of how learning from worked-out examples and from problems can take place. Second, they show that means-ends analysis, a central component of general problem-solving processes, plays a crucial role in learning. Hence, they give us some further insight into the respective roles of general skills and special knowledge in achieving high-level performance.

If our account is basically correct, then general skills (e.g., means–ends analysis) will be particularly important in the learning stages but will also show up implicitly in the *form* of the productions that are used in the skilled performance. The production system we have sketched for solving algebraic equations is a specialized instantiation of the general means-ends schema. Aside from their embodiment in specialized productions, general problem-solving techniques may not be evident in the professional's skilled behavior when he is engaged in relatively familiar tasks. They are likely to play an essential role whenever he has to move into new territory and attempt new learning.

The experiment also contains an important lesson for teachers and the writers of textbooks—a lesson that may be stated in terms of the classical dichotomy between rote learning and meaningful learning. Or perhaps I should call it a "hypothesis" rather than a lesson. It is a rule of algebra that you may subtract the same expression from both sides of an equation. The rule gives no hint of the circumstances under which a problem solver should avail himself of that privilege. The production system derived from the worked-out example gives exactly that kind of advice. It says that, if the goal is to solve the equation for X, and if there is a numerical constant on the left-hand side of the equation, the subtraction rule should be used to remove the constant. If you will examine some typical algebra textbooks, you will find that all of them make the rule of addition completely explicit, but you will almost surely find, also, that the *conditional* rule, which tells when the operation is appropriately applied, is only implicit or, if mentioned, is not much emphasized or elaborated.

This is not an isolated case. Generally speaking, textbooks are much more explicit in enunciating the laws of mathematics or of nature than in saying anything about when these laws may be useful in solving problems. The actions of the productions needed to solve problems in the domain of the textbooks are laid out systematically, but they are not securely connected with the conditions that should evoke them. It is left largely to the student (by examining worked-out examples, by working problems, or in some other way) to generate the productions, the condition-action pairs, that are required to solve problems. Hence, if the student mainly memorizes, learns by rote, the laws stated in the textbook, he or she will acquire little or no problem-solving ability.

We can see the same thing in diagnostic skills as in more general kinds of problem-solving skills. A physician needs to know how to go about treating the various diseases he may encounter in his patients. But that knowledge is of no use to him unless it is evoked by the presence of the appropriate symptoms. A large part of his medical knowledge has the form of condition-action pairs, in which the symptoms of disease are the conditions, and in which the treatments associated with the corresponding disease entities are the actions. On the basis of casual examination of textbooks in medicine and in various branches of taxonomic biology, I gather the impression that there is somewhat more attention given to expressing knowledge in these fields in the form of productions than there is in branches of science that are largely concerned with the exposition of natural laws.

Of course, I expect that productions, statements of the relation between conditions and recommended actions, can also be learned by rote—that is, by learning the verbal rules without acquiring the ability actually to perform the tests incorporated in the conditions. I am not suggesting that making the condition sides of our productions explicit provides any royal road to learning.

TEACHING PROBLEM SOLVING

The lessons for pedagogy that can be extracted from my remarks are clear enough that they don't need to be explicated at great length. A first lesson is that we can best think of most skills—both general skills and competence in specific subject matter—as being represented in productions rather than propositions. If this is true, then we need to teach our students that this *is* the form that most professional knowledge takes, so that they can monitor their own learning more effectively and not mistake learning propositions for acquiring basic skills. I continue to encounter many students who do not really know the difference between rote learning and learning with understanding, and who wonder why, after they have memorized some material with great thoroughness, they cannot pass examinations that require them to solve problems.

Second, there is a small arsenal of general problem-solving procedures that have been identified from research in psychology and artificial intelligence. In this chapter, I have particularly stressed one of them—means-ends analysis. Since we know that skills will be transfered only when the principles on which they rest are made explicit, these procedures need to be made evident for students, and then they need to be practiced, and practiced again. We are engaged here not merely in knowledge acquisition but in skill acquisition. All of our experiences tell us that people do not learn to ride bicycles simply by having the principles explained to them. What they must mostly do is to ride them. Similarly, the active practice of problem solving, but problem solving with awareness, should be the main core of a problem-solving course, or of problem-solving instruction in a subject-matter course.

Third, a considerable part of the training in problem solving *can* profitably take place in subject-matter courses. This has two advantages. First, it facilitates the transfer of skills that have been learned outside the subject-matter context. Second, it permits attention to be given to the whole continuum of skills, from the most general to the most specific technique, without focusing on just a single part of the spectrum. I think that at Carnegie-Mellon University, some of the most effective training in problem solving that has been done over the years has been carried out in engineering analysis courses within the Engineering College. In courses like these, students learn not only how to use means-ends analysis, or functional analysis, but also how to analyze a wide range of situations in terms of broad physical principles like conservation of energy.

Fourth, instruction in problem solving should place a considerable emphasis upon such techniques of self-instruction as learning from worked-out examples and learning by working problems; for it is precisely in the context of such learning situations that the student's general problem-solving techniques, as distinct from his special skills, are called upon most heavily. Such instruction will also help to alert students to the ways in which general techniques are embedded implicitly in the formats of the productions that carry their special knowledge.

Finally, the perceptual aspects of skill probably deserve increased emphasis. I have suggested that, in our textbooks and in our teaching, we tend to underemphasize the condition sides of the condition–action pairs. We need to help our students improve their skills of recognition and to help them acquire techniques for exercising those skills, so that if they have learned what to do, they will not be slow in recognizing when to do it.

Many of the points I have been making are familiar enough in current ways of teaching problem solving. From Polya down to recent textbooks, we have emphasized the working of examples and the explication of problem-solving principles. My emphasis upon perceptual skills, and upon the application of problem-solving to self-instructional tasks, are perhaps more novel. In any case, we are unlikely to find idle time on our own or our students' hands in either the general course on problem solving or in more special courses that try to teach problem-solving skills in a subject-matter context.

CONCLUSION

After entering all of the appropriate caveats about transfer, and after taking account of the essentiality of specialized knowledge for skilled professional performance, I believe that a strong case can still be made for teaching problem solving explicitly, as an important component of professional education. It must be taught, of course, in the context of a rich environment of problems—mostly but not entirely drawn from the professional field in question.

In teaching problem solving, major emphasis needs to be directed toward extracting, making explicit, and practicing problem-solving heuristics—both general heuristics, like means—ends analysis, and more specific heuristics, like applying the energy-conservation principle in physics. It is desirable for students to become aware of how heuristics are organized in memory, as sets of productions that provide not only a repertory of problem-solving actions but also conditions, associated with these, that serve to index the actions and to evoke them when they need to be used. A student's own learning processes will be enhanced if he understands that a large part of his professional skill resides in the ability to recognize rapidly the situational cues that signal the appropriateness of particular actions. In this learning, he needs to pay specific attention to the acquisition of these recognition skills.

A major goal in teaching problem solving should be to help the student acquire skills of self-instruction. In examining these skills in the context of recent AI research on learning, we see that they are intimately bound up with general problem-solving skills. Self-instruction involves being able to learn from worked-out examples and by working textbook problems. Doing this employs means-ends analysis, and the production systems that evolve during such learning embody means-ends analysis implicitly.

There is some empirical evidence that problem-solving skills can be taught, although there is regrettably little evidence that such instruction is cost-effective, as compared with equal effort devoted to subject-matter skills. As an empirical scientist, I could wish that the evidence were stronger. As a practical teacher, I am satisfied that, as we continue to learn more about the nature of problem-solving processes, we will be able to teach general problem-solving skills with increasing effectiveness and be able to circumvent the unsolvable problems of coverage and of predicting what specific knowledge our students will need 30 years hence.

ACKNOWLEDGMENT

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

REFERENCES

Anzai, Y., & Simon, H. A. Strategy transformations in problem-solving: A case study. *Psychological Review*, 1979, in press.

Feigenbaum, E. A. The art of artificial intelligence: Themes and case studies of knowledge engineering. *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, Pittsburgh, Pa.: Department of Computer Science, Carnegie-Mellon University, 1977.

Goldstein, I., & Papert, S. Artificial intelligence, language, and the study of knowledge, *Cognitive Science*, 1977, *1*, 84-124.

Neves, D. M. A computer program that learns algebraic procedures by examining examples and by working problems in a textbook. *Proceedings of the Second National Conference of the Canadian Society for Computational Studies of Intelligence*, 1978.

Newell, A., & Simon, H. A. *Human problem solving*. Englewood Cliffs, N.J.: Prentice-Hall, 1972.

Pople, H. Problem solving: An exercise in synthetic reasoning. *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, Pittsburgh, Pa.: Computer Science Department, Carnegie-Mellon University, 1977.

Rychener, M., et al. Problems in building an instructible production system. *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*. Pittsburgh, Pa.: Computer Science Department, Carnegie-Mellon University, 1977.

Shortliffe, E. *Computer-based medical consultations: MYCIN*. New York: Elsevier, 1976.

Simon, H. A. How big is a chunk? *Science*, 1974, *183*, 482-488.

Simon, H. A. The information-storage system called "human memory." In M. R. Rosenzeig & E. L. Bennett (Eds.), *Neural mechanisms of learning and memory.* Cambridge, Mass.: MIT Press, 1976.

Simon, H. A., & Gilmartin, K. A simulation of memory for chess positions. *Cognitive Psychology,* 1973, *5,* 29–46.

Thorndike, E. L. Mental discipline in high school studies, *Journal of Educational Psychology,* 1924, *15,* 1-22; 83-98.

Waterman, D. Generalization learning techniques for automating the learning of heuristics. *Artificial Intelligence,* 1970, *1,* 121-170.

Woodworth, R. S., & Schlosberg, H. *Experimental psychology* (2nd ed.). New York: Holt, Rinehart & Winston, 1954.