*Behavior Research Methods & Instrumentation*
*1979, Vol. 11 (2), 102-110*

# PRESIDENTIAL ADDRESS

DANIEL E. BAILEY, *University of Colorado, Presider*

# Teaching a computer to teach

RICHARD MILLWARD
*Brown University, Providence, Rhode Island 02912*

The long-range goal of computer-assisted instruction (CAI) should be to construct computer programs that interact with a student as a human tutor does. In order to make computers act as intelligent tutors, we must consider three aspects of teaching: (1) A tutor makes use of many different knowledge sources. (2) A tutorial session is a kind of problem-solving situation in which the tutor is trying to solve the problem of imparting understanding to the student. (3) There is an analogy between the process of communication and those of teaching in that both require the participants to make a number of inferences about each other's states and responses. A tutor designs an "instructional act," anticipating how the student will interpret and learn from it, and the student interprets and learns from the instruction in a manner that depends on his hypotheses about the intent of the tutor. The paper reviews the relevant literature and discusses how we will incorporate such ideas into CAI systems.

We are entering a new era in psychology, an era in which hard questions are being asked about complex processes for which the traditional experimental methodology is often inadequate. For example, the area known as human factors has for years relied primarily on psychophysics and learning paradigms. Today, human factors have given way to knowledge engineering. What is knowledge engineering? A simple definition might be "the building of intelligent systems to be run on computers." Some examples would be speech understanding systems (Klatt, 1977), diagnostic consultant systems for physicians (Shortliffe, 1976), and intelligent computer-assisted instruction systems (Brown & Burton, 1975). Each of these systems involves questions of concern to psychologists and each represents developments that are important to theoretical psychology.

My own interest (Millward & Spoehr, 1973; Millward & Wickens, 1974) in how people learn and use concepts led me to look at the work in artificial intelligence, where, it seemed, a different approach was being taken to how humans understand, represent, and utilize concepts. I felt that computer-assisted instruction (CAI) required the development of conceptual systems at least as complex as those I had previously been considering, and the rigor provided by implementing the CAI systems seemed salutory. So, about 3 years ago I turned some of my attention to artificial intelligence in general, and CAI in particular.

CAI, or teaching a computer to teach, requires three basic steps: (1) The problem must be stated exactly. (2) An algorithmic solution to the problem must be found or some plausible heuristic must be designed. (3) The solution must be implemented in some software system. The difficulty in teaching a computer to teach lies primarily in the first step, that is, in stating the problem. However, despite numerous recommendations for top-down programming, it did not seem plausible to try to work out all the details of the teaching process before I began to develop some simple tutorial programs.

I began with a plan to build a CAI system to teach BASIC (Millward, Mazzucchelli, Magoon, & Moore, 1978). However, our limited understanding about what a tutor *does* while teaching led me to modify my goal and to reduce the scope of the project. Two other factors also deterred efforts to build overly elaborate CAI systems. One was the inadequacy of current programming languages for representing abstract ideas about complex processes. The best existing language is still on the drawing boards and inaccessible (Bobrow & Winograd, 1977). The second factor was an insufficient behavioral analysis of how students learn textbook knowledge.

Therefore, instead of trying to do CAI directly, I shifted to an analysis of the behavior of a tutor in a tutorial situation, because one cannot put a tutor into a computer until he knows what a tutor does.

Now, let me explain the title of this talk. How are we going to teach a computer to teach? Actually, we are not going to. Instead we are going to have to discover what it means to teach and what it means to learn; then we will put that knowledge into the computer. But, like building bridges, one does not wait until he has a theory adequate to design the Verrazano Bridge before he builds any bridges at all. The theory involved in any engineering task often develops with the practice itself. Similarly, as we study how a tutor teaches, we try to integrate what we learn into a CAI system. Since both experiments and building model systems require a great

Copyright 1979 Psychonomic Society, Inc.

0005-7878/79/020102-09$01.15/0

deal of time, our strategy is to look at tutor-student systems involving, at most, a single hour session of tutorial. A sample problem might be to teach a student about binary numbers.

Figure 1 illustrates a situation in which a tutor and a student interact with a computer. We can constrain the interactions in various ways in order to look at different aspects of how people teach and learn and how computer aids to teacher-student interactions can be introduced most beneficially. The reason for the two computers is that the CAI system requires a list-processing language not available on the PDP-8. The PDP-8 is programmed to look at both student and tutor, as well as to interact with the IBM 360 at a 9,600-baud rate. In addition, the PDP-8 controls a large number of peripheral devices that are useful as teaching aids. Examples include projectors for slides of real scenes and oscilloscopes for line drawings.

There are a number of ways to vary this design. To indicate some of these, I will discuss four possible research paradigms. Of course, the variables that make these paradigms different can be combined in many ways, leading to a whole space of paradigms. At one extreme, the tutor and the student work side by side on the computer, allowing both visual and verbal communication.

A second, more constrained paradigm separates the tutor and the student completely, so that the student is unaware of the tutor's presence. Under this paradigm, the student believes the computer is doing all the teaching, even when the tutor actually is producing some of the output for the computer. The important point here is that the tutor can act intelligently on occasions when the CAI system cannot. The computer system can teach more smoothly, providing the atmosphere of proficiency

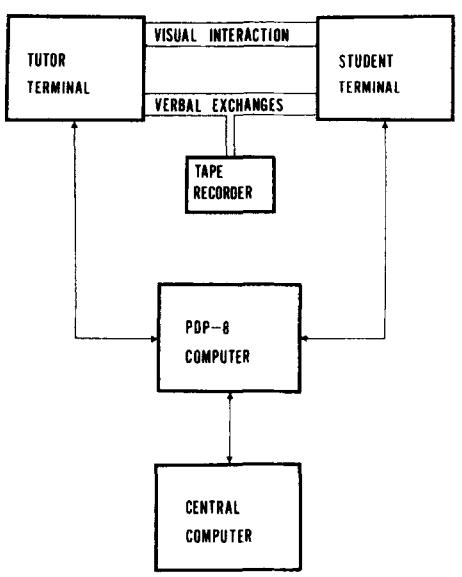


**Figure 1. An experimental system for monitoring tutor-student interactions.**

necessary to insure the student's confidence. In this paradigm we monitor and analyze the actions of the tutor in order to transfer the simpler of these actions to the CAI system. We would, of course, like to build a computer tutorial system that would mimic a tutor completely, but that is an ideal very far from being realized.

The third paradigm uses the computer primarily as an aid to teaching. The idea is to provide the computer with both informational and computational resources which the human tutor lacks. Such resources can be made accessible only to the tutor or only to the student, or to both. These resources include computational algorithms, graphic routines, information-retrieval systems, dictionaries, and statistics on the student's performance.

The fourth paradigm, a combination of Paradigms 1 and 2 is the one being used in our current research. The tutor-student interactions are limited to verbal exchanges and to messages transmitted through the computer. The student interacts with a limited CAI system on the central computer. However, the tutor monitors and modifies as necessary every student response before it is sent to the central machine. We usually let the output from the central machine go to the student without delay. The tutor can send meassages to the student and make comments for the record on the student's actions. The student can also ask questions of, or send messages to, the tutor. All computer interactions are recorded for later analysis. In our research, we record and monitor the behavior of the three components of this system—the tutor, the student, and the computer—and use the various constraints in the system as a kind of independent variable.

When a tutor and his student sit down together in a tutorial session, a number of things happen that span a wide range of psychological phenomena. First, there is almost always some chit-chat and general socializing. Then the lesson begins. The teacher asks some questions about previous work to determine whether the student has an adequate background for the material to be presented. The tutor then presents a number of ideas and asks more questions until the student has learned the lesson.

How do we characterize the tutor's behavior during these exchanges? A naive answer is that the tutor selects and presents a set of ideas about the subject matter and the student learns them. Although superficially this is a correct answer, it hardly captures the dynamics of the tutorial episode. But conventional CAI is too often based on just such an analysis. Conventional CAI usually focuses on the material to be presented and the sequence of presentation. The missing ingredient in CAI is understanding. A real tutor is not interested so much in whether the student remembers everything he said as in whether the student understands the subject. I think we all recognize the elusiveness of the word "understanding." The rest of my talk will concentrate on how

we might make a tutorial CAI system that knows when a student understands.

There are three factors in my analysis of the problem of creating an intelligent CAI system. Perhaps the most important factor is that of making explicit the types of knowledge used by both the tutor and the student. The second factor is the problem-solving behavior engaged in by the tutor. Finally, the third factor is natural language communication, without which there can be no effective teaching.

Figure 2 presents a listing of some of the types of knowledge which I believe are essential to the analysis of tutorial instruction. The tutor is characterized as having: (1) knowledge of the subject matter to be taught, (2) world knowledge, (3) knowledge of the student's knowledge and abilities, (4) self-knowledge, (5) knowledge of how to teach, and (6) knowledge of social conventions. A brief description of each of these types of knowledge should make it clear how they are used in the tutorial situation.

The tutor's knowledge of the subject matter is probably rather explicitly structured and is further supported by external reference material. The problem of putting such information into a computer has not yet been completely solved, since by "knowledge" we mean more than storing the material in text form. The CAI system must be able to use the material to generate questions for the student, to answer questions asked by the student, to make inferences, and to add new knowledge to old—uses that constitute understanding.

To some extent, general knowledge about the world is a catchall term for knowledge that cannot be conveniently placed anywhere else. Its essential characteristic is that it is knowledge shared by the tutor and the student. The importance of world knowledge is that a tutor builds new knowledge structures onto an old knowledge base, and, for some topics, a tutor may tap some very miscellaneous ideas in an attempt to teach a student a new idea. For example, in teaching binary arithmetic, it is natural to use the facts about decimal arithmetic. But when teaching about a push-down stack, the cafeteria-tray storage stack is an excellent analogy. We use analogy constantly to express ideas. If the analogy is to be effective, the relationship between the familiar and the unfamiliar must be clear. Most instruction depends heavily on world knowledge, and representing world knowledge is one of the hardest problems to solve in developing intelligent CAI systems.

TUTORIAL
KNOWLEDGE SOURCES

1. KNOWLEDGE OF THE SUBJECT MATTER TO BE TAUGHT
2. EVERYDAY KNOWLEDGE ABOUT THE WORLD
3. KNOWLEDGE ABOUT THE STUDENT'S KNOWLEDGE AND ABILITIES
4. THE TUTOR'S KNOWLEDGE ABOUT HIS OWN KNOWLEDGE
5. KNOWLEDGE ABOUT HOW TO TEACH AND COMMUNICATE
6. KNOWLEDGE OF SOCIAL CONVENTIONS

**Figure 2. Knowledge sources used by a human tutor.**

A tutor normally knows a great deal about his student's knowledge and abilities. This knowledge is generally rather implicit and is gained by the tutor as a function of experience. The cues a tutor uses for sizing up his student come from linguistic sources (the level of the student's grammar, his vocabulary, his articulateness), from socioeconomic information, from explicit facts about the student's history, from his very aptness for learning. All these inputs lead to different models for different students and, in turn, to different instructional strategies and expectations.

Another important component is the tutor's model of the student's knowledge of the subject matter. The tutor selects material to present to the student on the basis of a difference between the tutor's knowledge of the subject matter and the tutor's model of the student's knowledge of the subject matter. Included in the tutor's model of the student's knowedge are examples of potentially erroneous ideas which could produce certain kinds of errors.

One of the highly touted advantages of CAI is that it can be individualized. However, if none of these student variables is used explicitly by the system, a CAI program cannot be said to provide individualized instruction.

Both the tutor and the student make use of self-knowledge. A tutor uses self-knowledge to provide a stopping point in presenting material. Knowing what one knows keeps one from wandering off into areas where the facts are missing. Students use self-knowledge to focus their questions on areas where they are confused or sense a gap in their knowledge.

Knowledge of how to teach, or the techniques used by the tutor to transfer knowledge and understanding, is part of problem-solving and communication activities. Many of these techniques involve subjective judgments, for example, sensing that the student is confused. There are techniques for organizing material into meaningful structures. There are techniques for generating critical examples to clarify an idea being presented. Unfortunately, we simple do not have enough data yet to say much about these strategies.

The relationship between a student and his teacher reflects their knowledge of social conventions. For example, both tutor and student know that it is permissible for the teacher to ask some kinds of personal questions as a polite way of making conversation; for example, "Are you still having trouble with your room-

mate?" But by convention, a student does not have this option. Can you imagine a student asking his tutor, "Are you still having trouble with your chairman?" Does a computer need to capture any of these social interrelationships? At this time I am not sure whether we have to, or even whether we can, but these are certainly factors that create a difference between a machine tutor and a human tutor.

What are the choices we must make when we try to represent knowledge in a computer? One choice is of the appropriate representational formalism. We have a variety of possible formalisms: simple associational structures, set-theoretical structures, predicate calculus, featural and hierarchical organizations, networks, schemata-scripts-frames, beta structures, and production systems. (See Anderson, 1976; Rumelhart & Ortony, 1977; Shaw & Wilson, 1976; Winograd, 1977a.) These are not all equally easily implemented in a computer and our choice might depend on implementation restrictions.

The choice of a representational system is not independent of the function to which it will be put. In knowledge engineering, unlike more algorithmic programming, the functions are rather loosely defined. While a CAI system is functioning, knowledge must be used to control decisions, knowledge must be retrieved, and new knowledge must be added to the system. For example, the tutor's model of the student's knowledge must be continuously updated. Some types of knowledge must be reorganized.

The choice of a representation must also take into consideration the total amount of knowledge and the rate of increase in it. Since many of the types of knowledge mentioned above are large and changing, there are strong constraints on the representational formalism.

A final issue is that of the interrelationships among the various types of knowledge. A uniform representation has the advantages of allowing easy cross-referencing of the ideas represented in the different knowledge types and of helping to eliminate duplicate storage. However, multiple representations take advantage of the different requirements to which the knowledge will be put, as well as reflect the input modality of the information. This fact is the basis of the arguments for a human visual memory system different from a verbal or proposition memory. There are no easy solutions to this and the other problems of representation. Only experience will help us solve most of these problems.

The types of knowledge just described are fundamental to any analysis of the tutorial process. Representing any one of these types adequately is difficult; representing all of them and their interactions appears well beyond our abilities at this time. Many researchers in psychology, linguistics, and artificial intelligence are working on such problems. Although their work is too extensive to be summarized here, one example gives some idea of the state of the art at this time. Greeno's (Greeno, 1976, 1977, 1978) work is particularly interesting because he is explicitly considering the representation problem as it relates to education.

Greeno (1976, 1978) discusses a number of different types of representational systems for subject-matter knowledge. One is a procedural representation in which a concept is represented as a sequence of actions. For example, one can think of the actions involved in adding two fractions as a sequence of operations. These could easily be specified in a production system. However, geometry is more complex than the addition of fractions, and Greeno suggests a number of different representational ideas. The relationships between the elements of a geometric figure, the lines and angles, might be coded by a discrimination net. On the other hand, there are propositional facts about geometry whose interrelationships are such that a network is the appropriate structure for them.

Greeno also considers more advanced topics such as auditory psychophysics. Figure 3 shows these represented as a network of propositions, as might be done in Norman and Rumelhart's (1975) Memod theory. The nodes in Figure 3 are the relations, and the links are the events or objects upon which the relations act. Greeno's use of diagrams in Figure 3 (e.g., the pressure wave) is meant to indicate that students might have generative processes by which they can put together simple elements into a coherent picture-like representation. Figure 3 is obviously incomplete, but it nevertheless illustrates the kind of detailed structural organization required for more complex subject matters. In a more

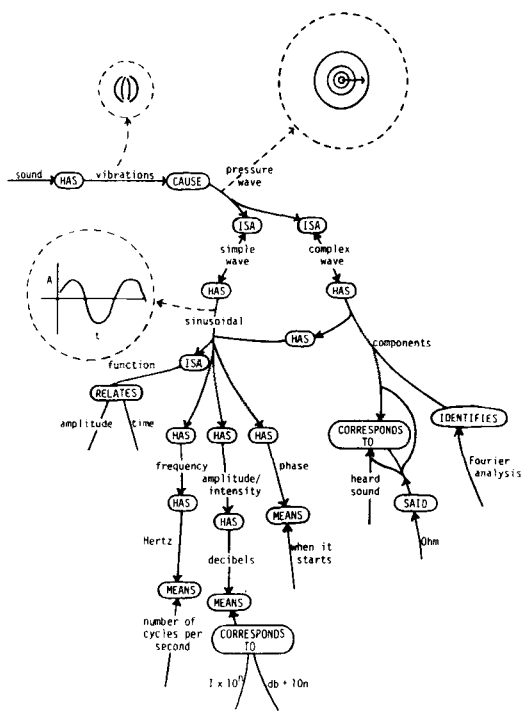


**Figure 3. Greeno's (1976) network of propositions for the physics of sound.**

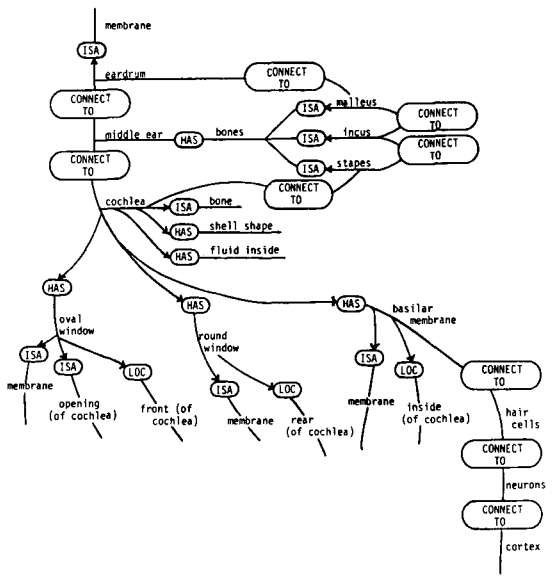


**Figure 4. Greeno's (1976) network of propositions tor the anatomy of hearing.**

complete representation, many of the nodes would themselves be unpacked into networks at least as complicated as the one illustrated in Figure 3.

Figure 4 shows a sample of knowledge about the anatomy of hearing. These networks are easily represented in a language like LISP, so these figures are translatable into data structures to be used by a tutorial model.

Figure 5 incorporates the two previous slides to explain what happens when sound is heard. The previous two networks are nodes in the current network. All these networks can be used to answer questions. For example, if a student was asked "How is frequency encoded in the nervous system?," there are two places where frequency enters into the explanation, and the network indicates both of these places. The student's answer would then presumably pick up one of the references; "it specifies the maximum point of the traveling bulge on the basilar membrane." Such an answer would require the program to perform a pattern match on the network after encoding the student's answer into a similar representation.

Greeno's chapter is an excellent first step in thinking about representation of subjects typically taught in schools. Obviously the first goals of more practical CAI will be school subjects, and his analysis of school subjects should help solve the subject-matter representational problem.

There are, however, the other knowledge structures to consider. How do we represent the tutor's knowledge of the student's knowledge? By a similar but incomplete network? How do we represent self-knowledge, or the knowledge a person has about his own knowledge? How do we know it is even present in the student? One idea is to tag facts in networks with confidence ratings, although such a device seems ad hoc to me. What do we do with new knowledge generated by the person himself on the basis of facts that he has learned? At least sometimes we know when we have inferred a fact rather than having learned it directly.

There is no doubt that we will have to store knowledge in computer systems in a great deal of detail. Knowledge engineers in CAI not only have the job of working out the details of each unit of a subject matter but also most participate in the development of good representational systems.

Newell and Simon (1972) define a problem as a situation in which a person wants something and does not know what series of actions he can perform to get it. Often a tutor wants his student to understand a given topic but does not know how to make him understand it. In fact, sometimes the tutor is not sure exactly what the "topic" is that he wants the student to understand in the sense that it is not a very sharply defined goal. Understanding how to prove a theorem in geometry is not very specific because there are many ways to do it. So, tutoring can be characterized as an ill-defined problem-solving situation.

The first step in solving a problem is to define a problem space, an internal representation of the external environment. The second step is to select a method to solve the problem. The third step is to apply the method. If the method succeeds, then the problem is solved. If it does not, then a new representation and a new method may be tried.

A tutor's problem is ill-defined in the sense that the

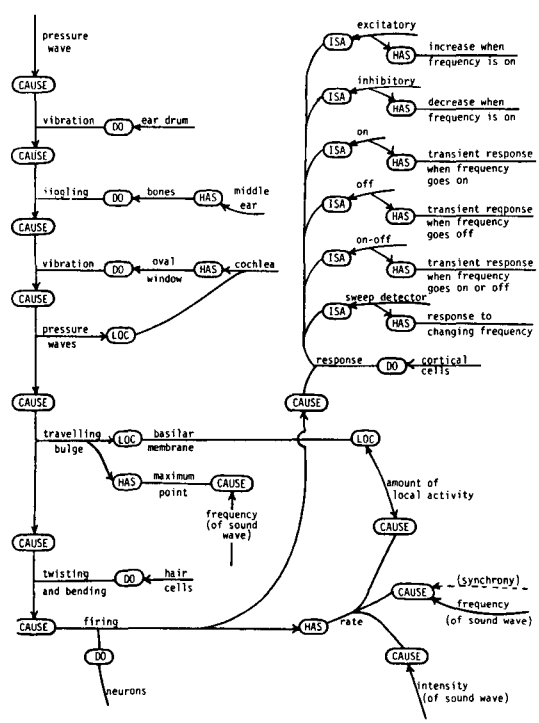


**Figure 5. Greeno's (1976) network of propositions about events that occur when a sound is heard.**

problem space and the set of operators, and sometimes even the goals, change during the course of the solution. Of course, with experience, especially after he has taught the same topic again and again to a homogeneous population of students, the tutor's problem space, operators, and goals become well specified and the problem ceases to be ill-defined. Good CAI programs will necessarily incorporate the results of such experience.

Let us consider the tutor's problem for a moment. First, he or she must ascertain the student's knowledge of the subject matter. This classification process places the student in one of the states in the tutor's hypothetical problem space. The tutor must then select a teaching technique as an operator to be used to change the student's knowledge. Which technique should be used? This will depend on the ability of the student (part of the tutor's knowledge of the student), on the state in the problem space, and certain factors associated with recent actions of the student and tutor. When the operator has been applied, the tutor can only assume that it succeeded. To make sure, the tutor must test the student to see that he has in fact moved to the new state. Testing helps to define the state precisely, since at no point does the tutor really have a very complete model of the student's knowledge. Finally, the tutor keeps alternating between attempts to change the student's knowledge state and testing what the student actually knows. Eventually, the tutor reaches a goal state or redefines a goal state appropriate to the progress that has been made.

What are some of the issues raised by looking at the tutorial process as a problem-solving one? One question is how the tutor infers a level of understanding or a model of the student's knowledge from the student's responses. One possibility is that the tutor has a model of the final knowledge structure and assumes that errors made by the student are caused by omissions and distortions in the student's knowledge structure. Each time a student makes an error, the tutor must examine his model to see what faults in it would produce that error. Then the tutor must make further tests to eliminate different possible explanations until a single explanation for the fault remains. The tutor can then proceed to make a correction to the student's knowledge state by adding more facts or clarifying something about the structure of concepts.

Another issue to consider in looking at the tutor's problem-solving behavior relates to the techniques or operators used to add knowledge to or change a student's knowledge structure. Obvious techniques come easily to mind: Simply present propositions to be learned. Present problems that make students apply rules in new situations. Ask questions that make students integrate ideas previously not considered together. Of course, a tutor does not use any one of these techniques exclusively, but rather switches back and forth among them. Why? What leads a tutor to select a question rather than telling the student a raw fact? These questions need to be answered if we are to build realistic tutorial systems.

Finally, a problem-solving situation usually requires the use of means-ends analysis and subgoals. Often the subgoals are worthy goals in and by themselves. Given the wide variation in past experiences, in interests and needs, and in the abilities of students, it would be unrealistic to have exactly the same set of goals for all students. How does a tutor modify his goals for his students? Should such modification be incorporated into tutorial programs? The existence of varying goals is part of the ill-defined nature of the problem faced by a tutor. A tenacious pursuit of a single uniform competence for all students is probably counterproductive to teaching.

Two recent research efforts have examined tutorial episodes extensively. Collins' and his colleagues' (Collins, 1977; Collins, Warnock, Aiello, Miller, 1975; Collins, Warnock, Passafiume, 1975) study of human inferencing illustrates how a tutor, using the Socratic method, elicits an answer from a student by having him examine his own knowledge. In addition to increasing a student's knowledge base, this technique may also teach a variety of reasoning skills. The second research example is Brown and Burton's (1975) SOPHIE, which leads a student through a session of hypothesis testing while the student is trying to debug a faulty power supply.

Collins' basic approach is to examine protocols of tutors and students discussing simple topics such as what makes rainfall and where rice grows. From these protocols, Collins extracts a large number of inference rules which are used by students to answer questions. Remember, students generally do not know the answer to these questions explicitly but must work out explanations on the basis of other facts they do know.

Collins (1978) incorporates his set of inference rules into a computer program as a set of productions. For example, the tutorial program has the following rule: *If* a student gives as an explanation for some event one or more factors which are not by themselves sufficient, *then* formulate a general rule asserting that the factors given are sufficient and ask the student if the rule is true. Collins' example is "If the student gives water as the reason they can grow rice in China, ask him 'Do you think any place with enough water can grow rice?'" This new question illustrates to the student that a single factor is not sufficient as an explanation since there are counterexamples to the stated rule. Enough examples of this kind may teach the student the nature of sufficiency.

Another inference rule that Collins suggests that students use is the "lack-of-knowledge" rule. If asked whether the Mekong River is as long as the Nile, a student, thinking aloud, immediately rejects the idea on the grounds that he knows almost nothing about the Mekong River, and, if it were as long as the Nile, he would have heard about it. This is based on the assump-

tion that he has heard about the Nile because it is so long. However, upon further reflection, the student realizes that he may have heard about the Nile because it is part of the western hemisphere and schools have ignored, until recently, southeast Asia. So the student thinks, "Maybe the Mekong is long and I just don't know it." His previous conclusion is thus weakened.

Brown and Burton's (1975) SOPHIE system is probably the most intelligent CAI system ever built. Its main component is a program that is an electronics expert. Given a description of a circuit and some fault in that circuit, the program can determine the effects of that fault throughout the circuit. The student interacting with SOPHIE is given the circuit diagram of a faulty power supply and asked to determine what is wrong with it. He can ask questions about the circuit as if he were testing it in the shop. What is the voltage at B4, the resistance at A9, etc.?

One elegant feature of the system is a very good natural English parser that understands the kind of dialogue used in such an informal situation (Burton, 1976). The program keeps track of what the student has learned and whether he has enough information to make a diagnosis of the problem. If the student has not enough information to justify his hypothesis, the program does not simply reject his hypothesis, but asks for a reason. Then it counters the student's explanation and makes the student do more thinking about the problem.

Both of these programs tutor by monitoring a subject's hypotheses and conclusions to make sure that they follow from facts the student knows. In the Collins program, this is done by encouraging the student to think about examples he already knows. In the Brown and Burton program, it is done by insisting that a hypothesis be clearly justified.

After the construction of a number of such systems, we may be in a position to generalize the concepts upon which they are based. Just as it is now relatively easy to build compilers because there have been 20 years of experimentation with them, so, in another 20 years, building tutorial programs for limited domains may be an engineering problem and not primarily a theoretical one. In the meantime, a great many questions remain to be answered about how tutors teach, how students learn, and how machines can be programmed to help in the process.

I have discussed types of knowledge and the problem-solving components of the tutorial situation. It must be readily apparent, however, that language processing is integrally linked to both and is absolutely essential in intelligent CAI. A tutor expresses his ideas in language, so there must be a translation from knowledge structures to communication acts. But natural language, unlike a sequence of computer instructions, is not denotationally exact. Instead, a statement made by a person to another person is an "act" by which the former wishes to change the latter's model of the world, or his knowledge state. Such an act, like tutoring, is not a guarantee that the recipient changes in the way intended. The listener must work to put together a coherent interpretation of the topic under discussion. To do this, he uses information about the content, past comments, his listener, knowledge of social conventions, and world knowledge.

There are many variables associated with human communication. Since teaching in different contexts involves a number of these variables, some understanding of how language processes differ in different environments is important. For example, a textbook is written with certain well-specified constraints in mind. The reader is free to vary his speed of reading, he can backtrack easily, and he can make use of external references if necessary (e.g., a dictionary). However, he cannot ask the writer for clarification of ideas not clearly stated. Therefore, a textbook is carefully structured to present information before it is required, to be as unambiguous as possible, and to avoid reference to knowledge that is not common to a wide audience. A textbook is self-contained.

In contrast, a casual conversation between friends may refer to events that only the two participants are aware of, is highly unstructured in a formal sense, allows interruptions for clarification, and is ephemeral in that the conversation cannot be "replayed" for further clarification. So conversation differs greatly from textbook writing.

The importance of this point for CAI can be emphasized by looking at the failure of programmed instruction. Earlier attempts at programmed instruction did not take this distinction into account. Books, by being self-contained, have a built-in coherence. By branching all over the place, programmed instruction lost this coherence. Branching was an attempt to capture the dynamics of a tutorial instructional session where the tutor shifted topics to take into account the success or failure of his communication and/or problem-solving efforts. But these dynamics cannot be written into a book or programmed statically in a computer. Providing students with "canned" frames (definitions, problems, questions, paragraphs, etc.) derived from textbooks is not tutoring and may actually be less effective than good textbooks.

Here is a summary of a tutorial CAI system: The higher level goals of a tutor are guided by the problem-solving component interacting with a model of the student. A difference between the tutor's model of the student's subject-matter knowledge and the tutor's own subject-matter knowledge generates a proposition to be transmitted to the student. The knowledge-of-teaching component is called to suggest a way to transmit this proposition; for example, ask a leading question, provide a hint, suggest an analogy, or tell the student the idea outright. Once the specific form of the proposition has been selected, the discourse generator constructs a sentence to accomplish the higher level goal.

The student's comments, answers, and questions are parsed into a representation that can be compared with existing knowledge systems. Assuming the student's comment is relevant to the subject matter, it is stored as a new piece of evidence about the student's knowledge. For example, if the student volunteers that a binary number has a base of two, this is added to the knowledge state as (KNOWS STUDENT (BASE BINARY)). Questions asked by the student trigger a sequence of actions directed at finding an answer. The student's answers to questions are compared to the expected answers.

Both the production and interpretation of discourse involve stupendous problems. The tutor-student interactions produce a kind of dialogue that has recently been the focus of a great deal of research by psychologists (Clark & Haviland, 1977; Kintsch, 1977; Rumelhart, 1975), linguists (Chafe, 1973; Searle, 1970), and computational linguists (Grosz, 1978; Levin & Moore, 1977; Schank, 1977; Winograd, 1977b). While this is not the place to review this abundant literature, some comments by Winograd deserve mention since they point up the complexity of the problem and illustrate a parallelism between the problem-solving component and the language processing component of the tutorial situation.

Winograd makes two preliminary points: First, the speaker combines multiple messages into a single structure; hence, as the listener unfolds the sentence, there are multiple points to be considered simultaneously. Second, language is a cooperative process: A "speaker designs a sentence anticipating how the hearer will interpret it and the hearer interprets it in light of hypotheses about the intent of the speaker." According to this model, a speaker is not just taking ideas from memory and expressing them in words but is organizing his ideas to fit his current model of the listener and his own communication goals. Similarly, the listener understands a sentence by constructing an interpretation that is consistent with his model of the situation being discussed by the tutor. For communication to occur, both the tutor and the student must have a number of types of knowledge to refer to, including models of each other's knowledge states.

A number of different natural language processing systems have been developed, and learning about these is difficult for those not working at one of the research laboratories where computer facilities dedicated to such research are available. However, since such systems are important for the knowledge engineer, psychologists interested in CAI will have to familiarize themselves with them. The most appropriate natural language understanding system for CAI appears to be the one used in Brown and Burton's SOPHIE (Burton, 1976).

Believe it or not, I think I have detailed how to teach a computer to teach. Get together a group of cognitive scientists, including a psychologist, a linguist, a computer scientist interested in artificial intelligence, and a teacher, and try to solve one of philosophy's oldest problems–epistomology. Give them a computer on which to experiment with large systems that cannot be explicitly axiomatized. The result should be some interesting intelligent CAI systems.

## REFERENCES


ANDERSON, J. *Language, memory, and thought*. Hillsdale, N.J: Erlbaum, 1976.

BOBROW, D. G., & WINOGRAD, T. An overview of KRL, a knowledge representation language. *Cognitive Science*, 1977, **1**, 3-46.

BROWN, J. S., & BURTON, R. R. Multiple representations of knowledge for tutorial reasoning. In D. G. Bobrow & A. Collins (Eds.), *Representation and understanding studies in cognitive science*. New York: Academic Press, 1975.

BURTON, R. R. *Semantic grammar: An engineering technique for constructing natural language understanding systems*, Report No. 3453. Boston, Mass: Bolt, Beranek, and Newman, 1976.

CHAFE, W. L. Language and memory. *Language*, 1973, **49**, 261-281.

CLARK, H. H., & HAVILAND, S. E. Comprehension and the given-new contract. In R. O. Freedle (Ed.), *Discourse production and comprehension*. Norwood, N.J: Ablex, 1977.

COLLINS, A. Processes in acquiring knowledge. In R. C. Anderson, R. J. Spiro, & W. E. Montague (Eds.), *Schooling and the acquisition of knowledge*. Hillsdale, N.J: Erlbaum, 1977.

COLLINS, A. Fragments of a theory of human plausible reasoning. In D. L. Waltz (Ed.), *Theoretical issues in natural language processing—2*. New York: Association of Computing Machinery, 1978.

COLLINS, A., WARNOCK, E. H., AIELLO, N., & MILLER, M. Reasoning from incomplete knowledge. In D. G. Bobrow & A. Collins (Eds.), *Representation and understanding studies in cognitive science*. New York: Academic Press, 1975.

COLLINS, A., WARNOCK, E. H., & PASSAFIUME, J. J. Analysis and synthesis of tutorial dialogs. In G. H. Bower (Ed.), *The psychology of learning and motivation* (Vol 9). New York: Academic Press, 1975.

GREENO, J. G. Cognitive objectives of instruction: Theory of knowledge for solving problems and answering questions. In D. Klahr (Ed.), *Cognition and instruction*. Hillsdale, N.J: Erlbaum, 1976.

GREENO, J. G. Process of understanding in problem solving. In N. J. Castellan, D. Pisoni, & G. R. Potts (Eds.), *Cognitive theory* (Vol. 2). Hillsdale, N.J: Erlbaum, 1977.

GREENO, J. G. Understanding and procedural knowledge in mathematics instruction. *Educational Psychologist*, 1978, **12**, 262-283.

GROSZ, B. J. Focusing in dialog. In D. L. Waltz (Ed.), *Theoretical issues in natural language processing—2*. New York: Association of Computing Machinery, 1978.

KINTSCH, W. On comprehending stories. In M. A. Just & P. Carpenter (Eds.), *Cognitive processes in comprehension*. Hillsdale, N.J: Erlbaum, 1977.

KLATT, D. H. Review of the ARPA speech understanding project. *Journal of the Acoustical Society of America*, 1977, **62**, 1345-1366.

LEVIN, J. A., & MOORE, J. A. Dialogue-games: Metacommunication structures for natural language interaction. *Cognitive Science*, 1977, **2**, 395-420.

MILLWARD, R. B., MAZZUCCHELLI, L., MAGOON, S., & MOORE, R. Intelligent computer-assisted instruction. *Behavior Research Methods & Instrumentation*, 1978, **10**, 213-217.

MILLWARD, R. B., & SPOEHR, K. T. Direct measurement of hypothesis-sampling strategies. *Cognitive Science*, 1973, **4**, 1-38.

MILLWARD, R. B., & WICKENS, T. D. Concept-identification

models. In D. H. Krantz, R. D. Luce, R. C. Atkinson, & P. Suppes (Eds.), *Contemporary developments in mathematical psychology* (Vol. 1). San Francisco: Freeman, 1974.

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

NORMAN, D. A., RUMELHART, D. E., & THE LNR RESEARCH GROUP. *Explorations in cognition*. San Francisco: Freeman, 1975.

RUMELHART, D. E. Notes on a schema for stories. In D. Bobrow & A. Collins (Eds.), *Representation and understanding studies in cognitive science*. New York: Academic Press, 1975.

RUMELHART, D. E., & ORTONY, A. The representation of knowledge in memory. In R. C. Anderson, R. J. Spiro, & W. E. Montague (Eds.), *Schooling and the acquisition of knowledge*. Hillsdale, N.J: Erlbaum, 1977.

SEARLE, J. R. *Speech acts: An essay in the philosophy of language*. Cambridge, England: Cambridge University Press, 1970.

SCHANK, R. Rules and topics in conversation. *Cognitive Science*, 1977, **1**, 421-442.

SHAW, R. I., & WILSON, B. E. Abstract conceptual knowledge: How we know what we know. In D. Klahr (Ed.), *Cognition and instruction*. Hillsdale, N.J: Erlbaum, 1976.

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

WINOGRAD, T. *Formalism for knowledge*. In P. N. Johnson-Laird & P. C. Wason (Eds.), *Thinking readings in cognitive science*. Cambridge, England: Cambridge University Press, 1977. (a)

WINOGRAD, T. A framework for understanding discourse. In M. A. Just & P. Carpenter (Eds.), *Cognitive processes in comprehension*. Hillsdale, N.J: Erlbaum, 1977. (b)