AI in Education

Beverly Park Woolf

COINS Technical Report 91-37

April 1991

To appear as an article in the *Encyclopedia of Artificial Intelligence,* Second Edition, edited by Stuart C. Shapiro, John Wiley & Sons, Inc.: New York, NY.

AI in Education [1]

Beverly Park Woolf

Department of Computer and Information Science
University of Massachusetts
Amherst, Massachusetts 01003

bev@cs.umass.edu

Contents

 Prepared for the *Encyclopedia of Artificial Intelligence*, 2nd Edition, Stuart C. Shapiro, Ed., Wiley-Interscience: NY.

Artificial Intelligence in Education

1 Intelligent Environments for Teaching and Learning

The field of AI in Education is concerned with development of Artificial Intelligence techniques for the study of human teaching and for the engineering of systems that facilitate human learning. The field addresses questions that are long-term in nature: how can systems facilitate learning and enable the measurement of learning progress (Lesgold, 1988). The term 'intelligent tutoring system' (ITS) is frequently used with regard to the engineering side of the discipline. Computational methods are used in support of AI activities such as planning, control, knowledge representation and acquisition, explanation, cognitive modeling, and dialog Management. Computational models are used to explore and evaluate alternative theories about learning. Research is motivated by the promise of building powerful teaching systems with greater knowledge about a domain, increased ability to make inferences about student behavior, and increased reasoning ability about topic selection and response generation. This entry reviews the current state of the field and discusses the history of the field. It addresses basic approaches to building teaching systems, recent developments in the field, and open research issues.

Three research goals have become apparent. The first is to use AI and cognitive science techniques to model experts who problem-solve in a domain, as well as tutors teaching and students learning in that domain. Computational models facilitate a degree of explicitness about learning theories and teaching strategies that is difficult to attain in classrooms. Such models encourage comparison of the results of using a teaching system with data from classroom style teaching (see Section 3.4) and aid in refining and evaluating a learning or problem solving hypothesis.

The second research goal involves explaining learning and teaching as parts of the human information-processing system. Since all intelligent beings learn, differences in learning rates might be due to a level of prior knowledge or to the quality of teaching. Teaching efforts are often critical to learning, as when a learner makes errors or has deep misconceptions. Modeling human teaching efforts should help us understand learning characteristics such as student motivation and performance.

The third research goal is to demonstrate completeness and reliability in the engineering side of the discipline and to show that intelligent instructional systems can be used effectively in training and classroom situations. Sometimes this goal is expressed in terms of providing each student with a computer-based tutor which has some of the qualities of a master teacher, such as scope and depth of subject matter, excellent knowledge of teaching, powerful communication skills, and the ability to inspire and motivate the student to learn. Clearly researchers have a long way to go to achieve this goal, and yet some progress can be shown. Subject matter expertise has been realized to a limited extent. Less progress has been made in modeling tutoring expertise and student knowledge. Even less progress has been made in simulating powerful communication skills. Human communicators use many verbal and non-verbal skills to convey information, including speech, informal sketches, and gestures. Computers are poor at verbal communication. On the other hand, they can be much better than (unaided) humans at visual or graphic communication of ideas and time-related processes. Many systems exploit this ability including STEAMER (Hollan et al., 1984) and IMTS (Towne et al., 1990) – see Section 3.3. To increase the communication bandwidth for teaching systems, it may be more cost-effective to exploit their inherent non-verbal communication strengths rather than to improve their poor verbal skills. The least amount of progress has been made in the affective area: for example, the crucial aspect of student motivation has yet to be seriously confronted. Several intelligent instructional systems have been used generally by many students and some systems result in significant learning gains (see Section 3.4). However, many others provide only limited coverage of a domain and have, by and large, demonstrated only simple knowledge engineering capabilities.

Research activities in this field are important to education, not only because such systems might someday become routine in classrooms, but also because such systems might support students in activities not available in traditional classrooms, such as extensive one-on-one collaboration with a tutor and freedom to explore hypothetical worlds, to make conjectures, and to test hypotheses.

Intelligent instructional systems are distinquished from earlier computer based training (CBT) systems, including computer aided instruction (CAI), simulations, and microworlds, in that they dynamically reason about and customize their response to the individual student. Traditional CAI systems are script-like: all machine responses, text or graphics, are organized into screens, and encoded branching instructions define both the topic and response to be presented. Every machine response, as well as every path of instruction for a typical student, is predefined by the author. Simulations and microworlds of this type lack student monitoring capability and show few positive learning effects (Alperson and O'Neil, 1990).

Conversely, intelligent instructional systems define a variety of knowledge types and then define pedagogical knowledge about how to teach that knowledge; the latter might contain rules of inferencing for reasoning about a variety of possible ways to teach the given content knowledge. The system reasons about its stored knowledge and then dynamically generates its own path through the knowledge in response to student behavior.

Current systems vary in the type of knowledge they teach. Some teach concepts, e.g., velocity and acceleration, as well as processes within whole systems, e.g., an emergency shut-down procedure in a boiler system (Woolf et al., 1986), or a hydraulic process involved in folding a helicopter's blades (Towne et al., 1990). Other systems teach meta-cognition, or the knowledge needed to reason about learning (Richer and Clancey, 1985) and how to debug solutions (McArthur et al., 1987). Some teach formal logic and formal knowledge, e.g., ALGEBRALAND (Foss, 1987) and the Geometry Tutor (Anderson et al., 1985).

2 Theoretical Basis

Researchers in this field ask under what conditions a computer system might effectively teach and if such a system might teach in the same way as does a talented teacher. Definitions of teaching are typically grounded in computational models of the human participants, including the *domain expert, teacher,* and *student.* Additionally, the definition frequently includes a model of effective *communication.* Typically, these four models work together to generate a system's intelligence. A model of the *expert* typically represents topics, concepts, definitions, or processes within the domain. A model of the *teacher* might include methods for providing remediation for errors as well as a selection of examples, analogies, and strategies for responding to idiosyncratic student behavior, and knowledge about when to interrupt. A model of the *student* or *cognitive processes* might represent general factors necessary for a person to learn in that domain, such as considerations about motivation. A general and abstract student model might be dynamically updated to include more specific attributes such as whether a particular student needs more challenges or more remedial advice. A *communication* model might include dialog and teaching principles in the domain couched within principles of good interface design.

Building an intelligent instructional system entails building each of these models using a variety of AI technologies, including knowledge representation, control and knowledge acquisition. These technologies are discussed below.

Knowledge Representation. In constructing intelligent tutors, two aspects of knowledge representation (qv) are important. First, what knowledge do teachers and trainers use to understand the domain, diagnose student behavior, and select new strategic approaches, and second, what are good representational schemes for encoding domain knowledge. Knowledge representation refers to how knowledge is used to model the domain, human thinking, learning processes, and teaching strategies. Knowledge bases might store concepts, activities, relations between topics, and other quantities needed to make expert decisions. They might also store a variety of lessons, topics, presentations, and response selections.

Control. Separation of the knowledge base (data structure) and instructional strategies control (algorithms or heuristics) is critical to expanding the power of these systems and to enabling the author to work with each component separately. This separation enables designers to experiment by adding new control schemes, such as Socratic teaching, while keeping the knowledge base fixed. Control refers to passage of an interpreter through knowledge bases and its selection of appropriate pieces of knowledge for making a diagnosis, a prediction, or an evaluation. Control structures might be defined separately for selection of lesson, topic, presentation, and response.

Knowledge Acquisition. Knowledge acquisition (qv) is a difficult problem for any AI system. It requires facilitating incremental additions to an existing knowledge base. Frequently this means building an interface that enables a domain expert to easily add new instantiations of a chosen data structure, e.g., a new frame or a new production rule, without requiring him or her to work in the implementation language. Knowledge acquisition requires and supports the identification and encoding of expertise. In constructing an intelligent tutor, knowledge acquisition should enable the expert to input questions, examples, analogies, and explanations; it should elicit not only tutoring primitives, such as topics and prerequisite knowledge, but also the reasoning the expert uses to know how and when to present each primitive. Tutoring knowledge might be described both in terms of content (i.e., topics, questions, and examples) and context (i.e., active tutoring strategies and current dialog interventions).

2.1 Reasoning about Expert Knowledge

Building a domain or *expert* model requires, to some extent, specification of the relative difficulty of topics, identification of the strategies and tactics used for tailoring instruction to an individual student, and a corpus of analogies, examples, and error diagnosis techniques for teaching in the domain. Without the aid of shells, e.g., expert systems shells, and authoring systems, which currently do not exist, this task is difficult. Even with such software tools, each new domain requires identifications of curriculum topics and prerequisite topics, causal and temporal relations between topics, and the relative difficulty of learning each topic.

Choice of representations is also a large issue. Two roles might be played by the chosen representation: in those systems which focus on teaching problem solving knowledge, the representation might first be used to solve the given problem and, second, used to communicate about the solution with the student. The first role suggests that the representation should be powerful enough for problem-solving, e.g., predicate calculus or rule-based language, and the second suggests that the language should be useful for explanations and should possibly contain a subset of terms a person might use to think about problems, e.g., spatial or temporal relations between topics. Few languages are powerful enough to include both such features.

2.2 Reasoning about Teaching Knowledge

Several styles of pedagogy or teaching knowledge have been explored in the process of developing customized responses to student behavior. Initially many systems were *despotic* in nature; incorrect student actions were quickly identified, based on reference to an error knowledge base, and quickly remediated. However, immediate help has several disadvantages: it may compete for short-term memory with newly learned material, and students might also become dependent on it (Schooler and Anderson, 1990). Researchers have moved away from building purportedly omniscient tutors and now focus on building *empathetic* environments that allow for independent exploratory behavior and also elicit information about student goals and plans. Such systems require less knowledge about the domain and can coach rather than tutor. These systems still reason about several forms of behavior before taking action using reasoning based on knowledge about how people solve problems or make inferences in the domain. Theoretical focus has shifted from exclusive diagnosis and remediation to identifying and supporting student management of their own cognitive processes.

Mixed-initiative dialog refers to human-computer communication in which a student can control the conversation or ask questions as desired. Such control is now assumed in responsive instructional systems. *Error diagnosis* refers to the system's ability to diagnose mistakes, plausible misconceptions, overgeneralizations, and missing information. A diagnostic tutor compares student behavior with that of an expert before reasoning about how to elicit better learning performance.

Tutoring style might vary across domain types. Thus it could include explanation, guided discovery learning, coaching, coaxing and critiquing. Although no one style is preferred over others, different domains will be better addressed with different primary styles. For example, didactic explanation is good for communicating a body of declarative knowledge shared by some community, e.g. biologists. Students need to learn the community's terminology and are not expected to rediscover all the principles in a field. On the other hand, more active discovery learning helps students "own" knowledge and is a better interaction style if students are expected to generate and test their own hypotheses, e.g., while troubleshooting a circuit.

Tutoring style might also vary *within* a tutorial domain. For example, a human teacher might support guided discovery learning at first and yet change his or her strategy to opportunistic one-on-one tutoring once the student requests a specific activity or shows the need for remediation. How and why human teachers change teaching style is an open research question. Machine teaching style can range from natural-language discourse to menu-selection, though the importance of natural language has diminished considerably in recent years now that structured command languages or menu selection structures appearto be as effective for communication as state-of-the-art spoken or written languages.

2.3 Reasoning about Student Knowledge

Building an intelligent instructional system requires identifying presumed student knowledge within the domain as well as making inferences about his or her grasp of meta-cognitive skills. Frequently, a system will use a student model to represent how the student has organized and incorporated new knowledge. The student model should dynamically indicate a system's changing views of the student's strengths and weaknesses as well as aspects of his or her currently (mis)understood knowledge.

2.4 Reasoning about Communication

Building a variety of intelligent tutors has uncovered some design considerations about the model of communication between machine and student (Burton, 1988). For instance, extensive research is required to uncover the cognitive nature of the task (see Anderson (1981) for research leading up to the Geometry Tutor and Woolf et al. (1986) for research leading to building a boiler tutor). Another design consideration results from the need to isolate key "tools" required for attaining expertise in the domain. For example, the economics tutor used specific tools for selecting parameters during experimentation and scientific inquiry (Bonar et al., 1986), and the Geometry Tutor used visualization tools to foster both forward and backward reasoning in the development of geometry proofs (Anderson et al., 1985) (see Section 3). The resulting tools were valuable motivational devices for learning even without help from any on-line tutor. A third consideration is the need to attain high fidelity to the modeled world (Hollan et al., 1984). Fidelity is measured by how closely the simulated environment matches the real world, and high fidelity means the situation is almost indistinguishable from the actual environment. High *conceptual* fidelity, which is possibly the most powerful attribute of a communication interface, means that the simulation reflects more a mental model used by an expert than an exact physical model.

3 Example Systems

Several intelligent instructional systems are discussed in this section along with the theoretical issues which they address.[2] Conventional computer programs for teaching have been around for decades. However, the field of AI in Education is said to have begun with Jaime Carbonell's program developed in 1970 (Carbonell, 1970).

[2] See Sleeman & Brown (1982) for a collection of the original papers discussed in the section Early Works. See Wenger (1987) for an excellent discussion of additional individual projects and a theoretical foundation of the field.

3.1 Early Works (1970-1982)

During the first phase of AI in Education, research was limited to building illustrations that showed ideas at work on toy domains. Jaime Carbonell's program SCHOLAR taught high school geometry and differed substantially from other teaching systems of its day in that it separated knowledge about how to teach from the content or subject matter of teaching. Carbonell represented objects and concepts in grade school geometry as nodes in a semantic network, suggesting that this was a feasible model of the way people store and access information. SCHOLAR held mixed-initiative dialogs (i.e., interactions initiated by either the student or the system) by traversing the network and asking or answering questions about the stored information. SCHOLAR reasoned about student answers, responded opportunistically, and, for the first time, was able to parse student questions.

Early research issues focused primarily on knowledge representation and grain size. Typically, ATNs (qv) (see Grammar, augmented transition), semantic networks (qv), and rule-based systems (qv) were used. SOPHIE – a Sophisticated Instructional Environment for electronic troubleshooting – reasoned about a trainee's solution for debugging an electronic circuit and decided whether a student's actions were appropriate given previous information about the circuit's behavior (Burton & Brown, 1982). The system used a simulated electrical circuit to test student hypotheses, typically conjectures about the cause of a malfunction, and "refused" to carry out probes that were irrelevant based on existing information.

SOPHIE was a landmark effort in the development of domain representations. It innovated the use of semantic grammars to parse student input and used multiple representations to help explain results to a student (Brown & Bell, 1982). Syntactically meaningful categories, such as "resistors," "transistors," and "measurements," were associated with grammar rules to parse the student's input. It answered hypothetical questions about circuit values, generated explanations about possible faults in the circuit, and used rules to solve electrical problems.

Early systems also focused on grain size or the way the world is divided up by the knowledge representation scheme. Grain size is often measured along an epistemological continuum beginning with "bits-and-pieces" and ending with "chunked elements." At the "bits-and-pieces" extreme, distinct and unconnected elements are used to represent elements in the subject area, as in WHY (Stevens et al., 1982), whereas at the chunked extreme, relations and morphisms between elements indicate temporal, logical, or pedagogic connections between the elements. GUIDON (Clancey, 1982) used a chunked representation to express connectedness and logical precedence of elements used to reason about medical diagnosis.

GUIDON (Clancey, 1979) is a tutor for medical students directed at teaching the process of diagnosis of infectious diseases. An early "bits-and-pieces" approach failed and a later implementation, based on chunked knowledge, succeeded. MYCIN (see MYCIN), a large medical expert system used to diagnose and prescribe remedies for infectious disease, was the basis for the original GUIDON. MYCIN has thousands of small rules for diagnosing diseases, and was adequate for a performance system – it worked to achieve a diagnosis. However, it did not work as the basis for a learning/teaching system because students need to know the underlying deep and causal knowledge (from which the "bits-and-pieces" rules are compiled). Deeper causal knowledge is an example of "chunked elements" mentioned above. MYCIN's rules were stripped of causal reasoning and cross-links needed by a student to cluster and learn the same rules. GUIDON was originally implemented by "reversing" the

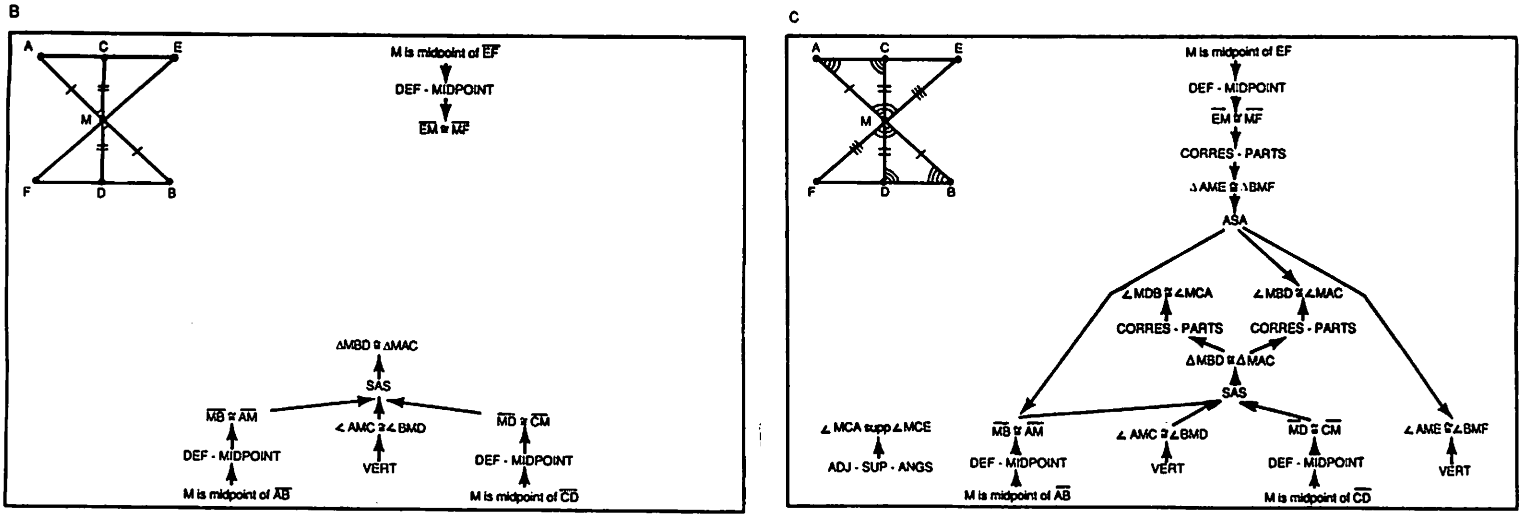

Figure 1: Geometry Tutor (Anderson et al., 1985)

rules of MYCIN. Such a method was not effective, according to Clancey, because medical diagnosis is not taught "cook-book" style and medical practitioners do not use perfect recall on a huge number of medical facts and rules (Clancey & Letsinger, 1981). In order to teach from such rules, GUIDON had to "decompile" and cross-index the stripped-down rules to provide students with the needed generalizations and references between the rules.

GUIDON did not communicate rules *per se*; rather, it presented explicit reasoning strategies described in terms of managing the student's set of hypotheses during problem solving. Thus a student might be asked to justify why she pursued an active hypothesis or be told to continue a specific line of reasoning. The shift in GUIDON's knowledge base was prompted by epistemological considerations, or the need to separate strategic tutoring knowledge from facts and rules. The strategic knowledge was expressed in terms of tasks that manipulated the space of hypotheses.

3.2 Broadening Interests (1982-1985)

During the next phase of AI in Education, knowledge bases were broadened in scope, allowing systems to be applied to real problems. Gradually ideas about tutoring were shown to be powerful enough to handle practical teaching problems, e.g., tutors in Pascal programming (Johnson & Soloway, 1984). Researchers began building systems in part as experiments to answer truly difficult questions about cognitive processes and learning, e.g., Anderson's LISP and Geometry Tutors (Anderson & Reiser, 1986; Anderson et al., 1985).

PROUST was designed to aid students in understanding non-syntactic bugs in their programs (Johnson & Soloway, 1984). It modeled the presumed intentions of the programmer and tried to match an agenda of goals, plans, and then code which the student 'intended' to use with the actual code produced. PROUST successfully discovered 81% of the bugs in 206 student programs. However, some students found PROUST's explanations difficult to use, suggesting the need for improved tutoring discourse.

Anderson used ACT theory (Adaptive Control of Thought), a well-specified cognitive model of learning, to develop geometry and LISP tutors (Anderson & Reiser, 1986; Anderson et al., 1985). These tutors demonstrated that research in cognitive psychology can produce useful insights for building tutoring systems. The theory states, in part, that cognitive functions can be represented as sets of production rules. In this context the system used model tracing to clarify student misconceptions and to redirect the problem-solving effort by setting new goals. It provided immediate feedback on errors and intervened as soon as a meaningful error was detected. However, recent work suggests that immediate feedback competes for working memory resources (Schooler & Anderson, 1990). However, more delayed feedback appears to foster the development of secondary skills such as error detection and self-correction, skills necessary for successful performance once feedback has been withdrawn. Thus Anderson was led to refine his cognitive model and to modify his tutoring systems. This hypothesize-test-evaluate cycle enabled him to demonstrate improvement both in the cognitive theory and the building of intelligent tutors.

Several tutors built at this time also illustrated the importance of the communication model. The Geometry Tutor (Anderson et al., 1985) made explicit use of several learning characteristics in geometry (see Figure 1) and enabled students to visualize features of problem-solving that are typically left implicit in textbooks, including graphic effects on a geometric figure, the tree-structured nature of geometric proof, and movement between two possible problem-solving strategies, forward and backward reasoning. Geometric reasoning is often not a simple linear logical chain, but rather a bushy tree, including many possible, and frequently not optimal, logical paths. By using the interface to click on relevant proof elements, students may work from the goal backwards or from the premises forward. These alternative paths are clearly articulated in the Geometry Tutor's communication model.

3.3 A Variety of Application Areas (1985-1991)

The next phase of tutor development indicated a clear emergence of new architectures and positive training results, leading to the belief that some progress has been made (see Section 3.4). Interest focused on communication and representation issues and on knowledge of the student. Systems were placed in elementary and high schools, universities, industrial sites, and military training sites.

Several systems focused on developing sophisticated interfaces which allowed a student to generate and test hypotheses. For instance, the Smithtown Economics Tutor (Shute & Bonar, 1986; Bonar et al., 1986) provided students with scientific inquiry tools that enabled them to collect, organize, and reason about data in the domain of economics. These tools, for example, allowed students to explicitly state laws such as that of "supply and demand." After setting population and income values, the price of tea and coffee, and the number of outlets or changes in the product quantity, the student made predictions about changes in economic variables such as supply and demand. Smithtown was implemented using a "bite-sized tutoring architecture" (Bonar et al., 1986) which organized the system around issues, called bites. Each bite was a miniature system containing some domain knowledge curriculum knowledge, and indications about the student's mastery of the bite. Thus the old functional components, such as diagnostic or expert modules, were still present, but distributed across the representation of the curriculum.

During this time, several intelligent tutors were developed to teach algebra, in part by displaying a student's solution on the trace window (McArthur et al., 1987; Foss, 1987;

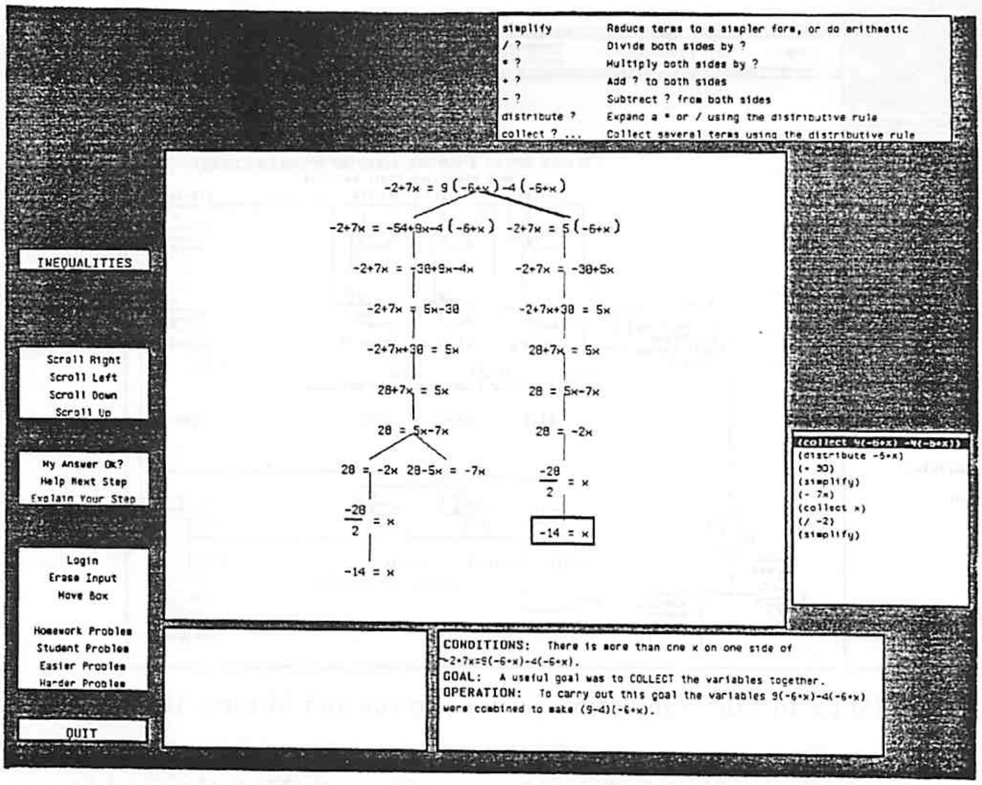

Figure 2: Algebra Tutor Interface

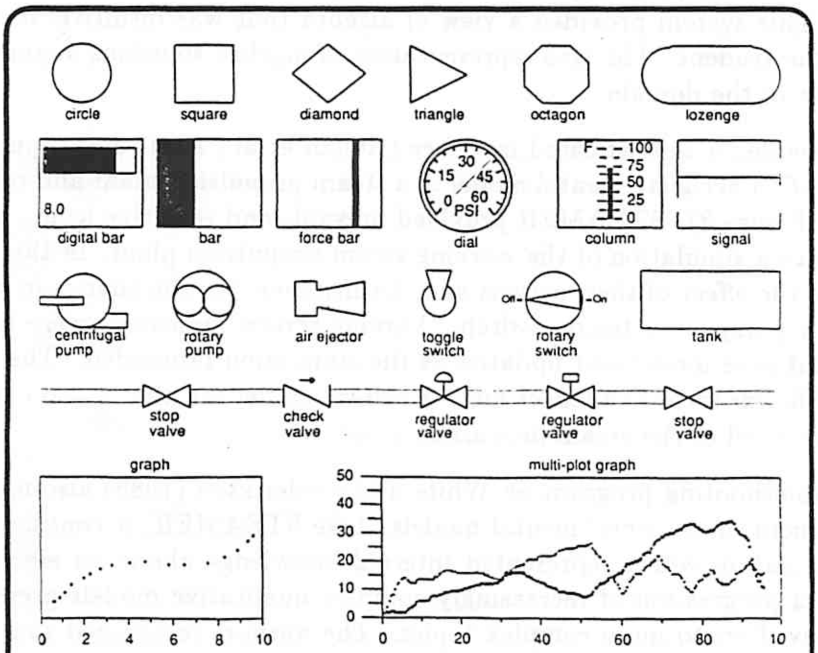

Figure 3: STEAMER Icons (Hollan et al., 1984)

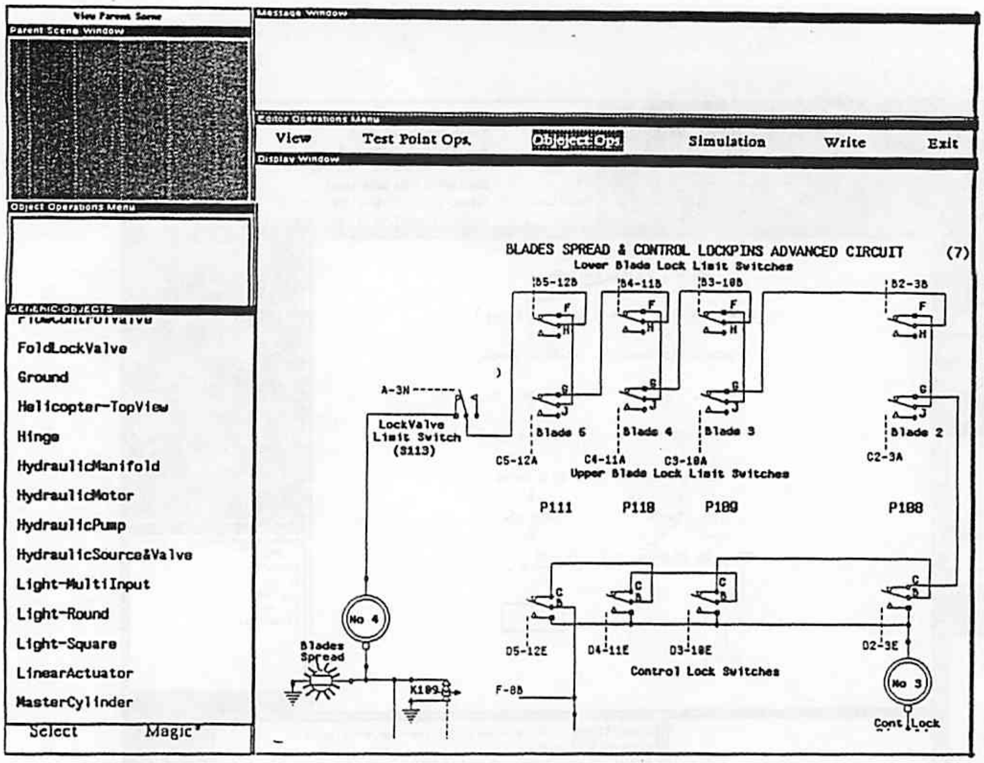

Figure 4: Using IMTS to Edit Simulation Scenes (Towne and Munro, 1989)

McArthur and Stasz, 1990). In the Algebra Tutor shown in Figure 2 (McArthur and Stasz, 1990), the system acted as a partner and encouraged the student to think about problem solutions at a high level, e.g., using operators such as "collect the variables," "simplify the expression," and "multiply both sides by ?". The system could invoke different tutoring styles and perform simple lesson planning or task sequencing. Students could easily change their own operations. This system provided a view of algebra that was intuitive, motivational, and helpful to the student. The tree representation of algebra solutions augmented a student's own abilities in the domain.

STEAMER also provided a sophisticated interface (Hollan et al., 1984) and supported students' development of an accurate mental model of a steam propulsion plant and related engineering principles (Figure 3). STEAMER provided movable and sensitive icons, which students could insert into a simulation of the working steam propulsion plant. In this way, students could measure the effect of their actions and, for instance, see the change in water pressure after adding a pump or a toggle switch. Various screen indicators were linked to an underlying quantitative model and updated as the simulation proceeded. The view presented by STEAMER was meant to reflect more a mental model (as used by an expert) than an exact physical model of the steam propulsion plant.

An electronics troubleshooting program by White and Frederikson (1986) also focused on supporting development of students' mental models. Like STEAMER, it contained an interactive graphic simulation which represented internal knowledge about an electronic circuit, in this case as a progression of increasingly complex qualitative models presented to students as they moved on to more complex topics. The authors recognized that true expertise does not consist of a simple progression of increasingly complex models, rather it requires sets of complementary models which vary along a set of dimensions, such as type, order, and degree (of sophistication).

The Intelligent Maintenance Tutoring System (IMTS) and its successor RAPIDS, provide a set of tools for constructing simulation-based tutoring environments (Towne et al., 1990). Object and scene editors, such as that shown in Figure 4, allow an author to use

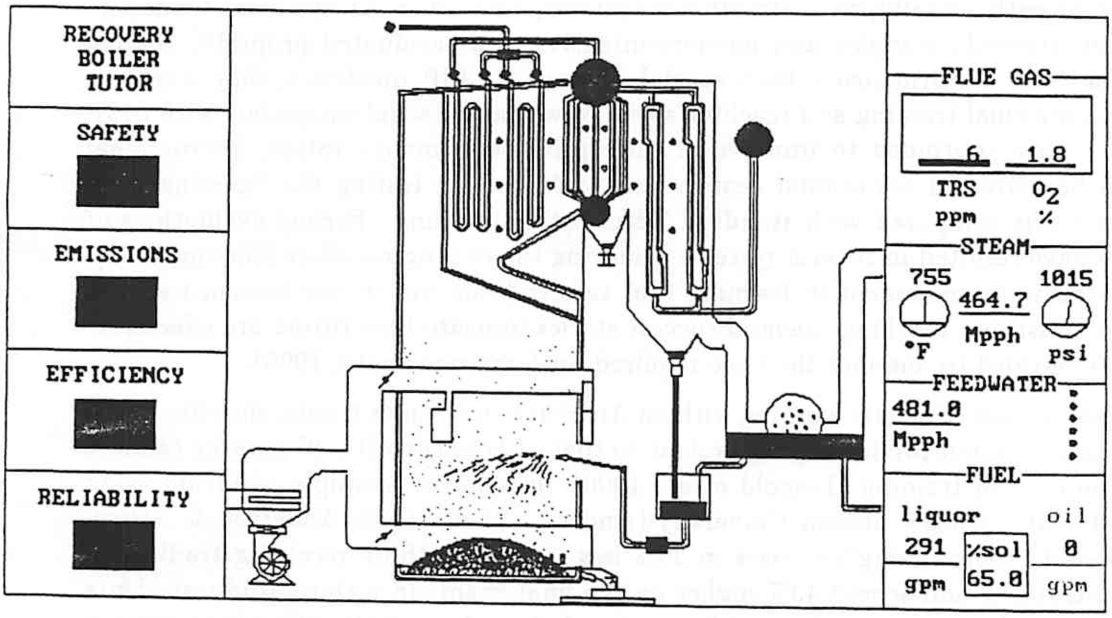

Figure 5: Sectional View of the Recovery Boiler

graphics of generic objects, including switches, pumps, motors, lamps and toggle switches to capture both the appearance and the behavior of the objects used in problem solving environments. Authors enter rules to express the relationship between input and output values of an object and indicate the appearance (e.g., light on, switch up) that objects should display under different conditions. Numerous simulations have been built using these systems; the largest contains over 300 objects and represents the electrical, hydraulic, and mechanical process of folding a helicopter's blades. This system was used to train students to troubleshoot components of the blades; it determined which problem the student should solve next, kept track of how much time was taken to solve each problem, and maintained a model of the student's presumed learning.

Another simulation-based tutor is currently in use in over one hundred industrial sites in America. The Recovery Boiler Tutor, RBT was built for a type of boiler found in paper mills throughout the United States (Woolf et al., 1986). It provided multiple explanations and tutoring facilities geared to the individual user, an operator in a control room. The tutor was based on a mathematically accurate formulation of the boiler and provided an interactive simulation complete with help, hints, explanations, and tutoring (see Figure 5). A user initiated any of 20 training situations or emergencies, or asked that an emergency or inefficient situation be chosen by the system. Once a situation had been initiated, the operator was encouraged to adjust meters and perform any of 40 actions on the simulated boiler to solve the emergency. The system identified optimal, less than optimal, and clearly irrelevant actions.

3.4 Evaluation

Until rather recently, intelligent instructional systems, like other AI systems, tended to be too computationally complex and memory-intensive to be evaluated properly. Having been built on high performance – even special purpose – LISP machines, they were not available to traditional training and teaching sites. Now that personal computers with LISP environments have continued to improve in their price-performance ratios, instructional systems can be tested in traditional centers and evaluated by testing the "goodness" of the teaching result compared with standard lecture style teaching. Formal evaluations of performance have resulted in several systems achieving the two-sigma effect (Bloom, 1984), which is the same improvement in learning that results from one-on-one human tutoring compared to classroom teaching. Several success stories indicate that tutors are effectively reducing by one-third to one-half the time required for learning (Shute, 1990).

In one special case, students working with an Air Force electronics troubleshooting tutor for only 20 hours gained proficiency equivalent to that of trainees with 40 months (almost 4 years) of on-the-job training (Lesgold et al., 1990). In another example, students using the LISP tutor at Carnegie Mellon University (Anderson et al., 1985; Anderson & Reiser, 1986) completed programming exercises in 30% less time than those receiving traditional classroom instruction and scored 43% higher on the final exam. In a third study, students using SMITHTOWN learned general scientific inquiry skills and principles of basic economics in one-half the time required by students in a classroom setting (Shute et al., 1989).

Although limited but encouraging results have been shown, the fact remains that classroom tests do not provide a measure of success for these systems because often the material presented is not the same as that taught in a traditional classroom, and a system's content cannot easily be integrated into a traditional curriculum. For instance, both the Geometry and Algebra Tutors discussed in Section 3 automate much of the symbol manipulation, e.g., addition and multiplication, in the domain and provide an environment for students to learn problem solving. The systems have, in part, redefined the curriculum content to focus on problem solving, and this makes evaluation of learning outcomes difficult since the original classroom focused on symbol manipulation.

Several aspects of evaluation remain major research issues. For example, if a system succeeds, which models should be assigned the credit, and how might the various models be fine-tuned to improve the next generation of systems? Portability between domains has been shown in only a few systems (Bonar et al., 1986; Anderson, 1990); most systems are less effective when rebuilt in another domain and generalizability has yet to be demonstrated.

4 Research Issues

Current research issues focus on traditional AI themes, i.e., knowledge representation and acquisition, control, planning and plan recognition, natural-language generation and recognition, explanation, dialog processing, and architectures. Ideally, cognitive and pedagogical theories should form the basis of design for new systems to test new AI technology, and when built, new systems should be used to test the theory. The crucial activity is iteration based on improved performance, which allows results from one iteration to inform and constrain development of the next: working tutors should foster refinement and evaluation of AI and cognitive theories and vice-versa. There is nothing so practical as a good theory. However, too little theory currently guides development of new systems.

Cognitive Modeling. Cognitive modeling is one research area that can make rich contributions to progress in this field. It should be applied at three stages of design within knowledge-based instructional systems: (1) development of pedagogical and subject-matter theories, (2) design of instruction, and (3) delivery of instruction (Woolf et al., 1991). Of these phases, the design of instruction is the one that seems to have achieved the most direct benefit from cognitive modeling thus far, including substantial benefits from modeling subject matter experts. For instance, Anderson et al. (1990) attribute much of the success of their tutors to the cognitive task analysis of experts in LISP, geometry and algebra.

Work on modeling good teachers and tutors has only just begun (with the exception of a few early classics, such as the work of Stevens and Collins on Socratic tutoring (1977)). Of the three phases of pedagogical work, the actual delivery of instruction is the area where cognitive modeling has thus far found the least fruitful application, mostly due to a historical accident. Working with classroom teachers and trainers seems to be a neglected activity during research and development of these systems.

Communication Modeling. Developing systems which are sensitive to student idiosyncrasies and able to customize their responses to teaching context is still a much sought goal. Achieving flexible mixed dialog between human and machine, whether text- or visually-based, is not yet possible. On the other hand, human-human dialog succeeds despite ambiguity and digressions because both participants model the dialog, the subject matter, and the other speaker; both actively work towards success of the dialog. This suggests that continuing efforts be made to enhance the machine's ability to do its part to model the user and dialog context. Techniques such as plan recognition and learning in active systems still play only a small role in current technology. Building responsive intelligent interfaces requires building mechanisms to support cooperative dialog and a deeper understanding of domain perspectives from the viewpoint of the learner.

Research issues in communication models include identifying new representations and control mechanisms. Thus, knowledge of didactic explanation might be represented and organized along with the basic domain knowledge. Indexing mechanisms for accessing different perspectives of a topic should be designed using abstractions appropriate for the content selection task.

Choosing and organizing domain knowledge for the communication effort provides the next set of research issues. Control should account for the tutor's ability to switch strategies dynamically according to multiple constraints and in a manner that is sensitive to features that human tutors use in tutorial interactions. Further work is required here to characterize "relevance" for selecting knowledge for didactic explanation, especially when multiple perspectives of the topic are available.

Other research issues center on development of an adequate model of the student and the pedagogical context, and then recognizing how a system might stimulate the student's own abilities and creativity. A separate issue concerns how relevant knowledge should be presented once it has been selected. Presentations, whether explanations or examples, should be delivered in a manner that helps the student understand new material and integrate it into an existing conceptual framework, or into one which has been built up during the preceding dialog.

In summary, despite much work attempting to do so, human-machine communication is not yet sensitive to dialog context and to what is known or knowable about the student's knowledge "state".

5 Design and Implementation Issues

Though some success has been seen, one might ask why more systems have not been deployed. The answer is that deep design and implementation issues remain, beginning with the lack of AI development tools, e.g., shells and frameworks, similar to those used to build expert systems. Tools would facilitate large-scale development; a simple tool, such as a simulation tied to an expert system or to a lock-step tutor, might be a practical way for designers to get started on a path of incremental design through feedback from the expert and student. A teacher should interact with a variety of tools, in much the same way that a conductor orchestrates a suite of instruments. Another reason for slow development is the need to reduce cognitive task analysis to engineering practice. An excessive amount of time is still required to analyze each task to the depth required for building these systems. Additionally, more information is needed about student motivation and cognitive development, specifically which activities engage particular students and how novice behavior is distinquished from expert behavior (Larkin et al., 1980; Chi et al., 1981).

The use of new knowledge representations should result in greater expressive power than that offered by first-generation expert system tools. For instance, qualitative simulations might be used to represent domain knowledge (Forbus, 1986; de Kleer & Brown, 1986), as when programming is seen as a space of problem solving plans selected and executed by a student (Johnson & Soloway, 1984). Qualitative physics might be used to reason about stored stereotypical plans to explain a complex physical device; these plans might then be selected and reformulated for presentation to the student.

Social issues provide the final set of barriers to producing generalizable and pervasive systems. As a result of educational changes mandated by the industrial revolution, traditional didactic, classroom-based teaching has become a strong cultural fixture in many parts of the world. Current classroom environments have grown progressively lecture based. Although one-on-one tutoring is in fact very old (see Plato, 1922), its reintroduction through this technology represents a radical change for modern educators and is strongly resisted. Intelligent instructional systems present a new tradition of one-on-one tutoring that is largely inquiry-driven in nature. Since educational change works very slowly, new systems will require curricula and infrastructure changes before they can be established within education and industry.

6 Discussion

Although much success has been evidenced, clearly much remains to be done to support the effective use of AI technology in education.

Current research has succeeded in exploring a large number of domains and has explored some non-traditional pedagogical strategies, such as partnering, mentoring, and scaffolding. However, many of the rich and detailed tutoring methods used by talented teachers still elude researchers, and the next generation of systems will require development of accessible shells and testbeds to facilitate further experimentation and development. For some modeling tasks, notably those representing students and tutors, work is far from complete and issues have only just been identified.

Several predictions can be made about future development of intelligent teaching systems based on near-term goals and long-term opportunities. As the computer price-performance ratio continues to improve, a wide expansion of tutoring systems will continue to be seen in new teaching and training arenas. New initiatives incorporated within these systems, such as qualitative reasoning, machine learning, case-based reasoning, general purpose architectures (e.g., blackboards), hypertext (Yankelovich et al., 1985), and multimedia (e.g., AI systems which include videodisc, speech, or film) will further enhance the effectiveness of these systems. Such technologies should facilitate the study of human learning and teaching as well as accelerate the emergence of new systems and the willingness of new authors to develop them.

Bibliography

J. Alperson and D. O'Neil, "The Boxscore: Tutorials 2, Simulations 0," *Academic Computing,* Feb. 1990, 18, 46-49.

J. Anderson, "Tuning of Search of the Problem Space for Geometry Proofs," *Int'l Joint Conf. Artificial Intelligence,* British Columbia, Canada, Aug. 1981.

J. Anderson, "Analysis of Student Performance with the LISP Tutor" in N. Frederickson and co-workers, eds., *Diagnostic Monitoring of Skill and Knowledge Acquisition,* Lawrence Erlbaum Associates, Hillsdale, NJ, 1990.

J. Anderson and B. Reiser, "The LISP Tutor," *Byte* 10, 159-175, 1986.

J. Anderson, C. Boyle and G. Yost, "The Geometry Tutor," *Proceedings of the International Joint Conference on Artificial Intelligence,* Los Angeles, CA, 1985.

J. Anderson, C.F. Boyle, A.T. Corbett and M.W. Lewis, "Cognitive Modeling and Intelligent Tutoring," *Artificial Intelligence* 42, 7-50, 1990.

B.S. Bloom, "The 2-Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring," *Educational Researcher* 13, 4-16, 1984.

J. Bonar, R. Cunningham, and J. Schultz, "An Object-Oriented Architecture for Intelligent Tutoring," *Proceedings of the ACM Conference on Object-Oriented Programming Systems, Language and Applications,* ACM, New York, 1986.

J.S. Brown and A. Bell, "SOPHIE: A Sophisticated Instructional Environment for Teaching Electronic Troubleshooting (An Example of A.I. in C.A.I.)" in D. Sleeman and J.S. Brown, eds., *Intelligent Tutoring Systems,* Academic Press, Cambridge, MA, 1982.

R. Burton, "The Environment Module of Intelligent Tutoring Systems," in M. Polson and J.J. Richardson, eds., *Foundations of Intelligent Tutoring Systems,* Lawrence Erlbaum Associates, Hillsdale, NJ, 1988.

R. Burton and J.S. Brown, "An Investigation of Computer Coaching for Informal Learning Activities" in D. Sleeman and J.S. Brown, eds., *Intelligent Tutoring Systems,* Academic Press, Cambridge, MA, 1982.

J.R. Carbonell, "Mixed-Initiative Man-computer Instructional Dialogs," Technical Report 1971, Bolt Beranek and Newman, Cambridge, MA, June 1970.

M. Chi, P. Feltovich and R. Glaser, "Categorization and Representation of Physics Problems by Experts and Novices," *Cognitive Science* 5, 121-152, 1981.

W. Clancey, "Case Management for Rule-Based Tutorials" in *Proceedings of the International Joint Conference on Artificial Intelligence*, 1979.

W. Clancey, "Tutoring Rules for Guiding a Case Method Dialog," in D. Sleeman and J.S. Brown, eds., *Intelligent Tutoring Systems*, Academic Press, Cambridge, MA, 1982.

W. Clancey and R. Letsinger, "Neomycin: Reconfiguring a Rule-based Expert System for Application to Teaching," *Proceedings of the Seventh International Joint Conference on Artificial Intelligence,* Vancouver, pp. 829-835, Morgan Kaufmann, Los Altos, CA, 1981.

J. de Kleer and J.S. Brown, "A Physics Based on Confluence," in D.C. Bobrow, ed., *Qualitative Reasoning about Physical Systems*, MIT Press, Cambridge, MA, 1986.

K. Forbus, "Qualitative Process Theory," *Artificial Intelligence* 24, 85-168. Reprinted in D.C. Bobrow, ed., *Qualitative Reasoning about Physical Systems*, MIT Press, Cambridge, MA, 1986.

C.L. Foss, "Learning from Errors in ALGEBRALAND," Technical Report IRL-87-0003. Institute for Research on Learning, Palo Alto, CA, 1987.

J. Hollan, E. Hutchins, & L. Weitzman, STEAMER: An Interactive Inspectable Simulation-Based Training System, *The AI Magazine*, 5(2), 15-27, Summer, 1984.

L. Johnson and E.M. Soloway, "Intention-based Diagnosis of Programming Errors," *Proceedings of the National Conference on Artificial Intelligence*, pp. 369-380, Austin, TX, 1984.

J. Larkin, J. McDermott, D. Simon and H. Simon, "Expert and Novice Performance in Solving Physics Problems," *Science* 208, 1335-1342, 1980.

A. Lesgold, "Toward a Theory of Curriculum for Use in Designing Instructional Systems," in Mandl & A. Lesgold, eds., *Learning Issues for Intelligent Tutoring Systems*, Springer-Verlag, New York, 1988.

A. Lesgold, S.P. Laijoie, M. Bunzo and G. Eggan, "A Coached Practice Environment for an Electronics Troubleshooting Job" in J. Larkin, R. Chabay and C. Shefic, eds., *Computer Assisted Instruction and Intelligent Tutoring Systems: Establishing Communication and Collaboration*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1990.

D. McArthur and C. Stasz, "An Intelligent Tutor for Basic Algebra," R-3811-NSF, RAND Corporation, Santa Monica, CA, 1990.

D. McArthur, C. Stasz, and J.Y. Hotta, "Learning Problem-solving Skills in Algebra," *Journal of Educational Technology Systems,* 15 (3), 303-325, 1987.

Plato, *Laches, Protagora, Meno, and Euthydemus*, W.R.M. Lamb, translator, Harvard Press, Cambridge, MA, 1922.

M. Richer and W.J. Clancey, "GUIDON-WATCH: A Graphic Interface for Viewing a Knowledge-Based System," *IEEE Computer Graphics and Applications,* 5, 51-64, 1985.

L. Schooler and J. Anderson, "The Disruptive Potential of Immediate Feedback," *Proceedings of the 12th Annual Conference of the Cognitive Science Society,* Lawrence Erlbaum, NJ, 1990.

V. Shute, "Rose Garden Promises of Intelligent Tutoring Systems: Blossom or Thorn?" Paper presented at the *Space Operations, Applications and Research (SOAR) Symposium,* 1990. Contact V. Shute, AFHRL, Brooks Air Force Base, TX 78235-5601.

V. Shute and J. Bonar, "Intelligent Tutoring Systems for Scientific Inquiry," *Proc. Cognitive Science Soc.,* Lawrence Erlbaum Associates, Hillsdale, NJ, 1986.

V.J. Shute, R. Glaser and K. Raghavan, "Inference and Discovery in an Exploratory Laboratory" in P.L. Ackerman, R.J., Sternberg, and R. Glaser, eds., *Learning and Individual Differences,* W.H. Freeman, New York, pp. 279-326, 1989.

D. Sleeman and J.S. Brown, eds., *Intelligent Tutoring Systems,* Academic Press, Cambridge, MA, 1982.

A. Stevens and A. Collins, "The Goal Structure of a Socratic Tutor," *Proceedings of the Association for Computing Machinery Annual Conference,* 1977.

A. Stevens, A. Collins and S. Goldin, "Misconceptions in Students' Understanding," in D. Sleeman and J.S. Brown, eds., *Intelligent Tutoring Systems,* pp. 13-24, Academic Press: Cambridge, MA, 1982.

D. Towne and A. Munro, "Artificial Intelligence in Training Diagnostic Skills" in D. Bierman, J. Breuker and J. Sandberg, eds., *Proceedings of the Fourth International Conference on Artificial Instruction and Education,* Amsterdam: IOS, 1989.

D. Towne, A. Munro, Q. Pizzini, D. Surmon, L. Coller, and J. Wogulis, "Model-Building Tools for Simulation-Based Training," *Interactive Learning Environments* 1, 33-50, 1990.

E. Wenger, *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge,* Morgan Kaufmann, Los Altos, CA, 1987.

B. White and J. Frederiksen, "Intelligent Tutoring Systems Based upon Qualitative Model Evolutions," *Proceedings of the National Conference on Artificial Intelligence,* Philadelphia, PA, 1986.

B. Woolf, E. Soloway, W.J. Clancey, K. VanLehn, and D. Suthers, "Knowledge-based Environments for Teaching and Learning," *AI Magazine,* 11(5), 74-77, 1991.

B. Woolf, D. Blegen, J. Jansen and A. Verloop, "Teaching a Complex Industrial Process," *Proceedings of the National Conference on Artificial Intelligence,* Philadelphia, PA, 1986.

N. Yankelovich, M. Meyrowitz and A. vanDam, "Reading and Writing the Electronic Book," *IEEE Computer,* 20(9), 15-30, 1985.