Empowering the Student: Prospects for an Unintelligent Tutoring System

by

Mitchell J. Nathan

Institute of Cognitive Science
University of Colorado
Campus Box 245
Boulder, CO 80309-0345

ICS Technical Report #89-11

To appear in Proceedings of the ACM Computer-Human Interaction, 1990 (Seattle).

Empowering the Student: Prospects for an Unintelligent Tutoring System

Mitchell J. Nathan

Institute of Cognitive Science
University of Colorado, Boulder, CO. 80309-0345
MNATHAN %CLIPR@BOULDER.COLORADO.EDU (303) 492-6492

ABSTRACT
Computer based instructional systems either direct students so modelling their actions is tractable, or provide them with total autonomy, but give little support to learning and problem solving processes. Instructional principles for empowering the student are emerging whereby more of the responsibility of diagnosis and goal-setting is placed on the student. Critical to this view is providing an environment which makes the ramifications of students' actions clear so students can meaningfully assess their own performance. In the domain of word algebra, the meaning of formal expressions can be reflected in computer animation which depicts the corresponding situation. An unintelligent tutor -- knowing nothing of the problem being solved and possessing no student model -- helps students to understand problems and debug formal expressions.

KEYWORDS: Active learning, intelligent tutoring systems, problem comprehension, discourse processing, mathematics instruction cognitive psychology.

INTRODUCTION
In his keynote address at the 1980 Carnegie Symposium on Cognition, Herbert Simon acknowledges the long, slow, tedious task that is human learning [22]. As arduous as the task is, it is made more so when we as educators direct not only what students learn but *how* they must learn it. In the field of Intelligent Tutoring System (ITS) development, cognitive scientists are applying information processing principles to produce computerized environments for instruction. The results are computer programs which guide the student through the learning and problem solving process.

This approach has been challenged recently on several counts by [9], [20], [21]. Scardamalia and others contend that ITSs do too much thinking for the student and do not encourage the student to develop many higher level cognitive skills and learn with one's own pace and style [20]. Systems, they argue, must engage students, encouraging them to use their intelligence and knowledge, rather than using knowledge and intelligence to merely guide learning. Computer systems that give students total autonomy are not the solution either. They can lead to poor performance because the system cannot help the student (a) learn to learn; (b) set cognitive goals; (c) facilitate problem comprehension; and (d) develop self-monitoring and knowledge organization skills.

In this study we present a set of principles derived from cognitive psychology along with recent experimental findings which suggest that a tutoring environment which *empowers* the student will be ultimately more beneficial than one which guides the student through a rigid problem solving process. By "empowering students," we mean that the tutor gives students the opportunity to address each task in their own style and pace, and authorizes students to assess their own performance. Such a tutor places more of the responsibility of learning, goal-setting, and diagnostics with the student than traditional ITSs. The tutor must also provide enough

structure to facilitate good knowledge organization and problem comprehension, and a rich enough environment to help the student perform self-diagnosis and correction. A tutoring system is being tested which embodies many of these principles. It is being used to teach students how to understand and solve word algebra problems. With it, students graphically construct a formal problem model which drives a computer animation of the perceived situation. Students compare the resulting animation (e.g. planes flying toward each other at different speeds) to their own situation or mental model in order to evaluate and, if necessary, alter the problem model which is then used to generate a solution. The system has no model of the student or knowledge of the problem being solved, yet experimental results indicate that it provides students with valuable cues for self-assessment of their understanding of a problem.

TUTORING SYSTEM DESIGN AND PRINCIPLES OF INSTRUCTION

Anderson, Boyle, Farrell, and Reiser [4] present several principles derived from experimental research in cognitive psychology which they argue are central to tutoring. Many of these have received little challenge. There is strong agreement, for instance, on the importance of minimizing students' working memory loads[1]. Also widely agreed upon is that making students' goals overt, instructing them in the context of the problem solving task, and providing support for iterative problem solving all help the student in task performance and skill acquisition [8].

One principle which has received a lot of attention states that the ITS feedback for error correction needs to be immediate. When students get off the proper solution track, Anderson [1] has shown that they can get hopelessly lost and must use tremendous cognitive resources to get back to their original goals. Episodes such as this do little to help students learn and can confuse the memory traces of correctly learned behavior. The *model tracing paradigm* is intended to prevent such occurrences. In this view, the system tries to trace the cognitive states of the student in real time, and provide immediate feedback when the student deviates from any of the expected (i.e. permissible) states. To support this, ITSs employ an internal knowledge base, the expert module [2], to classify student actions as acceptable or unacceptable. The approach makes two important presumptions: that designers can foresee all of the legitimate solution paths for a set of problems in a rich domain (such as programming); and that the ITS can appropriately diagnose how students have erred -- what they meant -- when solution attempts deviate from those recognized by the expert module[2].

A necessary precondition to building a tutor's expert module is a thorough task analysis of the legal transformations that can be performed on problems in a variety of representational forms. A system which omits from this analysis an unusual or highly stylized solution method is one which will fail to acknowledge valid solutions as legitimate. The source of students' errors can also be misjudged. Cummins, Kintsch, Reusser, and Weimer [6] have shown that errors made by first graders solving word arithmetic problems can be traced to misunderstanding the language of the problem. Behavior classified as the application of an inappropriate strategy was in some cases shown to be the correct strategy to a misunderstood problem. Feedback and remediation directed at the use of arithmetic strategies in this instance would be improper. The correct action is to help the student to better decipher the wording of the problem.

Learning By Erring

The model tracing paradigm also presumes that students are not capable of identifying and correcting their own errors. People can and do learn from their own actions. However, people can also *not* learn from their errors. It is this finding rather than the former that is the more pervasive in the human-computer interaction literature (e.g. [5], [10], [15]) and which supports the immediate feedback view.

It may be possible to provide people with an environment that helps them to diagnose their problem solving errors and put themselves "back on track" without impairing their learning. In his study of how users explain the actions of fictitious computer systems, Lewis [11] showed that a small but powerful set of causal reasoning heuristics can go a long way toward describing user behavior. When confronted with unfamiliar procedures, these heuristics seem to play a crucial role in organizing examples in memory and in generalizing from a small set of instances.

---

[1] Scardamalia and others [20], for instance, support practices that favor understanding and problem solving behavior over rehearsal and memorization.

[2] When discussing the underlying assumptions of the CMU LISP Tutor, Anderson [3] assumes that "when we interrupt students we correctly understand their internal states."

Educational programs which help students perform their own causal reasoning for diagnosis by exploiting these and other heuristics allow the student to take charge of the learning process and provide the tutoring system designer with a helpful ally in error diagnosis.

Two recent studies which manipulated tutor feedback seem to challenge the immediate feedback approach. Both studies suggest that withholding feedback from the student may engage the student more in the learning process and so lead to greater performance and transfer to novel situations. In a recent study [21], subjects in a LISP learning task received either no tutor (and so learned by exploration), a selective tutor (which intervened when two consecutive errors occurred), or a constant tutor. Subjects interacted with a set of examples in the LISP environment until they felt capable of solving related problems. Although post-test scores did not differ significantly, tutored subjects required significantly less time to solve test problems $F(2,14)=4.6$, $MSE=47.3$, $p<.05$. Furthermore, subjects in the selective tutor condition spent less time exploring examples and produced the fewest erroneous inputs, suggesting that learning is most efficient with selective tutoring. The authors concluded that tutor help, even in the form of correct solutions, interrupt students' thought processes and can therefore alter the entire memory trace (cf. [7]). Students, they suggest, learn most efficiently when they can see the effect of their errors and must reason causally about the source and meaning of error messages and unexpected system responses [21].

Lee [9] constructed a tutor for the domain of genetics which followed all of the instructional principles of Anderson and others [4] except feedback, which was either immediate or delayed. Subjects built diagrams on the computer to solve pedigree problems. Delayed feedback subjects had to produce complete diagrams before any errors were reported. They then had to rebuild some or all of it to correct any errors. Lee [9] found that the learning time was fastest for the immediate feedback group, $F(1,19)=31.95$, $p<.01$, which is consistent with many earlier findings (e.g. [13]). The post test revealed that delayed feedback students performed better overall than immediate feedback students $F(1,19)=9.13$, $p<.01$, primarily because of their performance on hard and novel problems. Lee concludes that immediate feedback subjects appear to employ a guessing strategy which interferes with learning: They construct a diagram and at certain points guess what to do next with confidence that if it is wrong the tutor will tell them how to correct it. Delayed feedback subjects had more at stake if the diagram was flawed since they would have to reconstruct all parts following the error. Consequently, delayed feedback subjects used more independent checks and considered their actions more carefully. They learn error detection and correction skills better than immediate feedback subjects who never have the opportunity to apply them (cf. [13]).

ANIMATE: AN UNINTELLIGENT TUTORING SYSTEM

A word algebra tutor has been developed which encourages active problem solving by the student. The tutor, ANIMATE, is an interactive computer system that runs on the Apple (R) Macintosh. It is intended to facilitate comprehension of a story problem by helping the student to construct both an animated situation model and an accompanying formal problem schema. Normally the problem schema is an implicit, intermediate mental structure in a long line of such structures generated from the initial stages of reading a problem to the eventual production of a solution [19]. By making this structure and its relation to the problem situation explicit, we hope to give the student a more concrete understanding of the conceptual relations in a problem and the cognitive tasks that need to be addressed to solve it.

Problem Schemata

ANIMATE uses a graphical arrangement of nodes and arcs to organize the information in word algebra problems (see Figure 1). Nodes serve as placeholders for numbers and unspecified values (variables) extracted from the problem statement. Arcs indicate the relations (constraints) among nodes (e.g. +, -, X, /, =). An uninterrupted horizontal or vertical sequence of nodes and connecting arcs forms an equation. Weaver and Kintsch [24] demonstrated the psychological reality of a problem schema level, similar to the network above, as intermediate to the cover story and underlying equations. They showed that subjects unfamiliar with the technique of first organizing the problem information into a problem schema, rated problem pairs significantly more similar when similarity was based on underlying schema structure, than when based on underlying equations, $F(1,17) = 26.6$, $p<.01$. When subjects were *trained* in the problem schema method, that difference increased significantly, $F(1,34) = 98.2$, $p<.01$.

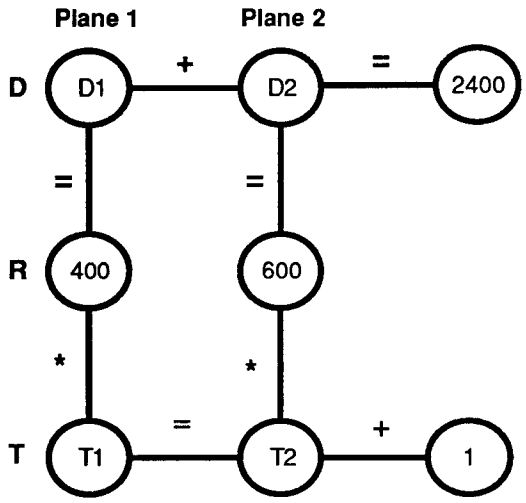

Figure 1: Organizing information of a typical collision problem (Problem 1) with the network problem schema.

Using ANIMATE

The student reads a problem such as Problem 1 below, and forms a semantic representation and situation model as is done with any text [23].

> *Problem 1.* A plane travels east toward Denver, which is twenty four hundred miles away, at four hundred mph. One hour later another plane travels from Denver on a collision course at six hundred mph. How much time will the air traffic controller have to avert an accident?

Students use ANIMATE to develop problem schemata and an animation by constructing and filling in an algebraic network. From the network they then extract the algebraic equations, as shown below.

$$D1 = 400 \times T1 \qquad \text{(Eqn. 1)}$$
$$D2 = 600 \times T2$$
$$D1 + D2 = 2400$$
$$T1 = T2 + 1$$

Consider the following problem.

> *Problem 2.* A supertrain leaves Boston headed cross-country at a rate of two hundred and fifty mph. If an airplane leaves the same city two hours later on a parallel course, how many miles outside of Boston will the plane overtake the train?

A student understands the situation as a plane and a train traveling at different rates. The plane leaves later and eventually overtakes the train. The student must infer that "overtake" means formally, "when the distances are equal." When equations are the only means to express these relations and no provisions for feedback are made, novice algebra students have tremendous difficulty [12].

The student may specify all of the information for the network of one character before building the network for the second (e.g. depth-first, as shown in Figure 2), or may interchangeably specify the network for both planes. Order is unimportant. Nodes, or "bubbles," and link operators are mouse sensitive and values and operators may be changed at any time. When a student selects a node or link, a calculator pops up indicatingt the student must enter a value. The animation will not run with an empty network; partial specification of an icon (such as the rate) is necessary. The RUN command starts the animation. The icons move and value gauges stop at the specified time or distance. The addition of a second character allows the student to compare relative rates of travel and build in constraints such as delays between icons.

ANIMATE provides the student with two ways to check his or her work. Syntactic (i.e. algebraic) checks are made by the system when the animation starts. Algebraically incorrect expressions are highlighted (Figure 2). The student must acknowledge the error and has the option to correct it then, or run the animation described by the incorrect problem model. Semantic checks are made by the student when assessing the animation which represents the situation model depicted in the network. Errors in conceptualizing the problem schema, such as interpreting lateness to mean "minus the delay" for the plane, result in animation which is counter to the student's expectations, such as the plane leaving before the train, as depicted in Figure 3. Mathematically the network may be correct; that is, algebraically consistent. Since the slower train can never catch up in this problem model, the net is situationally erroneous, however. Mismatches between the animation and the student's own situation model suggest how to correct the problem network. In the instance depicted in Figure 3, altering the delay operator will produce the intended display, so the plane leaves later and eventually passes the train, as shown in Figure 4.

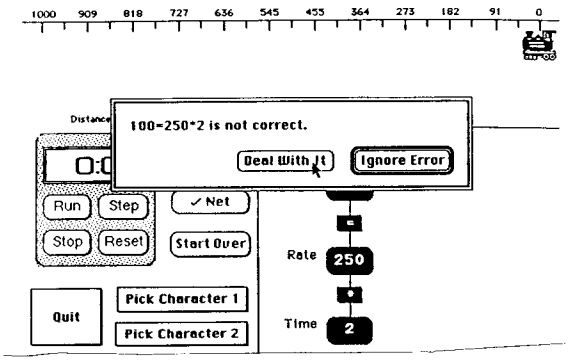

Figure 2: A partial problem schema for Problem 2. ANIMATE highlights syntactically incorrect expressions which the student may alter or ignore. At this point the student could be describing either a collision or an overtake problem situation. From [18].

It is largely the student who controls the interaction and the problem solving process. ANIMATE is incapable of assessing how well the animation matches the problem or the student's own internal situation model. It knows nothing of the specific problem being addressed, only whether or not the network is consistent. Evaluating the animation is simple for the student since it involves applying knowledge about common situations.

Experimental Findings

Twenty-four subjects from the University of Colorado Psychology 100 pool were randomly assigned to one of three groups: Equation, Network or Animation. The Animation group learned how the network method is used to construct animation on the computer. All subjects received three tasks. Task 1 presented a travel word problem and then three possible solutions, represented as either sets of equations (for the Equation group) or networks (the Network and Animation groups). Subjects used equations, networks, or the computer system to select from among the three choices the most appropriate formalism for the problem given. This measured subjects' ability to recognize a legitimate formulation of a solution. Task 3 resembled Task 1, except subjects were given a *solution* to a problem and told to select the appropriate *problem passage* being solved by the given solution. This task was further distinguished in that there were two correct solutions (choices A and B) which accounted for the given solution. It was predicted that Animation subjects may form a situational bias causing them to select choice B, which paralleled the situation given on the screen of two planes on a collision path, over A, describing two planes leaving the same place and travelling in opposite directions. A and B were both mathematically legitimate choices. Task 2 showed a formal solution that contained flaws. Subjects were to correct the formalism so the solution properly reflected the problem statement given.

Statistical analyses show that overall the Animation group performed superior to the other two groups. The Network method didn't lead to better problem solving performance than the equation method (with 30 min of tutoring) which is consistent with an earlier study [16]. The results of Task 1 -- which primarily tested

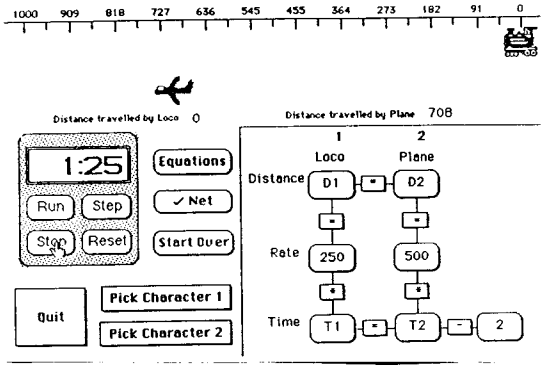

Figure 3: A (situationally) incorrect expression in the T (time) row for the schema of Problem 2. With the delay negated from T2, the animation (top) shows the (faster) plane leaving first, while the (slower) train waits. Overtake never occurs. From [18].

solution recognition -- show no group differences and a possible "ceiling effect" for this easy problem. Results from Task 2 show that the Animation group performed best in error detection and correction overall, $F(1,14)=6.7$, $p<.01$. This is not surprising considering this is the only group to receive feedback. A subsequent study which compared a computer version of the Network condition with syntactic (i.e., equation or network level) feedback but no animation, to the Animation condition [17] suggests that it is the situationally relevant feedback and not simply *any* feedback which helps subjects to find and correct errors in the network formalism.

When we look specifically at the two flaws in Task 2 we find that those operating with the

equations performed as well as the Animation group in correcting the "distance-equals" omission[3]. We speculate that equation users have learned to employ certain "checks" on their work, such as making sure the number of variables equals the number of (independent) equations. It is also possible that subjects simply remembered to include this from past episodes. Earlier episodes or rules may have been muddled for the Network group who used a novel approach.

The correction needed for the "delay" flaw involved recognizing that the minus sign in the equation must be a plus for the equation to faithfully represent the situation. Formally, this is a subtle error and is further hampered by the misleading use of the term "later" which to some suggests "minus" (cf. [14]). Situationally, however, this flaw is quite salient, depicting the fast plane leaving first trailed by the slower one. Not surprisingly, those in the two groups receiving no animation feedback performed significantly below the level of the Animation group, $F(1,14)=6.76$, $p<.01$.

The third task tested for a situational bias on the part of the Animation group using a problem pretested to be very easy. All of the subjects (as expected) selected a correct answer. Of greater interest is *which* correct answer subjects chose. Those in the Equation group were perfectly split, 50-50, on choosing either the statement describing a collision and one describing two planes flying apart, showing no preference for one situation over the other. In direct contrast, a full 100% in the Animation group chose the "collision" passage, passing up the "parting" story, choice A, as the problem best described by the given solution. This points to the power of situational reasoning on problem perception. The Network group fell in between, with 90% choosing the collision scenario, leaning toward the Animation group. Earlier work (e.g. [16]) suggests that subjects using the network formalism develop a stronger bias for situation model based reasoning about story problems than do those working with equations. The equation formalism, which is isomorphic to the network, did not demonstrate this same effect.

System Evaluation

[3] In the given problem, the formalism omitted the relation that the distances of the two planes were to be equal at the point of overtake. This is corrected in the network formalism by connecting the two distance ovals with an "equals" link, and by introducing the equation "D1 = D2" in the equation formalism.

In comparison to users receiving only syntactic feedback in the network technique and subjects using the traditional equation based approach, ANIMATE seems to help students to recognize and understand the formal expressions needed for word problem solving. ANIMATE does this in such a way as to empower the student, providing great flexibility in how problems are solved. It provides students with meaningful feedback that supports diagnosis and correction even though it has no stored knowledge base of the problems at hand or of its users.

ANIMATE addresses many of the principles of active learning and we are currently testing the role that each of them plays in problem solving and learning. It provides a way to organize problem information and keep track of one's goals. This frees up a student's cognitive resources for such activities as planning and causal reasoning about feedback. The system

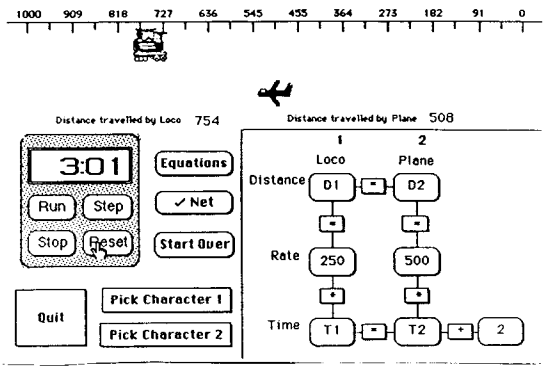

Figure 4: The T (time) row of the problem schema for Figure 3 is corrected so the train leaves first in the animation (top). The plane leaves after a two hour delay and catches up at the 1,000 mile point. From [18].

promotes use of varied knowledge representations since students are encouraged to reason situationally as well as formally. ANIMATE also allows for partial solutions which students iteratively refine, letting them focus on the different aspects of the problem as their own learning demands.

ANIMATE fails in a number of ways to promote active learning and address other recent concerns in computerized instruction. There is no "Help" facility for students. If they do get lost and either do not understand the system operation or the problem, students are left to flounder or seek out an instructor. An on-demand help facility would

likely lead to improved performance and lessen the burden of learning the computer interface. The tutor also does not provide selective or delayed feedback. Given the empirical findings cited above, such feedback for the student seems valuable. Students are also not provided with an environment that supports arbitrary representation building. Currently, network construction is subject to limitations, as is animation construction. A more general "construction kit" approach which retains the structure needed for self-diagnosis deserves greater consideration.

DISCUSSION

A number of questions arise regarding principles of instruction. How readily can one apply such principles to the design of tutoring systems? How do these principles interact with each other and with the domain? Which principles deserve the most weight? It may be evident when evaluating a system whether or not a set of principles has been adhered to. Yet is often unclear how to incorporate these principles into a particular system, since the principles themselves do not sufficiently constrain the system design. Only through experimentation can one determine the relative effects of individual principles on the learning and problem solving process.

In this paper we have addressed some theoretical issues of computer based instruction and examined them in reference to a tutor ignorant of the problems being solved by a student. The system operates in a domain that is outside of the current focus of intelligent tutoring research -- word problems -- since, to be intelligent, such a tutor must possess knowledge of students' interpretations and mis-interpretations of the language of word problems. We avoid the issue of "correct" or "allowable" solutions by placing the burden of assessment on the student. In this way the student's rich store of situations and skill in language understanding is exploited.

The system design regards problem solving as an active process to which students can bring to bear all of their cognitive and experiential resources. Its view of the student is as a diagnostician and explainer as well as a problem solver. The merits of empowering the student and the possible role one plays in assessing one's own problem solving behavior needs further attention. It is not simply by relying on findings from cognitive psychology but by application of these principles to such areas as computer based tutoring, that we can hone our theories of instruction.

ACKNOWLEDGEMENTS

I thank Clayton Lewis and Walter Kintsch for the many, many discussions regarding ANIMATE and the underlying theory of problem comprehension, and Peter Polson and Kurt Reusser for their valuable comments on this research. Also, my warmest appreciation goes out to Paul Johl and Emilie Young who gave up many free and not-so-free hours to help develop the theory and various versions of the tutor.

REFERENCES

1. Anderson, J. R. Acquisition of complex skill. *Psychological Review, 89,* (1982), 369-406.

2. Anderson, J. R. The expert module. In M.C. Polson and J.J. Richardson, Eds., *Foundations of intelligent tutoring systems.* Erlbaum, Hillsdale, NJ, 1988.

3. Anderson, J. R. Analysis of student performance with the LISP Tutor. In Frederickson, Glaser, Lesgold, and Shafto, Eds., *Diagnostic monitoring of skill and knowledge acquisition.* Earlbaum, Hillsdale, NJ, 1989.

4. Anderson, J. R., Boyle, C. F., Farrell, R., and Reiser, B. Cognitive principles in the design of computer tutors. Carnegie-Mellon Technical Report # ONR-84-1. 1984.

5. Carroll, J. M., and Rosson, M., B. Paradox of the active user. In J. M. Carroll, Ed., *Interfacing thought.* MIT Press, Cambridge, MA, 1987.

6. Cummins, D., Kintsch, W., Reusser, K., and Weimer, R. The role of understanding in solving word problems. *Cognitive Psychology. 20,* (1988), 439-462.

7. Ericsson, K. A. and Simon, H. *Protocol analysis: Verbal reports as data.* MIT Press, Cambridge, MA, 1984.

8. Glaser R., and Bassok, M. Learning theory and the study of instruction. University of Pittsburgh Learning Research and Development Center Technical Report no. 11, 1989.

9. Lee, A. Timing of feedback in tutoring systems. University of Colorado Technical Report 89-10, 1989.

10. Lewis, C. Understanding what's happening in system interactions. In D. A. Norman

and S. W. Draper, Eds., *User centered system design: New perspectives on human-computer interaction.* Erlbaum, Hillsdale, NJ, 1986a.

11. Lewis, C. Why and how to learn why: Analysis-based generalization of procedures. *Cognitive Science, 12,* (1986b), 211-256.

12. Lewis, C. H. Learning about computers and learning about mathematics. In J. M. Carroll, Ed., *Interfacing thought.* MIT Press, Cambridge, MA, 1987.

13. Lewis, M.W. and Anderson, J.R. Discrimination of operator schemata in problem solving: Learning from examples. *Cognitive Psychology, 17,* (1985), 26-65.

14. Lewis, K., and Mayer, R. E. Students' misconceptions of relational statements in arithmetic word problems. *Journal of Educational Psychology, 79,* (1987), 363-371.

15. Mack, R. L., Lewis, C. H., and Carroll, J. M. Learning to use word processors: Problems and prospects. *A C M Transactions on Office Information Systems, 1,* (1983), 254-271.

16. Nathan, M. J. Recall of stories and story problems. Unpublished Masters Thesis. University of Colorado, 1988.

17. Nathan, M. J., Kintsch, W., and Young, E. Toward a model of word problem comprehension and its implications for unintelligent tutoring. In preparation.

18. Nathan, M. J., and Young, E. Thinking situationally: Results with an unintelligent tutoring system for word algebra. Submitted.

19. Reusser, K. From text to situation to equation: Cognitive simulation of understanding and solving mathematical word problems. Research report no. 5. Universitat Bern, Switzerland, 1988.

20. Scardamalia, M. E., Bereiter, C., McLean, R. S., Swallow, J., and Woodruff, E. Computer-supported intentional learning environments. *Journal of Educational Computing Research, 5,* (1989), 51-68.

21. Schmalhofer, F., Kuehn, O., Messamer, P., and Charron, R. An experimental evaluation of different amounts of receptive and exploratory learning in a tutoring system. To appear in Proceedings of the 19th Annual Meeting of the Society for Computers in Psychology. (Atlanta, GA), 1989.

22. Simon, H. A. Why should machines learn? In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell Eds., *Machine learning: An artificial intelligence approach.* Tioga, Palo Alto, CA, 1983.

23. van Dijk, T. A., and Kintsch, W. *Strategies of discourse comprehension.* Academic Press, New York, 1983.

24. Weaver, C. A., and Kintsch, W. The conceptual structure of word algebra problems. University of Colorado Technical Report 88-12, 1988.