A Theory of Algebra Word Problem Comprehension and its Implications for Unintelligent Tutoring Systems

by

Mitchell J. Nathan and Walter Kintsch

Institute of Cognitive Science

University of Colorado
Campus Box 345
Boulder, CO 80309-0345

and

Emilie Young

U S West Advanced Technologies
Denver, CO

ICS Technical Report #90-02

A Theory of Algebra Word Problem Comprehension and its Implications for Unintelligent Tutoring Systems

Mitchell J. Nathan and Walter Kintsch

University of Colorado

Emilie Young

U S WEST Advanced Technologies, Denver, Colorado

Running Head: ALGEBRA WORD PROBLEM COMPREHENSION

Abstract

A tutoring approach is derived from a model of problem comprehension, based on the van Dijk and Kintsch (1983; Kintsch, 1988) theory of discourse processing. A problem statement is regarded as a text from which the student must glean propositional and situational information and make critical inferences. The student must coordinate this information with known problem schemata so that formal (i.e. algebraic) operations can be applied and exact solutions obtained. We argue that this task is a highly reading oriented one where poor text comprehension and an inability to access relevant long-term knowledge lead to serious errors. Further, formal algebraic expressions are so abstract their meaning is often elusive; this contributes to mistranslations and misinterpretations. We describe experimental results with ANIMATE, an unintelligent tutoring system which knows nothing of the problem at hand or of the student's actions. Subjects who build animations of situations described in typical word problems consistently outperformed tutor users with no animation and students using only equations in both training tasks and near and distant transfer tasks. Performance differences were greatest for novel problems. We conclude that by providing an environment which gives equations situation-based meaning and makes the ramifications of students' formal manipulations clear, students learn to relate formal expressions to the referent situations. This enhances problem comprehension and gives a stronger representational base to the problem solving process.

A Theory of Algebra Word Problem Comprehension and its Implications for Unintelligent Tutoring Systems

Cognitive scientists are currently paying a great deal of attention to the development of Intelligent Tutoring Systems (ITSs) for areas traditionally left to human instruction, such as computer programming instruction (Anderson, Boyle, Corbett, & Lewis, in press; Reiser, Kimberg, Lovett, & Ranney, 1989) and mathematics (e.g. Anderson, Bolye & Yost, 1985; Brown, 1985; Singely, Anderson, Gevins, and Hoffman, 1989). Intelligent tutors are computer programs intended to track the behavior of a student and provide meaningful feedback about performance errors. At the core of an ITS is an expert system for the field of concern (e.g. geometry). At the core of the ITS *philosophy* is the belief that by understanding the student's learning processes and feedback requirements, a knowledge-based system can be produced which will appropriately interact with the student to bring about heightened skill retention and problem comprehension. Yet there are areas of instruction -- such as word problem solving -- where we seem unable to provide much help to students. Furthermore, while it may be possible to build expert knowledge into computer systems, the educational intent behind a tutor is not simply to construct an expert capable of correcting student actions. Of paramount importance is to instruct the student so that his or her performance is enhanced when the tutor is removed.

Developing a better understanding of the psychological requirements for problem comprehension may alert us to students' instructional needs and provide us with the functional requirements for computer-based tutors. In this paper we present a theory of word problem comprehension based on research in the field of discourse processing (e.g. van Dijk & Kintsch, 1983). The theory focuses on the mental representations that are produced during reading comprehension. In solving word arithmetic or word algebra problems, errors made by students can be viewed as failures to produce the intended mental

representations, or failures to see how the situation described in the problem statement relates to the formal expressions one must work with in order to produce a quantitative (e.g. numeric) solution. Previous work has shown that text complexity and comprehension failures are central to the difficulty of word problems (Carpenter, Corbitt, Kepner, Lindquist, & Reys, 1980; Cummins, Kintsch, Reusser, & Weimer, 1988; Kintsch & Greeno, 1985). Indeed, empirical analyses of how college students actually solve word algebra problems have shown that reasoning explicitly about the situation, and not the algebra *per se*, plays a crucial role (Hall, Kibler, Wenger, & Truxaw, 1989). Here, our major theoretical claim is that in order to comprehend a problem *the student must make a correspondence between the formal equations and the student's own informal understanding of the situation described in the problem.* Our tutoring system facilitates this correspondence by making this relationship explicit. In doing so, we believe students will better comprehend the problem and ultimately improve their problem solving performance.

In this paper we describe a computer tutoring system, ANIMATE, which embodies our theory and present results of students' use of it when solving typical high school and college level algebra word problems. Our theory suggests that development of an *intelligent* tutor is not necessary, as long as the tutoring environment triggers the appropriate reasoning from the student, allowing the student to generate, manipulate and understand abstract, formal expressions. Students learn to construct a formal *problem schema* which organizes both information given in a problem statement and information inferred by the reader (Kintsch & Greeno, 1985). The information in the problem schema -- the values and mathematical relations -- is in turn used by the tutor to drive a computer animation. The student compares the activity shown on the computer screen to his or her own expectations based on the situation described in the problem story. Errors in specifying the problem schema lead to incorrect animations, the resolution of which is highly constrained by its

relation to the situation. Students "debug" the problem schema until they are satisfied that the appropriate animation appears on the screen. It is then a simple matter to derive the appropriate equations from the schema which can then be solved. The system we have developed is *unintelligent* in the sense that it knows nothing of the specific problems under consideration and has no model of the students' knowledge base, though it knows about algebra and the conceptual structure of algebra word problems. The experimental work to be reported below shows that even such a minimal system can help students to better understand and solve algebra word problems.

We begin this paper by illustrating the derivation of our model of problem comprehension, drawing on earlier work in this area. After highlighting the essential components of current theories, we discuss the distinctions we have drawn regarding students' mental representations of a problem. This is followed by an elaboration of the role that each of the model's components plays in the comprehension process. In conjunction, recent discussions concerning computer-based instruction, the role of student errors, and system feedback during student training, are presented. From this, and various theories of instruction, a set of instructional principles are derived which guide the design of the tutor ANIMATE. After illustrating how ANIMATE relies on our model of problem comprehension, the functionality and features of the tutor are demonstrated by way of an example problem solving session. We then present empirical results of students use of the tutor in a laboratory setting. These preliminary results are encouraging. They are discussed both in terms of problem solving performance and near and distant skill transfer. Finally, we present our findings in the context of modelling human problem solving.

Problem Solving and Comprehension

A (complete) theory of problem solving must include the language comprehension process, the resulting mental representations, the role of inferences and real world

knowledge, and the necessary formal calculations for deriving a solution. While models of problem *solving* behavior abound (e.g. Newell & Simon, 1972; Paige & Simon, 1966) few studies address the process by which solvers *comprehend* problems; that is, how they digest problem information and access the relevant real world knowledge so they may then apply solution strategies to a coherent problem representation.

In this paper we outline such a theory, extending the work of Kintsch and Greeno (1985), Reusser (1988), and Cummins et al. (1988). These studies have shown that arithmetic word problem comprehension can be understood within the framework of the general theory of discourse processing of van Dijk and Kintsch (1983). Thus, Kintsch and Greeno theorize that when reading a problem statement, a propositional representation, termed the *textbase*, is formed, as with any other text, to capture the meaning of the passage. The reader also forms a (qualitative) situation model which is the basis for understanding actions in a text (Reusser, 1988). Models for concrete situations and events are often based on imagery, especially dynamically changing imagery as in an animation. Problem solvers additionally must produce a representation of the problem structure which includes the conceptual relations among quantities in the problem. This schema, or *problem model* as it has been referred to from time to time, is the level at which students can apply formal calculation methods such as algebra for generating verifiable solutions.

Kintsch and Greeno (1985) equate the problem model with the situation model, while here we distinguish between a representation for events and one which is constructed with formal relations in mind. In our view, understanding and solving word problems, therefore, involves three mutually constraining levels of representation which must be constructed by the student: a representation of the textual input itself -- the *textbase*; a model of the situation conveyed by the text in everyday terms -- the *situation model*; and the formalization of that situation -- the *problem model.*

Kintsch and Greeno (1985) simulated the reading behavior of first graders solving arithmetic problems, using a production-rule model. Productions were used to construct a representation of a story problem and to select among possible problem solving strategies. The problems always consisted of several sets of objects (e.g. marbles) specified in a certain way (e.g. Fred gave some to Joe). The simulation demonstrated that the problem solving processes used by students could be reliably modelled by a computer program. Later work by Cummins et al. (1988) showed that incorrect problem solving behavior could be simulated by introducing faults into the program. Of the two classes of defects introduced -- incorrect arithmetic algorithms and linguistic deficiencies -- language processing errors modelled students' faulty behavior best. Cummins et al. (1988) also showed that solution performance was associated with the ability to recall the problem statements. Students who could correctly recall the stories were more likely to produce correct solutions. Students who incorrectly recalled the problems apparently encoded a different problem than the one presented, which they often then solved correctly. This study suggests that word problem solving is highly dependent upon language comprehension skills and prescribes that instruction must focus on linguistic as well as mathematical aspects of word problem solving.

Algebra Word Problem Solving

Processing algebra word problems can be viewed as similar to processing arithmetic problems. First, a propositional textbase is formed, just as with any other text. This textbase is organized into a (qualitative) situation model and mapped into an algebraic (quantitative) problem structure. A set of algebraic schemata -- templates for organizing problem relevant information -- serve as cues for constructing these problem models. The schemata which allow one to construct models for the word problems commonly found in college algebra texts are described in the following section. We then show how equations

can be derived from the relations specified in a problem model using a method of constraint propagation. We first illustrate this process with a single, worked-out example.

Problem 1: Distance-Rate-Time Word Problem

A plane leaves Denver and travels east at two hundred miles per hour. Three hours later a second plane leaves on a parallel course and travels east at two hundred and fifty miles per hour. How long will it take the second plane to overtake the first plane?

Generating a Propositional Representation

We show how a reader extracts the propositional representation from a problem. This step is not performed by the tutor but by the reader attempting to encode the presented information. We include it to give the entire workings of our model of problem comprehension, accounting for the transitions from a text, to a set of mental representations for the meaning of the text, to a final, mathematical description. When reading Problem 1, a list of propositions can be derived (as in van Dijk & Kintsch, 1983):

Prop1 LEAVE[PLANE1, DETROIT, Time1]

Prop2 RATE[Prop1, 200 mph]

Prop3 DIRECTION[Prop1, EAST]

Prop4 LATER[Prop1, Prop5, 3 HOURS]

Prop5 LEAVE[PLANE2, DETROIT]

Prop6 DIRECTION[Prop5, EAST]

Prop7 LOCATION[Prop5, Prop8]

Prop8 PARALLEL[COURSE]

Prop9 RATE[Prop5, 250 mph]

Prop10 HOWLONG[Prop11]

Prop11 OVERTAKE[PLANE2, PLANE1]

The top-level macroproposition for this text would be "PLANE2 overtakes PLANE1," with information about how fast they are going, when they are starting, and so on, subordinate to it. The corresponding situation model might involve a description of the two planes speeding along a parallel course, capturing the moment when the second plane just passes the first. All of this would be just as in understanding a story. We assume that students have sufficient experience to allow them to understand the situation adequately. This will be the case for many of the algebra problems we consider here, but not for all. For some problems (e.g. those involving the notion of monetary "interest," which is quite vague for many students) we shall have to teach the situational understanding first.

Generating a Problem Model

In solving an algebra word problem, a mathematical description must be composed which is consistent with the situation model. In constructing this description we note from our example that Prop2 specifies a relation between a distance and a time, thus eliciting a distance-rate-time (DRT) schema labelled PLANE1 (see Figure 1). This schema has slots for distance (D), rate (R), and time (T): Prop2 is assigned to Slot R, while Prop1 and Prop3 go into slots which help to further specify the problem. Prop9 initiates the construction of a second DRT schema for PLANE2, fills the R slot with 250 mph, and assigns Prop5 through Prop8 to the role of additional specifications. Prop4, because it relates the two DRT schemata, is tagged as a supporting relation; one which holds "delay" information. Supporting relations are special. They are not generally found alone, but are used to relate two or more stand-alone schemata. Prop10, which the problem solver must reason deals with the travel time of Plane2, is assigned to the T slot in PLANE2, indicating that it must be filled with a variable or "unknown," while Prop11 acts as another supporting relation. Thus, propositions in the text base aid in the construction or

instantiation of a (situation-specific) problem schema, which in turn organize the propositions into solvable algebraic formalisms.

We have chosen a simple graphical form to represent the problem schema[1]. Each "Distance = Rate * Time" schema is shown as three ovals, corresponding to the distance (D), rate (R), and time (T) slots, respectively, in a vertical arrangement, connected by line segments labelled by mathematical operators, such as "=" and "*". For clarity in analyzing this example, the text propositions organized by these schemata are written next to the ovals and/or lines to which they have been assigned (see Figure 1). Inside each oval quantitative information derived from the text propositions is shown. Line segments are labelled in such a way that the lines and ovals can be read as equations, either horizontally or vertically.

Insert Figure 1 about here

A traditional algebraic representation for the story problem given above along with the corresponding propositions and inferences from which it is derived would be

Equations	Propositions	Inferences
Dist1 = 200 x Time1	1, 2, 3	--
Dist2 = 250 x Time2	5, 6, 9	--
Time1= Time2 + 3	1, 4	"later" as "+ 3"
Dist1 = Dist2	5, 6, 7, 11	"overtake" as "="
T2 = ?	10	--

The distinction between understanding the situation in everyday terms, and understanding the problem structure in terms of an algebraic schema is central to our approach to tutoring: Students require the most support when constructing an algebraic problem model. They are often familiar with the everyday situations which these problems describe, and indeed, this familiarity can serve as a source of support for constructing the formal model. We do not propose a stage theory, however, in which situational understanding must precede the formation of the problem model. The kind of situation-based understanding that is required for word problems must from the very beginning be driven by the salient features of the problem text. The algebraic schema provides a perspective that is useful in constructing a situation model. Thus the nascent formalism actually helps the student to understand the situation in terms of the problem model and so to better comprehend the problem statement (Nathan, 1988). Consequently, we find a mutually supporting relationship, where situational understanding helps students to realize the episodic meaning of a formal problem model and, reciprocally, sensitivity to the problem model aids in the construction of a suitable situation model.

Generating an Equation

Ovals in Figure 1 which have not been given explicit values but are referenced as being critical for solving the problem can be filled in with variables -- placemarkers indicating their importance. Other empty ovals can be filled in by a process of constraint propagation where subsequent ovals, when filled, must be consistent with the expressions already specified. For instance, in Problem (1) we are asked for the travel time of the second plane, t. We can use the supporting relation which states that the second plane started 3 hours later (Prop4) to obtain "t+3" as the time for the first plane (slot T for the first column). The two vertical DRT schemata can supply entries for their respective

Distance (D) ovals. Applying the inferred equality relation (from Prop11) for the two distances yields the equation 200*(t+3) = 250*t.

Algebraic Schemata

Problem models and equations can be constructed in much the same way as in Figure 1 for most algebra word problems found in college and high-school textbooks. With a list of the required algebraic schemata, one can a priori construct all of the necessary problem models. Mayer (1981) has compiled a list of 1097 story problems from standard textbooks, which he classified into eight families. The first four families, comprising the majority of these problems, are various kinds of rate problems, while the others are Number problems (such as Age problems), Geometry (e.g. Area problems), Physics (e.g. Ohm's Law), and Statistics. The general rate schema is of the form

UNIT1 = Rate-of-UNIT1-per-UNIT2 x UNIT 2.

The differences among the four rate families are differences in the nature of the units:

	UNIT1	UNIT2
RATE-FAMILY 1	amount	time
RATE-FAMILY 2	cost	unit
RATE-FAMILY 3	portion	total cost
RATE-FAMILY 4	amount	amount

Number problems, on the other hand, generally are not so clearly tied to a particular schema; instead, relations tend to be specified in the text, except for certain abstract properties of numbers (e.g. that even numbers are always 2 x N). For physics problems, each physical law or formula corresponds to a schema in our analysis. Newton's Second Law, that the sum of all forces for a system in equilibrium must be zero, is one example. In

the Envisionment Machine, Roschelle (1986) graphically depicts a problem situation using the given and resultant force vectors as the formal problem model in order to help students to visualize a situation which is helpful for solving common kinematics problems. Similarly for geometry problems and statistics problems there is generally a schema for each formula and theorem.

In the Appendix, two representative problem types are shown for each family of rate problems. The many subtypes within each category listed by Mayer (1981) differ from the examples analyzed here in the nature of the supporting relations given and/or the unknown value being asked for. These variations are represented similarly. The rate schema in its various forms is, therefore, all we need for the construction of problem models for thousands of word algebra problems. Instead of over 1,000 problem types, we need to be concerned with only a limited set of building blocks for algebraic schemata. Our goal will be to teach students the building blocks which produce the appropriate schemata, and show them how to derive algebraic equations from these graphs.

The schematic formalism that we have developed is supposed to help students better understand the conceptual structure of an algebra word problem, but it does not understand it for the student. Students must still make the hard inferences, still apply their knowledge of the situations described and still ultimately generate algebraic equations. The graphic representation reduces student memory load by constraining the sort of information that is to be considered, providing cues for what might be important, and making it apparent when vital information is absent from the problem model. In the problem model of Figure 1 some important inferences had to be made so that the verbally represented problem could be put into a solvable form: Prop11 says only that one plane overtakes the other. In our situation model we have one slower plane leaving initially and, after some delay, a faster plane approaching the first plane from behind. One can infer from the situation that the distances

travelled by the two planes must be equal at the point of "overtake" and this must be represented mathematically. Prop10 states that we want to know how long it is before the overtake occurs. It is not from the textbase but again from the situation model that a reader realizes that this corresponds to the travel time of the delayed plane. Inferences of this type are crucial for understanding a problem so that the appropriate information is gathered and incorporated into a solvable problem model. We find that failure to include such situation-directed inferences leads to problem solving errors on the part of students.

From this discussion it should be clear that one of the goals of our tutor will be to help students organize and search for missing information by making otherwise covert structures overt in a graphical form. A second purpose will be to provide the student with a way to check what has been done. An animation driven by the student's formal problem representation makes apparent the situational implications of that structure. For example, seeing the fast plane overtake the slower one and continue across the screen indefinitely may suggest to the student that incorporation of additional constraints in the problem model will bring about the intended animation: namely, that the two planes must not pass their ultimate destination. The top horizontal arc of Figure 1 which equates the distances travelled by the two planes represents such a constraint.

The most important benefit that we claim for our tutor is that it may help to avoid the isolation of algebraic knowledge by forcing students to consider formal expressions with reference to a particular situation.

Making Formalisms Situationally Meaningful

Word problem solving is both an exercise in text processing and mathematics skills. Teachers, however, often focus their efforts on developing students' skills in manipulation of formal mathematical expressions, rather than on strategies of problem

comprehension (Mayer, 1985), though there is strong evidence (e.g. Cummins et al., 1988) that the latter is what makes word problems so notoriously difficult.

It seems that students readily learn to manipulate the necessary formal relations. They demonstrate early familiarity with "translation rules" which help map keywords found in word problems into the language of algebraic expressions, e.g. "by" mapping to times; "altogether" as plus (Nesher & Teubal, 1975). But students often do not learn the principles which underlie the algebra; nor do they see the formulae as mathematical models (i.e. descriptions) of the situations described (Greeno, 1989). Thus, students will assign a meaning to the symbols of a formal expression, though it may not be consistent with the situation described. When given situationally impossible problems, for example, some students blindly translate passages into formal expressions and produce answers which, while mathematically correct, are in reality absurd (Paige & Simon, 1966). Those who form situational representations of the symbols and expressions realize either the impossibility of the problem setting, or, upon calculation, the impossibility of the solution (e.g. requiring a negative amount of money in one's pocket). Similar findings are available from research on physics problem solving (Caramazza, McCloskey & Green, 1981; Larkin, 1983), geometry (Schoenfeld, 1987), and programming (Pennington, 1987). Wertheimer (1946) with his famous discussion about teaching students to find the area of a parallelogram, was probably the first to draw attention to this disassociation between formal expressions and the situation described.

Greeno (1989) explains many of these results as instances where abstract *symbol*-space representations and transformations, and real world or *situation*-space representations coexist as disconnected systems. Consequently, students learn to procedurally solve symbolic expressions without learning how to properly reason about related "real world" problems. When these representations exist as disconnected entities, it is possible for

problem solvers to perform operations on symbolic expressions which are no longer faithful to the situations to which the expressions are intended to refer. It is by establishing a correspondence of the symbols to the situation that one roots the formalisms into the space of permissible and expected events (Greeno, 1989). Unfortunately, institutional instruction sometimes ignores this and fosters the acquisition of procedures which are not anchored to real world situations.

The tutor which we describe below was designed to help students see how a formal problem description, such as a set of equations, relates to a concrete situation. We cannot always bring the real world into the classroom -- flying real planes and mixing real liquids, but we do have the technology to provide reasonable animations of these events that can be substituted for the real thing.

Figure 2 sketches this bridging function of our tutor. The lower portion of the graph depicts how, according to our model, algebra word problems are solved without the aid of the tutor. The text of the word problem is comprehended, and both a situational representation and a formal algebraic problem model are constructed. General world knowledge enters into the former process, and more specialized algebraic knowledge determines the latter. The point in our model where decontextualization can occur is indicated in the sketch by the broken line labelled "coordination." The tutor ANIMATE does two things. First, it requires the student to construct an explicit, graphical representation of the conceptual problem model, as implied by the theory, before deriving an equation to actually solve the problem. We shall discuss below the benefits expected from introducing this intermediate step. Second, by having an animation illustrate the actions implied by the student's problem representation, ANIMATE provides a link between the formal domain of algebra and a situation in the "real world." It is not only that the student can use the animation to check whether the problem was conceptualized

correctly or not (the "Is it right?" link). That is important. But equally important is to establish the link between the concepts and structures in the formal model to the events in the animated world. Changes to the conceptual network make something happen in the animation; so the animation provides a context for the formalism and grounds it in a depiction of reality.

Insert Figure 2 about here

There are other tutoring systems with similar goals, notably the Envisionment Machine (Roschelle, 1986) and the TRIP tutor (Gould & Finzer, 1981). Envisionment, like ANIMATE, supports multiple representations. One representation simulates (real) situations, while the other provides a manipulable formal problem model (force diagrams, in this case). A problem solver must coordinate these two representations and resolve conflicts so that they are consistent. Furthermore, the resulting situation must be plausible for the objects behaving in the world as we know it, and the symbolic expressions must be legal and follow all known constraints.

The TRIP tutor (Gould & Finzer, 1981) was designed to help students formulate the appropriate Distance-Rate-Time equation from a written problem statement. TRIP lets students build computer animated descriptions of collision problems and then evaluate their own equations based on the behavior of the animation. Unlike our tutor, TRIP has no intermediate conceptual representation mediating between the verbal formulation of the problem and the equations. Also, no attempts were made to base TRIP on an explicit theory of problem comprehension. It is the teacher in TRIP who provides the criteria for correctness of the constructed picture, while ANIMATE exploits the *student's* ability to assess this. ANIMATE also provides the student with greater flexibility in the construction

of situation and problem models, whereas TRIP contains explicit knowledge of each problem assigned -- the distances, rates, travel times. Students can only enter values which are correct for that problem, and the animation will run only when the problem description is correct. ANIMATE has no notion of a correct problem description, only an internally consistent one. The student can arbitrarily vary a problem to have different values, constraints and unknowns. Indeed, even partial and incorrect problems can be run. This contributes to the exploratory nature of ANIMATE.

The work of Hall et al. (1989) has educational implications similar to our own. Relying on extensive empirical analyses of how college students actually solve such problems, the authors observe that "model based" reasoning plays a crucial role in word algebra problem solving. They conclude that "integrating dual representations of a problem at situational and quantitative levels is a central aspect of competence" (p. 269). To represent the quantitative level of a problem Hall et al. (1989) use a graphical scheme based on the work of Shalin and Bee (1985) and Greeno et al. (1986) which has many similarities with the schematic representation proposed above. Their situational representation, however, is not an animation but is based on a more abstract description of situational relations. It is a "learning system" in the sense of Nesher (1989), intermediate to the formal problem model and to the situation itself. In contrast, we use animations as a substitute for a directly experienced situation.

A Strategy for Tutor Design: Discourse Comprehension Theory and Instructional Principles

Our tutor rests on a double foundation: a theory of how students understand and solve word problems, and a set of instructional principles.

The Theory

ANIMATE is directly based on the theory of word problem understanding originally proposed in Kintsch & Greeno (1985) and elaborated by Reusser (1988),

Cummins et al. (1988), and Kintsch (1988). While previous work dealt with arithmetic problems, the extension of this theory to algebra, which we have described above, is quite straightforward. In place of the "set" schema and its variants, we introduce a larger set of algebraic schemata, primarily the various rate schemata described above (see Appendix). We need to replace the counting strategies employed in arithmetic problems with more powerful calculational strategies, viz. the constraint propagation procedures for generating equations from the problem model, and the algebraic procedures for solving these equations.

Kintsch and Greeno's (1985) theory of arithmetic word problem solving has been formalized as a computer simulation and tested empirically, so we have some assurance that it captures the essential features of how first-grade students solve simple arithmetic word problems. It does not at present seem possible to formalize and test empirically our model for algebra word problem comprehension in the same way. In order to construct a simulation for word algebra one would need a knowledge base that included an enormous amount of general world knowledge, in addition to knowledge about algebra. It is not clear at present how to construct such a large and general knowledge base, nor how to operate with it if we had one (but see Lenat & Guha, 1989, for a presentation of a system with this goal in mind). Hence a simulation is impractical at this time and we have no ready means to derive and test empirical predictions from the model.

Instead, we can consider our tutor as a test of the theory following Anderson, Boyle, & Yost (1985) in this respect. Obviously this is a risky strategy. If the tutor fails to improve student problem solving performance and comprehension, we don't know if the underlying theory was wrong, or if we made poor system design decisions that are unrelated to the theory. Similarly, if subjects using the tutor demonstrate elevated comprehension and solution performance, we have only an indirect test of the theory.

In building a tutor, we need to address both the question of *what* to tutor and *how* to go about it. The answer to the first question can be obtained by taking our model seriously: We must make the relation between certain mental representations explicit so that students understand that symbolic expressions have a situation-based interpretation. In principle this model can be worked out to the same level of detail as was done for arithmetic word problems, making explicit all the steps involved, as in the examples above. Many of these steps, of course, people do not need help with -- we presume our students can read, for instance. Taking our earlier discussion and the results of Cummins et al. (1988) as a cue, we focus here upon the task of helping students to learn the translation from the problem text into a formal, conceptual structure[2].

The answer to our second question -- how to tutor -- cannot be derived solely from the process model of how students solve word problems. A host of questions arise concerning various aspects of tutoring and user interface principles that are outside the domain of our process model. Recently much has been done to apply the laboratory findings of cognitive psychology to the field of instruction. We borrow from this work to help us determine the principles which will guide our computer-based tutor.

Instructional Principles

Anderson, Boyle, Farrell, and Reiser (1984) present several principles derived from experimental research in cognitive psychology which they argue are central to tutoring. There is a strong case and widespread agreement, for instance, on the importance of minimizing students' working memory loads during problem solving (cf. Polson & Richardson, 1988; Scardamalia et al., 1989). Also widely agreed upon is that making students' goals and processes overt, instructing them in the context of the specific task, and providing support for successive approximations toward a solution, all help the student in task performance and skill acquisition (Glaser & Bassok, 1989).

A more controversial principle states that feedback from a tutor needs to be immediate if students are to optimally improve (e.g. Anderson et al., 1984; Reiser et al., 1989). This is based on the view that "...an error comes close to being a necessary and sufficient condition for tutorial intervention" (Anderson, 1989; p. 343) and on experimental findings which have shown that when subjects get "lost" they must use tremendous cognitive resources to get back to their original goals (Anderson, 1982). Such episodes, it is argued, do little to help students learn and can confuse the memory traces of correctly learned behavior. However, recent studies on computer-based feedback lessen the strength of these claims. Schmalhofer, Kuehn, Messamer, and Charron (1989) have shown that a "selective" tutor -- one which provides feedback only after two consecutive errors have occurred -- had certain advantages in introductory LISP learning and problem solving over an "immediate" tutor -- the type advocated by Anderson and others. Similarly, Lee (1989) showed that delayed feedback can lead to better overall problem solving performance than immediate feedback, especially on hard and novel problems. Immediate feedback in these domains seems to foster "blind learning" where students can use trial-and-error to complete a problem, but have no notion as to the reasoning process underlying a solution. Since the reasoning remains hidden, it is difficult for the student to learn it.

The term "error" may have different meanings in these two bodies of research. Anbderson (1982; 1989) is primarily concerned with skill acquisition and the compilation of the steps of a task into a single, efficient procedure. It is through this that he hopes to bring about error-free performance. In reasonably constrained domains, such as algebraic manipulation and introductory programming and geometry, this seems to be a fruitful approach. In open-ended domains, such as life sciences, however, one is more concerned with developing critical thinking abilities and hypothesis generation and testing. In open domains errors are valuable, often serving as learning opportunities. In such arenas, we

may wish to reward students for trying out possibilities, even though many may initially prove to be erroneous, since it may force students to restructure their knowledge and alter their reasoning.

Instructional principles recently laid out by Scardamalia et al. (1989) focus on the need for tutoring systems to encourage students' active participation in processes such as planning, self-assessment and monitoring, problem relevant inferencing, goal-setting, knowledge organization and problem solving. It is the *student*, not the tutor, that needs to perform diagnosis, goal-setting, and planning, to obtain the maximum opportunity for learning. Systems must provide the facilitating structures and tools which enable students to use their intelligence and knowledge, rather than providing knowledge and intelligence to simply guide learning. However, Scardamalia et al. (1989) note also that giving students total autonomy can lead to poor results 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. This view seems to be supported by experimental evidence with environments such as LOGO -- the prototypical learning-by-exploration system (Papert, 1980): Work with elementary school-aged students using LOGO has not upheld the expectations for improvements on reading and math tests (e.g. Clements, 1986; Battista & Clements, 1986; though see Fischer, Boecker, & Eden, in press, for a more thorough discussion). Thus, neither an all-knowing system that takes control and initiative away from the student, nor unguided exploration which offers the learner no support at all seem optimal for instruction.

Why Un-intelligent Tutoring May Work

We make two points here: that the time for intelligent tutoring systems has not yet come for many instructionally significant domains; and that unintelligent tutors (systems that do not try to understand the student's actions or the problems) are sufficient to enhance

students' learning, provided they are capable of supporting students in an active learning process.

For a program to adaptively customize its behavior, the tutoring system must understand what the student is doing. We need a psychological process model of the behavior in question. This is often not available, or too sketchy to be of much use. Intelligent tutoring without the necessary intelligence is a risky business. If the system misclassifies a student's response or behaves inconsistently, it can easily confuse or discourage the student (Holland, Holyoak, Nisbett, & Thagard, 1986). For most tasks, including mathematics, there is no single, unique way to solve a problem. It is insufficient to merely evaluate a student's answer; instead it is the method which leads to the correct answer that intelligent tutors need to impart. A program that misclassifies minor errors (or typos) as major conceptual errors, or lets an erroneous method go uncorrected because it led to a correct answer, may do more harm than good. In the absence of a principled method for task analyses, and of psychological process models which account for varied learning styles and which allow students to perform subtasks in many different orders, intelligent tutoring for many domains doesn't look very promising in the near future. Intelligent tutoring is possible today in closed domains, such as geometric proofs (Anderson, Boyle, & Yost, 1985), but not in open domains such as word problems which require a huge store of general world knowledge.

Robust, flexible tutoring systems which are expected to exhibit intelligence may best be regarded as a long term goal. Our goal is the more immediate development of a computer system which aids students in problem understanding and learning. We propose an approach which assigns to the computer program tasks that it is good at doing *today*. Among these are executing simple, time-varying graphics, bookkeeping, and processing formally described relations (e.g. equations). Tasks which lie outside of today's

technology -- natural language processing, inductive reasoning -- are left for the student. Other systems, such as EUCLID (Smolensky & Fox, 1987), a computer system for specifying argumentation, and CSILE (Scardamalia et al., 1989), share this philosophy. In our system the computer program serves as an informative worksheet for mathematics and animation. The student reads a problem, constructs a formal problem model, and obtains feedback from the animation as to its correctness. The intelligence is all the student's. The system is there merely to help organize problem information around selected schemata and allow the student to see the situational correlate of specified formal relations.

But quite apart from questions of feasibility, the idea of intelligent tutoring in many cases represents intellectual overkill. The good tutor, human or machine, does not spoonfeed and prechew, but makes it possible for students to do the work themselves and learn from their mistakes (Scardamalia et al., 1989). A body of instructional theory has evolved that emphasizes "procedural facilitation:" Learning must be done by the students themselves, and the function of instruction is to facilitate that learning (Bereiter, & Scardamalia, 1989; Brown, & Palincsar, 1989; Collins, Brown, & Newman, 1989). We believe that a machine can support this, without actually having to understand in any deep way either the problem or the student.

ANIMATE

ANIMATE is an interactive tutor that facilitates comprehension of word problems by helping the student construct a formal problem network, which is then used to run a simple animation of the problem. It is written in HyperCard™ and runs on Apple™ Macintosh computers. It requires a minimum of a Macintosh Plus™ with two drives. Currently, ANIMATE only tutors problems in the AMOUNT-PER-TIME RATE family of word problems (Mayer, 1981). These are problems which employ the "distance = rate x time" equation and includes "overtake," "collision" and "distance apart" problems.

The tutor was designed so that students can use it with little or no instruction. All available commands and network values are displayed on the screen as buttons. Students choose commands and change values by selecting the appropriate button with a free rolling mouse-cursor and pushing the mouse button. Consequently, this situation driven interface reduces working memory load by only allowing legal commands. While students are guided in the correct use of the system, ANIMATE gives them a great deal of control over how to decompose each problem and the order in which to achieve subgoals. A student may run an animation on partial and incorrect networks, or interrupt the network building process to enter new animation parameters. He or she may also construct new problems or make problem variants to explore algebra and develop more experience about mathematics.

An effective way to explore the functionality of the system is with an example of how a student would use ANIMATE to solve a specific algebra problem. We assume that the student has been given the instruction to produce the equations necessary to solve the following:

Problem 2: Collision Word Problem

A huge ant is terrorizing San Francisco. It travels east toward Detroit, which is twenty four hundred miles away, at four hundred miles per hour. The Army learns of this one hour later and sends a helicopter west from Detroit at six hundred miles per hour to intercept the ant. If the ant left at 2 PM, what time will the helicopter and the ant collide (ignoring any time changes).

The student decides on an initial goal of describing the movement of the ant. He or she pushes the button "Pick Character 1" and is presented with pictures representing the possible characters (Figure 3). After choosing the ant, buttons appear at the bottom of the

screen which allow the student to select a pointing hand for the ant's starting location and travel direction. The student chooses the leftmost, east facing button and the ant appears in the corresponding position on the screen.

Insert Figure 3 about here

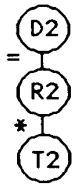
To specify the motion of the ant, the student must build and customize a problem model. Pushing the "Equations" button causes a palette of suggested network equations to appear (see Figure 4). The equation palette provides the student with small schema building blocks to organize problem information rather than requiring the student to work only at the level of entire schemata (for justifications of this approach, see Schank, 1984; Kintsch, 1988). The student chooses

and the first **D=R*T** equation in the network appears. This equation may be made the *stopping condition* for the animation, if the student chooses. The animation will then stop when it is satisfied by the position and travel time of the ant. The selection of a stopping condition encourages the student to link the problem model and the situation-based goal since real events need a specified end to be properly described. The student then selects the rate node (R1) of the network and, in accordance with the information of Problem 2, uses the calculator to enter the number 400 (see Figure 5).

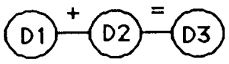
Insert Figures 4 and 5 about here

To check his progress, the student pushes the "Run" button which first checks the problem schema for algebraic and network errors. The student could check for problem model errors without also running an animation by selecting the "Check Net" button; but running an animation always initiates a preliminary network check. The tutor finds no errors and so begins the animation by moving the ant across the screen at a correlate of 400 miles per hour. The animation, along with the clock that shows the elapsed (animation) time, the calibrated ruler at the top of the screen, and the distance gauges, all provide the student with valuable situation-based feedback for assessing the correctness of the problem model. Because distance and time variables are unspecified, the animation will continue until the student pushes the "Stop" button. After determining that the initial goal of describing the ant has been met, the student develops the remainder of the problem model.

To build the rest of the network the student chooses from the palette



and an animation stopping condition of

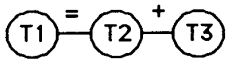


The final equation appears in the network with a box around it, indicating the relation as the situation's stopping condition. The student enters the remaining values given in the problem, putting 2400 in the D3 node to show the characters' initial distance apart, and 600 as the helicopter's rate (R2). From the text the student knows the helicopter leaves 1 hour after the ant and so mistakenly believes that this delay (T3) should be *subtracted* from the helicopter's time. The student knows also from the text that the ant leaves at 2 PM and so erroneously enters 2 for T1. Since the helicopter leaves one hour later, 3 is entered for the T2 node. He or she then pushes the "Check Net" button and the tutor finds no math errors or obvious net errors, such as an incomplete equation.

The student then enters the distance values by multiplying the rates and times in the network (400*2, 600*3). When "Check Net" is pushed now, the tutor warns that **800 + 1800 = 2400** is not correct, and highlights the flawed equation (Figure 6). This feedback indicates that there must be an error in the network. If the distance values were changed so that they correctly add to 2400 then the vertical D=R*T equations would be incorrect. The student decides to ignore the error and, with no further errors found, pushes the "Run" button.

Insert Figure 6 about here

The student immediately notices that the helicopter, not the ant, starts moving first (Figure 7) which is contrary to any situation-based expectations. The mismatch suggests that the delay for the characters is improperly specified in the network. The Time equation in the bottom row is suspect. He or she stops the animation and picks the equation



The equation now means situationally that the delay is attributed to the helicopter (i.e. the ant travels for more total time). The Time equation in the network changes from **2=3-1** to **2=3±1**, which is obviously wrong arithmetically. The student realizes now that he or she has entered "starting times" for the characters whereas the network needs "travel times" to produce an appropriate animation. Since these times are unknown, the student enters variables in the associated distance and time bubbles of the network.

When the animation is run this time, the ant leaves an hour before the helicopter. As the ant and helicopter near the point of collision (Figure 8) the student decides that the animation matches his or her mental image of the word problem. The following verified equations are taken from the net.

$$D1 = 400 \times T1$$

$$D2 = 600 \times T2$$

$$D1 + D2 = 2400$$

$$T1 = T2 + 1$$

Insert Figures 7 and 8 about here

The above example illustrates how a problem solution is reached through a series of successive approximations. ANIMATE supports this by helping the student to generate and

then diagnose a formal problem network. 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. By making this structure overt, and explicitly tying it to a situation, the student gains a more concrete understanding of the conceptual relations of a problem.

ANIMATE knows nothing of the problem being solved and so cannot tell the student if the network being built is correct. We rely upon the student's understanding of the situation described in the problem statement to set his or her expectations for how the animation should appear. In Figure 2, this is represented in our model by the "Is It Right?" link from the student's situation model to the computer animation. This question is addressed *by the student*; it is he or she who decides whether the animation matches the problem representation, and so answers the question, "Is It Right?" A mismatch suggests how the student may alter the problem model to create the expected animation. In this way, the student also addresses the "What Went Wrong?" link. Thus, we can restate our goal in designing the tutor as providing situation-based feedback so that students can detect and correct their formal problem models -- that is, successfully address the "Is It Right?" and "What Went Wrong?" links.

In order to encourage active participation by the student in the construction of the network and evaluation of the animation, the tutor provides *visual*, *auditory*, *problem model level*, and *situation level* feedback. *Visual* feedback is demonstrated not only by the animation but through the "button" based interface. When a student selects a button or network node, the button briefly highlights to indicate it has been selected. For example, network nodes are highlighted as the subject enters new values. *Auditory* feedback includes short musical pieces indicating when the animation has begun, is interrupted or is completed. *Problem model* feedback is provided by an equation palette as a network

building tool and the arithmetic and network level checks. The animation specified by the network is the source of the tutor's *situation level* feedback. It includes the characters which move in specified directions, the clock, ruler and other gauges.

Preliminary System Evaluation

Is ANIMATE a good tutor -- better than traditional classroom instruction, perhaps even approaching an experienced human tutor? Even more importantly, does ANIMATE truly help the student to bridge the gap between real world situations and formal operations? Is it a way to avoid decontextualized learning, as we had hoped for? We do not have definitive answers to these questions. In particular, we do not have a large-scale, long-term classroom evaluation of the effectiveness of ANIMATE, nor do we yet have a system complete enough and sufficiently robust that it could be used for such an evaluation. Nevertheless, the question of the empirical adequacy of the tutor is important enough, so that even the preliminary results that we have so far obtained are of interest.

In a study by Nathan (1988) subjects were given limited training in the Network approach to solving distance-rate-time problems. After only a 20-minute training session, these subjects produced significantly more problem-related inferences in their recall protocols than did subjects who used the traditional equation method or who simply read the problem for comprehension. These subjects did not, however, show significantly better problem solving performance. This suggests that the network method facilitates problem organization in terms of the situation model but it does not support solution generation to a greater degree than equations do. Another study investigated the psychological reality of the problem schema level, similar to the network described above, as an intermediate representation between the cover story and the underlying equations. Weaver & Kintsch (1988) showed that subjects rated problem pairs significantly more similar when similarity was based on structure of the schemata than when based on the form of the underlying

equations. Given the wide-spread finding that subject are very poor at detecting structural similarities among such problems (e.g. Reed, 1987), Weaver and Kintsch (1988) argue this may be in part due to the way we define "structure." Subjects are indeed sensitive to the conceptual structure of a problem though not to its equation form. In the next section we report empirical results from a study that explores two components of our model: the problem model as a distinct level of representation and its effect on problem solving performance; and the impact that animation will have on problem solving performance and near and far transfer tasks when that animation serves to anchor the necessary mathematical formalisms to a concrete situation model.

The Role of the Animation in ANIMATE

Like the two studies mentioned above (Nathan, 1988; Weaver & Kintsch, 1988), the experiment which follows is limited because of the extremely brief training period, and because problems of only one type (distance-rate-time problems) were used. The experiment was designed to explore the overall effectiveness of the animation component of the tutor in facilitating problem comprehension and giving meaning to the abstract symbols used for formal problem solving. For that purpose, four training groups were compared: an Equation group, which served as the control group, reviewed traditional procedures for solving algebra word problems; an Animation group which was trained in the use of the ANIMATE tutor; and two groups which used non-animation based tutors. The Stopping-condition group used a tutor which was identical to the ANIMATE tutor in that it included all the graphical problem representation machinery, character selection, and stopping condition selection, but did not provide a way to run the animation; and the Network group which was identical to Stopping-condition group except that it did not ask users to select a stopping condition when equations were chosen from the palette. The three "tutor" conditions -- Animation, Stopping-condition and Network groups -- were designed to tease

apart any effects due to the different aspects of setting up and running an animation. The four groups of subjects solved word problems in quite different ways, and hence they may be expected to react differently to the various types of problems. We tested what our subjects had learned with a range of transfer problems, measuring near and distant transfer and involving situational elements to a greater or lesser extent.

Method

Subjects. Fifty-six undergraduates from the University of Colorado participated in a two hour experiment as part of their course requirements for Introduction to Psychology.

Design and materials. Subjects were randomly assigned to either the Equation, Stopping-condition, Network or Animation treatment and run in groups of 14 students at a time to mimic a classroom setting. All groups received identical pretests, analogous task booklets which constituted the active part of their training sessions, and identical post-tests which contained two near transfer tasks, one distant transfer task and one which may be considered an intermediate transfer task. Those in the Network, Stopping-condition and Animation groups additionally used tutoring programs which ran on Apple™ Macintosh computers, distributed one to each subject.

The paper-and-pencil pretest consisted of four problems. The first two problems were standard distance-rate-time problems. Students were told to write down a set of equations that could be used to solve each problem. They did not need to do the algebra to actually solve the problems. Problem 3 was a debugging problem: four potentially flawed equations involving travel times and distances were given after the verbal statement of a problem. Subjects were to find and correct any errors. The final problem consisted of a set of time and distance equations. Subjects were asked to write a short story which could be a legitimate word problem for the given equations.

Four different problems of the same type and difficulty as given in the pretest were used in the post-test to assess the transfer of skills presented during the training session. Students were not permitted to use the tutors during this exam, although they were encouraged to apply any new methods or principles that they thought were appropriate. The standard equation generation problems are taken to be near transfer tasks. The debugging problem may be regarded as intermediate as it contained a word problem statement and equations in a familiar form, although the expressions were not suitable for the given situation. Identifying and correcting formal expressions is likely a task that subjects performed in producing a final solution, although they may not have had the task expressed in exactly this way before. The story-writing problem is considered to be a distant transfer task since it was not reviewed during training, and had probably not been encountered before this experiment. The post-test consequently provides three levels of transfer upon which we can evaluate each treatment's effect on performance.

Three problems arranged as a task booklet were used during training. The first task was a standard multiple choice word problem with three possible solutions from which students had to select the correct set of equations. For the second training task students had to generate a complete set of equations which would lead to a numerical solution of the given word problem. In the third task a "buggy" set of equations was given as the possible solution to a word problem. Students were asked to find and correct any errors.

Training methods. Subjects all received a training session which reviewed the basic principles of the Distance = Rate X Time equation, its physical and mathematical interpretation, and how to apply it to several representative examples. Whereas subjects in the Equation group used the strictly algebraic approach of deducing equations directly from the problem texts, the three tutor groups learned to use the graphic-based network approach presented earlier an intermediate step in the translation (see Figure 1). Equations were then

derived from the network. The network approach was taught first by demonstration and then by practice on the respective computer tutor (either ANIMATE, Stopping-condition or the Network-only tutor).

Procedure. All groups were pre-tested on their knowledge of algebra. Subjects then had either a brief review of traditional algebra (Equation group) or a 30-min. tutorial on the network method. Subjects then solved 3 problems in a task booklet using the method introduced in their training sessions. Those working with a computer tutor used either the complete ANIMATE computer program, or slightly modified versions which provided the Stopping-condition program -- which is identical in every respect to the ANIMATE tutor except that it gives no situation-based feedback (i.e. no animation) -- or the Network-only version. Finally, all subjects took an identical post-test administered without use of the computer tutors. An independent experiment revealed that the pretest and the post-test were of comparable difficulty, $F(1,31) < 1$.

Scoring. Stories had to be completely correct to receive credit, and, likewise, multiple choice problems received either a score of 1 or 0. Partial credit was assigned for all other problems involving the writing or correcting of formal expressions (equations or networks). Analyses of student performances scored with no partial credit yielded similar results to those scored with partial credit, so all subsequent results reported are for scores with partial credit assigned for all but the story-writing and multiple choice problems.

Results and Conclusions

We first consider the results of the pretests and post-tests. Since these two tests were in every respect identical for the four groups of subjects, it provides the fairest measure for the overall effectiveness of the treatment. Table 1a shows the results of these tests. All groups improved from pretest to post-test, $F(1,55) = 48.65$, $p<.0001$. But ANIMATE tutor users improved far more than the Equation group, $F(1,55) = 13.6$,

$p<.0005$, which showed an 11.25% improvement. Furthermore, the Animation group improved more than the non-animation tutors, by 52.75%, versus a 23.6% average for the other tutor groups, $F(1,55) = 8.8$, $p<.005$. The non-animation tutor users did not improve appreciably more than equation users, $F(1,55)=1.64$, n.s., nor did inclusion of the stopping condition lead to greater improvements over the Network-only condition, $F(1,55)<1.0$. These results are complicated by the fact that the Equation group had a significantly higher pretest score than the three tutor groups, $F(1,55) = 4.3$, $p<.05$, which did not differ from each other. Although this difference can only be attributed to random effects in the selection of subjects, it does affect the interpretation of these results somewhat. Nevertheless, even 30-min. of training with ANIMATE leads to more than four times the improvement of standard procedures for solving problems in the word algebra tests we gave -- the effect of ANIMATE is rather impressive.

One could easily suppose that after more than a thousand hours of practice with standard algebraic methods, a brief session with this new approach might only serve to confuse the students. Instead, the exposure to ANIMATE proved quite helpful on the post-test where problems were solved without the help of the tutor. This is true in particular for the near and distant transfer tasks. The distant transfer task entailed writing a story to correspond to a given set of equations. ANIMATE helped subjects perform this novel task quite successfully, leading to significantly greater improvement than the other three conditions, $F(1,55)=4.8$, $p<.05$. Furthermore, the animation component of ANIMATE really appeared to matter. The Network and Stopping-condition groups, which used the same mathematical approach as the Animation group, but which were not exposed to situation-based feedback, showed improvements in line with the equation group.

There were no measurable differences in problem solving performance attributable to treatment for the intermediate transfer task. This was the debugging task which required

that students correct a flawed set of equations intended to describe (i.e. solve) a given story. Our hypothesis was that the Animation group would receive advantageous feedback regarding the situational inappropriateness of the given equations. The link that they would have formed between the animation and components of the formal problem model was thought to constrain the search involved in error detection and correction. Although improvement was greatest for all groups on the debugging problem, the ceiling on performance was not nearly achieved. The overall correct performance on the post-test for this problem was about 60%, with only 64.3% of the highest scoring groups (Network and Animation) solving the it correctly (see Table 1b).

Two near transfer tasks were given in the post-test. These were both conventionally structured problems where students read a story and then wrote down either the network or the set of equations appropriate to the solution. Students practiced this both during the training presentation and in their training booklets. Since no reliable differences were found between the two problems, all subsequent analyses are based on their combined scores. Students in the Animation group showed highly significant improvement on these problems, greater than all other groups, $F(1,55)=12.7$, $p<.001$. The large effect is attributable both to the difference between animation and non-animation students, $F(1,55)=15.45$, $p<.0005$, and to the relatively small improvement (4%) by the Equation group (see Table 2). Compared to this, both non-animation tutors exhibited significantly greater improvement, $F(1,55)=4.15$, $p<.05$. Recall that the Equation group showed a significantly higher pretest score than all other groups. This initial bias is in part due to their high pretest performance on the same type of problem, which exceeded all other groups, especially the Animation group, $F(1,55)= 8.78$, $p<.05$.

All analyses were redone taking into account whether or not students in the Equation group used pictures to solve or set up the various problems. This factor did not

reveal any new effects, a finding that is consistent with the earlier work by Mayer (1982). As with Mayer's study, picture drawing received no feedback on correctness or appropriateness. Nor was there a compelling reason, as with the animation-based tutor, to make sure that the picture jived with the formal expressions. ANIMATE, in contrast, provides the opportunity for students to "scaffold" from animated pictures to an appropriate mental representation which has a tie to the formal equations.

We find that while *improvements* differed significantly, there are no reliable group differences in students' post-test performance on the near transfer tasks. Thus, while all tutors, and the ANIMATE tutor in particular, helped students to improve on conventional equation generation tasks, they do not appear to bring students' level of performance above that of the initially more-competent students in the Equation group.

Insert Tables 1a and 1b about here

For problems solved during training, the results are in agreement with the pre- and post-test scores. Table 2 shows that overall the Animation group strongly outperformed all of the other groups, $F(1,55)=14.1$, $p<.0005$. The Equation group and non-animation tutor groups performed at a comparable level, with no significant differences apparent. Animation students, more specifically, achieved much better problem solving performance than either those using the modified tutors, $F(1,55)=10.1$, $p<.0025$, or those in the Equation treatment, $F(1,55)=13.7$, $p<.0005$; this difference was not due to any one group's inferior performance.

Table 2 also shows the results for the three individual training tasks. The Animation group was superior to the other groups in solving the multiple choice problem, $F(1,55)=12.3$, $p<.001$, while those groups showed similar performance, $F(1,55) < 1.0$. When subjects had to generate the appropriate equations themselves, the tutor groups all surpassed the (Equation) control group, $F(1,55)=11.72$, $p<.0001$. While the Animation group clearly outperformed the Equation group, $F(1.55)=10.3$, $p<.0025$, there were no reliable differences among the tutors, although the difference between the non-animation tutor users and the ANIMATE users approached significance.

The effect of animation on problem solving seems clearly positive up to this point. Although debugging skills (found to be the most difficult task regardless of treatment, $F(1,55)=15.7$, $p<.0002$) were not significantly better for Animation subjects, these data are in line with the findings for all of the other tasks. All that may be lacking to reliably show this difference is greater statistical power.

Insert Table 2 about here

For every task, subjects who worked with the complete ANIMATE tutor outperformed those who did not have access to the animation. Even the debugging task, while not showing demonstrable improvement for animation students above and beyond the improvements gained by other students, shows performance which parallels the other types of problems. As shown in Table 3, the Equation group had similar performance improvements. These improvements, we speculate, are based on familiarity of the problem. Students all saw the debugging problem three times, during training, pretest and post-test. When solving this problem, students may draw upon general strategies of formula recognition rather than true mathematical understanding. The 53.6% improvement by

animation students as compared to a 35.7% improvement by the control group is still noteworthy, however, and may indicate a deeper understanding of the problem. Animation subjects, for example, were found to be more likely than Equation subjects to appropriately correct a buggy equation if they detected it, $F(1,55)=5.0$, $p<.05$. Students outside of the Animation condition fail to show a similar performance improvement on the Story problem, for which there was no training practice (Table 3). For this task, it is less clear what general strategies students may have access to, especially given the extreme novelty of the problem. With regard to the Solve problem, the room for improvement demonstrated by Animation subjects appears to have been used up in the high pretest performance shown by Equation subjects.

Solving algebra word problems is not just a question of getting the formalism right, it seems to depend on the student's understanding of how this formalism is rooted in a real-world situation. What is encouraging about these results, above all, are the indications that ANIMATE may indeed help overcome students' tendency in traditional mathematics instruction to learn decontextualized formal procedures, achieving the conceptual understanding necessary to reason mathematically. The little amount of training our subjects received with ANIMATE helped them excel in the problem solving tasks, indicating that it is an effective aid in solving distance-rate-time word problems. The improvement that Animation subjects exhibited on the transfer test, where they did not have access to the tutoring system, was also remarkable. ANIMATE, while particularly useful on the most common type of problem, proved highly effective in boosting performance on the unusual and unfamiliar story-writing problem.

The experimental results seem to support the notion that by providing a weakly structured environment which makes an explicit correspondence between the symbols of an algebraic equation and a simple depiction of the situation described, students improve their

ability to recognize and generate solutions to distance-rate-time problems. This advantage appears to persist even after the training wheels are removed. Subjects exposed to the animation-based treatment performed consistently better than those using non-animation based tutors and those in the control group, on both near and distant transfer tasks.

Although many questions remain, we have obtained some understanding of how students solve word problems with and without the tutors we provided. Additional exploration is still needed to determine what facets of problem solving support help students most. It is noteworthy that using ANIMATE achieved more for our subjects than providing them with "just another trick" (i.e. the network) to solve word problems. Furthermore, it was not enough to have students merely consider what a situation might demand (as was done in the Stopping-condition group). Many Equation group subjects drew pictures; their performance was still consistently below that of the Animation group. The results are encouraging and from them greater development and testing of ANIMATE in an educational setting seems in order.

References

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

Anderson, J. R. (1988). The expert module. In M.C. Polson and J.J. Richardson (Eds.) *Foundations of Intelligent Tutoring Systems.* Hillsdale, NJ: Erlbaum.

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

Anderson, J. R., Boyle, C. F., Corbett, A. T., & Lewis, M. W. (in press). Cognitive modelling and intelligent tutoring. *Artificial Intelligence.*

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

Anderson, J. R., Boyle, C. F., & Yost, G. (1985). The geometry tutor. In A. Joshi (Ed.), Proceedings of the Ninth International Joint Conference on Artificial Intelligence (pp. 1-7). Los Altos, CA: Morgan Kaufmann.

Battista, M. T., & Clements, D. H. (1986) The effects of Logo and CAI problem-solving abilities on mathematics achievement. *Computers and Human Behavior, 2,* 183-193.

Bereiter, C. & Scardamalia, M. (1989). Intentional learning as a goal of instruction. In L. B. Resnick (Ed.) *Knowing, learning and instruction.* Hillsdale, NJ: Erlbaum.

Brown, A. L., & Palincsar, A. S. (1989). Guided, cooperative learning and individual knowledge acquisition. In L. B. Resnick (Ed.) *Knowing, learning and instruction.* Hillsdale, NJ: Erlbaum.

Brown, J. S. (1985). Process versus product: A perspective on tools for communal and informal electronic learning. *Journal of Educational Computing Research, 1,* 179-201.

Caramazza, A., McCloskey, M., & Green, B. (1981) . Naive beliefs in "sophisticated" subjects: Misconceptions about trajectories of objects. *Cognition, 9*, 117-123.

Carbonell, J. G. (1983). Learning by analogy: Formulating and generalizing plans from past experience. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach.* Palo Alto, CA: Tioga.

Carpenter, T. P., Corbitt, M. K., Kepner, H. S., Lindquist, M. M., & Reys, R. E. (1980). Solving verbal problems: Results and implications for national assessment. *Arithmetic teacher, 28*, 8-12.

Clements, D. H. (1986). Effects of Logo and CAI environments on cognition and creativity. *Journal of Educational Psychology, 78*. 309-318.

Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics. In L. B. Resnick (Ed.) *Knowing, learning and instruction.* Hillsdale, NJ: Erlbaum.

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

Fischer, G., Boecker, H.-D., & Eden, H. (in press). Interactive problem solving with LOGO. Hillsdale, NJ: Erlbaum.

Glaser R., & Bassok, (1989). Learning theory and the study of instruction. LRDC Technical Report.

Gould, L., & Finzer, W. (1981). A study of TRIP: A system for animating Time-Rate-Distance problems. *Proceedings of the 3rd World Conference on Computers in Education.* (Lausanne).

Greeno, J. G. (1989). Situation models, mental models, and generative knowledge. In D. Klahr and K. Kotovsky (Eds.), *Complex information processing: The impact of Herbert A. Simon.* Hillsdale, NJ: Earlbaum.

Greeno, J. G., Brown, J. S., Foss, C., Shalin, V., Bee, N. V., Lewis, M. W., & Vitolo, T. M. (1986). Cognitive principles of problem solving and instruction. (Technical Report No. 41, Berkeley Cognitive Science Report Series). Berkeley: University of California.

Hall, R., Kibler, D., Wenger, E., & Truxaw, C. (1989). Exploring the episodic structure of algebra story problem solving. Cognition and Instruction, 6. 223-283.

Kintsch, W., & Greeno, J. G. (1985). Understanding and solving word arithmetic problems. *Psychological Review. 92*. 109-129.

Larkin, J. H. (1983). Expert and novice differences in solving physics word problems. In D. Gentner, D., & A. L. Stevens (Eds.) *Mental models*. Hillsdale, NJ: Erlbaum.

Lee, A. (1989). Timing of feedback in tutoring systems. Institute of Cognitive Science Technical Report No. 89-10, University of Colorado.

Lenat, D. B., & Guha, R. V. (1989). *Building large knowledge-based systems: Representation and inference in the Cyc project*. Reading, MA: Addison-Wesley.

Lewis, M., & Anderson, J. R. (1987). In Greg Kearsley (Ed.), *Artificial intelligence and instruction: Applications and methods*. Reading, MA: Addison-Wesley.

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

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

Mayer, R. E. (1981). Frequency norms and structural analysis of algebra story problems into families, categories, and templates. *International science. 10*. 135-175.

Mayer, R. E. (1982). Memory for algebra story problems. *Journal of Educational Psychology, 74*, 199-216.

Mayer, R. E. (1985). Mathematical ability. In R. J. Sternberg (Ed.), *Human abilities: An information processing approach* (pp. 127-150). New York: Freeman.

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

Nathan, M. J., Johl, P., Kintsch, W., & Lewis, C. (1989). An unintelligent tutoring system for solving word algebra problems. In D. Bierman, J. Breuker, and J. Sandberg, (Eds.) *Proceedings of the 4th International Conference on AI and Education.* Amsterdam: IOS. pp. 169-176.

Nathan, M. J., & Young, E. (submitted). Thinking situationally: Results with an unintelligent tutor for word algebra problems.

Nesher, P. (1989). Microworlds in mathematical education: A pedagogical realism. In L. B. Resnick (Ed.), Knowing, learning, and instruction: Essays in honor of Robert Glaser. Hillsdale, NJ: Erlbaum.

Nesher, P., & Teubal, E. (1975). Verbal cues as an interfering factor in verbal problem solving. *Educational Studies in Mathematics, 6*, 41-51.

Newell, A., & Simon, H. A. (1972). *Human problem solving*. Englewood Cliffs, NJ: Prentice-Hall.

Paige, J. M., & Simon, H. A. (1966). Cognitive processes in solving algebra word problems. In B. Kleinmuntz (Ed.), *Problem solving*. New York: Wiley.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books.

Pennington, N. (1987). Comprehension strategies in programming. In G. Olson, S. Sheppard, & E. Soloway (Eds.), Empirical studies of programmers: Second workshop. Norwood, NJ: Ablex.

Polson, M. C., & Richardson, J. J. (1988). *Foundations of Intelligent tutoring systems.* Hillsdale, NJ: Erlbaum.

Reed, S. K. (1987). A structure-mapping model for word problems. *Journal of Experimental Psychology: Learning, Memory and Cognition, 13*, 124-139.

Reiser, B., Kimberg, D. Y., Lovett, M. C., & Ranney, M. (1989). Kowledge representation and explanation in GIL, an intelligent tutor for programming Princeton University Cognitive Science Laboratory Technical Report No. 37.

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

Roschelle, J. (1987). Envisionment, mental models, and physics cognition. Presented at the International Conference on AI and Education (Pittsburgh, PA).

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

Schank, R. C. (1984). *Dynamic memory: A theory of reminding and learning in computers and people.* New York: Cambridge University.

Schoenfeld, A. H. (1987). What's all the fuss about metacognition? In A. H. Schoenfeld (Ed.), *Cognitive science and mathematics education* (pp. 189-216). Hillsdale, NJ: Erlbaum.

Schmalhofer, F., Kuehn, O., Messamer, P., & Charron, R. (1989). 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).

Shalin, V., Bee, N. V. (1985). Analysis of the semantic structure of a domain of word problems. (Technical Report No. APS-20). Pittsburgh: University ofPittsburgh Learning Research and Development Center.

Singely, M. K., Anderson, J. R., Gevins, J. S., & Hoffman, D. (1989). The algebra word problem tutor. In D. Bierman, J. Breuker, and J. Sandberg, (Eds.) *Proceedings of the 4th International Conference on AI and Education.* Amsterdam, Netherlands: IOS. pp. 267-275

Smolensky, P., & Fox, B. (1988). Computer-aided reasoned discourse. In R. Guidon (Ed.), *Cognitive science and its application to computer human-interaction.* Hillsdale, NJ: Erlbaum.

van Dijk, T. A., & Kintsch, W. (1983). *Strategies of discourse comprehension.* New York: Academic Press.

Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes.* Cambridge, MA: Harvard University Press.

Weaver, C. A., & Kintsch, W. (1988). The conceptual structure of word algebra problems. Institute of Cognitive Science Technical Report No. 88-11, University of Colorado.

Author Notes

The authors gratefully acknowledge Denise Cummins, Clayton Lewis, Peter Polson, and Kurt Reusser for their many comments on this work. We also thank Paul Johl for his contributions in developing the ANIMATE system and Eileen Kintsch for her comments on earlier versions of this manuscript. This work was supported in part by Grant MH 15872 from the National Institutes of Mental Health to Walter Kintsch.

Communications concerning this manuscript can be sent to the attention of Mitchell J. Nathan, University of Colorado, Institute of Cognitive Science, Campus Box 345, Boulder, CO 80309-0345.

Footnotes

1. For related approaches, see Shalin and Bee (1985), Greeno, Brown, Foss, Shalin, Bee, Lewis, and Vitolo (1986), as well as Hall et al. (1989).

2. Although students also need help in solving the algebraic equations, we shall only be concerned with the construction of conceptually correct equations. Other tutors (e.g. Brown, 1985; Singely et al., 1989) exist that focus on solving equations.

Table 1a: Percent correct solutions (and raw scores) on pre- and post-tests for the four treatments

	Animation	Stopping-condition	Network	Equation	Total
Pretest	18.75 % (.75)	35.25 % (1.41)	24.25 % (.97)	41.5 % (1.66)	30 % (1.2)
Post-test	71.5 % (2.86)	60.25 % (2.41)	46.5 % (1.86)	52.75 % (2.11)	57.75 % (2.31)
Improvement (Pretest - Post-Test)	52.75 % (2.11)	25 % (1.0)	22.25 % (.89)	11.25 % (.45)	27.75 % (1.11)

Table 1b: Percent improvement on pre- and post-test for each problem, by treatment

	Animation	Stopping-condition	Network	Equation	Total
Solve (Near transfer)	63.4 %	24.1 %	35.7 %	4 %	31.9 %
Debug (Medium transfer)	53.6 %	14.3 %	46.4 %	35.7 %	37.5 %
Story-writing (Far transfer)	50 %	35.7 %	7 %	-7 %	21.4 %

Table 2: Percent correct solutions (and raw scores) on training tasks for three training conditions.

	Animation	Stopping-condition	Network	Equation	Total
Total Scores	85.7 % (2.57)	58.3 % (1.75)	60 % (1.8)	50 % (1.5)	63.3 % (1.9)
Task 1: Mult. Choice	31 % (.93)	16.7 % (.50)	12 % (.36)	14.3 % (.43)	18.3 % (.55)
Task 2: Set up	29.7 % (.89)	27.3 % (.82)	28 % (.84)	19 % (.57)	26 % (.78)
Task 3: Debug	25 % (.75)	14.3 % (.43)	20.3 % (.61)	16.7 % (.50)	19 % (.57)

Table 3: Raw scores on Debug (intermediate transfer) and Story (Distant transfer) and Solve (near transfer) problems on the training task booklets and pre- and post tests, for Animation and Equation groups.

	Problem	Pretest	Training	Post-Test
Equation Group	Debug (intermed.)	.21	.50	.57
	Story (distant)	.29	N/A	.21
	Solve (near)	.62	.57	.66
Animation Group	Debug (intermed.)	.11	.75	.64
	Story (distant)	.21	N/A	.71
	Solve (near)	.21	.89	.85
Network Group	Debug (intermed.)	.18	.61	.64
	Story (distant)	.07	N/A	.14
	Solve (near)	.38	.84	.71
Stopping-condition Group	Debug (intermed.)	.39	.43	.54
	Story (distant)	.07	N/A	.43
	Solve (near)	.48	.82	.72

Figure Captions

Figure 1. A conceptual problem model for Problem 1 using the Network Method. Nodes serve as placeholders or slots for values and variables (e.g. t) while arcs are labelled with operators. This network is additionally labelled with the propositions derived from the textbase of Problem 1 to show how text comprehension supports problem model generation.

Figure 2. A graphical depiction of our theory of problem comprehension. From the bottom moving up, problem-relevant and situation-relevant information are both extracted from a problem text and form independent and potentially isolated mental representations. True problem comprehension is achieved when these representations cohere. By having students explicitly link a problem model network to a situation model-like animation, we expect to facilitate their coordination. Students manipulate the network to produce changes in the animation until the animation matches their expectation of the problem situation.

Figure 3. Pushing the "Pick Character 1" button presents a selection of characters for playing out the situation on the computer screen. Students select first a character and then a screen location and direction. Here, a student has just selected the ant character for the passage of Problem 2.

Figure 4. Problem model building is supported by a palette which presents network components. Any chosen component may stipulate the mathematical conditions for which

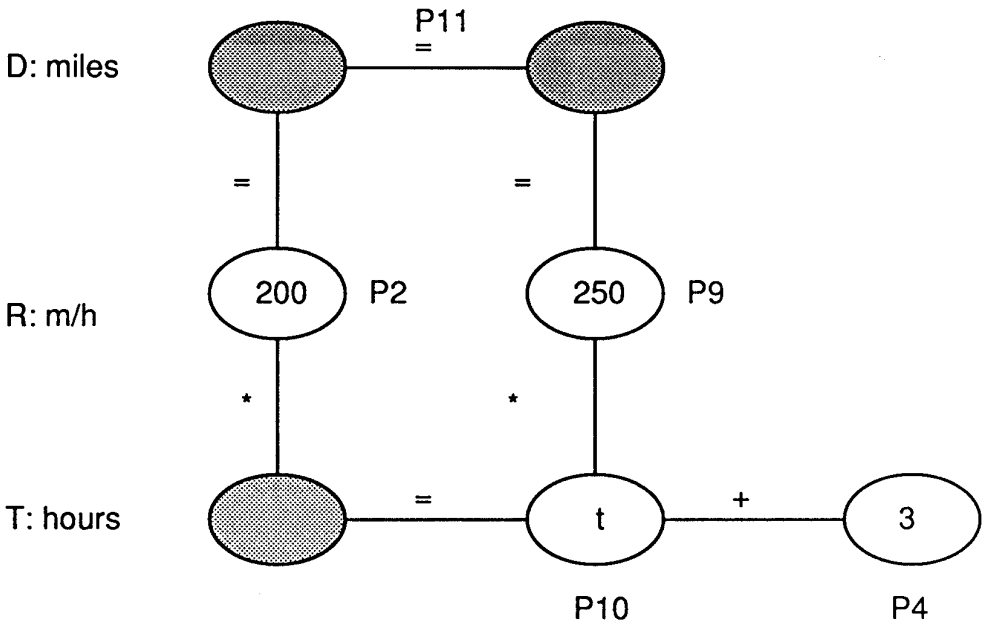
the animation should stop. Deciding this may be an important part of forming a sound situation model.

Figure 5. Values and variables are entered into a selected node by way of a calculator interface. Here, the rate of the ant given in Problem 2, 400 mph, is being entered.

Figure 6. A fully-developed problem model, including the ant, the helicopter and all the associated values and relations. The Distance equation is boxed indicating its role as the stopping condition for the animation. This equation is also highlighted in black to indicate that the tutor has detected an algebraic error which the student may choose fix or to ignore.

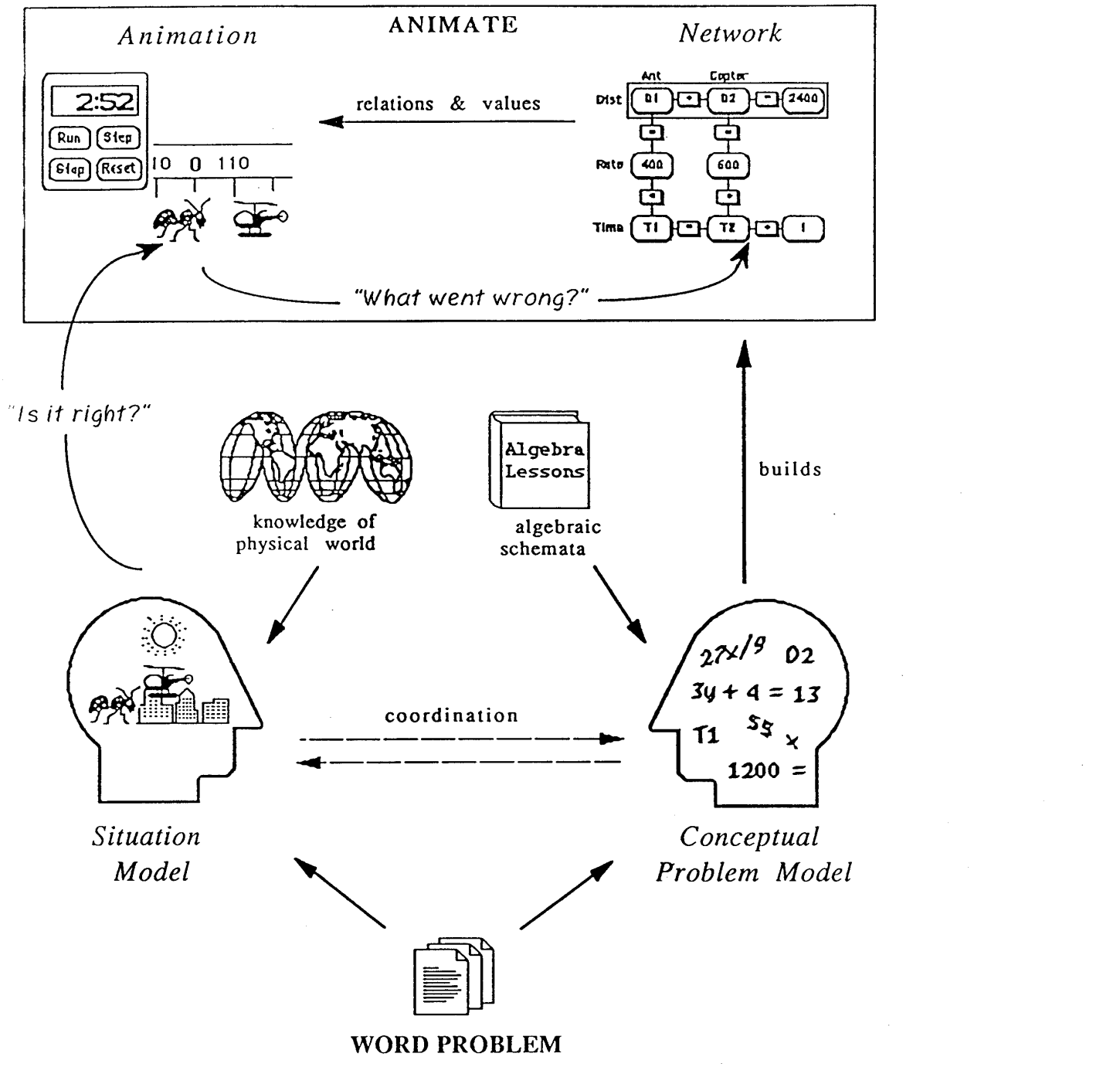
Figure 7. The student elected to ignore the error message shown in Figure 6. After 19 sec of running the animation only the helicopter has moved while the ant waits, behavior which is specified by a negative delay value in the horizontal Time equation. This contradicts the situation described in Problem 2.

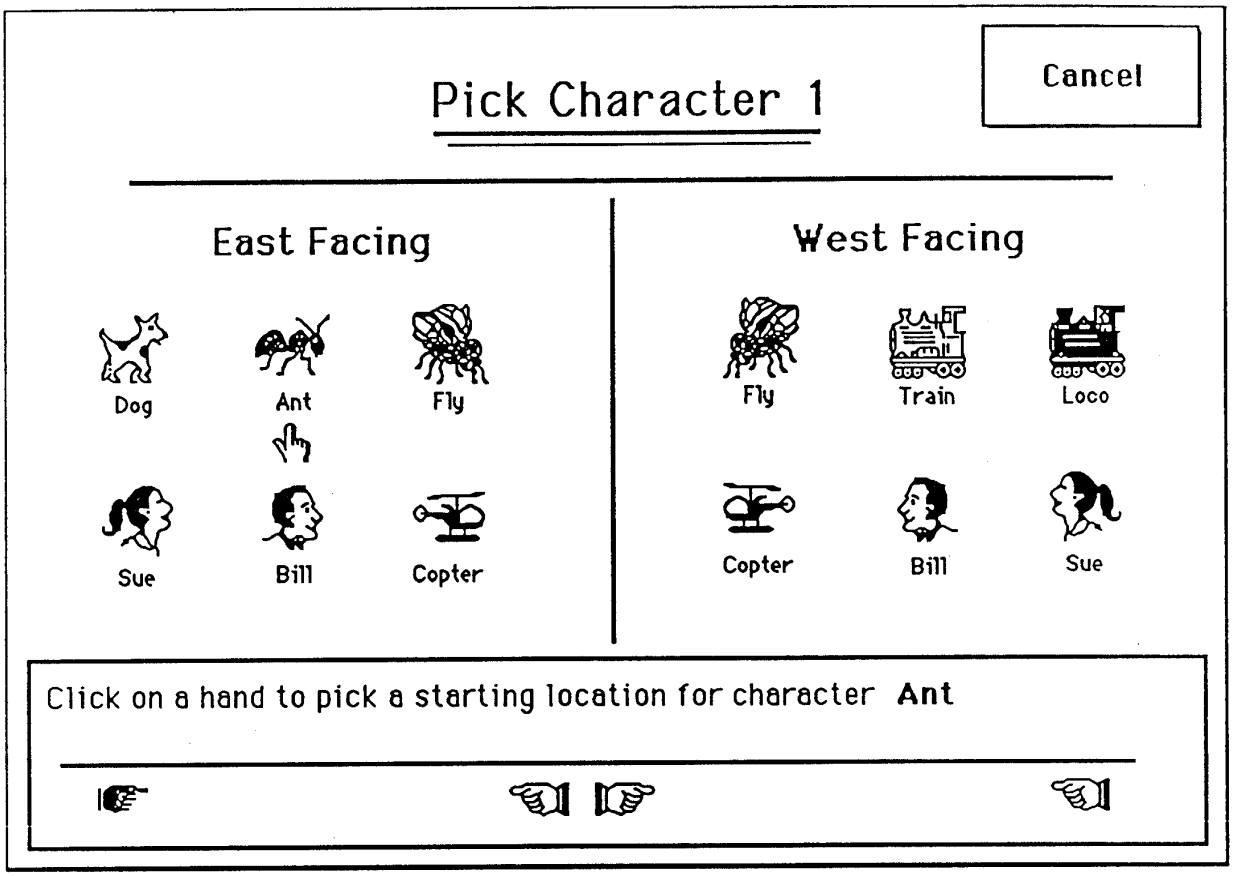
Figure 8. After debugging the problem model network, and introducing variables for unknown quantities, the animation is run and shows a situation in accordance with expectations. Eventually, the ant and helicopter meet. Time and distance gauges show solution values while the network contains the problem formalism. The final algebraic equations can be read from the network horizontally and vertically.

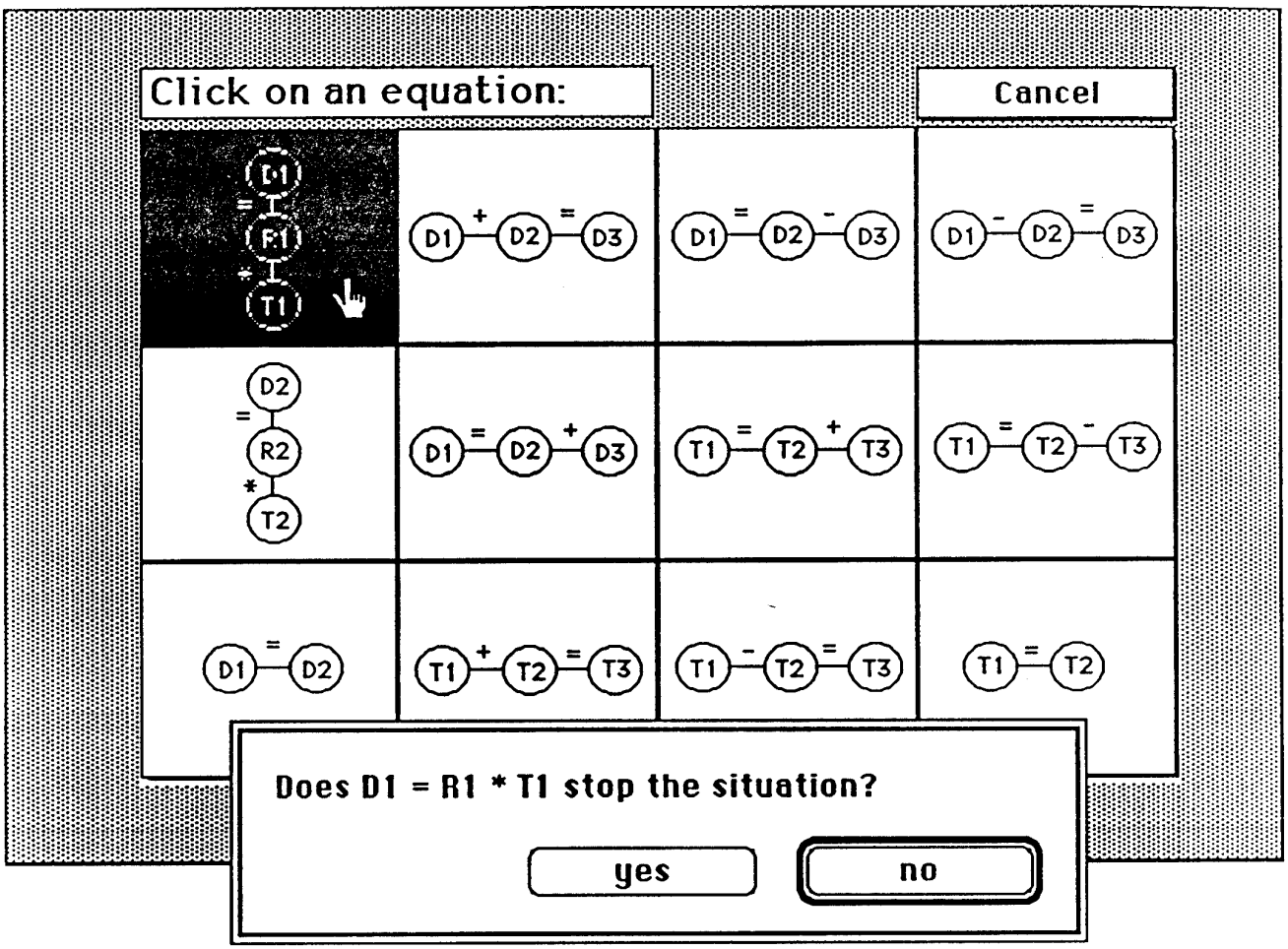


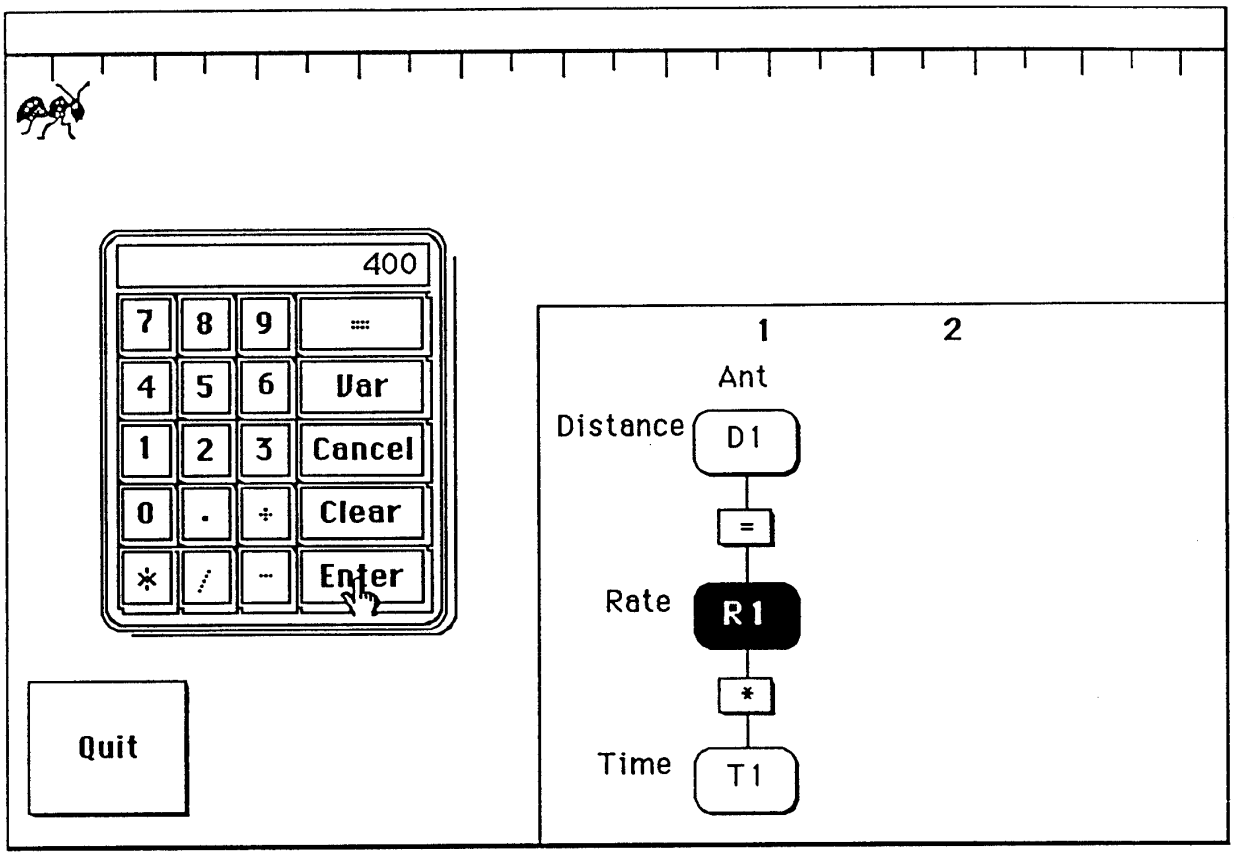
PLANE 1
P1
P3

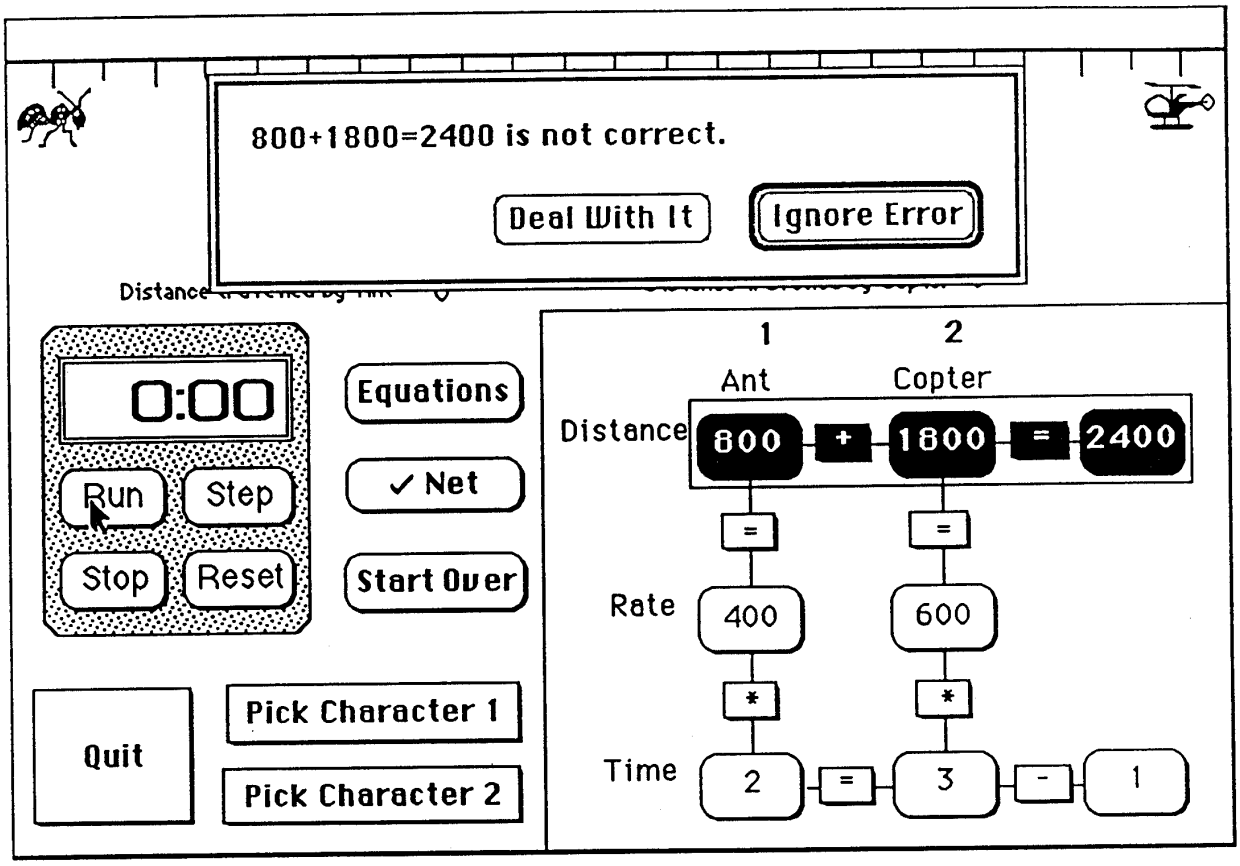
PLANE 2
P5
P6
P7
P8

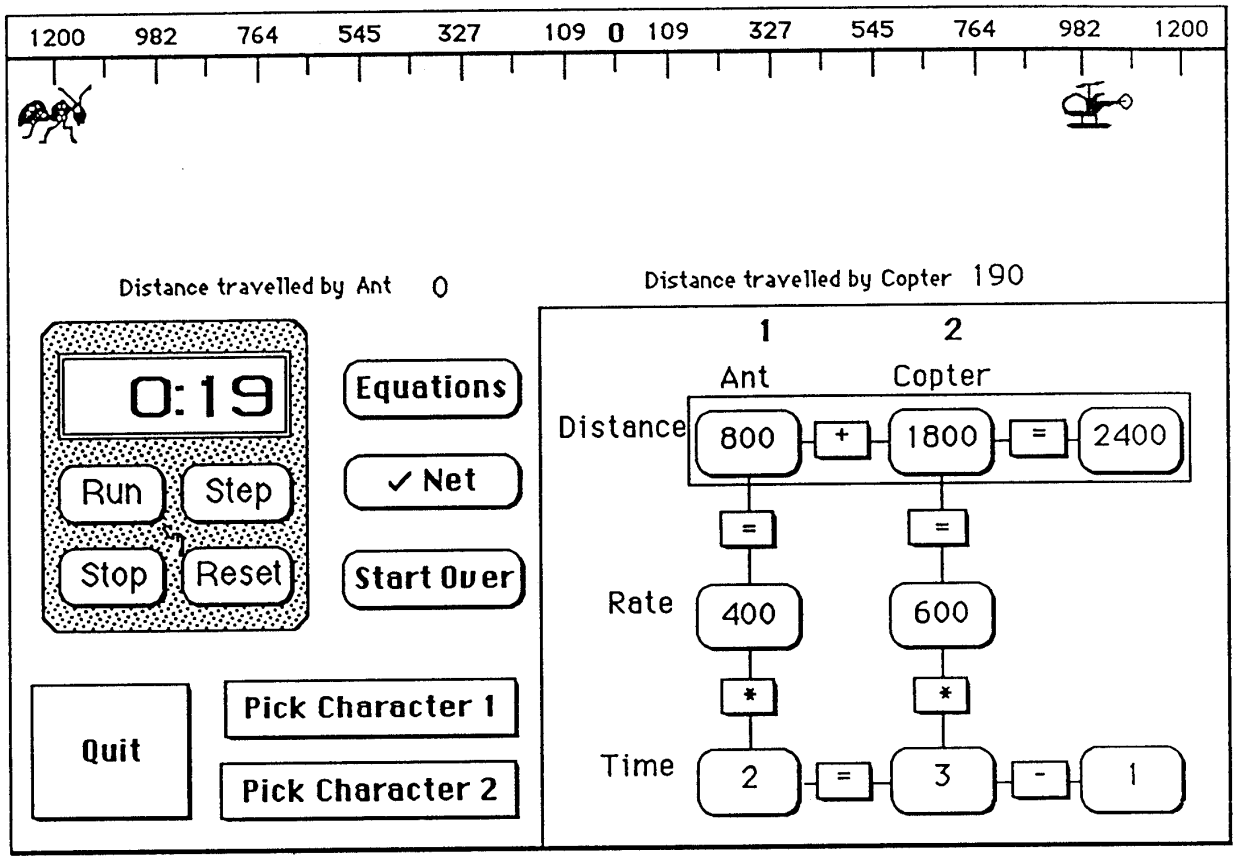

Animation
ANIMATE
Network
2:52
Run
Step
Stop
Reset
10 0 110
relations & values
Ant
Copter
Dist
D1
D2
2400
Rate
400
600
Time
T1
T2
1
"What went wrong?"
"Is it right?"
knowledge of physical world
Algebra Lessons
algebraic schemata
builds
coordination
27x/9 D2
3y + 4 = 13
T1 5s x
1200 =
Situation Model
Conceptual Problem Model
WORD PROBLEM

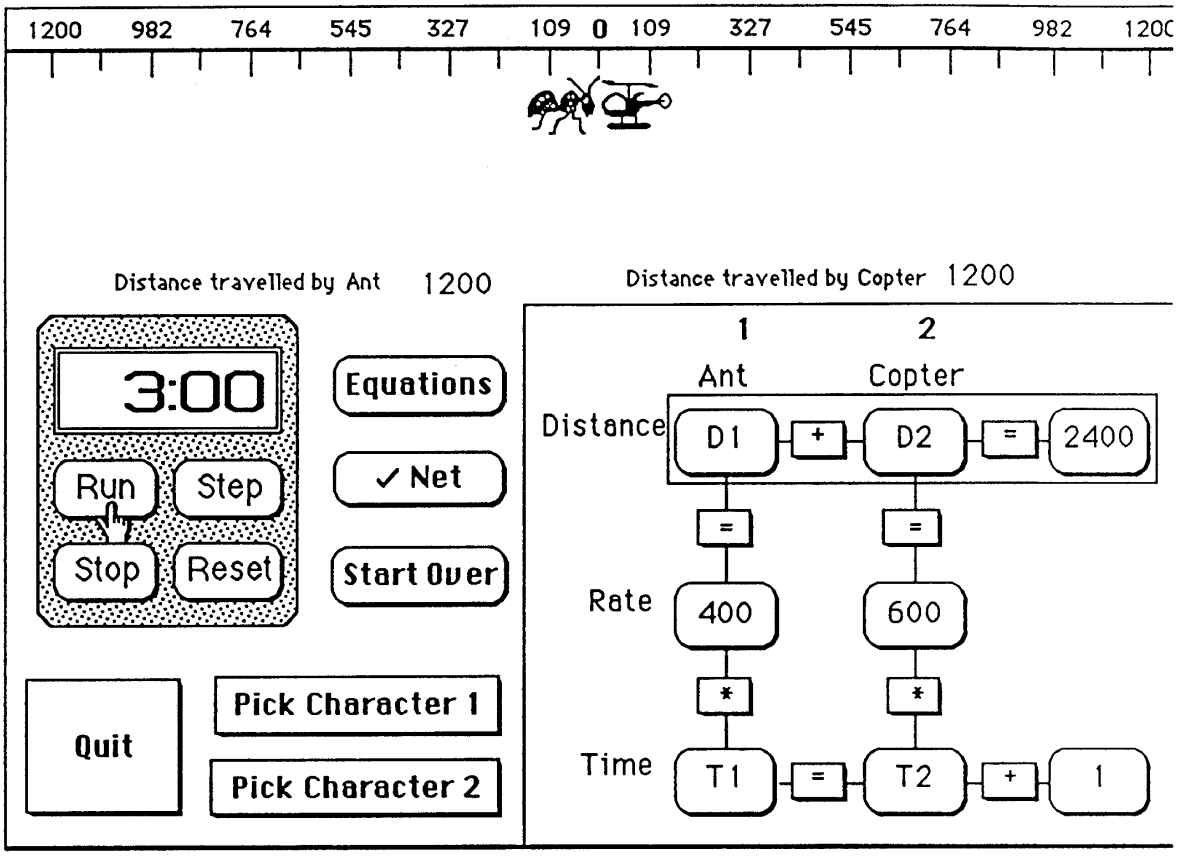

Pick Character 1
Cancel
East Facing
Dog
Ant
Fly
Sue
Bill
Copter
West Facing
Fly
Train
Loco
Copter
Bill
Sue
Click on a hand to pick a starting location for character Ant


Click on an equation:
Cancel
D1 = R1 * T1
D1 + D2 = D3
D1 = D2 - D3
D1 - D2 = D3
D2 = R2 * T2
D1 = D2 + D3
T1 = T2 + T3
T1 = T2 - T3
D1 = D2
T1 + T2 = T3
T1 - T2 = T3
T1 = T2
Does D1 = R1 * T1 stop the situation?
yes
no


400
7
8
9
4
5
6
Var
1
2
3
Cancel
0
.
Clear
Enter
Quit
1
2
Ant
Distance
D1
=
Rate
R1
*
Time
T1


800+1800=2400 is not correct.
Deal With It
Ignore Error
0:00
Run
Step
Stop
Reset
Equations
✓ Net
Start Over
Quit
Pick Character 1
Pick Character 2
1
2
Ant
Copter
Distance
800
+
1800
=
2400
=
=
Rate
400
600
*
*
Time
2
=
3
-
1


1200 982 764 545 327 109 0 109 327 545 764 982 1200
Distance travelled by Ant 0
Distance travelled by Copter 190
0:19
Run
Step
Stop
Reset
Equations
✓ Net
Start Over
Quit
Pick Character 1
Pick Character 2
1
2
Ant
Copter
Distance
800
+
1800
=
2400
=
=
Rate
400
600
*
*
Time
2
=
3
-
1


1200 982 764 545 327 109 0 109 327 545 764 982 1200
Distance travelled by Ant 1200
Distance travelled by Copter 1200
3:00
Run
Step
Stop
Reset
Equations
✓ Net
Start Over
Quit
Pick Character 1
Pick Character 2
1
2
Ant
Copter
Distance
D1
+
D2
=
2400
=
=
Rate
400
600
*
*
Time
T1
=
T2
+
1

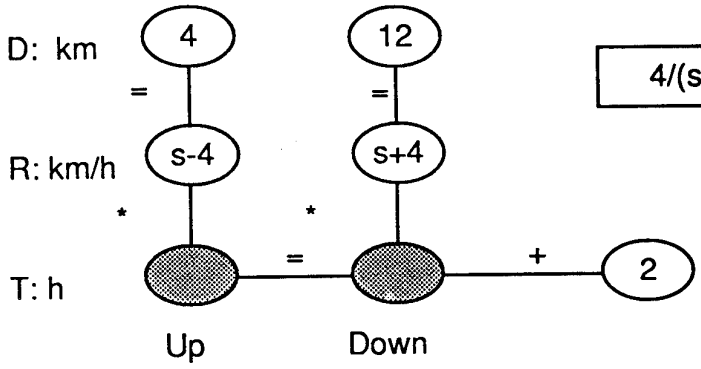
Appendix

Problem Models for Two Examples From Each of The First Four of Mayer's (1981) Seven Algebra Word Problem Families

FAMILY I: AMOUNT-PER-TIME RATE

Current

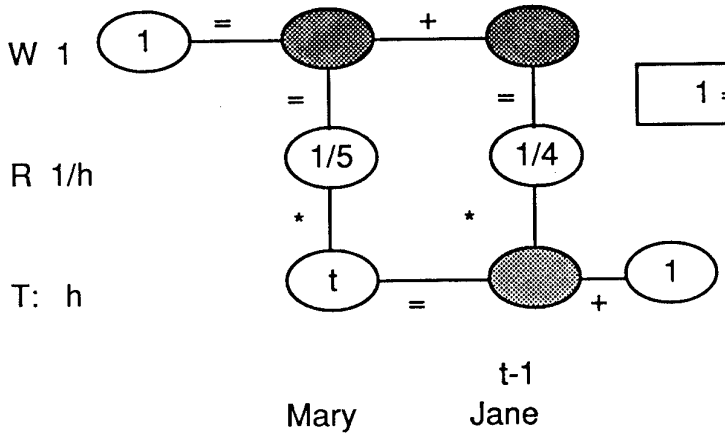
The current in a stream moves at a speed of 4 km/h. A boat travels 4 km upstream and 12 km downstream in a total time of 2 hours. What is the speed of the boat in still water?



$$4/(s-4)+12/(s+4) = 2$$

Work

Mary can do a job in 5 hours and Jane can do the job in 4 hours. If they work together, how long will they take to do the job if Jane starts 1 hour after Mary?

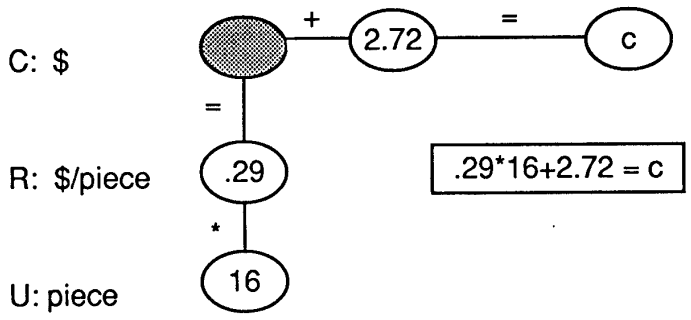


$$1 = 1/5*t+1/4*(t-1)$$

FAMILY II: COST-PER-UNIT RATE

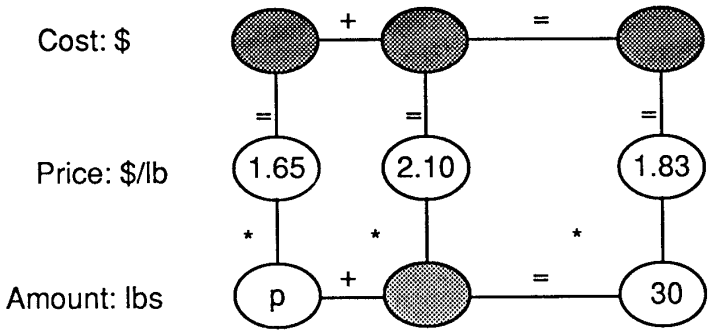
Fixed Cost

Sixteen balls of yarn can be bought from a mail order housefor 29c each plus $2.72 for postage. What does the total order cost?



Dry Mixture

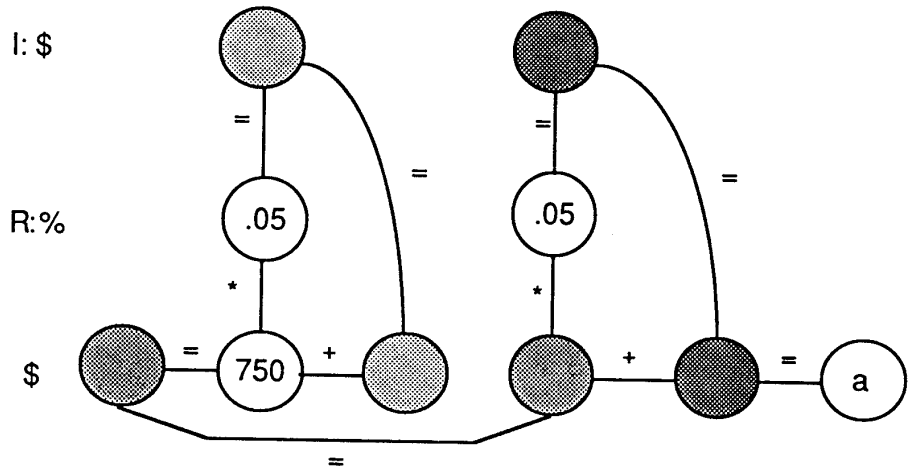
A grocer mixes peanuts worth $1.65 a pound and almonds worth $2.10 a pound. She wants 30 pounds of the mixture worth $1.83 a pound. How many pounds of each should the grocer include in the mixture?



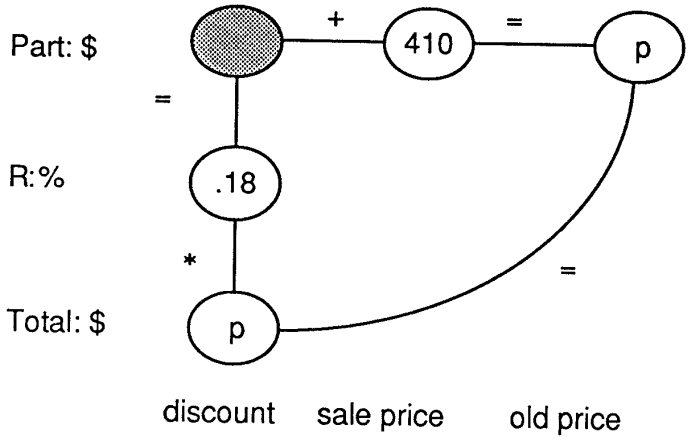
$$1.65p + 2.10(30-p) = 1.83*30$$

FAMILY III: PORTION-TO-TOTAL-COST RATE

Suppose $750 is invested at 5% annually. What amount will be in the account at the end of 2 years?



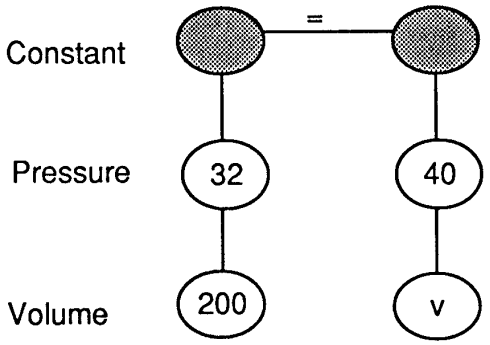
An appliance store drops the price of a certain TV 18% to a sale price of $410. What was the former price?



FAMILY IV: AMOUNT-TO-AMOUNT RATE

Inverse Variation :

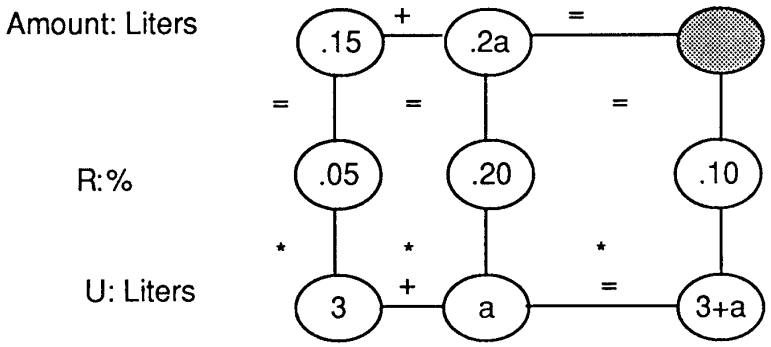
The voluime of gas varies inversely with the pressure on it. The volume of gas is 200 cc under a pressure of 32 kg/sqcm. What will be its volume under a pressure of 40 kg/sqcm ?



32*200 = 40v

Wet Mixture :

A chemist has 3 liters of a 5% acid solution. How many liters of a 20% acid solution must be added to make a mixture which is 10% acid?



.05*3 + .20*a = .10*(3+a)