COGNITION AND INSTRUCTION, 1992, *9*(4), 329–389

A Theory of Algebra-Word-Problem Comprehension and Its Implications for the Design of Learning Environments

Mitchell J. Nathan and Walter Kintsch
University of Colorado

Emilie Young
U S WEST Advanced Technologies
Denver, Colorado

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 competent student must coordinate this information with known problem models so that formal (i.e., algebraic) operations can be applied and exact solutions can be obtained. We argue that this task is a highly reading-oriented one in which poor text comprehension and an inability to access relevant long-term knowledge lead to serious errors. In particular, poor students often omit from their solutions or misspecify necessary mathematical constraints that are based on reading inferences needed to describe fully the problem situation. Furthermore, formal algebraic expressions are so abstract that their meaning is often elusive; this contributes to mistranslations and misinterpretations. The competent approach is teachable, however.

We describe experimental results with ANIMATE, a learning environment that knows nothing of the problem at hand or of the student's actions. Subjects encouraged to reason explicitly about the situations described in typical word problems consistently performed as well as or better than those who were not, in both training and transfer tasks. We conclude that, by using an environment that gives equations situation-based meaning through computer animation, 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 call for evaluation methods beyond just algebra problem-solving performance is made. The implications of this work for the design of future computer-based tutors and other learning environments are also discussed.

Requests for reprints should be sent to Mitchell J. Nathan, Learning Research and Development Center, University of Pittsburgh, Pittsburgh, PA 15260.

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. This article presents a theory of word-problem comprehension based on research in the field of discourse processing (e.g., Kintsch, 1988; van Dijk & Kintsch, 1983). The theory focuses on the mental representations 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 and failures to relate the situation described in the problem statement to the formal expressions needed 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; Lewis & Mayer, 1987) and mathematics in general (Resnick, 1988). Indeed, empirical analyses of how college students actually solve word algebra problems indicate 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: *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.* A necessary assumption of our theory is that students are capable of understanding the stories given and that they form an appropriate situation model. We hypothesize that correspondence is teachable and describe a computer tutoring system, ANIMATE (Nathan, Johl, Kintsch, & Lewis, 1989; Nathan & Young, 1990), which embodies our theory. We present results of students' use of it when solving typical high-school-level and college-level algebra word problems. This tutoring system facilitates the correspondence between the situation and the formal equations by making their relation explicit through the use of equation-driven computer animation. In doing so, it helps students to better comprehend the problem and ultimately to improve their problem-solving performance.

We begin by presenting a theory of word-problem comprehension, drawing on earlier work in this area. The role of situation-based reasoning when solving problems in formal domains (of central importance in our theory) is then discussed. The theory of problem comprehension, however, does not sufficiently define how to teach students to solve word problems. For this, we need a theory of computer-based instruction for word algebra problems, addressing the role of student errors and system feedback during training. From this discussion, we adopt a set of instructional principles that guide the design of ANIMATE, a computer tutor. After illustrating how ANIMATE relies on our model of problem comprehension, the functionality and features of the tutor are demonstrated by way of a sample problem-solving session. We then present empirical results of students' use of the tutor in a laboratory setting as an indirect test of our theory of problem comprehension. The results obtained from students using ANIMATE are compared with students operating under differing levels of experimental control. In this study, problem-solving performance is measured by students' scores on a variety of typical motion–distance (D) = rate (R) $\times$ time (T)–problems before, during, and after training. Problem comprehension is evaluated by examining students' errors and assessing students' performance on some nontraditional tasks such as writing a story problem from a set of algebraic constraints. These preliminary results for the tutor group are encouraging. They are discussed in terms of models of human problem solving, methods of problem-solving evaluation, and the design of future learning environments.

A PSYCHOLOGICAL MODEL OF WORD-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. Although models of problem-solving behavior abound (e.g., Newell & Simon, 1972), few studies (e.g., Bobrow, 1968; Simon, 1979) 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. A significant component of a comprehension model is how one's mental representations of the problem situation inform and help to constrain the formal expressions necessary for a solution (Greeno, 1989; Singley, Anderson, Gevins, & Hoffman, 1989).

We outline a theory of word-problem comprehension, 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; Kintsch, 1988). Kintsch and Greeno theorized 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 representation for the actions in a text, termed the *situation model* (van Dijk & Kintsch, 1983).

Using a production-rule model, Kintsch and Greeno (1985) simulated the reading behavior of young students solving arithmetic problems. Productions were used to construct a representation of a word problem–a textbase and situation model–and to select among possible problem-solving strategies. The situation model for the problem-solving task was highly task specific, capturing the set relations and arithmetic operations needed for solving the problem. They thus termed this representation the *problem schema* and hypothesized that it subsumed the situational nature of the problem text. The arithmetic 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 modeled 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 modeled students' faulty behavior best. Cummins et al. also showed that solution performance was associated with the ability to recall the problem statements. Students who could correctly recall the story textbases were more likely to produce correct solutions. Students who incorrectly recalled problems apparently encoded a different problem textbase than the one presented, which they often then solved correctly. This study suggests that the development of an appropriate representation of the formal aspects of the problem is highly dependent on language comprehension skills and the formation of a correct textbase and situation model. This in turn affects word-problem-solving performance. The study prescribes that instruction focus on language comprehension processes as well as on the mathematical aspects of word-problem solving.

In the current discussion, we distinguish between a representation for events (the *situation model*) and one that is constructed with formal relations in mind (the *problem model*), teasing apart distinctions not made by Kintsch and Greeno (1985). Processing algebra word problems is viewed here as similar to processing arithmetic word 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 a (quantitative) problem model that captures the algebraic problem structure. A set of algebraic problem schemas (templates for organizing problem-relevant information) provides the explicit, graphical cues to guide the construction of these problem models. The problem schemas for 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 schema using a method of constraint propagation. In our view, the process of understanding and solving word problems involves three mutually constraining levels of representation that must be constructed by the student: (a) a representation of the textual input itself–the textbase, (b) a model of the situation conveyed by the text in everyday terms–the so-called situation model, and (c) the formalization of that situation–the problem model.

In the problems under consideration here, situation models are often based on imagery, although a situation model may also be represented propositionally, as has been found with users' representations of computer processing tasks (Mannes & Kintsch, 1991). In the tasks we have selected, computer animations provide an effective way to characterize much of the content of the situation model for a text. We illustrate this process with a single worked-out example.

Generating a Propositional Representation and a Situation Model

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

According to the theory, a reader extracts the propositional representation from a problem. When reading Problem 1, a list of propositions is derived, as in van Dijk and Kintsch (1983):

Prop1 LEAVE[PLANE1, DENVER, Time1]
Prop2 RATE[Prop1, 200 mph]
Prop3 DIRECTION[Prop1, EAST]
Prop4 LATER[Prop1, Prop5, 3 HOURS]
Prop5 LEAVE[PLANE2, DENVER]
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" (Prop11), with propositions denoting such information as how fast the planes are going and when they start subordinate to it.

The corresponding situation model consists of a representation of the two planes speeding along a parallel course, capturing the moment when the second plane passes the first. Such a representation involves details that may differ among readers. We have no way of specifying these details, and there is no need to do so. All we need to do is represent the essential, common parts of everyone's situation model: one moving object overtaking another.

The situation model draws on a reader's knowledge of the world to "fill in the gaps" left by a sparse story. In the example text, nothing is said about the relative distances the two planes will travel when "overtake" occurs, yet the situation model reveals that they will have flown equal distances at that point (van Dijk & Kintsch, 1983; Weaver & Kintsch, 1987). Furthermore, the situation model tends to present things in relative terms, qualitatively, without regard to the precise quantities often found in the textbase of a problem statement and needed in a problem model (Nathan, 1988).

All this is no different from understanding a story. To understand a story problem, the reader must have sufficient knowledge to understand the situation adequately and appropriate strategies to generate necessary inferences and elaborations to make the story complete (van Dijk & Kintsch, 1983). This is the case for many of the problems we consider here but not for all. For those problems where students lack a situational understanding (e.g., physics problems that challenge students' naive notions, as reported by McCloskey, 1983), we have to teach this first. When there is situational understanding, it can help provide the student with a solution-enabling algebraic interpretation for the problem text. It does this by helping the student access and apply the real-world knowledge associated with a situation when setting up the formal relations or problem schema. The situation model further helps the student by providing situational constraints, against which formal constraints may be checked (cf. Singley et al., 1989).

An erroneous problem schema, as shown in the next section, violates or omits certain aspects of a situation model. In terms of our theory, problem schema omissions are expected to match information that is unstated in the text but necessary for complete understanding of the situation. Violations often parallel miscomprehensions of the text or misapplication of algebraic principles. Along with the textbase and situation model representations often attributed to readers, problem solvers additionally must produce a representation of the problem structure that includes relations among quantities in the problem. This problem schema is the representational level at which students can apply formal calculation methods such as algebra for generating verifiable solutions (Kintsch & Greeno, 1985; Reusser, 1988). The situation model can support detection and even correction of formal problem-schema errors but only when correspondence between it and the problem model representations is established.

Generating a Problem Model

In solving an algebra word problem, the psychological theory we have been developing maintains that a mathematical description must be developed that is consistent with the reader's situation model. Construction of such a description from our example theoretically might go as follows: Prop2, from the prior list of propositions, specifies a relation between a distance and a time, thus eliciting in the reader a distance–rate–time ($D = R \times T$) schema labeled PLANE1 (see Figure 1). This schema has slots for distance (D), rate (R), and time (T): Prop2 is assigned to slot R, whereas Prop1 and Prop3 go into slots that help to further specify the problem situation. Prop9 initiates the construction of a second $D = R \times T$ 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 $D = R \times T$ schemas of the two planes, is tagged as a supporting relation, one that holds the relative starting times ("delay" information) of the two planes. Supporting relations are a special kind of problem schema. They are not often found

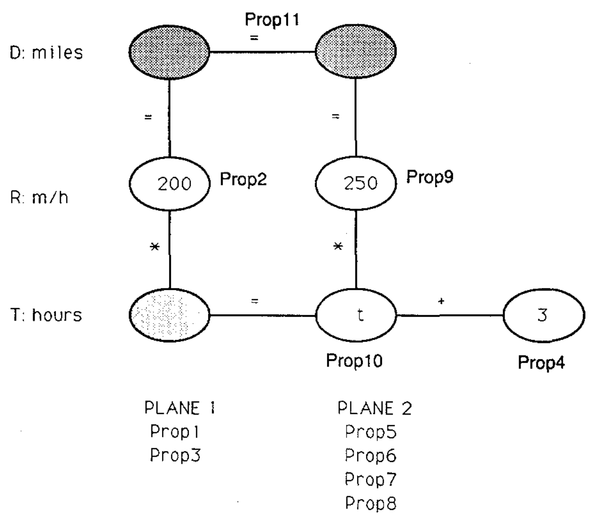


FIGURE 1 A conceptual problem model for Problem 1 using the network method. The network is labeled with propositions (e.g., Prop9) from the textbase of Problem 1 to show how text comprehension supports problem model generation.

alone but are used to relate two or more independent schemas and tend to capture more general relations (such as the notion of a "delay") than do the problem-specific schemas such as $D = R \times T$.

Supporting relations can be based on aspects of a situation that are unstated in the original text and so are absent from the reader's textbase. When this is so, these relations need to be derived through an inference supported by the reader's general knowledge and situational understanding of the text (Singley et al., 1989). These inferences make large demands on the reasoning and memory of a problem solver (van Dijk & Kintsch, 1983). We thus come to the first explicit prediction of our word-problem comprehension theory:

Prediction 1a: Because of the added cognitive demands of inference making, readers will make inferences only when they seem necessary. Poor problem solvers will tend to *omit* them from their representations, and so they will omit the associated equations (supporting relations) from their solutions to story problems. Problem solvers who reason situationally will tend to include these inference-based equations.

Situation-based reasoning can produce the inferences necessary for understanding a problem situation. A correspondence between the situation model and the problem model constrains one's representations to be faithful to both. This leads to our second prediction:

Prediction 1b: Initially poor problem solvers who have subsequently learned to make an adequate situation model of the cover story and to link this representation to their mathematical understanding will exhibit in their solution protocols a marked decline in their misspecification of supporting relations.

Predictions 1a and 1b mark our initial attempts to specify students' problem-solving behavior within the framework of the problem comprehension model. Together they make claims about what we expect to see in students' initial solution protocols and how we expect them to change when a student would assume our ideal, model-based reasoning approach. As we develop the theory more fully, we refine and add to this set of predictions.

Returning to our example, the problem solver must realize that Prop10 of the textbase deals with the travel time of Plane2, so it is assigned to slot T in the PLANE2 $D = R \times T$ schema (indicating that the slot must be filled with a variable or unknown). Prop11 requires that an inference be made regarding the state of the planes at the point of overtake before it can be tied to a supporting relation. Thus, propositions in the textbase and inferences drawn from the situation model aid the subject in constructing and instantiating (situation-specific) problem schemas. This in turn organizes the propositions into solvable algebraic formalisms. Alternative ways of forming a solution-enabling problem schema are, of course, possible (e.g., one may prefer a catch-up scenario that focuses on the difference in rates of the two planes).

A simple graphical form, shown in Figure 1, acts as a correlate of the problem model.[1] Each $D = R \times T$ schema is shown as three ovals, corresponding to the D, R, and T slots, respectively, in a vertical arrangement. Ovals are connected by line segments labeled with mathematical operators, such as "=" and "*" (multiplication). For clarity in analyzing this example, the text propositions organized by these schemas are written next to the ovals and/or lines to which they have been assigned. Inside each oval, information derived from the text propositions is shown. Line segments are labeled so that the lines and ovals can be read as equations, either horizontally or vertically.

An algebraic representation for the word problem given before, along with the corresponding propositions and inferences from which it is derived, would be:

Equations	Propositions	Inferences/Supporting Relations
$D1 = 200 \times T1$	1, 2, 3	–
$D2 = 250 \times T2$	5, 6, 9	–
$T1 = T2 + 3$	1, 4	"Later" as "+ 3," because PLANE2 travels less time than PLANE1

[1]See, for example, the verbal protocols that capture the problem-solving process of experts and novices rectifying a faulty situation model of simple mechanics and geometry problems by relying on their understanding of the problem model (Larkin, 1983; Schoenfeld, 1985).

Equations	Propositions	Inferences/Supporting Relations
$D1 = D2$	5, 6, 7, 11	"Overtake" as "=," because this occurs when the two planes have traveled equal distances
$T2 = ?$	10	–

The distinction between understanding the situation in everyday terms and understanding the problem structure in terms of a formal algebraic problem schema is central to our model: Students require the most support when constructing a problem model representation. They are often familiar with the everyday situations that 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 required for word problems must from the very beginning be driven by the salient features of the problem text. Thus, the nascent formalism actually helps the student to understand the situation in terms of the problem model and to comprehend better the problem statement (Nathan, 1988). Consequently, we find a mutually supporting relationship in which situational understanding helps students realize the episodic meaning of a formal problem model, and, reciprocally, sensitivity to the requirements of a problem schema aids in the construction of a suitable situation model. These claims of our model can be more clearly stated as experimental predictions for students' problem-solving behavior.

Prediction 2: Students encouraged to interpret story situations mathematically by relating the characters, events, and relations in a given cover story to their knowledge of formal symbols and expressions needed for a quantitative solution will be more competent in generating solution-enabling equations for word problems than their counterparts who use a straight translation-based approach of mapping story phrases to equations.

It follows from this claim that students who initially use the straight mapping approach but subsequently learn to use the situation-based approach are expected to demonstrate marked improvements in their performance, greater than one would expect from just reviewing algebra. Because we are looking at the formation of a two-way mapping from situational understanding to mathematics, the complementary claim can also be made.

Prediction 3: Students encouraged to interpret algebraic equations situationally by relating the formal symbols and expressions to their knowledge of characters, events, and relations in the given cover story will be more competent in the generation of situational descriptions of algebraic equations than their counterparts who use a straight translation-based approach of mapping story phrases to equations.

As with the type of behavior mentioned earlier, we expect to see students who are situationally trained demonstrate improvement that is not explained by a review of algebra alone. Adequate situation-based reasoning will support the formation of the inferences necessary for a complete understanding of a problem situation. The situation model can support detection and even correction of formal-problem-schema errors, but only when there is a correspondence made between the situation model and the problem model representations. From this discussion and these predictions, one further claim can be made:

Prediction 4: Problem solvers who reason situationally are better equipped than their phrase-oriented counterparts to recognize the situational appropriateness or inappropriateness of a set of equations that may accompany a cover story. Two subclaims follow.

Prediction 4a: These students will be better than their phrase-oriented counterparts at matching a mathematical description to its referent situation and at detecting mathematical descriptions inconsistent with a given situation.

Prediction 4b: These students will be better able to correct improperly specified formal expressions so that they correctly describe the intended situation.

Generating an Equation

Ovals in Figure 1 represent schema slots (e.g., $T1$ represents the travel time for Character 1). When a slot is referred to in a problem statement with an associated value, the oval for that slot can be filled. Otherwise, a variable is introduced and put in the oval as a place marker, indicating the unfinished state of the problem network. Other empty ovals can be filled in by a process of constraint propagation, so that 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. We can use the proposition stating that the second plane started 3 hr later (Prop4) to obtain the supporting relation "$t + 3$" as the time for the first plane (slot T for the first column). The two vertical $D = R \times T$ schemas can supply entries for their respective D ovals. Applying the situationally inferred equality relation (from Prop11) for the two distances entered in the problem schema yields

$$200 * (t + 3) = 250 * t. \qquad \text{(Equation Set 1)}$$

Algebraic Problem Schemas

Problem schemas 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 schemas, one can construct a priori all the necessary problem models. Mayer (1981) compiled a list of 1,097 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, whereas 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 following form:

$$\text{UNIT1} = \text{Rate-of-UNIT1-per-UNIT2} \times \text{UNIT2}.$$

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

Rate Family	UNIT1	UNIT2
Amount per time	Amount	Time
Cost per unit	Cost	Unit
Portion to total cost	Portion	Total cost
Amount to amount	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 $2 \times N$). For physics and geometry problems, physical laws, theorems, and axioms correspond to schemas. Newton's second law—that the sum of all forces for a system in equilibrium must be zero—is one example.

In Appendix A, 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 requested. These variations can be 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 schemas like those of Figure 1.

The schematic formalism we developed is designed to help students understand the conceptual structure of an algebra word problem, but it is not the only formalism possible in the domain of algebra problems.[2] Furthermore, it does not understand the problem for the student. Students must still make the hard infer-

[2]For related approaches, see Shalin and Bee (1985), Greeno et al. (1986), and Hall et al. (1989).

ences, still apply their knowledge of the situations described, and still ultimately generate algebraic equations. The graphic network representation reduces students' 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.

From this discussion, it should be clear that a goal of any instructional approach that grows from our model will be to help students organize and search for missing information by making otherwise covert structures overt, using, for example, a graphical arrangement. 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 faster plane overtake the slower one and continue across the screen indefinitely may suggest to the student that incorporation of additional constraints in the problem schema will bring about the intended animation: namely, that the two planes must not pass their ultimate destination. The top horizontal line segment in the example (Figure 1), which equates the distances traveled by the two planes, represents such a constraint.

THEORY OF LEARNING: MAKING FORMALISMS SITUATIONALLY MEANINGFUL

Word-problem solving is both an exercise in text processing and in 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; Willis & Fuson, 1988), although 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 also demonstrate early familiarity with "translation rules" that help map key words 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 underlying principles of algebra or do not see the formulas 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, although it may be inconsistent with or unrelated to the situation described. When given situationally impossible problems, for example, some students blindly translate passages into formal expressions and produce answers that, although 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, on 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 (1945/1982), in his famous discussion about teaching students to find the area of a parallelogram, was probably the first to draw attention to this dissociation of formal expressions from the situation described.

Greeno (1989) explained many of these results as instances where abstract *symbol*-space representations (e.g., equations) and transformations, and real-world or *situation*-space representations coexist as disconnected systems. Consequently, students learn to solve symbolic expressions procedurally without learning how to reason about related real-world events. When these representations exist as disconnected entities, it is possible for problem solvers to perform operations on symbolic expressions that are no longer faithful to the situations to which the intended expressions refer. By establishing a correspondence of the symbols to the situation, one roots the formalisms in the space of permissible and expected events (Greeno, 1989). Unfortunately, institutional instruction sometimes ignores this and fosters the acquisition of procedures not anchored to real-world situations.

The theory of problem comprehension we have presented shows how a formal problem description, such as a set of equations, relates to a concrete situation. Clearly, it is not always practical to bring the real world into the classroom, having students observe real flying planes, for example. But we do have the technology to provide reasonable animations of these events that can be substituted for the real thing. This general idea is not new. The most well-known educational computing environment providing this capability is the LOGO system (Harel, 1990; Papert, 1980). LOGO provides students with the capability of driving computer graphics and robotic motors to test formal hypotheses. LOGO, however, makes no explicit link to the formalisms underlying the subsequent behavior, a link that we regard as crucial to overcoming decontextualization. The most important benefit that we claim for an instructional approach that emerges from our theory is that it will help to avoid the decontextualization of algebraic knowledge by forcing students to consider formal expressions with reference to a particular situation.

Others have done theoretical work with educational implications similar to our own. Relying on extensive empirical analyses of how college students solve such problems, Hall et al. (1989) observed that model-based reasoning plays a crucial role in algebra-word-problem solving. They concluded 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. used a graphical scheme based on the work of Shalin and Bee (1985) and Greeno et al. (1986) similar in many ways to the schematic representation proposed earlier. Lewis (1989) taught students a diagrammatic representation system intended to make the formal relations of a problem more accessible to the student. This approach highlights the semantic and situational structure of arithmetic problems in a form that is independent of the consistency of the problem's language. Willis and Fuson (1988) used diagrams to emphasize the state changes taking place for the quantity of concern in the problem situation. The diagrams then helped students to select the appropriate solution strategy for common word arithmetic problems.

These studies used representations that are not in the form of an animation but are based on more abstract descriptions of situational and mathematical relations. They are "learning systems" in Nesher's (1989) sense, intermediate to the formal problem schema and to the situation itself.

A STRATEGY FOR TUTOR DESIGN: DISCOURSE COMPREHENSION THEORY AND INSTRUCTIONAL PRINCIPLES

Our tutoring approach rests on a double foundation: (a) a theory of how students understand and solve word problems and (b) a set of instructional principles. Each of these components is laid out in the following sections.

The Theory of Word-Problem Understanding: What to Tutor

As stated earlier, ANIMATE is directly based on the theory of word-problem understanding originally proposed in Kintsch and Greeno (1985) and elaborated by Reusser (1988), Cummins et al. (1988), and Kintsch (1988). Although previous work dealt with arithmetic problems, our extension of this theory to algebra, just described, is straightforward. In place of the "set" schema and its variants, we introduce a larger set of algebraic schemas, primarily the various rate schemas described (see Appendix A). We need only replace the counting strategies employed in arithmetic problems with more powerful calculational strategies, namely, 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 young students solve simple arithmetic word problems. At present it does not seem possible to formalize and empirically test our model for algebra-word-problem comprehension in the same way. To construct a simulation for word algebra, one would need a knowledge base that included an enormous stock 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 or 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, and Yost (1985) in this respect.[3] Obviously, this is a risky strategy. If the tutor fails to improve student problem-solving performance and comprehension, we will not know if the underlying theory was wrong or if we made poor system design decisions 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 *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 facilitate the construction of students' mental representations that relate the formal and the situational aspects of a word problem. We do this by making certain 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 earlier examples. Taking our earlier discussion and the results of Cummins et al. (1988) as a cue, we focus here on the task of helping students to learn the translation from the problem text into a formal, conceptual structure.[4]

With many of these steps, of course, people do not need help. We presume our students can read, for instance. When problems are familiar or the level of the problem solver's expertise is high, the transition from problem text to solution may be more direct and the relevant inferencing may seem automatic. For experts, the situation model and the problem model effectively constrain each other, so that no major gaps can arise between them, or, if they do, they are quickly corrected.[5] Experts see the world filtered through their problem models and solidly anchor their formalisms in the real world. Novices have to learn to do this. Their problem models may correspond poorly to the situational constraints of a given problem, and this lack of correspondence may go unnoticed. With the situational component of our tutor, we try to get students to coordinate their naive situational understanding with their formalizations. We are not concerned with the experts, because they do not need much help and encouragement. They can generate solvable problem representations directly from the problem text, leaving much of the conceptual structure implicit. But for the many students who need the steps spelled out, we believe that making the intermediate representations concrete and the relations among them explicit will provide the right kind of assistance.

[3]For example, some algebraic instruction makes use of *role tables*, which similarly organize the equation information, making the role of each algebraic term explicit (Anderson, 1989b).

[4]Note, however, that we do not claim to have a process-level model of problem comprehension to the same degree of detail as proposed by others (e.g., Anderson, Conrad, & Corbett, 1989; J. S. Brown & VanLehn, 1980).

[5]Although students also need help in solving the algebraic equations, we are only concerned with the construction of conceptually correct equations. Other tutors (e.g., J. S. Brown, 1985; Singley et al., 1989) exist that focus on solving equations.

The answer to our second question—how to tutor—cannot be derived solely from the process model of how students solve word problems. Numerous 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 that will guide the design of our computer-based tutor.

Instructional Principles: How to Tutor

Anderson, Boyle, Farrell, and Reiser (1984) presented several principles derived from experimental research in cognitive psychology that, they argue, are central to tutoring. There is a strong case for and widespread agreement on some of these principles, such as the importance of minimizing students' working memory loads during problem solving (cf. Polson & Richardson, 1988; Scardamalia, Bereiter, McLean, Swallow, & Woodruff, 1989). Also widely agreed on 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 help the student in task performance and skill acquisition (Glaser & Bassok, 1989).

Another principle states that feedback from a tutor needs to be immediate if students are to improve optimally (e.g., Anderson et al., 1984; Anderson, Conrad, & Corbett, 1989; Reiser, Kimberg, Lovett, & Ranney, 1989). This is based on the view that "An error comes close to being a necessary and sufficient condition for tutorial intervention" (Anderson, 1989a, p. 343) and on experimental findings showing 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.

It seems clear that, in a variety of reasonably well-constrained domains (e.g., algebraic manipulation, introductory programming, and geometry), immediate feedback is the most successful approach (cf. Anderson et al., 1989). In open-ended domains (e.g., the life sciences or the comprehension of word problems), however, there is less evidence that is compelling. For a program to adaptively customize its behavior and provide immediate feedback, the tutoring system must understand what the student is doing. One needs a detailed psychological process model of the behavior in question. With the dependence on language comprehension implicit in word-problem solving, this level of analysis is either not available or too sketchy to be of much use (Anderson et al., 1989). 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 complete psychological process model that accounts for varied reading and problem-solving styles in this domain, we have opted for a minimalist system. Our goal is the development of a computer system that aids students in problem understanding and learning. We propose an approach that assigns to the computer program tasks such as executing simple, time-varying graphics, bookkeeping, and processing formally described relations (e.g., equations). More complex, knowledge-intensive tasks such as natural language processing and inductive reasoning are left for the student. Other systems—such as EUCLID (Smolensky & Fox, 1988), a computer system for specifying argumentation, and CSILE (Scardamalia et al., 1989), a system of networked computers that supports scientific discussion—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 regarding its correctness. The system primarily helps to organize problem information around selected schemas and allows the student to see the situational correlate of specified formal relations. Such a learning environment does not try to understand in any deep way the student's actions or the problems. Yet we believe and present promising results to the effect that this is sufficient to enhance students' learning, provided it is capable of supporting students in an active learning process.

Instructional principles 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 (cf. Papert, 1980). This approach emphasizes "procedural facilitation," in which learning must be done by the students themselves and the function of instruction is to facilitate that learning (Bereiter & Scardamalia, 1989; A. L. Brown & Palincsar, 1989; Collins, J. S. Brown, & Newman, 1989). In this view, it is the *student*, not the tutor, who performs diagnosis, goal setting, and planning to give the maximum opportunity for learning. Tutors provide the facilitating structures and tools that enable students to use their intelligence and knowledge, rather than providing knowledge and intelligence to guide the learning. This view has met with success in other arenas, most notably with students using the LOGO environment (Battista & Clements, 1986; Clements, 1986; Harel, 1990). However, Scardamalia et al. (1989) also noted 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.

We draw heavily on these concepts to provide a system (described in the next section) in which the student is an active participant, not only in the problem-solving process, but also in solution assessment, error diagnosis, and recovery. We cannot provide an adequate software environment for intelligent, knowledge-based feedback and explanation for the domain of word problems. We have neither the knowledge base for this domain nor an all-encompassing task analysis.

However, by enlisting the students in the process, having them close the feedback loop for us, we expect to obtain the types of learning and problem-solving improvements ordinarily expected from intelligent tutoring systems.

THE ANIMATE LEARNING ENVIRONMENT

ANIMATE is an interactive tutor that facilitates comprehension of word problems by helping the student to construct a formal problem network, which is then used to run a simple animation of the problem. It represents our approach to providing situational support for the construction and evaluation of algebraic expressions and is greatly influenced by the work reported earlier on situation-based reasoning and on the principles for effective instruction. The detailed empirical observations of competent problem-solving behavior made by Hall et al. (1989) are very encouraging for this approach. We provide students with an environment that supports the construction of what Hall termed *model-based solutions*. Ideally, ANIMATE users will produce the iterating, simulation type of reasoning that Hall's competent problem solvers produce with paper and pencil. We additionally expect that the more facile medium of computer animation will be motivating for students.

Primarily, ANIMATE is rooted in our discourse-processing-based theory of word-problem comprehension. The major claim of this theory is that problem comprehension rests on the student's formation of a representational structure linking his or her understanding of the problem situation to a solution-enabling formalism. This permits the student to express mathematical ideas situationally and interpret events in a mathematical form.

Figure 2 sketches this linking function of our tutor. The lower portion depicts how, according to our model, algebra word problems are solved without the aid of a situation-based tutor. The text of the word problem is comprehended, and both a situational representation and a formal algebraic problem model are constructed. This step is not performed automatically by the tutor or simulated by some underlying computational model. Rather, it is a hypothesis of our psychological theory of problem comprehension that these steps are followed by the reader attempting to encode the presented information. We include it to give the entire workings of the theory. That is, the theory accounts for the transitions from a text to a set of mental representations for the meaning of the text, to a final mathematical description.

General world knowledge enters into the former process, and more specialized algebraic knowledge determines the latter. The point in our model at which decontextualization can occur is indicated in the sketch by the broken line labeled *coordination*. The tutor, ANIMATE, does two things. First, it requires the student to construct an explicit, graphical representation of the conceptual problem model (the algebraic problem schema) before deriving an equation that

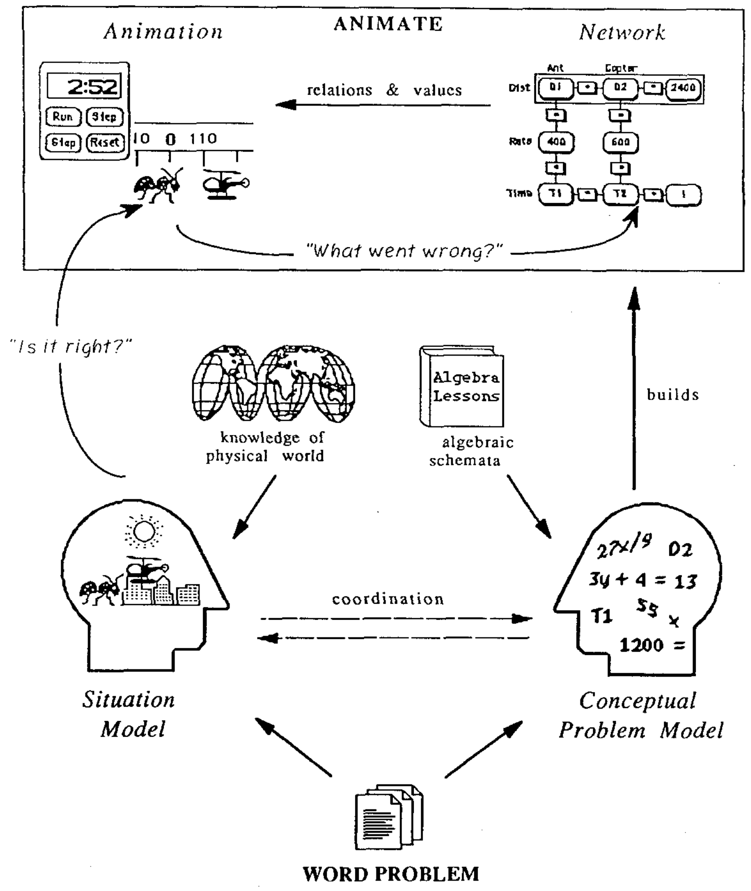


FIGURE 2 A graphical depiction of our theory of problem comprehension. From the bottom moving up, problem-relevant information and situation-relevant information are each extracted from a problem text, forming potentially isolated mental representations. Problem comprehension is achieved when these representations cohere. When a student explicitly links a problem model (such as the network) to a situation model (such as an animation), their coordination is facilitated. Students manipulate the network to produce changes in the animation until the animation matches their expectation of the problem situation.

will actually solve the problem. Later we discuss the benefits expected from introducing this intermediate step. Second, by using an animation to 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 or not the problem was conceptualized correctly (the "Is it right?" link of Figure 2). That is important. But equally important is establishing the link between the concepts and structures in the formal model and the events in the animated world (top arrow), so that initial problem comprehension will prevent students from making certain errors or omissions in the first place (Nathan & Young, 1990).

The animation serves as the externalization of a student's situation model. It is not the case that any externalization will do. According to the theory, the events must be represented in a form that allows students to operate on the situation model and to go through a process analogous to actually manipulating or observing the physical objects themselves. Animations of the kind used in our tutor have this feature of being analogous to real-world events.

Any unexpected behavior in the animation can then be traced back to the formal problem representation along the student's causal link structure, addressing "What went wrong?" and further strengthening the situation–equation correspondence. Changes to the algebraic network make something happen in the animation; so the animation provides a context for the formalism and grounds it in a depiction of reality. It is a necessary assumption of our theory that students are capable of understanding the stories given and that they form an appropriate situation model. Without this, subjects would be unable to evaluate the feedback given by the system. They would be looking at two potentially meaningless representations – a set of abstract symbols and an unfamiliar animation – and their coupling, no matter how extensive and explicit, would not teach them anything about the situational meaning of mathematical expressions.

There are other tutoring systems with similar goals, notably the Envisionment Machine (Roschelle, 1987), and the TRIP tutor (Gould & Finzer, 1982). The Envisionment Machine, like ANIMATE, supports multiple representations. One representation simulates physical situations, whereas the other provides a manipulable formal problem schema (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 mathematically legal and follow all known constraints.

The TRIP tutor (Gould & Finzer, 1982) 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, no attempts were made to base TRIP on an explicit theory of problem comprehension. Furthermore, it is the teacher in TRIP who provides the criteria for correctness of the constructed picture, whereas ANIMATE exploits the *student's* ability to assess this (Nathan, 1990).

ANIMATE also provides the student with greater flexibility in the construction of situation and problem models. TRIP contains explicit knowledge of each problem assigned (distances, rates, and travel times), and students can only enter values that are correct for that problem. The animation in TRIP 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, partial and even incorrect problem model networks can be run. This contributes to the exploratory nature of ANIMATE.

ANIMATE 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 that employ the $D = R \times T$ equation and include "overtake," "collision," and "distance apart" problems. Other areas of instruction, such as "compound interest," planetary physics, and computer programming, are under investigation, but we do not expect the approach outlined here to address all topics in the current educational curriculum. We feel, however, that the *principles* guiding this approach are fundamentally beneficial to many students and extendable to a variety of domains often left out of the realm of Intelligent Tutoring System development.

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, the interface reduces working memory load by only allowing legal commands. Although 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.

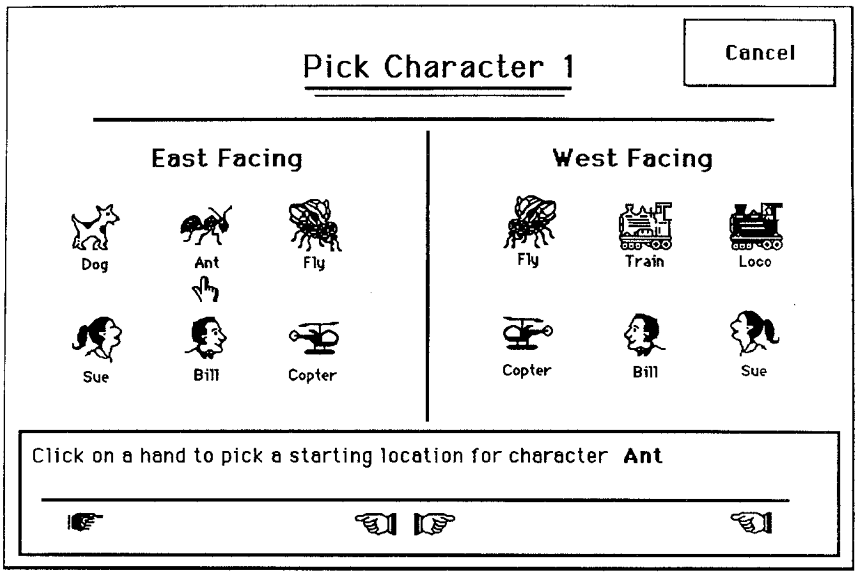
An effective way to describe the functioning of the system is with an example of how a student would use ANIMATE to solve a specific algebra problem. We assume that the student has been given instructions to produce the equations necessary to solve the following problem:

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 p.m., what time will the helicopter and the ant collide (ignoring any time changes)?

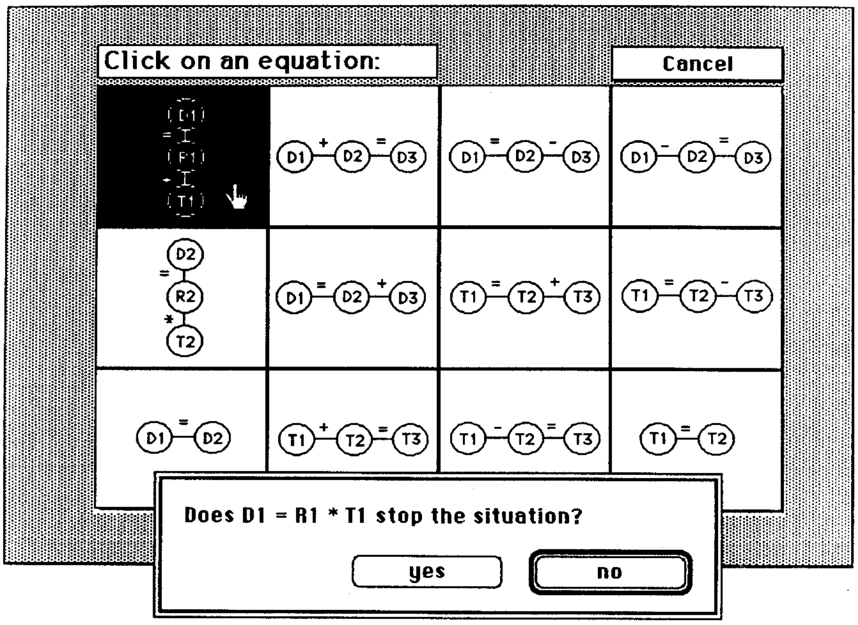
The student decides on an initial goal of describing the movement of the ant, pushes the button *Pick Character 1*, and is presented with pictures representing the possible characters (Figure 3). After choosing the ant, buttons appear at the bottom of the screen allowing the student to select a pointing 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.

To specify the motion of the ant, the student must build and customize a problem schema. Pushing the *Equations* button causes a palette of suggested network equations to appear (see Figure 4). The equation palette provides the student with



Click on a hand to pick a starting location for character **Ant**

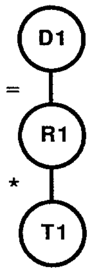
FIGURE 3 Pushing the *Pick Character 1* button presents a selection of characters for playing out the situation on the computer screen. Students select first a character and then a screen location/direction arrow.



Does D1 = R1 * T1 stop the situation?

FIGURE 4 Problem model construction is supported by an equation palette that presents network components that are then linked together on the screen.

small schema-building blocks to organize problem information rather than requiring the student to work only at the level of entire schemas (for justifications of this approach, see Kintsch, 1988; Schank, 1984). The student chooses



and the first $D = R \times T$ 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 representation and the situation-based goal, because real events need a specified end to be properly described. This type of behavior, where problem solvers are explicitly addressing the boundary conditions of the underlying situation of a problem that is described mathematically, has been identified with competent problem solving (e.g., Hall et al., 1989). The student then selects the rate node (*R*1) of the network and, in accordance with the information of Problem 2, uses the calculator to enter the number 400 (see Figure 5).

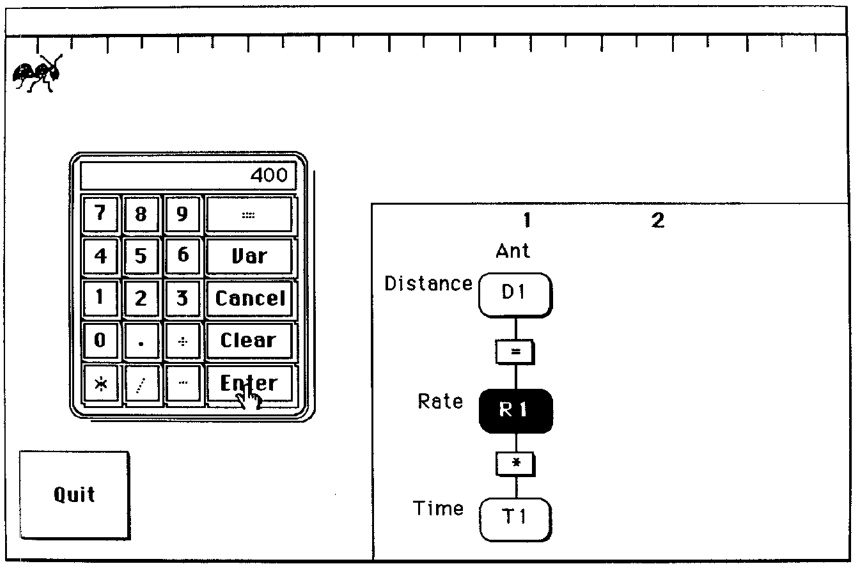
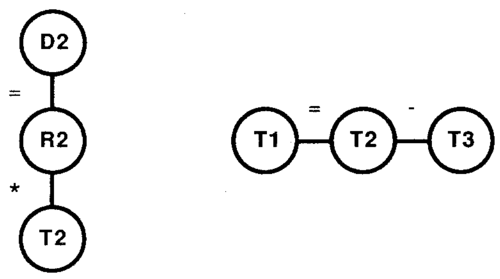


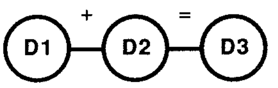
FIGURE 5 Values and variables are entered into a selected node by way of a calculator interface. Here, the rate of the ant given in Problem 2 (400 mph) is being entered.

To check one's progress, the student pushes the *Run* button, which first checks the problem schema for algebraic and network errors. The student could check for problem schema errors without 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 mph. The animation, 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 schema. 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 schema.

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. Conceptually, this last equation is a more appropriate choice as a stopping condition than, say $D1 = R1 \times T1$, because it emphasizes the situational nature of the story—characters traveling a certain combined distance—rather than a relation that is static and true throughout the problem situation.

The horizontal equations are supporting relations. As described earlier, they specify the relative behavior of the two characters and are often based on inferences made by the problem solver. Our tutoring approach, by emphasizing the situational nature of the problem and its tie to algebraic schemas, is intended to help the student generate these inferences, specify them mathematically, and diagnose them.

The student enters the remaining values given in the problem, putting 2400 in the *D*3 node to show the characters' initial distance apart and 600 in *R*2 as the helicopter's rate. From the text, the student knows that the helicopter leaves 1 hr after the ant and so mistakenly believes that this delay (*T*3) should be *subtracted* from the helicopter's time. The student knows also from the text that the ant leaves at 2 p.m. and so erroneously enters 2 for $T1$. Because the helicopter leaves 1 hr 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 ($400 * 2$, $600 * 3$ for each character, respectively). 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, the vertical $D = R \times T$ equations would be incorrect. The student decides to ignore the error and, with no further errors found, pushes the *Run* button.

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 uses one of two methods to alter this expression. The student may use the mouse cursor to change the "−" operator directly to a "+" using the

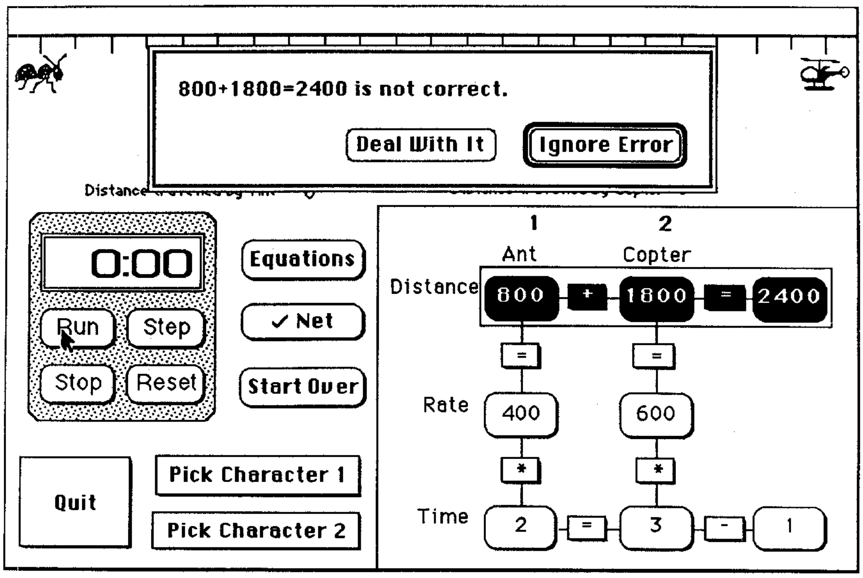

800+1800=2400 is not correct.

FIGURE 6 A fully developed problem model, including characters and their associated values and relations. The distance equation is boxed, indicating its role as the stopping-condition for the animation. The tutor detected an error in the distance equation that the student chose to (explicitly) ignore.

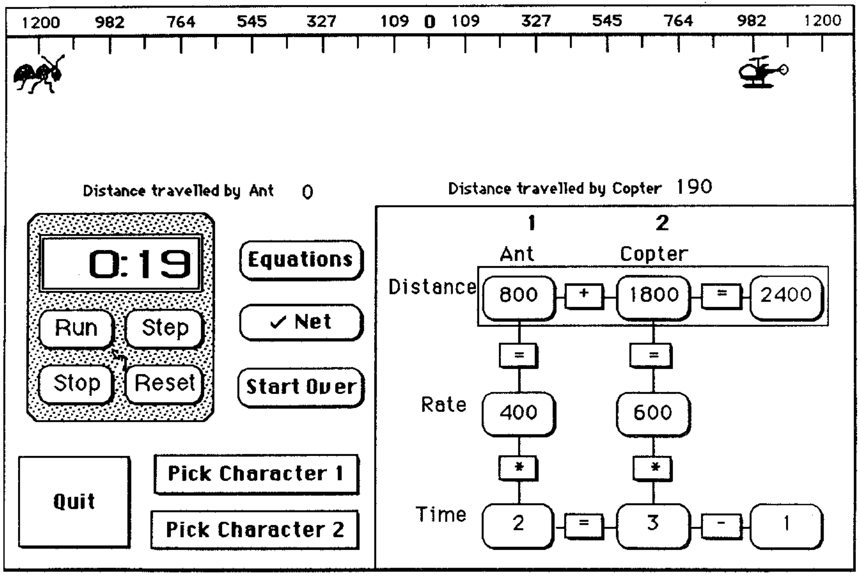
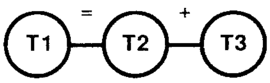


FIGURE 7 After 19 sec, only the helicopter has moved—behavior specified by a negative delay value in the time equation—in contrast to the situation of Problem 2. The erroneous distance equation is still present in the problem model network, the student having chosen to ignore it.

calculator, just as values and variables are changed in the ovals. Or the student may pick the equation



Having executed either of these two methods, 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. Because 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 equations, verified syntactically and situationally, are taken from the subgraphs of the net.

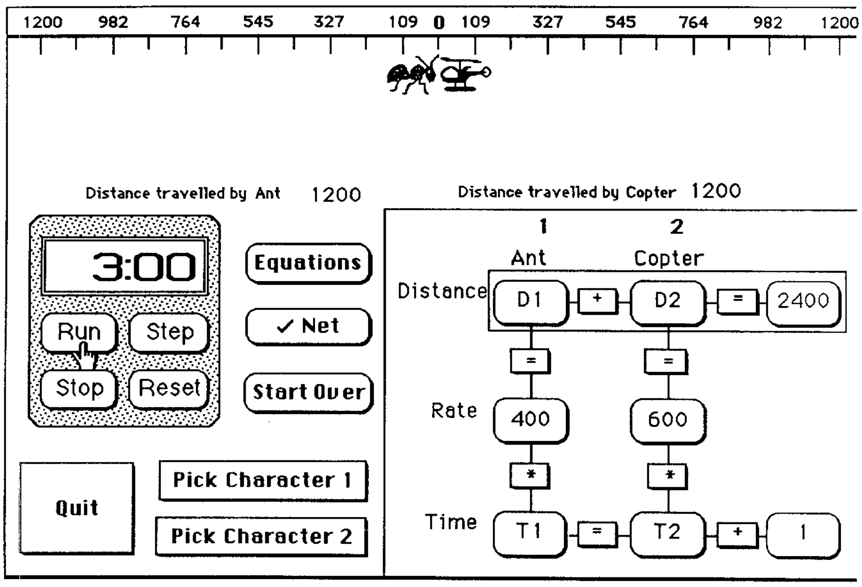


FIGURE 8 The animation is run on a debugged network. It shows a collision situation in accordance with expectations. Time and distance gauges show solution values, whereas the network contains the problem formalism. The final algebraic equations can be read from the network horizontally and vertically.

$$
\begin{aligned}
D1 &= 400 \times T1 \\
D2 &= 600 \times T2 \\
D1 + D2 &= 2400 \\
T1 &= T2 + 1
\end{aligned}
\qquad \text{(Equation Set 2)}
$$

The preceding 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 on 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 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 if the animation matches the problem representation and so answers the question, "Is It Right?" A mismatch suggests how the student may alter the problem schema 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.

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 also 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 as 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 that move in specified directions, the clock, the ruler, and other gauges.

PRELIMINARY EMPIRICAL SYSTEM EVALUATION

Is ANIMATE a good tutor—better than traditional classroom instruction, perhaps even approaching an experienced human tutor? Even more important, does ANIMATE truly help the student bridge the gap between real-world situations and formal operations? Is it a way to avoid decontextualized learning, as the underlying model claims? 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 the ANIMATE system or 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 our preliminary results are of interest.

In a study by Nathan (1988), subjects with high-school-level algebra experience were given limited training using the graphical network approach of Figure 1 to solve distance–rate–time problems. After only a 20-min 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 a 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 before, as an intermediate representation between the cover story and the underlying solution equations. Weaver and Kintsch (1992) showed that subjects rated problem pairs significantly more similar when similarity was based on schema structure (i.e., the intermediate level) than when based on the form of the underlying equations. Given the widespread finding that subjects are very poor at detecting structural similarities among such problems (e.g., Reed, 1987), Weaver and Kintsch (1992) argued that this may be due in part to the way we define *structure*. Subjects are indeed sensitive to the conceptual structure of a problem although not to its equation form.

Like the two studies already mentioned (Nathan, 1988; Weaver & Kintsch, 1992), the experiment that follows is limited because of the extremely brief training period and because problems of only one type (distance–rate–time) were used. The experiment was designed to explore ways to facilitate problem comprehension and to give meaning to the abstract symbols used for formal problem solving. For that purpose, the performances of four different problem-solving treatment groups on a variety of problems were compared with reference to our theoretical predictions. We restate them briefly here.

Prediction 1 focuses on the types of errors students are expected to make when there is no firm foundation for relating their mathematical knowledge to their situational understanding. Prediction 1a states more specifically that students without this foundation will tend to omit from their solution protocols expressions that are inferable but not immediately present in the textbase. In our terms, these errors will appear as omissions of uncued supporting relations. Prediction 1b holds that, when students subsequently learn to relate their mathematical knowledge to their situational understanding, we witness a large decrease in the frequency of omissions and misspecifications of these inference-based relations.

Prediction 2 expresses our expectations that the animation-based feedback will be an effective aid in the generation of solution-enabling equations for word problems. We expect this advantage to continue, even when the feedback is removed. A like advantage is not expected merely from experience with the network method. Prediction 3 similarly maintains that the ANIMATE learning environment will effectively support the situational interpretation of formal equations and that this advantage will also continue when solving problems without the aid of the tutoring system; that is, a lasting change in these students' problem comprehension strategies will occur.

Predictions 4a and 4b address the ability to recognize the appropriateness of a set of equations to a situation. We predict that the situation-based reasoning encouraged by the ANIMATE system will support greater discriminability of equations in its users than in those who work only with equation-based structures (4a). We further expect that the feedback component, which supports error diagnosis and correction, will enhance students' performances in reformulating erroneous and mismatched equations (4b).

In all, we hypothesize that students who exhibit early difficulties in problem solving will benefit from exposure to an environment that explicitly relates one's situation model to the formal problem model. This will affect students' problem-solving behaviors and comprehension processes and will continue to benefit these students when they solve problems outside the tutoring environment.

METHOD

Subjects

Fifty-six undergraduate students from the University of Colorado participated in a 2-hr experiment as part of their course requirements for Introduction to Psychology. All had a background in algebra through the high school level, and in a postexperiment interview, all reported familiarity with the kinds of problems given.

Design and Materials

Subjects were randomly assigned to one of four treatments—equation, stopping condition, network, or animation—and were tested in groups of 14 students at a time to mimic a classroom setting. All groups received identical pretests, an experimentally manipulated training session that included a review of algebra word problems and three training exercises, and identical posttests. Examples of the materials provided in each of the experimental tasks are presented in Appendix B. During the training, subjects in the network, stopping-condition, and animation groups additionally used tutoring programs that ran on Apple™ Macintosh computers, distributed one to each subject. Thus, the experiment was a 4 (Treatment, between subjects) × 3 (Problem Type, within subject) × 2 (Time of Test, between subjects) repeated measures design.

The tutoring programs varied in functionality. The ANIMATE tutor, with animation causally linked to the equation network, as described before, was used by the animation treatment group. The stopping-condition group used a tutor closest in functionality to the ANIMATE system. Subjects set up an equation network and a situation on the computer screen and selected an equation that, when true, was to stop the animation. Stopping-condition subjects, however, were not able to "run" an animation to receive situation-based feedback on their equations. They set up a static depiction of the situation, and their only feedback was at the level of the algebraic self-consistency of the network. The network-only tutor did not require selecting a stopping condition or setting up a situation. With this tutor, subjects constructed an algebraic network in a computer environment with an interface that was otherwise similar in layout to the animation and stopping-condition tutors. The functionality of the various tutors is summarized in Table 1. The network-only and stopping-condition tutors can be grouped together as

TABLE 1
Differences in Functionality of the Tutors Used in Experiment 1

Tutor	*Network Formalism?*	*Set Up Problem Situation?*	*Run Animation?*
ANIMATE 1.0	Yes	Yes	Yes
Stopping condition	Yes	Yes	
Network only	Yes		

nonanimation tutors, whereas ANIMATE and the stopping-condition tutor can be viewed as (differing levels of) *situation-based tutors*. The three "tutor" conditions—animation, stopping condition, and network—were designed to tease apart effects due to the different aspects of constructing the network, setting up a situation, and running an equation-driven animation on the computer. Subjects under the equation treatment constituted the control group, controlling for any improvement that may have resulted from a review of the algebraic concepts and repeated exposure to the algebra word problems.

The paper-and-pencil pretest and posttest consisted of four distance–rate–time problems. Two were of the *solve* variety, where students were told to write down an equation or set of equations that could be used to solve each problem. Subjects were told that they did not need to do the algebraic manipulation that would lead to a numerical answer. This problem type was intended to test a subject's ability to generate a (set of) mathematical expression(s) that describes a situation presented in the form of a story. This ability to translate a qualitative, informal representation into a solution-enabling, formal representation is seen as a crucial part of word-problem comprehension. It also serves as an operational test of Prediction 2.

As a test of the complementary skill and Prediction 3, we gave subjects a set of time and distance equations with no accompanying story line. Subjects were directed to write a short story to serve as a legitimate word problem for the given equations. We refer to this as the *story* problem and use it to ascertain a subject's ability to extract and report the situational interpretation or "meaning" of a formal representation. Here, subjects' understanding of the mathematical symbols is assessed by their ability to express them in an alternative form, specifically, in the language of simple events.

Finally, subjects were instructed to examine and, if necessary, correct potentially flawed equations that appeared with a word problem. Only errors in supporting relations were introduced, because earlier studies revealed subjects readily recognized the form of the governing $D = R \times T$ equation. The erroneous equations described situations not present in the word problems given. This is referred to as the *debug* problem. Its intended value is as a partial test of Prediction 4, examining subjects' abilities to evaluate the appropriateness of a formal expression to a referent situation and, if necessary, to alter the expression as the situation demands.

The training exercises made use of the solve and debug problems but not of the story problem. Instead, a multiple-choice task—the *choice* problem—was given in which three possible solutions appeared along with a word problem statement. Students had to select the correct solution set from among equations describing other situations. Although there is clearly a much greater element of chance in this problem type than in the other three, this problem tested subjects' discrimination ability of formal expressions, further testing Prediction 4a.

Group Training

Subjects all participated in a review session that presented the basic principles of the $D = R \times T$ equation, its physical and mathematical interpretations, and its application to several representative examples of the solve variety of word algebra problems. Whereas subjects in the equation group reviewed the strictly algebraic approach of deducing equations directly from the problem texts without the use of intermediate structures, the three tutor groups learned to use first the graphic-based network approach presented earlier (see Figure 1). Equations were then derived by copying down collinear subgraphs from the network. The network approach was taught first by demonstration and then by practice on the respective computer tutor (ANIMATE, the stopping-condition tutor, or the network-only tutor).

Procedure

Fifty-six subjects organized into four groups of 14 each were pretested on their knowledge of algebra. The pretest consisted of four problems: two of the solve variety, one debug, and one story. Subjects then had either a brief review of traditional algebra (control group) or a 30-min tutorial on the network method. Subjects next solved three training tasks—a solve, a debug, and a choice problem—using the method introduced in their tutorial. Those working with a computer tutor used either the complete ANIMATE computer program, the stopping-condition tutor, or the network-only tutor. Finally, all subjects took an identical posttest, with a format identical to that of the pretest, administered without use of the computer tutors. An independent experiment revealed that the pretest and the posttest were of comparable difficulty, $F(1, 31) < 1$.

RESULTS

Scoring

Pretest and posttest scores were measured by performance on two solve problems, one debug problem, and one story problem, for a maximum score of 4 points on each test. Training task score was measured by performance on a debug problem, a solve problem, and a choice problem, for a maximum score of 3 points. Written stories had to describe completely the equations provided to receive 1 credit. Likewise, multiple-choice problems received a score of either 1 or 0. Partial credit was assigned for all other problems involving the generating or correcting of formal expressions (equations or networks). Full credit was determined by the number of relations needed to specify the solution, so that an experienced problem solver could, through algebraic manipulation, produce the correct numerical answer for the quantity requested in the problem. In a typical "overtake" problem, students could provide two correct distance–rate–time equations, for example, but fail to specify correctly the "equal-distance" or "relative travel-time" (i.e., delay) supporting relations and still receive partial credit of a ½ point. Equations could be specified separately (as in Equation Set 2) or as a single expression with proper substitutions made (as in Equation Set 1).

Error Analysis

An analysis of the types of errors produced and of their relative likelihood indicates more specifically *how* each treatment affects student performance. We identify two classes of errors, those by omission and those by specification or form.[6] An error of *omission* was coded when there was the absence of a needed formal relation, such as a relation linking travel times (e.g., $T1 = T2$) or distances. An error of *specification* or *form* was coded if a relation between terms was present in students' solution protocol but that relation was inconsistent with the situation described by the problem text. Along another dimension, errors may be made with either (a) governing equations (i.e., $D = R \times T$, a *DRT error*) or (b) inference-based supporting relations. Thus, for the analyses to follow, we rely on four categories of errors. An example of an error of specification for a governing relation commonly made in subjects' pretest solution protocols is the expression $R/T = D$. These relations are cued by the overall problem schema associated with the problem texts (cf. Kintsch & Greeno, 1985). A common specification error for supporting relations is that $D1 = D2$ for a collision problem (such as Problem 2 shown earlier). An omission error was coded if constraints such as these were left out of a subject's solution protocol entirely.

The frequency of errors for each of the four categories is broken down by treatment and presented in Table 2 (for pretest) and Table 3 (for posttest). The *subtotal* row contains the sum of all classifiable errors, that is, errors that fall into one of the four categories just defined. The *total* row is the sum of all classifiable and unclassifiable errors and represents all errors made during problem solving. These tables exclude any contribution from debug problems, because the form in which these problems were presented heavily biased the data. In particular, debug problems never included an erroneous form of the governing

[6]This class of errors is similar to the "errors of commission" referred to by Hall et al. (1989).

TABLE 2
Frequency of Pretest DRT and Supporting-Relation Errors for All Treatments (Excluding Debug Problems)

	Treatment				
Error Type	*Animation*	*Equation (Control)*	*Network*	*Stopping Condition*	*Total*
DRT					
Form	20	11	10	17	58
Omit	7	11	3	7	28
Total	27	22	13	24	86
Supporting relations					
Form	18	14	24	17	73
Omit	13	14	14	19	60
Total	31	28	38	36	133
Subtotal	58	50	51	60	219
Unclassified	10	2	5	0	17
Total	68	52	56	60	236

equation ($D = R \times T$), only errors in the supporting relations. Also, these problems always included some form of a supporting relation, which biased subjects away from making errors of omission.

Table 2 reveals that the greatest number of pretest errors involved inference-based supporting relations, a result that our model explicitly predicts (Prediction 1). In the 236 errors made by all subjects during the pretest, 17 were unclassi-

TABLE 3
Frequency of Posttest DRT and Supporting-Relation Errors for All Treatments (Excluding Debug Problems)

	Treatment				
Error Type	*Animation*	*Equation (Control)*	*Network*	*Stopping Condition*	*Total*
DRT					
Form	0	5	7	2	14
Omit	1	5	6	10	22
Total	1	10	13	12	36
Supporting relations					
Form	12	19	20	10	61
Omit	5	26	23	14	68
Total	17	45	43	24	129
Subtotal	18	55	56	36	165
Unclassified	1	0	1	1	3
Total	19	55	57	37	168

fied. Of those classified, 133 (or about 61%) were labeled as errors pertaining to uncued supporting relations, and 86 (about 39%) were errors concerning governing relations (i.e., $D = R \times T$ schemas). Using the frequency data of Table 2 to determine the likelihood for subjects to have made certain types of errors, we found that, from the 133 supporting relation errors, it was equally likely with a 95% confidence interval for subjects to have made errors of omission (73, or 55% $\pm$ 8.5%) and of improper form specification (60, or 45% $\pm$ 8.5%).

Students tended to include $D = R \times T$ relations in their pretest performance but had difficulty in specifying these correctly. Form errors and omissions at pretest time were equally likely in the control group (which produced 50% of each kind of error), whereas errors of form, 28 in all, were more likely for all other groups. Tutor groups (with a range of 11% to 25% and an average of 20% $\pm$ 15%) were as likely as control group subjects (with a 39% $\pm$ 17% likelihood) to omit $D = R \times T$.

In contrast to their pretest performance, students on average made fewer errors with governing relations during the posttest. Of the 384 classifiable errors made in both tests (219 in the pretest and 165 in the posttest), 86 or 22.4% were pretest DRT errors. This percentage dropped dramatically at posttest, when only 36, or 9.3%, concerned DRT expressions. This can be attributed largely to the exposure students had during training when they encountered many properly specified $D = R \times T$ equations. Even though we made no predictions concerning the relative likelihood of erring on governing equations, it is worthwhile comparing the network, stopping-condition, and equation groups, which were equally likely to make DRT errors at posttest. Table 3 shows that, of the 165 classifiable posttest errors made by all subjects, stopping-condition subjects showed a 7.3% $\pm$ 4% likelihood of making DRT errors (with a frequency of 12), which was not reliably different at the 5% level from either the network group (with 7.9% $\pm$ 4%, and a frequency of 13) or the control group (with 6.1% $\pm$ 3.6%, and a frequency of 10). Although tutor users were always presented with correct governing relations through the "equation palette" (Figure 4), this did not give them any advantage over the free-form equation group. The group that stands out in its performance is the animation group, which demonstrated a substantially lower probability of making DRT errors (with a likelihood of only 0.6% $\pm$ 1%, and a frequency of 1, which is not significantly different from zero).

Our theory makes some very clear predictions about the probability of making errors with supporting relations after exposure to the different treatments. Prediction 1a holds that those who used situation-based tutors – the animation and stopping-condition groups – should be expected to have a substantially lower likelihood of omitting inference-based supporting relations. Prediction 1b states further that those who explicitly relate their situational understanding to their mathematical knowledge, as animation students have done, should exhibit the greatest decline in their misspecification of these relations.

A comparison of the posttest data of Table 3 and the pretest data of Table 2 shows that the probability of making a supporting-relation error at posttest increased for those groups that did not use a situation-based tutoring environment. The control group experienced increases from pretest to posttest in both errors of specification (by some 35%, from 14 to 19) and errors of omission (by more than 85%, from 14 to 26). Network subjects were similarly more likely to omit supporting relations at posttest, increasing 64%, from 14 errors at pretest to 23 errors at posttest. By contrast, stopping-condition and animation tutor users showed substantial drops in the likelihood of omitting these inference-based equations, a result that supports Prediction 1a. The stopping-condition group went from 19 to 14 errors, a 26% drop, whereas the animation group went from making 13 errors at pretest to 5 at posttest, a 61.5% decrease. Students in the animation group, with 5 errors and a 3% ± 2.6% probability of omitting supporting relations, were less likely than students in any of the non-situation-based conditions to omit a supporting relation from a solution, but not reliably different at the 5% level from the 14 errors and 8.5% ± 4% probability of the stopping-condition group. The network and control groups, with a 14% ± 5% probability and a 16% ± 5.6% probability of making this error, respectively, did not make substantially more errors than the stopping-condition group.

Although these results support the problem model, this last finding runs counter to the expectations expressed in Prediction 1a. The stopping-condition group showed serious difficulty in generating supporting relations. Their pretest performance was poorest in this arena, and, although they showed substantial improvement, they failed to perform better than the network and control groups at posttest. The omissions made by the stopping-condition group were, predictably enough, not for stopping-condition expressions but for expressions regarding the relative starting times of characters (i.e., delay equations), for which they received no obvious advantage.

When stopping-condition subjects did include supporting relations in their protocols, however, they tended to get them right for the given situation. They also showed the greatest decrease in making errors of formulation for supporting relations, some 41%, dropping from 17 errors to 10 errors. The animation group demonstrated comparable improvement, dropping 33%, from 18 to 12 errors. The network group experienced a less extreme decline, 17%, in the frequency of these errors, whereas, as noted previously, the control group had a marked *increase*. These improvements are partially consistent with Prediction 1b. We anticipated that exposure to the ANIMATE environment would improve students' ability to formulate situationally inferred expressions more effectively than any other treatment. Apparently, the nonanimation tutors, particularly in the stopping condition, helped substantially in this regard. Yet, even with these large improvements and a 2:1 difference in error frequency between the most error-prone network group (with a 12% ± 5% probability of erring) and the least error-prone stopping-condition group (with a 6% ± 3.6% probability), there were no reliable group differences (at the 5% level) in the likelihood of making these errors.

These findings largely support our theoretical predictions (1a and 1b) for the role that a situation-based environment can play in improving the generation and formulation of a formal problem model. Exposure to the ANIMATE environment clearly helped students to generate the necessary inferences for a solution; but a similar advantage was not consistently afforded by the stopping-condition tutor.

Pretest and Posttest Performance

We first consider the results of the pretests and posttests. Because these two tests were in every respect identical for the four groups of subjects, they provide the fairest measure for the overall effectiveness of the treatment. Table 4 shows the results of these tests, as well as subjects' performance on the three training tasks for the four experimental treatments. For each treatment group, we see its size (*n*), the set of problem types, and the proportion correct for each type of problem when it appeared in the pretest, the experimental training, and then the posttest. For training tasks, choice problems were used instead of story-type problems; their measures appear in square brackets to highlight this substitution. Data for each treatment include a row describing the test or training task totals for that treatment. The solve figures reported for pretest and posttest are obtained by averaging subjects' performance on two solve problems. Thus, pretest and posttest totals are obtained by adding *twice* that figure to the reported score for debug and story problems, resulting in a maximum of 4 test points. Totals for the training tasks are computed by adding the proportion correct for each of the solve, debug, and choice problems. The last section of Table 4 reports the mean performance for all treatment groups together on each of the problem types.

A one-way within-subject analysis of variance (ANOVA) with test score as the dependent measure produced a highly significant effect of time of test, indicating that, overall, subjects improved from pretest to posttest, $F(1, 52) = 47.7$, $p < .0001$, $MS_e = 7.7$. A two-way ANOVA with treatment group as a between-subjects variable, time of test (pretest vs. posttest) as a within-subject variable, and solution performance as the dependent variable showed a significant Time of Test × Treatment interaction, $F(3, 52) = 4.5$, $p < .01$, $MS_e = 0.7$. This indicates a difference in the effectiveness of the training for the four groups.

Post hoc comparisons showed the particular differences responsible for the ANOVA results. A Newman–Keuls test comparing improvement scores indicated that ANIMATE and network-only tutor users improved significantly ($p < .05$) more than those in the control group (57.5% and 30%, vs. a 9% improvement, respectively). ANIMATE users also improved significantly more than nonanimation tutor users: the stopping-condition and network-only treatments. The mean performance improvements of the nonanimation tutor users did not differ significantly from each other, and although the stopping-condition group

TABLE 4
Mean Proportion Correct (and Standard Deviations) on Pretest, Training, and Posttest Problems for Each Treatment

			Performance Scores		
Treatment Group	*n*	*Problem Type*	*Pretest*	*Training*	*Posttest*
Equation (control)	14	Solve	.62 (.25)	.57 (.39)	.66 (.30)
		Debug	.21 (.32)	.50 (.39)	.57 (.51)
		Story/[Choice]	.29 (.47)	[.43 (.51)]	.21 (.43)
		Total[a]	1.74 (.78)	1.5 (.90)	2.1 (.93)
Animation	14	Solve	.21 (.38)	.89 (.13)	.85 (.17)
		Debug	.11 (.29)	.75 (.38)	.64 (.50)
		Story/[Choice]	.21 (.42)	[.93 (.27)]	.71 (.47)
		Total[a]	.74 (1.2)	2.57 (.48)	3.05 (.84)
Stopping condition	14	Solve	.48 (.35)	.82 (.28)	.72 (.31)
		Debug	.39 (.36)	.43 (.27)	.54 (.50)
		Story/[Choice]	.07 (.27)	[.50 (.52)]	.43 (.51)
		Total[a]	1.42 (.82)	1.75 (.74)	2.41 (1.2)
Network	14	Solve	.38 (.46)	.84 (.19)	.71 (.32)
		Debug	.18 (.29)	.61 (.40)	.64 (.50)
		Story/[Choice]	.07 (.27)	[.36 (.50)]	.14 (.36)
		Total[a]	1.01 (.97)	1.81 (.87)	2.2 (1.1)
All groups	56	Solve	.42 (.39)	.78 (.28)	.74 (.28)
		Debug	.22 (.32)	.57 (.27)	.60 (.49)
		Story/[Choice]	.16 (.37)	[.55 (.50)]	.38 (.49)
		Total[a]	1.22 (1.0)	1.9 (.85)	2.46 (1.1)

[a]Total pretest and posttest scores are computed by adding two solve problems (twice the reported proportion), one debug problem, and one story problem. Percentages reported for total test scores are out of 4 points total. Total training task scores are computed by adding one solve problem, one debug problem, and one choice problem. Percentages for total training task scores are out of 3 points.

improved more than the control group, this difference was not significant at the .05 level.

These results must be tempered by the performance of the equation group on the pretest, which, in a one-way ANOVA with test scores as the dependent measure, was found to score significantly higher than the average of the three tutor groups, $F(1, 52) = 4.3, p < .05, MS_e = 0.7$, whereas the other treatments did not differ from each other. Although this difference can only be attributed to random effects in the selection of subjects, because the pretest preceded even the algebraic review, it does affect the interpretation of these results somewhat. Nevertheless, even 30 min of training with ANIMATE led to more than six times the improvement of standard procedures for solving problems in the word algebra tests we gave. Furthermore, the performance improvement of the animation group was greater than that of the other tutoring groups. This finding directly supports the view that exposure to situation-based feedback for equations serves as an effective learning tool for algebra students, its effects lasting beyond the removal of that feedback.

One could easily suppose that, after more than a thousand hours of school-based practice with standard algebraic methods, a brief session with this new approach might only serve to confuse the students. Instead, the exposure to the network approach proved quite helpful on the posttest, where problems were solved without the help of the tutor. Although our finding that network students improved more than control group subjects is counter to our theoretical expectations and to previously reported results (Nathan, 1988), this must be tempered with the understanding that the sheer novelty of a computer-based training environment for mathematics may have advantages of its own. Our interpretation is that there is an attentional advantage incurred from use of the network-only tutor for mathematics problem solving. This made the mathematics review more salient to students and likely increased their motivation to learn and perform.

Improvements can be specifically ascribed to posttest performance on the solve and story problems. Two solve problems were given in the posttest. Because no reliable differences were found between the two problems, all subsequent analyses were based on their combined scores. Statistical analyses revealed a main effect of problem type, $F(2, 104) = 12.9, p < .001, MS_e = 1.9$, with no interaction with time of test. A Newman–Keuls post hoc comparison ($p < .05$) revealed that solve problems were easiest at pretest and continued to be so at posttest. A postexperiment interview revealed that this was a familiar task for all the students.

A one-way between-subjects analysis of covariance (ANCOVA) was conducted, with posttest performance on the average score of the two solve problems as the dependent measure, subjects' pretest score as the covariate, and treatment as the between-subjects variable. This test revealed a performance difference as a function of treatment, $F(3, 52) = 2.9, p < .05, MS_e = 0.32$. A subsequent post hoc comparison of means adjusted for pretest performance (Judd & McClelland, 1989) revealed that this significant difference was due to differences between the animation group and each of the other groups, as indicated by the Newman–Keuls critical difference test ($p < .05$). This finding supports Prediction 2 that the animation-based feedback is an effective aid in generating solutions for word problems, even when the feedback is removed.

Further post hoc analyses were performed on data from the control group, comparing those who used pictures to solve or set up the various problems with those who did not. This comparison is important because it may reveal that some members of the equation group naturally apply some of the situation-based principles incorporated in the ANIMATE tutor. A contrast-coded factor (Judd & McClelland, 1989) that encoded the use or absence of pictures as part of the solution process was used. If picture use had a reliable effect, the mean performance of picture users would differ significantly from that for subjects who did not use pictures. This factor did not reveal any new main effects or interactions.

This finding is consistent with earlier work by Mayer (1982). As in Mayer's study, picture drawing received no feedback for correctness or appropriateness. Further, there was no compelling reason for control subjects to make sure that the picture fit with the formal expressions. In contrast, the equation-driven animation in the ANIMATE program allows students to "scaffold" from animated pictures to a mental representation having a strong tie to the formal equations.

There were no measurable differences in problem-solving performance attributable to treatment for the debug problem. Our initial hypothesis (Prediction 4a) was that subjects in the animation group, by receiving advantageous feedback regarding the situational inappropriateness of the flawed equations, would learn to constrain their search for the detection of errors. In addition, we expected that an improved ability to interpret erroneous equations, through practice in the ANIMATE learning environment, would enhance subjects' ability to correct these found errors (Prediction 4b). Although improvement was greatest for the animation group, this difference was not significant at the .05 level.

The overall correct performance on the posttest for the debug problem was 60%, with only 64% of the highest scoring groups (network and animation) solving it correctly (see Table 4); so the ceiling on performance was not nearly achieved. The analyses indicate that subjects in all treatment groups performed comparably on the debug problem at pretest time, $F(3, 52) < 1.0$, and continued to do so at posttest. A post hoc test comparing students' behavior in this problem reveals, however, that animation subjects were significantly more likely than any other treatment group to correct a buggy equation once they detected it, $t(52) = 2.24, p < .05, MS_e = 1.0$. This suggests that, although the learning environment may not support transfer of superior error-detection skills (owing perhaps to the role the animation plays in highlighting this for the student), it does aid them in respecifying erroneous expressions. This latter finding provides partial support for Prediction 4b.

For the final problem, a one-way between-subjects ANCOVA was performed with posttest performance on the story problem serving as the dependent measure, pretest score as the covariate, and treatment as the between-subjects variable. This analysis gave a significant main effect of treatment as a predictor of performance in solving the story problem, $F(3, 52) = 2.6, p < .05, MS_e = 0.18$. A subsequent post hoc comparison using the Newman–Keuls test revealed that the animation group performed significantly higher than the network and control groups ($p < .05$) but not higher than the stopping-condition group. The stopping-condition group was found, however, to perform no better than the network or control groups.

This result largely supports Prediction 3. This story-writing problem is perhaps the most situation-oriented of the three tasks. Thus, it may come as no surprise that the greater exposure to situation-level processing provided by the animation and stopping-condition tutors helped subjects most in this novel task. Although we expected the animation component to be the most effective means for teaching students to interpret equations situationally, it appears that merely setting up the situation and assigning a situational role to at least one supporting relation, as is done in the stopping-condition tutor, also provided an added advantage. This advantage, however, is not large enough to distinguish it statistically from the performance of the control group. The *highest* level of performance seemed to arise from exposure to an environment that provided both setting up a situation and executing a simulation of the equations, however.

This result is very informative because it seems to identify an area in which the explicit feedback through animation does not produce a clear advantage above and beyond setting up a situation. This is because students in the latter context are already engaged in the necessary situation-based reasoning. Such a finding is consistent with results of Lewis (1989), who used a simple diagram-based method that highlighted the relations within the problem situation to improve students' ability to construct proper arithmetic expressions from word problems. Findings such as we report here and those reported by Lewis are useful for the designers of mathematics curricula and computer-based learning environments. It is not necessarily the case that the most interactive, simulation-based environments will lead to the greatest problem-solving performance. This result does not conclude otherwise, either, especially with the small number of subjects we have used in an experimental setting. Rather, it leaves the question open and demands that further experimentation be done.

Training Task Performance

For problems solved during training, the results are in agreement with the pretest and posttest scores. The fifth column of Table 4 shows the results for the three individual training tasks solved by the experimental treatment groups. Because choice problems were used instead of story problems, as in the pretest and posttest, they appear in the story problem row in square brackets. A two-way ANCOVA (with treatment as the between-subjects variable, problem type as the within-subject variable, pretest score as the covariate, and training task score as the dependent variable) shows a main effect of problem type, $F(2, 104) = 8.17$, $p < .005$, $MS_e = 0.9$. A Newman–Keuls test ($p < .05$) comparing problem performance indicated that the solve problem was least difficult, with no measurable difference between the debug and choice problems. There was also a main effect of treatment, $F(3, 51) = 5$, $p < .005$, $MS_e = 1$. A subsequent Newman–Keuls test ($p < .05$) revealed more specifically that animation students performed better than either those using the nonanimation tutors or those in the control group. There were no other statistical differences at the .05 level, although the control group tended to perform below the level of the network and stopping-conditions groups.

Additionally, there was a significant Treatment × Problem Type interaction, $F(6, 104) = 2.2$, $p < .05$, $MS_e = 0.24$. Post hoc comparisons using the Newman–Keuls test ($p < .05$) showed that this interaction was the result of two factors. First, the animation group demonstrated superior performance over all the other groups in solving the choice problem, with no other detectable group differences for this problem type. Thus, they showed greater ability at discriminating among solutions for a given problem situation. In fact, ANIMATE users, with a proportion correct of $.93 \pm .13$, were the only ones to perform reliably (at the 95% level) above chance. The confidence intervals around the performances of the stopping-condition group (at $.5 \pm .26$), the network group (at $.36 \pm .25$), and the control group (at $.43 \pm .26$) all included the .33 proportion correct one would expect from chance selection. Second, the equation group showed inferior performance compared with all other groups in generating solution-enabling equations, as measured by performance on the solve problem. No other differences in solve-problem performance were found among the three tutor groups, although the difference between the nonanimation tutor users and the ANIMATE users approached statistical significance. The solve-problem results with the various tutors were expected to be substantially different, the expectation being that situational feedback would act as a superior aid in performance. This was not the case. Instead, all tutors seemed to help users substantially, especially compared with control-group subjects. Even the stripped-down network-only tutor, with no situational component, helped students in the easiest of the problems. As reported earlier, however, this higher level of performance stays with ANIMATE users far more than with the other tutor users when the problem-solving aids are taken away.

Skills for detecting errors in buggy expressions were not found to be significantly better for any one group, as measured by performance on the debug problem. These performance data are in line with the findings for the posttest results for debug problems reported earlier. As with those results, we failed to find strong support for Prediction 4, that debugging ability is superior for ANIMATE users. Performance on the choice problem partially supports this in the area of solution discrimination, but it is not compelling enough to refute the null finding with students' performances on the debug problem.

For every task, subjects who worked with the complete ANIMATE tutor scored at the same level or at a higher level than those who did not have access to the animation. Their problem comprehension—as measured by their inferencing abilities, performance in generating formal equations from stories, and writing stories to match formal equations—was strongest, despite their brief exposure to this novel learning environment. Stopping-condition students, as encouraged by their problem-solving environment to reason situationally about certain equations, showed performance improvements similar to those of the animation students for the story-generation task. Even in the debugging task, animation students showed performance changes paralleled by their improvement in other problems. These improvements were matched by all subjects, however, and so led to no measurable treatment differences at posttest time. This lack of advantage in debug performance, we speculate, is due to subjects' general strategies of formula recognition rather than pure mathematical understanding. The 53% improvement by animation students compared with a 36% improvement by the control group is still noteworthy, however. Animation subjects' greater likelihood to correct buggy equations once they were detected suggests that there are effects due to the training, although we cannot make strong claims about what they are from these data.

DISCUSSION

Drawing on earlier work in this area, we have presented a theory of the cognitive processes that we believe are necessary to achieve competent word-problem-solving behavior. This theory states that problem comprehension relies on a situational understanding of the problem at hand and its relation to one's mathematical knowledge. The theory assumes that students first understand the situations described by the cover story of the problems. Problem comprehension, in turn, provides strong support for the reasoning that is often needed to produce a proper solution. We have also reviewed interpretations of learning and instruction that specify how this competent behavior can be taught to students. A tutoring environment based on this approach was then described; it actively engaged students in the process of explicitly coordinating a formal representation of the problem solution with a qualitative representation of the situation, as depicted by simple computer animation. This situation-based tutor served as an indirect test of our model of problem comprehension. Performance of students using it to solve a variety of problems was compared with subjects in other conditions that provided varying levels of integration of the situation and the formal problem constraints.

The analysis of students' errors and problem-solving performance helps to highlight the ways a situation-based learning environment can and cannot support improvement in problem comprehension and solution generation. From these results, we find that some of our model predictions are correct. The inferences that students must make to fully specify the solution to a word problem are indeed difficult and contribute greatly to students' low pretest performances. The model explicitly predicted (Prediction 1a) our finding that students' pretest solutions would tend to lack those statements that depend on making and expressing inferences necessary for solutions. We also expected (Prediction 1b) that, by working in an environment that encourages situation-based reasoning as a normal part of the solution process, students would demonstrate an improved ability to make these inferences and specify them correctly. Our empirical findings bear this out. The frequency of inference-based errors at posttest for students working with either of the two situation-oriented tutors decreased substantially, whereas the errors of students in the non-situation-oriented treatments—the network and control groups—actually increased. One possible reason for this increase is that the fatigue level of students performing mathematics with no externally provided engagement may have been much higher at posttest time. Although the animation group did not make substantially fewer inference-based errors than the stopping-condition group (which set up a static depiction of the problem situation), it was consistently the group least likely to make such errors in general.

We also made a series of predictions about subjects' problem-solving behaviors. We expected that the situation-based reasoning conveyed by exposure to the ANIMATE tutoring environment would help students generate equations from texts (Prediction 2), generate texts from equations (Prediction 3), and detect and correct erroneous solution equations for a given situation. Evidence to support Predictions 2 and 3 is found in the data from performance on the solve and story problems. In support of Prediction 4a, the ANIMATE group outperformed all others in its ability to recognize a correct solution set for a situation, as measured by performance in the choice problem. In partial support of Prediction 4b, we found that animation subjects had the greatest error-correction ability once an error had been detected. This suggests that some important changes in conceptual understanding occurred for animation students that were not present in the other conditions. However, the expected advantage of animation subjects in detecting errors without the use of the tutor, as stated in Prediction 4b, was not found in the debug problem data.

Work on skill transfer in LISP programming has shown similar findings. Positive skill transfer between solution generation (writing a LISP function) and debugging skill (correcting a buggy function) is specific to the components these two skills have in common. Kessler (1988, reported in Singley & Anderson, 1989) found that writing the appropriate fragment of a function was common to both tasks and so exhibited a high degree of positive transfer. Uncommon components, such as isolating the error of a buggy function, did not transfer as strongly. In algebra-word-problem solving, we have found that subjects who perform well and demonstrate large improvements in solution generation are also good at generating the correct expressions for a set of buggy equations. The detection of these errors does not transfer, however.

Solving algebra word problems is not just a matter of getting the formalism right. It seems to depend on understanding how this formalism is rooted in a real-world situation. Most encouraging about our results are the indications that a situation-based learning environment may indeed help overcome students' tendencies in traditional mathematics instruction to learn decontextualized formal procedures. Instead, such an environment may help them to achieve the conceptual understanding necessary to reason mathematically. The brief training our subjects received while using ANIMATE enabled them to excel in certain 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 posttest, where they did not have access to the tutoring system, was also remarkable. ANIMATE, although particularly useful on the most common type of problems involving translation from words to formal equations, proved as effective in boosting performance on the unusual and unfamiliar story-writing problem.

It is not clear what general strategies students make use of in the story-writing task, given the novelty of the problem. This task is the most situation-oriented, demanding from students, not a formal expression, but a linguistic description of a situation initially specified mathematically. We view this task as reciprocal to the solve task for problem comprehension. That the animation and stopping-condition groups show no reliable performance differences in this task suggests that animation feedback is of limited additional impact for such a task, once subjects are engaged in the reasoning needed for setting up a static depiction of the situation. This may have important pedagogical implications when one is seeking to develop a software environment with minimal functionality. For more traditional word algebra problem-solving tasks, as exemplified by the solve problems, the impact of animation-based feedback is apparent. The ANIMATE tutoring system is clearly helpful in bringing about significant performance improvements.

The experimental results seem to support the notion that providing a weakly structured environment that makes an explicit correspondence between the symbols of an algebraic expression and a simple depiction of the situation described enables students to improve their performance with distance–rate–time problems. The problem model–situation model correspondence seems teachable to students. When the situation is represented by a computer animation, subjects perform best of all. This advantage appears to persist even after the training wheels are removed. We have tested this on a limited set of problems to date, and the results are encouraging enough to invite experimentation on a wider range of problems.

Although many questions remain, we have obtained through this theory a new understanding of how students solve word problems. A text-comprehension-based view of problem solving highlights the origin of certain types of errors and is a valuable perspective when seeking their remedy. Further exploration is needed to determine what facets of problem-solving support help students most. It is noteworthy that using ANIMATE gave more to our subjects than "just another trick" (i.e., the intermediate network representation) for solving 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 final performance was still below that of the animation group. Greeno's (1983) view of the role of *conceptual entities* in problem solving is relevant here. Conceptual entities are cognitive structures used to represent and reason about a problem in a direct way. The set of conceptual entities held by a problem solver determines the kind of information available for reasoning about a problem using general methods. Certain instructional settings can facilitate the construction of a solution-enabling set. Greeno identified two means of acquiring these: observation and computation. ANIMATE provides students with an opportunity to acquire the concepts of travel time, relative starting times, stopping conditions, and so on, through both of these means. Our aim has been to provide students with these conceptual entities so that the general reasoning methods they have can be applied when "canned" problem-solving strategies (e.g., translation rules) prove to be difficult to apply or ineffective.

What can the work presented here tell us about the design of learning environments? The last 10 years have witnessed a move toward computer-based instruction. Some of the most promising systems have embedded in them large amounts of domain-specific knowledge for assessing student performance and supporting meaningful feedback and explanation (e.g., Anderson et al., 1989; Reiser et al., 1989). The development of such systems represents an enormous effort in knowledge engineering, computer system development, and user testing. These systems are restricted to the domains for which there are detailed models of students' problem-solving behavior. Although some of these systems make extensive use of experimental work to determine the amount and form of knowledge they use, these are the exception rather than the rule (Wenger, 1987). This is not a beneficial state of affairs, but it is understandable, because the experimental effort needed to accumulate and codify the domain-specific knowledge can be greater than the task of building the tutor itself.

In the design of the ANIMATE tutoring system, we have attempted to tease apart the learning environment from any system-provided guidance. ANIMATE contains little in the way of domain knowledge and, outside its ability to inform students about the self-consistency of the network (e.g., as in Figure 6), ANIMATE uses none of this knowledge to assess performance or direct a student's solution approach. Rather, ANIMATE relies on students' abilities to assess their own performance in mathematical problem solving by presenting the mathematics in a form that is more accessible to them. In this sense, it is an attempt at a minimalist system for supporting a difficult area of instruction–algebra-word-problem solving. It is possible that some students could also benefit from knowledge-based explanation or guidance during the construction of a formal problem model or specification of the associated animation. It may be advantageous to introduce knowledge-based guidance selectively at a later time and determine empirically where such support is most helpful. The architecture of the ANIMATE system supports such experimentation.

It is no accident that the form we chose for presenting mathematical ideas is a graphic one, intended to be like students' situation models of the story problems. In our view, comprehension failures are central to the difficulty of word problems. Consequently, we have built on text comprehension research in our attempt to model student performance. Within this framework, the kinds of mental representations that students form are key to predicting their performance in answering questions, drawing inferences, and using their knowledge to solve problems. We have hypothesized that two mental representations are most helpful in supporting mathematical reasoning: the problem model and the situation model. Furthermore, it is not sufficient that students form both of these representations in the course of reading and solving a word problem. They must also integrate information from both to assign situation-based meaning to formal expressions and interpret mathematical statements situationally. From this perspective, we have developed a tutoring environment that attempts to bring students' representations of the situation and the mathematics together through animation-based feedback. Our experimental results are limited but promising.

The experimental findings support our psychological model, as instantiated by the ANIMATE tutoring environment, in four of six predictions. But, there are several important shortcomings. The ANIMATE system does not produce students with superior debugging behavior or bring about exemplary problem-solving behavior. Even though the animation group showed marked improvements in a variety of problem types, the error rate was still high at posttest. ANIMATE students' reasoning, too, left much room for improvement. Although their abilities to generate and specify inference-based supporting relations improved appreciably, even in the face of declining performance for some groups, the likelihood of misformulating supporting relations was still high. The ANIMATE system has many limitations at this time. Students do not receive all the flexibility possible in a computer system (cf. the LOGO system), and the situational support is not always presented in the most ecological manner. The persistence of errors in the animation students' solution protocols may thus be due to limitations of our model of problem comprehension or to our instantiation of that model in the tutor.

What does this work tell us about methods for evaluating students' problem-solving abilities? Had we used only the most typical measures of problem-solving ability (performance on the solve problem), we would have drawn many erroneous conclusions with regard to the effect of the ANIMATE system on students' learning. With just these results, it would seem that the ANIMATE tutor was clearly superior to the other treatments. By including problems that had students generate stories from equations and evaluate and debug problem solutions, a more complete, albeit more complicated, picture has emerged. We also found it valuable to look not only at problem-solving performance measures but also at measures of problem comprehension, which were chiefly linked to the errors students made in their solution protocols. Detailed analyses such as those presented by Hall et al. (1989) can provide further understanding of the reasoning processes of students and can be used to further inform cognitive models and instructional approaches.

There were many statistical comparisons that did not clearly favor the problem comprehension model. Yet, in all, the performance of the animation group on a range of tasks consistently ranked at or near the top, and these subjects consistently improved. Most notably, students using the experimental ANIMATE environment for less than an hour performed better than control subjects on the most common form of translation task (the solve problem). Underlying this is the improved ability of subjects to make certain necessary situation-based inferences and express them algebraically. We take these findings as evidence in favor of our model. Still, the empirical results are clearly wanting. As stated earlier, we chose an indirect method for testing our theory, using tutor users as our spokespeople. There are several layers between our theoretical predictions and the data. It is not possible to tell if these inadequacies are due to inherent limitations of the model or to problems with the current instantiation of it. These results are very promising, however, and we conclude from them that it is worthwhile to seek more direct methods for testing this theory.

ACKNOWLEDGMENTS

The authors gratefully acknowledge Denise Cummins, Stephanie Doane, 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, Ann Bovenmyer Lewis, Rogers Hall, and an anonymous reviewer for helpful comments made regarding an earlier draft of this article. This work was supported in part by National Institutes of Mental Health Grant MH-15872 to Walter Kintsch.

REFERENCES

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

Anderson, J. R. (1989a). Analysis of student performance with the LISP tutor. In N. Frederickson, R. Glaser, A. Lesgold, & M. G. Shafto (Eds.), *Diagnostic monitoring of skill and knowledge acquisition* (pp. 27–50). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Anderson, J. R. (1989b). Psychology and intelligent tutoring. In D. Bierman, J. Breuker, & J. Sandberg (Eds.), *Proceedings of the Fourth International Conference on AI and Education* (p. 1). Amsterdam: IOS.

Anderson, J. R., Boyle, C. F., Farrell, R., & Reiser, B. (1984). *Cognitive principles in the design of computer tutors* (Tech. Rep. No. ONR-84-1). Pittsburgh: Carnegie Mellon University.

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.

Anderson, J. R., Conrad, F. G., & Corbett, A. T. (1989). Skill acquisition and the LISP tutor. *Cognitive Science, 13*, 467–505.

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* (pp. 361–392). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Bobrow, D. G. (1968). Natural language input for a computer problem-solving system. In M. Minsky (Ed.), *Semantic information processing* (pp. 135–215). Cambridge, MA: MIT Press.

Brown, A. L., & Palincsar, A. S. (1989). Guided, cooperative learning and individual knowledge acquisition. In L. B. Resnick (Ed.), *Knowing, learning, and instruction* (pp. 393–452). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

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.

Brown, J. S., & VanLehn, K. (1980). Repair theory: A generative theory of bugs in procedural skills. *Cognitive Science, 4*, 389–426.

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

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* (pp. 453–494). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

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

Glaser, R., & Bassok, M. (1989). Learning theory and the study of instruction. *Annual Review of Psychology, 40*, 631–666.

Gould, L., & Finzer, W. (1982). A study of TRIP: A computer system for animating time–rate–distance problems. *International Journal of Man–Machine Studies, 17*, 109–126.

Greeno, J. G. (1983). Conceptual entities. In D. Gentner & A. L. Stevens (Eds.), *Mental models* (pp. 227–252). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

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

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* (Tech. Rep. 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.

Harel, I. (1990). *Constructionist learning*. Cambridge, MA: MIT Media Laboratory.

Judd, C. M., & McClelland, G. H. (1989). *Data analysis: A model-comparison approach*. San Diego: Harcourt Brace Jovanovich.

Kessler, C. (1988). *Transfer of programming skills in novice LISP learners*. Unpublished doctoral dissertation, Carnegie Mellon University, Pittsburgh.

Kintsch, W. (1988). The use of knowledge in discourse processing: A construction–integration model. *Psychological Review, 95*, 163–182.

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 & A. L. Stevens (Eds.), *Mental models* (pp. 75–98). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

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

Lewis, A. B. (1989). Training students to represent arithmetic word problems. *Journal of Educational Psychology, 81*, 521–531.

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

Mannes, S., & Kintsch, W. (1991). Routine computing tasks: Planning as understanding. *Cognitive Science, 15*, 305–342.

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.

McCloskey, M. (1983). Naive theories of motion. In D. Gentner & A. L. Stevens (Eds.), *Mental models* (pp. 299–324). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Nathan, M. J. (1988). *Recall of stories and story problems.* Unpublished master's thesis, University of Colorado, Boulder.

Nathan, M. J. (1990). Empowering the student: Prospects for an unintelligent tutor for word algebra problem solving. In J. C. Chew & J. Whiteside (Eds.), *Proceedings of CHI '90* (pp. 407–414). New York: ACM.

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

Nathan, M. J., & Young, E. (1990). Thinking situationally: Results with an unintelligent tutor for word algebra problems. In A. McDougell & C. Dowling (Eds.), *Computers in education* (pp. 425–430). New York: North-Holland.

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* (pp. 187–216). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

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* (pp. 51–118). 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 for intelligent tutoring systems.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

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). *Knowledge representation and explanation in GIL, an intelligent tutor for programming* (Cognitive Science Laboratory Tech. Rep. No. 37). Princeton, NJ: Princeton University.

Resnick, L. B. (1988). Treating mathematics as an ill-structured discipline. In R. I. Charles & E. A. Silver (Eds.), *The teaching and assessing of mathematical problem solving* (pp. 32–60). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Reusser, K. (1988). *From text to situation to equation: Cognitive simulation of understanding and solving mathematical word problems* (Research Rep. No. 5). Bern, Switzerland: Universität Bern, Department of Educational Psychology.

Roschelle, J. (1987, May). *Envisionment, mental models, and physics cognition.* Paper presented at the International Conference on AI and Education, Pittsburgh.

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 Press.

Schoenfeld, A. H. (1985). *Mathematical problem solving.* New York: Academic.

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: Lawrence Erlbaum Associates, Inc.

Shalin, V., & Bee, N. V. (1985). *Analysis of the semantic structure of a domain of word problems* (Tech. Rep. No. APS–20). Pittsburgh: University of Pittsburgh, Learning Research and Development Center.

Simon, H. A. (1979). *Models of thought.* New Haven, CT: Yale University Press.

Singley, M. K., & Anderson, J. R. (1989). *The transfer of cognitive skills.* Cambridge, MA: Harvard University Press.

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

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

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

Weaver, C. A., & Kintsch, W. (1987). Reconstruction in recall of prose. *Text, 7,* 165–180.

Weaver, C. A., III, & Kintsch, W. (1992). Enhancing students' comprehension of the conceptual structure of algebra word problems. *Journal of Educational Psychology, 84,* 419–428.

Wenger, E. (1987). *Artificial intelligence and tutoring systems: Computational and cognitive approaches to the communication of knowledge.* Los Altos, CA: Morgan Kaufmann.

Wertheimer, M. (1982). *Productive thinking.* Chicago: University of Chicago Press. (Original work published 1945)

Willis, G. B., & Fuson, K. C. (1988). Teaching children to use schematic drawings to solve addition and subtraction word problems. *Journal of Educational Psychology, 80,* 192–201.

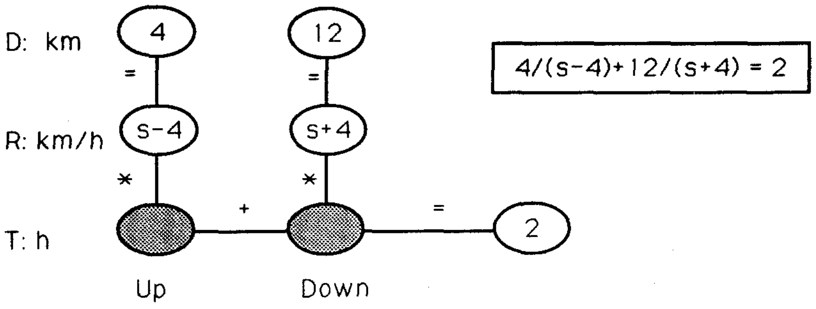
APPENDIX A

Problem Models for Examples From Each of Mayer's (1981) Rate Families

FAMILY I: AMOUNT-PER-TIME RATE

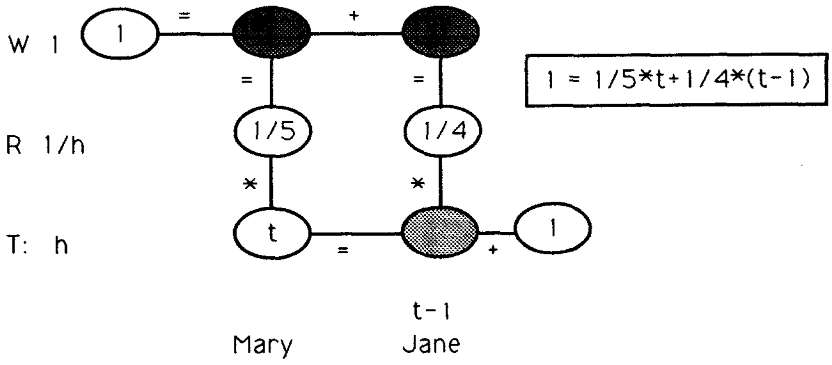
Current

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?



Work

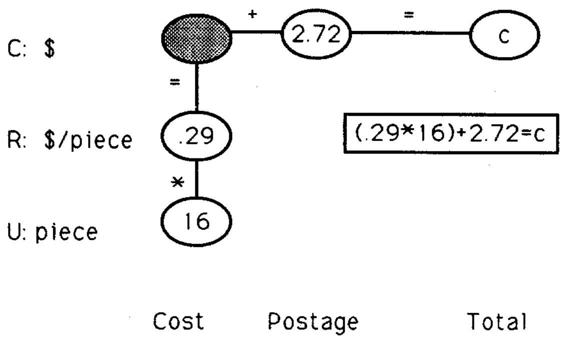
Mary can do a job in 5 hours and Jane can do the job in 4 hours. If they work together, how long will they take to do the job if Jane starts 1 hour after Mary?



FAMILY II: COST-PER-UNIT RATE

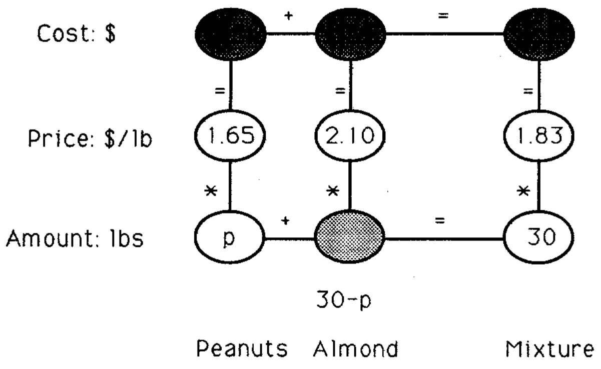
Fixed Cost

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



Dry Mixture

A grocer mixes peanuts worth $1.65 per pound and almonds worth $2.10 per 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?

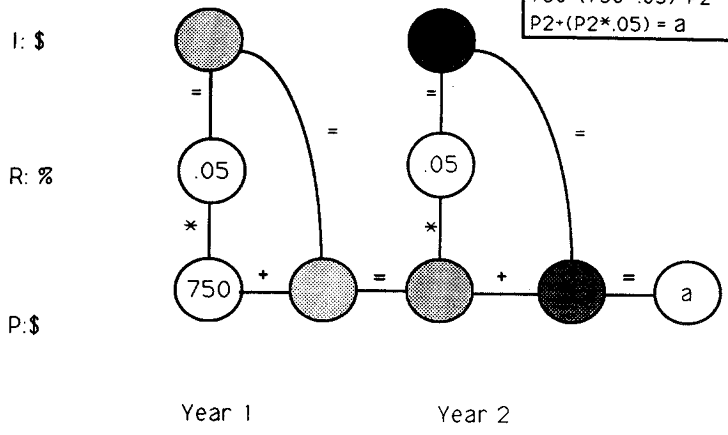


$$\boxed{1.65p + 2.10(30-p) = 1.83*30}$$

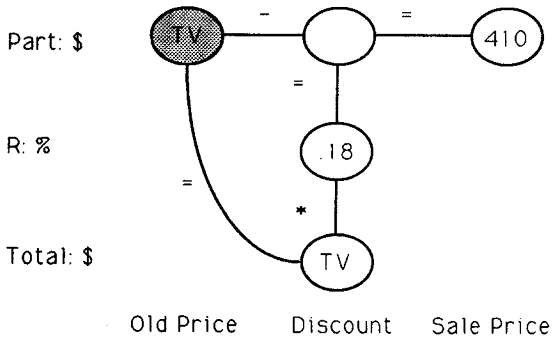
FAMILY III: PORTION-TO-TOTAL-COST RATE

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

750+(750*.05)=P2
P2+(P2*.05) = a



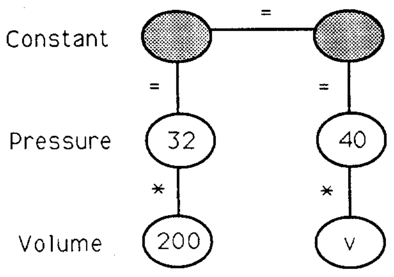
An appliance store drops the price of a certain TV 18% to a sale price of $410. What was the former price?



FAMILY IV: AMOUNT-TO-AMOUNT RATE

Inverse Variation :

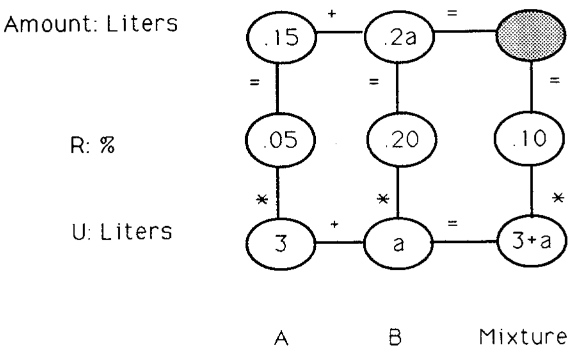
The volume of gas varies inversely with the pressure on it. The volume of gas is 200 cc under a pressure of 32 kg/sqcm What will be its volume under a pressure of 40 kg/sqcm ?



32*200=40*v

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?



.15 + .2*a = .10*(3+a)

APPENDIX B

Experimental Materials for Pretest, Posttest, and Training Tasks

Pretest

Problem 1. Write the appropriate equation or set of equations that can be used to solve the following problems. (*You do NOT need to do algebra or solve for any values*)

Problem 1a. Two planes start at the same time leaving from cities thirty six hundred miles apart. They travel toward each other, one at four hundred miles per hour and the other at seven hundred miles per hour. If the planes leave at Noon, how long will the air traffic controller have to warn the planes of a collision?

Problem 1b. Alan leaves San Francisco at 6 a.m. on a train to Los Angeles to give a presentation. The two cities are about six hundred miles apart. The train travels at one hundred and fifty miles per hour. Phyllis realizes that he left his slides at home and will need them when he arrives. If Phyllis leaves at 8 am on a plane that travels at three hundred miles per hour and the plane and train arrive at the same station, how long will Phyllis have to wait for Alan to arrive?

Problem 2. Below you are given a word problem and a possible solution to it in the form of a set of equations. The equations may be flawed, however. Please correct the equations so that someone could use them to solve the problem as given.

Vanna and Crystal are in a four thousand mile cross country car race. Vanna starts immediately and travels at eighty five miles per hour. Crystal starts in the second wave of cars because she did not place as high in the time trials. The second wave leaves one hour later. If Crystal drives at one hundred and fifteen miles per hour, how long will it take her to pass Vanna?

$$\begin{aligned} D1 &= 85 \times T1 \\ D2 &= 115 \times T2 \\ T1 &= T2 - 1 \\ D1 + D2 &= 4000 \end{aligned}$$

Problem 3. Below you are given a set of equations that describes a situation mathematically. Write a short story that could be a word problem for the given equations. Use Bob as character 1 and Rebecca as character 2 in the word problem. Try to keep it to 3 sentences or less.

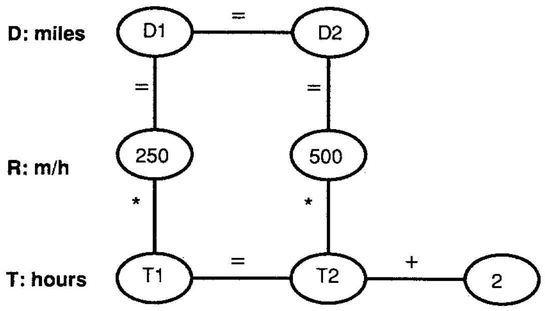
$$D1 = 25 \times T1$$
$$D2 = 20 \times (T1 + 1)$$
$$D1 + D2 = 80$$

Training Task: Tutor Groups

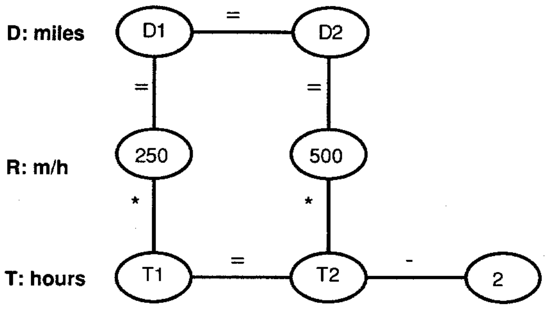
Problem 1. Select one of the three choices below that you feel is the best solution for the word problem given.

A train travels west at two hundred and fifty miles per hour. Another train leaves on a parallel course two hours later and travels west at five hundred miles per hour. How far will the second train travel when it overtakes the first train?

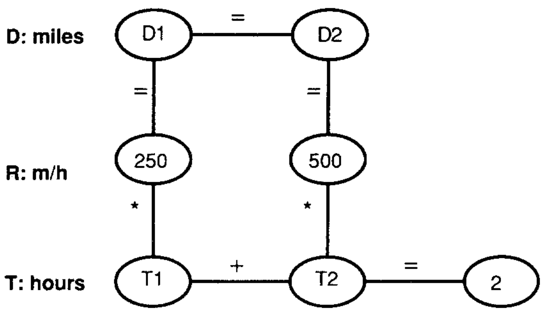
A.



B.



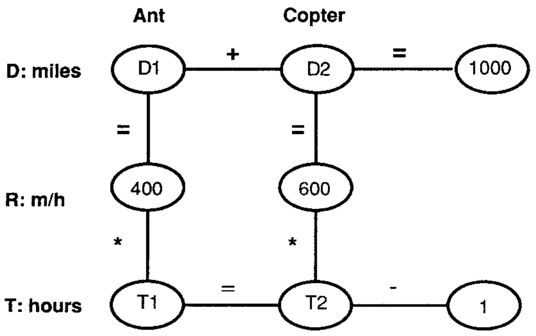
C.



Problem 2. Please construct the correct network for the word problem given. Draw any diagrams or pictures that may help. When you are satisfied with the drawn network, write down the equations that can be extracted from the network to solve the problem.

Bill hates flies. There is a fly next to Bill. His scream startles the fly, which takes off to the left immediately, traveling at one hundred and fifty feet per minute. Bill freezes for one minute and runs to the right at one hundred feet per minute. How long will it take Bill to be three hundred feet away from the fly?

Problem 3. The following network is intended to describe the word problem below. The network may have flaws, however. Correct any flaws by writing directly on the network. When done, copy over the resulting equations from the corrected network.



A huge ant terrorizes San Francisco. It travels east toward Denver, which is twenty four hundred miles away, at four hundred miles per hour. The Army learns of this and sends a helicopter from Denver an hour later to intercept. It travels at six hundred miles per hour. If the ant left at two p.m., ignoring time changes, when will the ant and helicopter meet?

Training Tasks: Equation Group

Problem 1. Select one of the three choices below that you feel is the best solution for the word problem given.

A train travels west at two hundred and fifty miles per hour. Another train leaves on a parallel course two hours later and travels west at five hundred miles per hour. How far will the second train travel when it overtakes the first train?

A.

$$\begin{aligned} D1 &= 250 \times T1 \\ D2 &= 500 \times T2 \\ D1 &= D2 \\ T1 &= T2 + 2 \end{aligned}$$

B.

$$\begin{aligned} D1 &= 250 \times T1 \\ D2 &= 500 \times T2 \\ D1 &= D2 \\ T1 &= T2 - 2 \end{aligned}$$

C.

$$\begin{aligned} D1 &= 250 \times T1 \\ D2 &= 500 \times T2 \\ D1 &= D2 \\ T1 &+ T2 = 2 \end{aligned}$$

Problem 2. Please construct the correct network for the word problem given. Draw any diagrams or pictures that may help. When you are satisfied with the drawn network, write down the equations that solve the problem.

Bill hates flies. There is a fly next to Bill. His scream startles the fly, which takes off to the left immediately, traveling at one hundred and fifty feet per minute. Bill freezes for one minute and runs to the right at one hundred feet per minute. How long will it take Bill to be three hundred feet away from the fly?

Problem 3. The following equations are intended to describe the word problem below. The equations may have flaws, however. Correct any flaws by writing directly on the equations. When done, copy over the resulting equations.

$$\begin{aligned} D1 &= 400 \times T1 \\ D2 &= 600 \times T2 \\ D1 &= D2 + 1000 \\ T1 &= T2 - 1 \end{aligned}$$

A huge ant terrorizes San Francisco. It travels east toward Denver, which is twenty four hundred miles away, at four hundred miles per hour. The Army learns of this and sends a helicopter from Denver an hour later to intercept. It travels at six hundred miles per hour. If the ant left at two p.m., ignoring time changes, when will the ant and helicopter meet?

Posttest

Problem 1. Below you are given a word problem and a possible solution to it in the form of a set of equations. The equations may be flawed, however. Please correct the equations so that someone could use them to solve the problem as given.

Arsenio and Ronald start in the same city. Ronald starts immediately and travels west at forty five miles per hour. Arsenio leaves one hour later, traveling east. If Arsenio drives at fifty miles per hour, how long will it take him to be two hundred miles away from Ronald.

$$\begin{aligned} D1 &= 45 \times T1 \\ D2 &= 50 \times T2 \\ T1 &= T2 - 1 \\ D1 + D2 &= 200 \end{aligned}$$

Problem 2. You are given a set of equations that describes a situation mathematically. Write a short story that could be a word problem for the given equations. Use Linda as character 1 and Kathy as character 2. Keep it to 3 sentences or less.

$$\begin{aligned} D1 &= 250 \times T1 \\ D2 &= 100 \times T2 \\ T1 &= T2 - 1 \\ D1 &= D2 \end{aligned}$$

Problem 3. Write the appropriate equation or set of equations that can be used to solve the following problems.

(You do NOT need to do algebra or solve for any values)

Problem 3a. An ant and a fly start at the same time leaving from homes thirty feet apart. They travel toward each other, the ant at four feet per hour and the fly at seven feet per hour. If the insects both leave at 1 p.m., how long will it take for them to meet?

Problem 3b. Bill leaves for a bike ride at 6 a.m. riding at sixteen miles per hour. He breaks a wheel and has to walk back with his bike at four miles per hour. If he arrives home eight hours later, assuming he took no rests or detours, how far did Bill initially ride?