Illustrating Principled Design: The Early Evolution of a Cognitive Tutor for Algebra Symbolization

KENNETH R. KOEDINGER
JOHN R. ANDERSON

INTRODUCTION

This paper provides an illustration of the principled design of an interactive learning environment. It provides a view of the early stages of this process where design, testing, and redesign are most critical. The long term goals of principled design are twofold: (1) to create a system that can be convincingly demonstrated as an effective and practical learning aid and (2) to provide a replicable account of how and why the system is effective. A principled design is one that is *both theoretically guided and empirically supported.* A principled design is guided by a set of theoretical principles and specific pedagogical hypotheses (Anderson, Corbett, Koedinger & Pelletier, 1995; Koedinger & Anderson, 1993). It is informed by user testing early and often. The design process is iterative: theory, design, test, redesign. Tests that fail lead first to redesign and then, if principled redesigns fail, to changes in the theory. It should be the natural expectation of the field that no Interactive Learning Environment will be fully effective in its initial implementations and that early demonstrations of limitations have a positive, not a negative, bearing on the value of the final system. *The only systems immune to some failure are ones that are never tested.* Unfortunately, these are all too common. (For example, Corbett, Koedinger and Anderson (in press) report that only 25% of the papers at recent ITS conferences include any kind of empirical evaluation.)

We describe the design of a particular kind of intelligent tutoring system called a cognitive tutor (Anderson, Corbett, Koedinger & Pelletier, 1995). In addition to employing artificial intelligence techniques, cognitive tutors have the defining feature of containing a psychological model of the cognitive processes behind successful and near-successful student performance. This cognitive model provides the core functionality. The cognitive model is used by a

Interactive Learning Environments
Volume 5, pages 161-179

ISSN: 1049-4820

Kenneth R. Koedinger, Human-Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA 15213; email: koedinger@cmu.edu.

technique called *model tracing* to provide students with individualized tutoring support as needed and in the context of problem solving activity. The cognitive model is also used with *knowledge tracing* to monitor students' evolving strengths and weaknesses and to adapt the selection of activities to provide maximal learning opportunities. These features appear central to the demonstrated benefits of cognitive tutors. In comparisons with alternative instructional approaches, the use of cognitive tutors has been shown to accelerate learning by as much as three times, increase posttest performance by a standard deviation, and lead to documented improvement in student motivation (Anderson, Corbett, Koedinger, & Pelletier, 1995).

Early cognitive tutors for mathematics in the domains of geometry theorem proving and categorical word problems were criticized by mathematics educators for not addressing issues of current emphasis in math education reform (cf. NCTM standards, 1989). These systems were largely designed as proofs-of-concept and for learning experiments. A large current effort is to apply the lessons learned from these early cognitive tutors to new areas of mathematics that are consistent with the recent curriculum standards of the National Council of Teachers of Mathematics (NCTM, 1989). We are taking a *client-centered* approach that combines our cognitive tutor technology and principled design experience with the subject-matter and curriculum experience of local and national educators. We have focused on first year algebra at the high school level both because algebra is fundamental to further mathematics and science and because it is a critical gatekeeper to future success (Pelavin, 1990).

THEORY: ACT AND INDUCTIVE SUPPORT

The cognitive model at the core of cognitive tutors is based on the ACT theory (Anderson, 1983; 1993). The ACT theory distinguishes declarative knowledge from procedural knowledge. Declarative knowledge is knowledge that can be directly accessed including facts, concepts, pictures, and stories. It can be acquired in a number of ways including by instruction, by example, by discovery, or by derivation. Procedural knowledge is performance knowledge that cannot be directly accessed. It can only be acquired by doing, that is, by applying declarative knowledge in the process of problem solving. By itself declarative knowledge is "inert" (Whitehead, 1929; CTG, 1990)–it doesn't do anything. Procedural knowledge is needed to make use of it. A math student may know that the inverse of multiplication is division, but without procedural knowledge of how to apply this fact in problem solving, may be unable to solve a problem like "If you paid $38 for jeans that were reduced to 80% of their original price, what was the original price?" Procedural knowledge is bound to the problem solving contexts in which it is acquired. A student who has "proceduralized" the inversion of multiplication in the context of algebra equation solving (e.g., $0.8x = 38$), may not have the procedural knowledge to solve problems like the one above. Similarly, a store manager may do fine on a problem like the one above, but may not be able to solve a similar problem in different context (e.g., "If the sales tax rate is 5%, what tax base is necessary to yield one million dollars in tax revenues").

The ACT theory claims that procedural knowledge is learned from analogy to examples. The theory does not deny the possibility of learning by being told, but claims that such learning is done indirectly and depends on students' competence to interpret instructions to create examples for themselves. Such examples can then provide the basis for analogy-based learning. Cognitive tutors are designed to give students opportunities to apply declarative knowledge in a variety of problem solving contexts. According to ACT, such learning by doing results in the acquisition of production rules–units of procedural knowledge that tie problem solving strategies and actions to particular problem contexts. A key step in the principled design of a cognitive tutor is an analysis of the cognitive strategies and actions that learners use in the domain of interest (cf. Koedinger & Anderson, 1993).

We set out to build a cognitive tutor for high school algebra and, in particular, were focused on the algebraic "symbolization" process, that is, the translation of problem situations into algebra notation. The first step in an ACT-based analysis of this domain is to consider how symbolization could be learned from analogy to examples. One approach is to have students see examples of translations of the problems to symbols. However, with such examples this is an "enigmatic domain" in that these examples do little to reveal the cognitive processing steps needed to make the translation (Koedinger & Anderson, 1993). Students are left to make shallow or "superstitious" analogies (c.f. Lewis, 1988), like "altogether means add," that often lead to knowledge that does not generalize. For example, consider the problem:

> Mary and John have eight marbles altogether. Mary has two marbles. How many marbles does John have?

A common student error is to put 8 and 2 "altogether" and answer 10 (cf. Cummins, Kintsch, Reusser & Weimer, 1988).

We decided to explore an alternative approach that both draws on what students already know and makes visible some of the intermediate processing steps in the symbolization process. A critical step in this alternative approach is to view algebra as *generalized arithmetic.* Algebra provides a way to generalize a series of arithmetic procedures (e.g., 0.8 * 40, 0.8 * 45, 0.8 * 50) in a single, concise statement (e.g., $0.8x$). Viewing algebra this way suggests an alternative to direct translation examples as a source for analogical learning. Concrete arithmetic procedures can be used as a intermediate step in translation. Arithmetic examples provide a source for more meaningful examples that draw on students' prior experience with arithmetic. Further, because of this prior experience, students can create these examples for themselves. The next section describes our cognitive analysis in more detail and, in particular, how we arrived at this approach.

The Domain: Algebra Problem Solving

We began our client-centered approach working with a Pittsburgh math teacher and with the textbook that was being used by the Pittsburgh public schools for their Algebra 1 course (Forester, 1984). The Forester textbook was published prior to the new standards of the National Council of Teachers of Mathematics (NCTM, 1989), but contained new approaches consistent with these reforms. In particular, the text makes a serious attempt to elaborate the instruction of word problem solving. In traditional algebra textbooks, word problems are common, but instruction on how to solve them is rare and quite limited.

Table 1a shows an algebra word problem that could be seen in the beginning of a traditional algebra textbook. The normative strategy for solving such problems has two stages. First, define a variable (e.g., x = the hours worked) and write an equation (e.g., $42x + 35 = 140$). Second, solve this equation for the variable (e.g., $x = 2.5$). Table 1b shows Forester's variation of this problem. In his "Foreword to Teachers" Forester describes his motivation for such problems:

> The word problems involve variables that really vary, rather than standing for unknown constants. Some problems have multiple parts in which students are forced to write an expression representing a variable quantity. Then they evaluate the expression for several values of the variable, and write and solve equations involving the expression. By so doing, even students who can do the problem in their heads get practice in proper algebraic techniques so that they can work the harder, less structured problems later on.

The Forester problem (1b) represents a change from the traditional problem (1a) in the insertion of questions 1-3 between the problem statement (first sentence in both problems) and the question posed (last sentence in both problems). Question one explicitly asks students to symbolize, that is, to translate the problem statement into symbols. Although the intent of a traditional problem (1a) is for students to begin by symbolizing, the Forester problem is different in making this step explicit to students in question one. Questions two and three in the Forester problem (1b) give the number of hours worked (the x or *start variable*) and ask for the pay the company receives (the y or *result variable*). Question four turns this around and, like the traditional problem, now gives the pay received (the result variable) and asks for the hours worked (the start variable). We refer to questions like two and three as *result-unknown* problems because the unknown, the pay received, is the result of the process or events described in the problem. Questions like four are *start-unknown* problems because the unknown, the hours worked, is the start of the process or events described in the problem. The use of result-unknown and start-unknown is present in the elementary arithmetic literature (Carpenter, Corbitt, Kepner, Lindquist & Reys, 1980; Riley & Greeno, 1988; Briars & Larkin, 1984) to describe problems where there is a single arithmetic operator, but we extend its use here to include problems, like those in Table 1, that involve two operators.

Cognitive Analysis of Symbolization

In previous work (Koedinger & Anderson, 1990; 1991), we showed that even for mathematical experts in a decidedly *deductive* domain, geometry theorem proving, problem solving knowledge has a fundamental *inductive* character. While much of mathematical reasoning in it's externalized written form is the deductive manipulation of symbols, the underlying cognitive processes that support effective reasoning draw on induction from prior perceptual experience (cf. Cheng & Holyoak, 1985). If expert mathematical knowledge is fundamentally organized as inductive abstractions, not deductive rules, then perhaps instruction that supports and encourages such inductive reasoning would more effectively lead to expertise. We will refer to this conjecture as the *inductive support hypothesis.*

To illustrate how this hypothesis applies to algebra, we contrast two views about algebra development and, in particular, about how problems like those in Table 1 are solved:

1. The normative-deductive view, which is implicit in traditional algebra textbooks, is that people solve verbally-presented problems by first translating to symbols and then manipulating the symbols to find answers.

2. An alternative conceptual-inductive view, which derives from the cognitive analysis and empirical work of ourselves and others, is that people can solve many verbally-presented problems without recourse to normative symbolic strategies either by using internal verbal representations or by using simpler arithmetic symbols

Figure 1 illustrates the student strategies consistent with these two views. Figure 1a shows the normative-deductive strategies encouraged by Forester textbook problems like the one shown in Table 1b. The left of Figure 1a illustrates the question parts of a Forester problem and on the right is an example solution. The three strategies are shown in boxes with arrows indicating their inputs and outputs. According to the normative-deductive view, a student answers the symbolization question (1) by *algebra translation*, a strategy that draws on knowledge of how certain phrases in problem statements translate into algebraic symbols (e.g., if the problem says "42 dollars per hour" and h is the variable for hours, then translate to $42h$). Algebra translation produces an expression for the cost of the bill, "$42h + 35$." The result-unknown questions (2 & 3) can then be answered by *evaluate expression*. This strategy is the familiar *plug-and-chug* where the student substitutes the given number of hours for h and then does the arithmetic (e.g., plug $h = 3$ into $42h + 35$ and solve to get 161). The start-unknown question (4) is answered by *solve algebra*, that is, by setting the expression equal to the given value of the bill ($42h + 35 = 140$) and solving the equation for h.

Table 1c shows a modification of the Forester problem guided by the inductive support hypothesis. The change is simply to move the result-unknown questions prior to the symbolization question. This change encourages an alternative strategy for symbolizing whereby

Table 1. Different versions of the same algebra problem

a. Traditional textbook problem:
Drane & Route Plumbing Co. charges $42 per hour plus $35 for the service call. Find the number of hours worked when you know the bill came out to $140.

b. Forester textbook problem:
Drane & Route Plumbing Co. charges $42 per hour plus $35 for the service call.
1. Create a variable for the number of hours the company works. Then, write an expression for the number of dollars you must pay them.
2. How much would you pay for a 3 hour service call?
3. What will the bill be for 4.5 hours?
4. Find the number of hours worked when you know the bill came out to $140.

c. Inductive support problem:
Drane & Route Plumbing Co. charges $42 per hour plus $35 for the service call.
1. How much would you pay for a 3 hour service call?
2. What will the bill be for 4.5 hours?
3. Create a variable for the number of hours the company works. Then, write an expression for the number of dollars you must pay them.
4. Find the number of hours worked when you know the bill came out to $140.

the answers to the result-unknown questions can be used to induce the symbolic expression.

Figure 1b illustrates this alternative strategy and other alternatives to the normative-deductive strategies described above as they apply to questions in an inductive-support problem. For these problems, the result-unknown questions (1 & 2) come first. Figure 1b shows an alternative strategy (not evaluate expression as in Figure 1a) for solving result-unknowns. This arithmetic translation strategy draws on students' pre-algebraic knowledge of how phrases in problem statements translate into arithmetic steps (e.g., if the problem says "42 dollars per hour" and the hours is "3," then multiply 42 and 3). In performing arithmetic translation on questions one and two, the student twice performs the process of multiplying the hours by 42 and then adding 35. This process provides the basis for an alternative strategy for answering the symbolization question. In the *Induce pattern* strategy the student generalizes from the pattern of the arithmetic steps ("42 * 3 + 35" and "42 * 4.5 + 35"), replacing the specific values for the hours by the variable *h*, to produce the algebraic expression "42 * *h* + 35". Using the expression, the student can now answer the start-unknown question (4), as in the normative-deductive view, via the *solve algebra* strategy. The conceptual-inductive view, however, recognizes alternative strategies for solving start-unknown problems that work directly from the problem statement. The *unwind strategy* is illustrated in Figure 1b. In this strategy, the student works backward from the result-value (the cost of the bill, 140 dollars) inverting the operations and applying them in reverse (e.g., subtract off the service charge, 140 - 35 → 105, and then divide by the hourly rate, 105 / 42 → 2.5). A second strategy, not illustrated in Figure 1b, is to *guess-and-check*: guess at the start value, do the arithmetic, and check if it yields the result. If not, guess again. Such strategies have been observed in elementary students on one operator start-unknown problems (Briars & Larkin, 1984). Koedinger & Tabachneck (1995) have also observed students performing these strategies on two-operator start-unknowns.

The key focus of the experiment described here is on the alternative strategies for symbolization. The Forester problem encourages the algebra translation strategy as the only approach to symbolization. In contrast, the inductive support problem encourages a second strategy. The student is first encouraged to answer the result-unknown questions via the arithmetic translation strategy that draws on students' pre-algebraic knowledge. The answers to the result-unknown questions then provide data for the induce pattern strategy to produce the symbolic expression. In this way, algebraic symbolization can occur as a generalization of arithmetic rather than as a new translation process.

A second potential benefit of the inductive support problems is to make more clear how creating an algebra expression can aid the solution of the start-unknown questions (e.g., finding hours worked given the total bill). In contrast to the Forester problems where the result-unknown questions interrupt the connection between the symbolization and the start-unknown questions (see the right of Figure 1a), in the inductive support problems the symbolization step directly precedes the start-unknown question (see the right of Figure 1b). In this way, the student may better appreciate the value of creating the expression as it serves to help solve

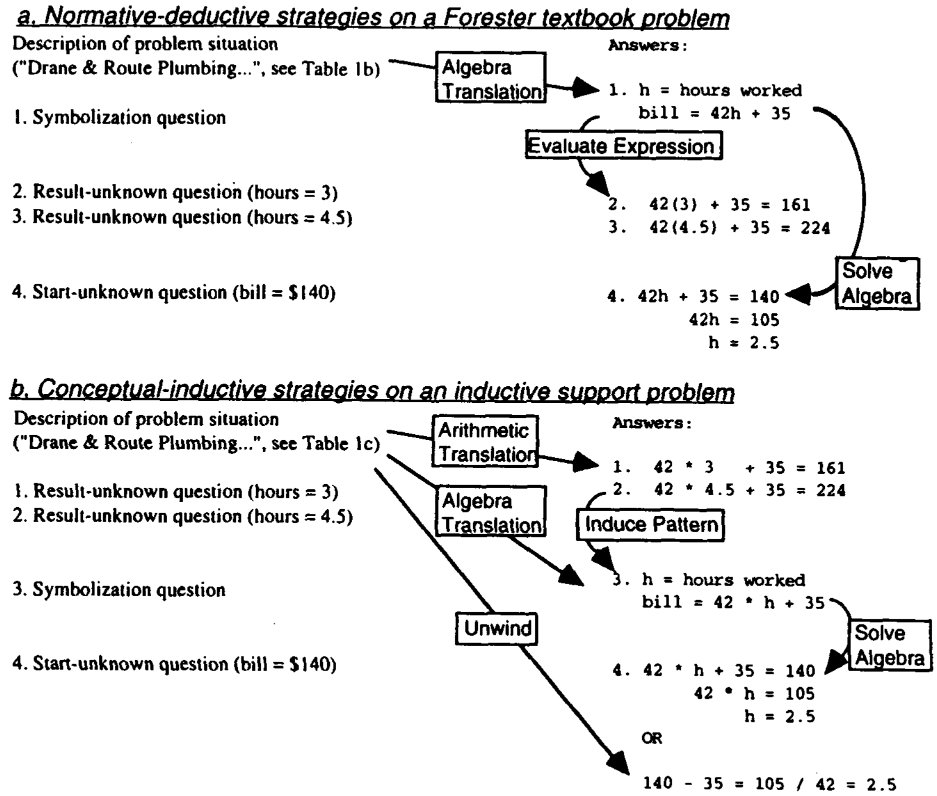

Figure 1. Strategies (in boxes) for Different Problem and Question Types: (a) a Forester Problem is Solved by Algebra Translation, Evaluating an Expression, and Solving the Algebra, (b) An Inductive Support Problem Provides a Second Way to Answer the Symbolization Question by Inducing the Pattern from the Result-Unknown Solutions. (Note: For simplicity sake, we have left off lines that would represent the information flow from the questions to the strategies.)

the presumably more difficult start-unknown question. This line of reasoning presumes that result-unknown problems are easier than both symbolization and start-unknown problems–hypotheses we test in the experiment described below.

DESIGN: A COGNITIVE TUTOR FOR ALGEBRAIC SYMBOLIZATION

For any significant education or training domain, a fully adequate cognitive analysis cannot be achieved a priori. Fast prototype development and early testing with students is a critical complement to cognitive analysis methodologies like task analysis, protocol analysis, and cognitive modeling. The creation of a prototype allows experimentation to test high level hypotheses and help refine the cognitive analysis.

The Prototype Tutor

In the initial prototype we developed, students worked through problems like those presented in chapters one and two of the Forester textbook. All problems had four questions like those in Table 1b&c. Students answered the questions in these problems by filling in the rows of a table.

Tutor curriculum. The tutor curriculum was in two lessons. The problems in lesson 1 involved one arithmetic operator and were of the form "$y = x + a$" or "$y = ax$" where a was an integer or fraction. Lesson two problems involved two operators and were of the form "$y = ax + b$" and "$y = b - ax$" where a and b where integers or fractions. Students worked on a lesson until they reached mastery of the skills in it. Skill mastery was determined by the knowledge tracing algorithm (Corbett, Anderson, & O'Brien, 1995).

Nature of interaction. Figure 2 shows the prototype tutor screen on the first problem from lesson one. The student has already labeled the columns of the Worksheet with the relevant quantities, "width" and "length", and she has answered questions one and two by filling in rows one and two of the Worksheet. To finish, she must answer the second result-unknown question (3) and the start-unknown question (4). As part of answering the start-unknown question, she will use the Equations window to find the width by setting the given length of 43 cm to the symbolic expression for length. By entering the equation "$w + 7 = 43$" into the Equations window, she can use it's symbolic calculation capabilities to automatically find the value of w. This first problem requires simple calculations, but later problems increase in difficulty.

Cognitive tutors support learning by doing (Anderson, 1983; Anzai & Simon, 1979). Like a good personal human tutor, cognitive tutors try to minimize the support they provide. Ideally the tutor does nothing and just watches as the student works through a problem. The underlying cognitive model characterizes the cognitive objectives of the instruction, that is, the range of solution approaches that curriculum advisors tell us they would like to see students achieve. As long as students' actions are within this range of good solutions, the tutor is silent. However, when students make clear logical errors, perform actions characteristic of misconceptions or otherwise go outside the range of reasonable solutions, the tutor indicates the error. Following the principle of maximizing students' participation, errorful actions are indicated without comments that might otherwise distract them or discourage them from correcting the error themselves. When a student's action matches a *buggy rule*, a comment is made that indicates the action is an error. Buggy rules characterize common student slips or misconceptions which are often difficult for students to recognize and thus, they merit particular comment.

At any point a student can request a hint. Students' requests for hints can come at any time, but they often follow an error. To maximize the cognitive engagement of students, the first hint given is vague. Sometimes this hint is enough for a student to develop and pursue an approach on their own. If not, they can ask for

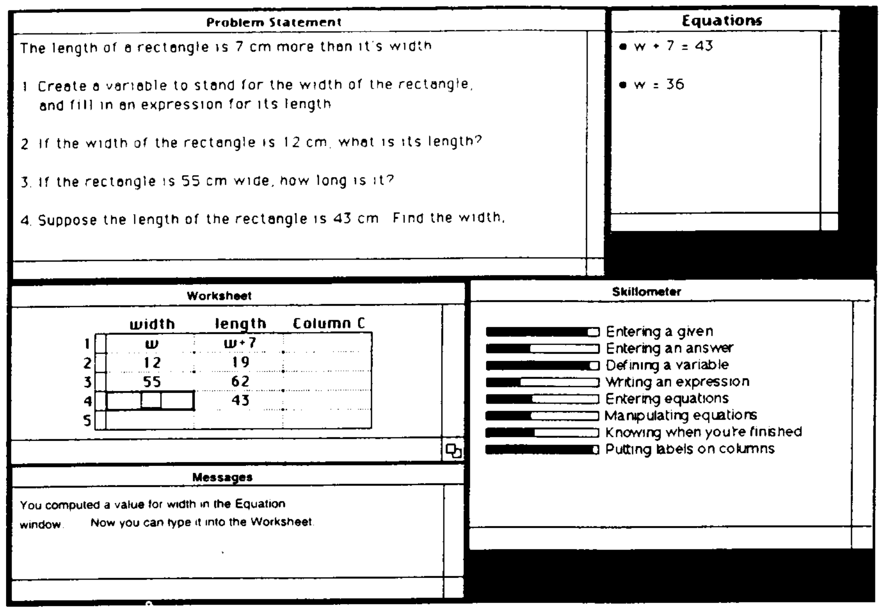

Figure 2. A Screen from the Prototype Tutor Showing the Textbook Tutor Variant

further hints. Hints get successively more specific, culminating in a suggestion to take a specific action.

Tutor Variants

Three variations of a basic algebra tutor were created that were identical except for the order of the questions within each problem. The *Textbook* variant had the questions ordered as in Table 1b with the result-unknown questions coming *after* the symbolization question, but before the start-unknown question. The *Inductive-Support* variant had the questions ordered as in Table 1c with the result-unknown problems coming first *before* the symbolization question. The third *Traditional-Plus* variant was implemented to be more like the traditional problem presentation (Table 1a) where the main focus is on translating to an algebraic expression in order to solve start-unknown problems. However, the Traditional-Plus variant was different from the problem in Table 1a in that the symbolization is explicitly requested prior to asking the start-unknown question. Further, the result-unknown questions were appended at the end of each problem so that all three types of questions appeared in each tutor variant. To summarize, the tutor variants differed only in how the result-unknown questions were positioned between, before or after the symbolization and start-unknown questions as shown in Table 2. Figure 2 shows the Textbook tutor variant–the result-unknown questions (2 & 3) come between the symbolization question (1) and the start-unknown question (4).

TEST: A PARAMETRIC EVALUATION STUDY

Hypotheses

The first hypothesis of this study is simply that this early cognitive tutor prototype would aid student learning. In particular, we expected to see students performing significantly better

Table 2. Difference between the 3 tutor variants

Tutor Variant	*Position of Result-unknown Questions*
Textbook	Between: symbolize, find result, find start
Inductive-Support	Before: find result, symbolize, find start
Traditional-Plus	After: symbolize, find start, find result

on the post-test than on the pre-test, particularly on the targeted skill of symbolization: translating problem statements into algebraic symbols. This hypothesis is predicted as a consequence of the individualized learning support facilitated by cognitive tutors: model-tracing facilitated feedback and hints and knowledge-tracing facilitated problem selection and lesson promotion.

In contrast, the central hypothesis of this study, inductive support hypothesis, is predicted as a consequence of the alternative cognitive analyses underlying the tutor variants. In particular, the hypothesis is that the inductive-support tutor will lead to greater student learning (bigger gains from pre to post-test) than the other tutor variants. This hypothesis depends on the following assumption. For inductive support to be effective, it must be the case that the two strategies involved in performing it, arithmetic translation and induce pattern (see Figure 1b), are easier to perform than the direct algebra translation strategy. Thus, the study was designed to also test this *non-normative strategy hypothesis*. Putting it in terms of the question types the hypothesis is that prior to instruction students will perform better on result-unknown questions (using arithmetic translation) than on symbolization questions (using algebra translation). The normative strategy hypothesis predicts the opposite, that result-unknown questions should be harder, because according to this view (see Figure 1a) the student must first do algebra translation to produce a symbolic expression and then evaluate this expression to get an answer.

To summarize, the study tests the following three hypotheses:

H1: Non-normative strategy hypothesis: Result-unknown problems are easier than symbolization problems because they can solved without translation to algebraic symbols.

H2:Tutor effectiveness hypothesis: The tutor will lead to significant learning from pre-test to post-test particularly on symbolization questions.

H3:Inductive support hypothesis: Students encouraged to use the inductive-support strategy will learn more than students encouraged to use normative strategies.

Method

Thirty high school students participated during the summer after having completed an algebra course in the Pittsburgh Public Schools. Students were randomly assigned into one of the three tutor-variant conditions. Students attended 1.5 to 2 hour sessions over 3-4 days. Figure 3 illustrates the experimental procedure. On the first day, students were given a 30 minute pre-test and then started working on the tutor. In subsequent sessions they worked on the tutor until they graduated from both of the tutor lessons. After graduating from lesson 2, students took a 30 minute post-test.

Test Items

There were two forms of the test, A and B, which were counter-balanced across pre- and post-testing. That is, half the subjects received form A as a pre-test and then form B as a post-test while the other half received form B as a pre-test and form A as a post-test. This guarantees that any pre to post-test improvements

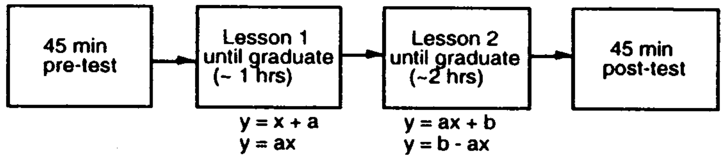

Figure 3. The Experimental Procedure

are due to learning and not a consequence of the post-test being easier than the pre-test.

Each test form had eight problems, four corresponding to tutor lesson one and four to tutor lesson two. One of each set of four problems was a traditional word problem as illustrated by the problem in Table 1a. The other three problems corresponded to the three tutor variants; that is, one had the result-unknown questions between the symbolization and start-unknown questions, another had result-unknowns first, and the third had result-unknowns last. The scoring metric used for the analysis yielded a total of 26 points for the eight problems on each test. The two traditional problems were scored one point for each correct answer. The other six problems accounted for 24 points as the four questions in each problem were scored one point each.

Results and Discussion

The Non-normative Strategy Hypothesis. To test whether students find result-unknown questions easier to solve than symbolization questions, we looked at their performance on the pre-test. Looking at 48 questions on the two forms used (2 forms × 6 problems × 4 questions), we performed a two factor ANOVA with number of operators, one or two, as one factor and question type, symbolization, result-unknown or start-unknown, as the other factor. Table 3 shows the average percent correct on the six question categories. There were significant main effects of both number of operators ($F(1, 42) = 39$, $p < .001$) and question type ($F(2, 42) = 4.9$, $p = .01$). Two operator problems were only 30% correct while one operator problems were 68%. As predicted, result-unknown questions (55%) were easier than symbolization questions (35%). A Scheffe's S post-hoc test shows this difference is statistically significant ($p = .01$). The difference is much larger on the one operator problems (79% vs. 42%) than on the two operator problems (31% vs. 28%), a significant interaction ($F(2, 42) = 3.7$, $p < .05$). The smaller difference on the two operator problems may be a consequence of a *floor effect*, that is, the experienced difficulty of two operator problems on the pre-test left little room for the students to do worse on the symbolization questions.

Students' success on the result-unknown questions even when the symbolization question is answered incorrectly indicates they tend not to use the normative strategies illustrated in Figure 1a, that is, translating to algebra and then evaluating the algebra to get the result. Instead, it appears they can solve result-unknowns directly as illustrated in Figure 1b.

Table 3. Percent correct (and standard deviations) on the six question categories

	Question Types		
Number of ops	*Symbolization*	*Result-unknown*	*Start-unknown*
one operator	42% (19%)	79% (19%)	73% (19%)
two operators	28% (12%)	31% (16%)	30% (26%)

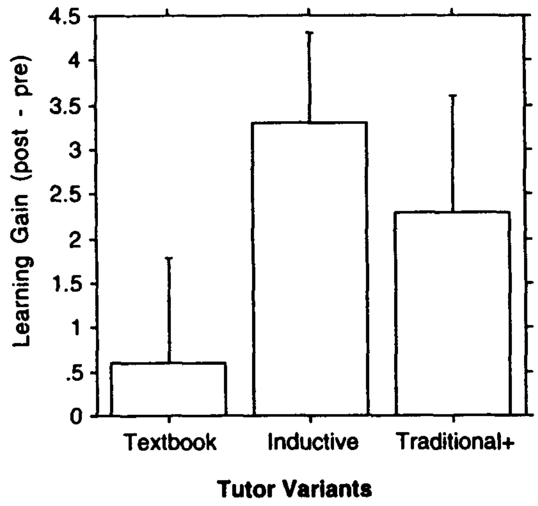

Figure 4. The Mean Learning Gains of Students in the Three Tutor Variant Conditions (error bars are 95% confidence intervals). Learning Was Greatest with the Inductive-Support Tutor Variant

The Tutor Effectiveness Hypothesis

Taken together, all three tutor variants resulted in significant learning over a relatively short period of time demonstrating the general positive effect of cognitive tutors. Students graduated the two lessons of the tutor in 3 hours on average, ranging from 0.8 to 5.0 hours. On average students scored 12.8 (out of 26) on the pre-test and 14.9 on the post-test, a large 16% improvement that is statistically significant ($F(1, 29) = 9.7$, $p < .005$).

Consistent with the focus of the tutor, the biggest gains were on the symbolization question in the test problems, while there was little gain on the result-unknown and start-unknown questions (the test-time by question-type interaction was statistically significant $F(3, 87) = 7.9$, $p < .0001$). The improvement in symbolization was large, increasing 71% from 2.1 (of 6) on the pre-test to 3.6 on the post-test. The start-unknown and particularly the result-unknown problems exercise arithmetic skills that students learn in earlier grades and that can be performed by a calculator. Since students did not practice these skills during tutoring and did not use calculators during the paper-and-pencil testing, it is no surprise that there were not large increases on the start-unknown and result-unknown problems.

The Inductive Support Hypothesis. Because time-on-task is such a critical variable in learning, it is important to first establish that students using the three tutor variants spent an equivalent amount of time on their tutor. Indeed, there was no significant difference in the total amount of time spent on three tutor variants ($F(2, 22) = .21$, $p = .8$). Students using the three tutor variants, textbook, inductive-support, and traditional-plus, averaged 2.8, 2.9, and 3.3 hours respectively.

Figure 4 shows the learning gains of students in the three tutor variants. The learning gain was computed by subtracting the pre-test score from the post-test score. As hypothesized, students that used the inductive-support tutor showed the greatest learning gains. In particular, they performed much better than the textbook group showing a 26% improvement over their pre-test scores compared to the 5% improvement of the textbook group. An ANCOVA with post-test score as the dependent measure, pre-test as the covariate, and textbook versus inductive-support tutors as a between-subjects factor, showed significant difference ($F(1, 17) = 4.4$, $p = .05$). The learning gains of the traditional-plus group fall ambiguously between the other two, neither significantly greater than the textbook group nor significantly less than the inductive-support group.

The data collected on student performance during tutoring provide corroborating evidence for the inductive-support hypothesis. We first describe how these data were collected. Each of the possible solutions to any problem involves some set of individual actions the student performs in the tutor interface. The tutor can provide feedback or hints on any action along the solution path that the student is pursuing. In addition to recording the particular solution path that a student takes, the tutor records the following information for each action: 1) the time taken to perform the action, 2) what errors, if any, are made, and 3) what hints, if any, are requested. Each action is categorized according to the cognitive model so that actions that reflect the same hypothesized skill can be investigated in aggregate. Thus, we can examine whether students appear to be getting better at a particular skill or whether the tutor variants are having different effects on students' learning of an aggregate skill.

The inductive-support hypothesis predicts that students should do better at the symbolization action as a result of working out the arithmetic (result-unknown) problems first. The data are consistent with this hypothesis. As illustrated in Figure 5A, students using the inductive-support tutor are faster at symbolization on the harder problems in Lesson 2 than students using the other tutor variants. A 2 way ANOVA with tutor variant as a between-subjects factor and lesson number as a within-subjects factor shows an overall interaction of marginal statistical significance ($F(2, 27) = 2.6$, $p = .09$). On the harder problems in lesson 2, students using the inductive-support tutor complete the symbolization step in about 28 seconds on average, while students using the textbook tutor take about 48 seconds on average to symbolize.

By itself, the symbolization speed-up is consistent with a competing hypothesis. It was shown in two previous cognitive tutors that when all other factors are equal, actions tend to get faster as a student gets further into a problem (Anderson, 1993, pp. 152 & 177)–a *later-faster* effect. This alternative hypothesis would suggest that symbolization is faster in inductive-support simply because it is second in the sequence of major actions in that condition while it is first in the textbook and traditional-plus conditions. The trouble with this later-faster account, however, is that it makes similar predictions regarding the ordering of the other major actions (result-unknown and start-unknown) that are not consistent with the data. According to the later-faster account, result-unknown actions should be slowest in inductive-support, where they are performed first, and fastest in traditional-plus, where they are performed last. Similarly, start-unknown actions should be slowest in traditional-plus where they are second and faster in inductive-support and textbook where they are last. As shown in Figures 5B and 5C there are no significant differences between tutor variants on result-unknown latencies nor on start-unknown latencies. Thus, the later-faster hypothesis can be rejected and the inductive-support hypothesis appears to be the only consistent explanation of faster symbolization of students in the inductive-support condition.

As shown in Figure 5B, students in all conditions had more difficulty with the result-unknown problems in lesson 2 than those in lesson 1. This difference is not surprising given that lesson 2 problems involved two arithmetic operations (multiplication and either addition or subtraction) and lesson 1 problems involved just one operation (either multiplication or addition).

We would expect a similar difference on start-unknown problems (Figure 5C) if students were

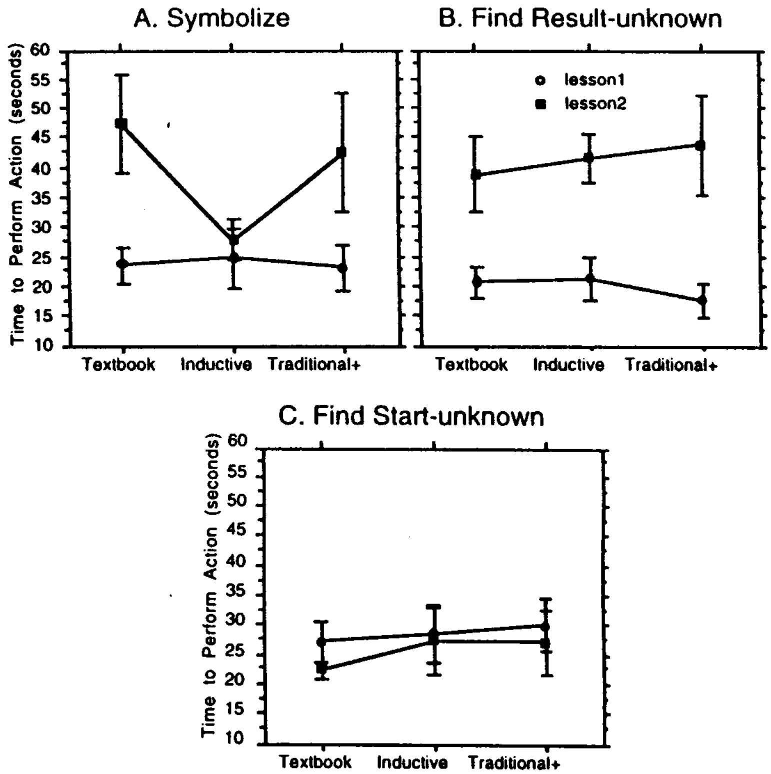

Figure 5. Student Performance on Question Types During Tutoring for Each Tutor Variant. The Only Difference is Shown in Panel A, Where Inductive-Support Resulted in Faster Symbolization in Lesson 2

solving these problems on their own, however, the tutor provided students with a symbolic calculator that they used to solve start-unknown problems. The times in Figure 5C reflect the time it takes students to select the "solve" menu item in the symbolic calculator. (The process of writing the equation, recall, is captured in the symbolization time shown in Figure 5A.) Although lesson 2 problems are otherwise more difficult than lesson 1 problems, it is not any more difficult in lesson 2 to select the "solve" menu item. Thus, we see no difference in Figure 5C between lesson 1 and lesson 2 times to solve the result-unknown problems.

There were also no significant differences between the tutor variants in students' accuracy within any question type. The only significant on-line difference between tutor variants, that symbolization is easier after doing analogous arithmetic, appears uniquely consistent with the inductive-support hypothesis.

REDESIGN: TOWARDS PAT

The results from this parametric study provide support for the inductive-support hypothesis that students are best taught to build mathematical abstractions on a foundation of common sense achieved through concrete reasoning. Nevertheless, it appeared that some students occasionally had difficulty in making the transition from concrete reasoning (e.g., 55 + (1/4) * 20) to the abstract formula (e.g., 55 + (1/4)*x*). This difficulty seemed to stem from the fact that the arithmetic steps were performed in students' heads. Informally we have observed that students are often capable of coming up with answers to concrete questions and yet have significant difficulty in articulating how they did so.

In redesigning the system, we hypothesized that if we have students write down the arithmetic steps that lead to answers, these concrete expressions could be the source for *visually-supported* induction of the abstract expression. Figure 6 shows the Problem Statement and Worksheet windows at the beginning of a problem in the current system, called PAT (Koedinger, Anderson, Mark & Hadley, 1995). This mobile phone problem illustrates an effort to create problems with real world currency from which students can better learn how to apply algebra and appreciate its relevance.

Figure 7 shows the student having difficulty in coming up with an expression. The student requests a hint and PAT's response is shown in the Messages window. The suggestion is to use the Equation Support window and try to articulate how to get the resulting cost for a 2 minute phone call.

As shown in Figure 8, the Equation Support window scaffolds the inductive support strategy. Here, students work out the arithmetic recipe for a sequence of small integer values (2, 3, 4). For example, they show how to find the cost for a 2 minute call, then for a 3 minute call and finally for a 4 minute call. Seeing the results of these steps, the abstraction to algebraic symbols is fairly straight-forward. The student simply needs to notice what is varying and replace it with the single letter variable.

In addition to the results of this parametric study, we were further encouraged to implement the Equation Support window when we observed that, without any prior discussion with us, classroom teachers were using this same strategy when they helped students to symbolize.

In apparent contrast with our results, PAT's Worksheet window has the Formula row above the rows for the specific questions (labeled 1-5 in Figure 8)–appearing more like the Forester textbook than the inductive support tutor variant. The reason for this is that we wanted the Worksheet to have the functionality and look of a spreadsheet. In later lessons, the Worksheet window operates like a spreadsheet whereby as soon as the student enters the expression, result-values (e.g., 15 in row 1) are automatically computed as soon as the corresponding start-value (e.g., 10) is entered. We made this compromise with respect to the inductive-support hypothesis, but in all other ways the presentation and tutoring advice are consistent with the inductive support hypothesis. First, as in the inductive support condition in this experiment, the problem statement has the concrete questions first and the request to symbolize comes

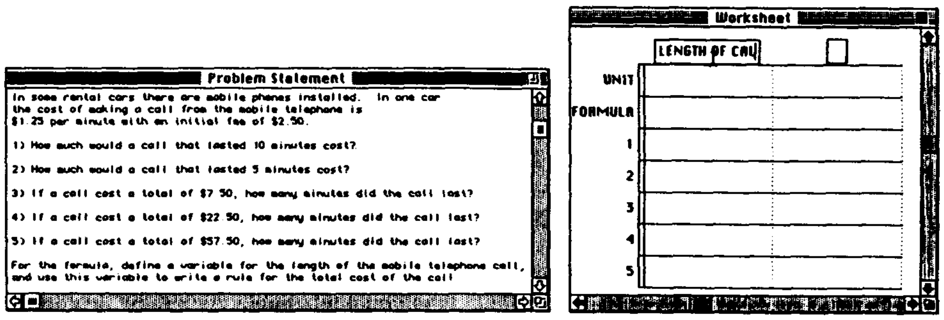

Figure 6. Windows from PAT. The Student Reads the Problem Situation from the Problem Statement Window and Begins by Labeling the Columns in the Worksheet

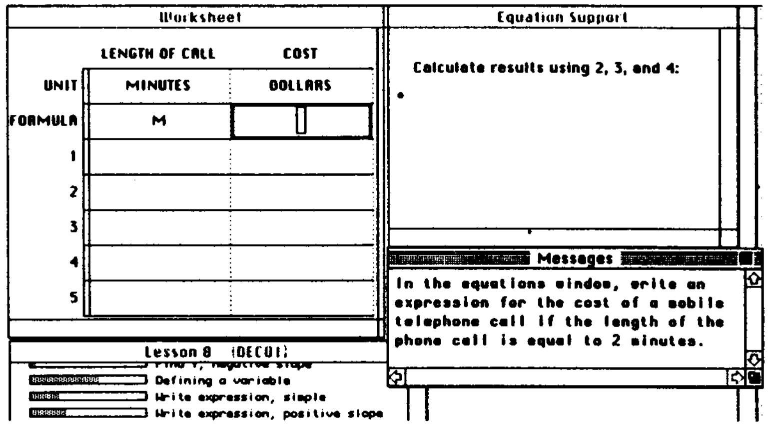

Figure 7. Some PAT Windows After Entering the Column Labels, Units, and
Defining a Variable. The Equation Support Window Scaffolds the Inductive Symbolization Strategy

afterwards. Second, when a student asks for a hint, the system will recommend answering the concrete questions before writing the expression. Third, when students first use the system, a printed handout walks them through the first problem in the recommended order, that is, concrete instances before abstraction. Finally, we recommend that teachers reinforce this order when they are helping students.

This example is a nice illustration of the conflicts that can arise in the process of principled design of interactive learning environments. Two principles were in conflict:

1. Design the interface to support student learning.
2. Design the interface for external validity, that is, so that transfer to working with

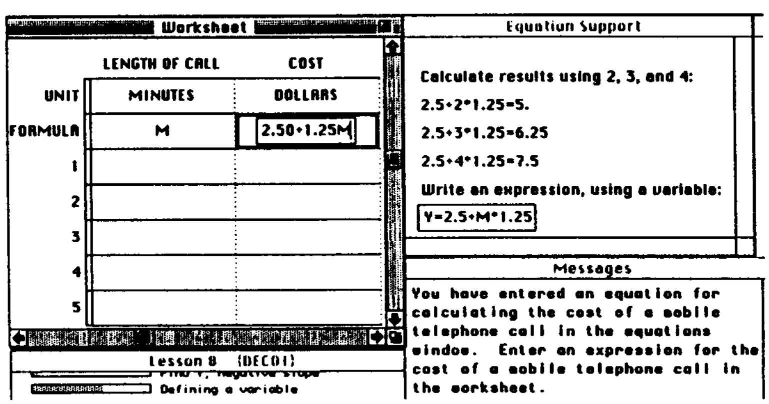

Figure 8. The Completion of the Equation Support and the Copying Over of the Resulting Expression in the Worksheet

tools in the target environment (e.g., workplace) is most natural.

The evidence for the effectiveness of inductive support suggested putting expression entry at the bottom of the Worksheet, while the desire to have the look and feel of a spreadsheet suggested putting it at the top. In our resolution of this conflict, principle 2 took primacy–the interface was made to be more externally valid. However, to achieve principle 1 all aspects of the surrounding instructional context, the tutoring approach being just one of them, are consistent with the inductive support hypothesis. This may prove to be a reasonable strategy for addressing conflicts of this kind in general.

Further laboratory experiments with PAT are in progress. In addition, PAT is now being used by over 500 students in classrooms in three Pittsburgh Public schools. Our early in vivo evaluation has shown dramatic learning gains by students in classes using PAT compared with traditional algebra classes (Koedinger, Anderson, Mark, & Hadley, 1995).

CONCLUSION

We have presented the early stages of a principled design of a cognitive tutor for algebra. Initial design was driven by a cognitive analysis of the kinds of algebra problems presented in the first chapters of the popular Forester textbook. Our first pass cognitive analysis attempted to identify the highest level processing elements that lead to student answers. We focused at the strategic level in this first pass. We identified the normative strategies implied by the textbook, wrote ACT theory production rules to perform these normative strategies, and implemented a prototype cognitive tutor to support students in learning them.

The cognitive analysis and tutor development was particularly focused on the *symbolization* process, that is, producing symbolic expressions given verbal descriptions of quantitative relationships, because we felt this step was the most important and most difficult. This symbolization process has always been a critical part of mathematical competence but is gaining an increasing role as technology takes over more and more of the tasks of symbol manipulation. Learning how to model or "mathematize" situations is appropriately a major emphasis of mathematics reforms in the US. Symbolization is a critical part of mathematizing that deserves special attention as an end in itself, and not just as a means to finding numeric solutions as is its traditional role.

In the process of tutor development and through interaction with teachers and informal observation of students, it appeared that students might have alternative, non-normative strategies for symbolization. The normative strategy for symbolization, the *algebra translation* strategy, involves the student learning to directly translate from words to algebraic symbols. In contrast we proposed an alternative *inductive-support* strategy that introduces an intermediate step in the symbolization process which may aid student performance and learning. Instead of going directly from words to symbols, we hypothesized students could first perform *arithmetic translation*, as they do when they answer simple numeric questions (result-unknowns), and then *induce* from the pattern of arithmetic operations they performed the analogous symbolic expression.

This inductive-support strategy led to a couple of predictions in contrast with the normative strategy, first, that students should initially be better at answering numeric result-unknown questions than at symbolizing and, second, that a cognitive tutor that encouraged inductive-support strategies would lead to better student learning than the initial tutor designed to encourage the normative strategies. A study was designed as both a formative evaluation of our early prototype (would students learn from it) and as a parametric test of the inductive-support versus normative predictions. The pre-tests results of this study showed that indeed symbolization is a difficult step for students and that students are initially able to correctly answer numeric questions without first symbolizing. The overall pre-test to post-test gains by all students indicated that the core cognitive tutor was an effective learning aid. The gains were largest on the symbolization questions, the real focus of the tutor.

In comparing the tutor variants, we found that students learned significantly more from the inductive-support tutor than the textbook tutor. The advantage appeared both in aiding student performance during tutoring–inductive support students were faster to symbolize–and in leading to increased student learning–inductive support students showed greater test score gains from the pre-test to the post-test. These results indicate that inductive support may provide an instructional approach that is more effective than alternatives in helping students acquire the difficult and important skill of symbolizing.

The inductive support approach has been incorporated in the more complete PAT system that evolved from the tutor prototype tested in this study and which has been shown to be an effective part of improved algebra instruction at the high school (Koedinger, Anderson, Hadley, & Mark, 1995) and college (Koedinger & Sueker, 1996) level. This study served a critical formative evaluation step and, in particular, provided guidance in system redesign that resulted in a new "Equation Support" window and the corresponding tutor knowledge that guides students in using and learning the inductive support strategy.

Pervasive Situatedness or Induction of Abstract Knowledge from Concrete Experience

There has been much emphasis in recent cognitive science and learning environment research on the role of authentic problem solving situations in student performance and learning (e.g., Cognition and Technology Group, 1990). The extreme view is that all instruction should be situated in authentic or real world experience. Although there appear to be advantages of situated or anchored instruction, the risks and limitations of situatedness need to be identified.

Situating or anchoring mathematics in real world experience is clearly important for helping young adults both to appreciate the relevance of mathematics to everyday experience and to develop the cognitive skills for applying mathematics to real problems. A focus on connecting mathematics to real world problem situations is a critical feature of PAT as well as of the surrounding curriculum and student assessments (see Koedinger, Anderson, Mark, & Hadley, 1995). Such a focus is not only helpful to students, but is also critical to communicate and reinforce to teachers, administrators and parents the importance and relevance of mathematics to out-of-school concerns. The widespread distribution and popularity of the Jasper Woodbury series (Cognition and Technology Group, 1993) is indicative of the strong desire in some schools and communities to find compelling justification for mathematics instruction.

In addition to these benefits in situating mathematics, there also some real risks and limitations. Requiring all instruction to be in the context of real world or authentic project situations, as some advocate, has the potential dangerous consequence of slowing learning to a snail's pace of case-by-case concrete investigations. Exclusive use of large, authentic projects can waste precision learning time on non-problematic aspects of student performance and provide few opportunities for growth on the more sophisticated and problematic aspects. Such an approach is like learning to play tennis by always playing games and never practicing one's serve. A second risk is that over-emphasis on situated problems can lead students to acquire overly-situated skills that can only be applied in limited situations (e.g., Williams, Bransford, Vye, Goldman, & Carlson, 1992). The categorical word problem approach that NCTM has cautioned against (coin, work, mixture, etc.) could appear again but with different, more real, categories (profit margin, population decline, school fund raising). What is needed in addition to situated problems is instruction in cognitive strategies that generalize across problem categories and encourage students to develop abstract skills that will make them more flexible in the face of novelty.

The inductive-support condition appeared to help students to develop a more general strategy for aiding the symbolization process. This strategy is not tied to the wording of any particular problem situation or problem type. Reminiscent of one of Polya's (1957, p. xvii) classic heuristics for effective problem solving: "Could you imagine a more accessible related problem?", the inductive support strategy can yield more flexible and adaptive problem solving in the face of novelty.

More cognitive research is needed to better understand the relative merits of two alternative approaches to instruction: concrete project-based instruction within authentic situations versus abstract, principle-based instruction within small targeted exercises. The focus should be on identifying the conditions that best achieve the benefits and avoid the limitations of both approaches. Such conditions must be specified at a level of detail sufficient to guide the principled design of effective learning environments.

REFERENCES

Anderson, J. R. (1983). *The Architecture of Cognition.* Cambridge, MA: Harvard University Press.

Anderson, J. R. (1993). *Rules of the Mind.* Hillsdale, NJ: Erlbaum.

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *The Journal of the Learning Sciences, 4* (2), 167-207.

Azai, Y., & Simon, H. A. (1979). The theory of learning by doing. *Psychological Review, 86,* 124-140.

Briars, D. J., & Larkin, J. H. (1984). An integrated model of skill in solving elementary word problems. *Cognition and Instruction, 1,* 245-296.

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.

Cognition and Technology Group (1990). Anchored instruction and its relationship to situated cognition. *Educational Researcher, 19*(6), 2-10.

Cheng, P. W., & Holyoak, K. J. (1985). Pragmatic reasoning schemas. *Cognitive Psychology, 17,* 391-416.

Corbett, A. T., Anderson, J. R., & O'Brien, A. T. (1995). Student modeling in the ACT programming tutor. In P. Nichols, S. Chipman, & R. Brennan (Eds.), *Cognitively diagnostic assessment* (pp. 19-41). Hillsdale, NJ: Erlbaum.

Corbett, A. T., Koedinger, K. R., & Anderson, J. R. (in press). Intelligent tutoring systems. To appear in *The Handbook of Human Computer Interaction*, 2nd Edition.

Cognition and Technology Group at Vanderbilt (1993). The Jasper Experiment: Using video to furnish real-world problem-solving contexts. *Arithmetic Teacher: Mathematics Education Through The Middle Grades, 4,* 474-478.

Cummins, D. D., Kintsch, W., Reusser, K., & Weimer, R. (1988). The role of understanding in solving word problems. *Cognitive Psychology, 20,* 405-438.

Forester, P. A. (1984). *Algebra I: Expressions, equations, and applications.* Menlo Park, CA: Addison-Wesley.

Koedinger, K. R. (1991). On the design of novel notations and actions to facilitate thinking and learning. In *Proceedings of the International Conference on the Learning Sciences,* (pp. 266-273). Charlottesville, VA: Association for the Advancement of Computing in Education.

Koedinger, K. R., & Anderson, J. R. (1990). Abstract planning and perceptual chunks: Elements of expertise in geometry. *Cognitive Science, 14,* 511-550.

Koedinger, K. R., & Anderson, J. R. (1991). Interaction of deductive and inductive reasoning strategies in geometry novices. In *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society,* Hillsdale, NJ: Erlbaum.

Koedinger, K. R., & Anderson, J. R. (1993). Reifying implicit planning in Geometry: Guidelines for model-based intelligent tutoring system design. In Lajoie, S. P. & Derry, S. J. (Eds.), *Computers as Cognitive Tools.* Hillsdale, NJ: Erlbaum

Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1995). Intelligent tutoring goes to school in the big city. In *Proceedings of the 7th World Conference on Artificial Intelligence in Education,* (pp. 421-428). Charlottesville, VA: Association for the Advancement of Computing in Education.

Koedinger, K. R., & Sueker, E. L. F. (1996). PAT goes to college: Evaluating a cognitive tutor for developmental mathematics. In *Proceedings of the Second International Conference on the Learn-*

ing Sciences. Charlottesville, VA: Association for the Advancement of Computing in Education.

Koedinger. K. R.. & Tabachneck. H. J. M. (1995). Verbal reasoning as a critical component in early algebra. Presented at the 1995 annual meeting of the American Educational Research Association, San Francisco, CA.

Lewis. C. H. (1988). Why and how to learn why: Analysis-based generalization of procedures. *Cognitive Science, 12*, 211-256.

National Council of Teachers of Mathematics (1989). *Curriculum and Evaluation Standards for School Mathematics*. Reston, VA: The Council.

Pelavin. S. (1990). Changing the odds: Factors increasing access to college. Report #003969. New York: College Board Publications.

Polya. G. (1957). *How to Solve It: A New Aspect of Mathematical Method*, 2nd edition, Princeton, NJ: Princeton University Press.

Riley. M. S., & Greeno, J. G. (1988). Developmental analysis of understanding language about quantities and of solving problems. *Cognition and Instruction, 5*, 49-101.

Tabachneck. H. J. M., Koedinger. K. R., & Nathan. M. J. (1995). A cognitive analysis of the task demands of early algebra. In *Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum.

Whitehead, A. N. (1929). *The aims of education*. New York: MacMillan.

Williams, S. M., Bransford, J.D., Vye, N.J., Goldman, S. R., & Carlson, K. (1992). Positive and negative effects of specific knowledge on mathematical problem solving. Paper prepared for the annual meeting of the American Educational Research Association, San Francisco, CA, April, 1992.