Cognitive Tutors: Technology Bringing Learning Science to the Classroom

Kenneth R. Koedinger

Albert Corbett

Introduction

Individual tutoring is perhaps the first instructional method. It dates back at least to Socrates and the Socratic method. Although one-to-one tutoring by expert human tutors has been shown to be much more effective than typical one-to-many classroom instruction (Bloom, 1984), it has not been economical to provide every child with an individual tutor. Lectures and books became pervasive in education to spread knowledge at lower cost. However, increasing capabilities of computer hardware and software have been creating new opportunities to bring one-to-one tutoring to more students. Furthermore, computer technology provides an opportunity to systematically incorporate advances in learning science into the classroom, to test associated principles of learning, and best adapt them to the needs of students and teachers.

Early attempts to use computers for instruction included Computer-Aided Instruction (Eberts, 1997) and then, later Intelligent Computer-Aided Instruction or Intelligent Tutoring Systems (Sleeman & Brown, 1982; Wenger, 1987; Corbett, Koedinger, & Anderson, 1997). Computer-based instruction has been shown to be effective in increasing student learning beyond normal classroom instruction (e.g., Kulik & Kulik, 1991), however, not to the level of human tutors (Bloom, 1984). Early attempts at Intelligent Tutoring Systems included mimicking Socratic dialog in teaching electronics trouble-shooting, adding intelligent questioning to an existing Computer-Aided Instruction system for

learning South American geography, adding tutoring strategies to an existing "expert system" for medical diagnosis, and adding tutoring strategies to an existing educational game for mathematics (Sleeman & Brown, 1982).

In a parallel development that dates back even earlier, cognitive theories of human learning, memory, and problem solving were being implemented as computational models with computers (Newell & Simon, 1972). In the mid-1980s, John R. Anderson and colleagues merged these two strands and introduced a more interdisciplinary approach to Intelligent Tutoring System development and testing (Anderson, Boyle, & Reiser, 1985) that added the discipline of cognitive psychology to the discipline of artificial intelligence that had previously been the prime mover. The Intelligent Tutors emerging from this approach were constructed around computational *cognitive models* of the knowledge students were acquiring and began to be called "*Cognitive Tutors*" (Anderson et al., 1995). These cognitive models represent learner thinking or *cognition* in the domain of interest, whether it is algebra, programming, scientific reasoning, or writing essays. The cognitive model also includes a representation of the kinds of early learner strategies and misconceptions that are steps in the trajectory from novice to expert.

Full-scale Cognitive Tutors have been created to help students learn in a variety of domains including middle and high school mathematics (Koedinger, Anderson, Hadley, & Mark, 1997; Koedinger, 2002), computer programming (Anderson et al, 1995; Mathan & Koedinger) and college-level genetics (Corbett et al 2005). Cognitive Tutors typically speed learning or yield greater learning relative to conventional problem-based instruction (Anderson, et al., 1995) and

approach the effectiveness of good human tutors (Corbett, 2001). The most widely distributed Cognitive Tutor is one for algebra, which is part of a complete course for high school algebra and in 2004-2005 was in use in some 2000 schools across in the United States. As described later in the paper, students in Cognitive Tutor Algebra I have been shown to score twice as high on end-of-course open-ended problem solving tests and 15% higher on objective tests as students enrolled in a traditional Algebra course. A few of these schools are high performing, resource rich suburban schools, but most of them are urban or rural schools, with average teachers and with a relatively large number of economically disadvantaged, minority or learning disabled students. We roughly estimate some half million students have used the tutor for a total of about 20 million student-hours.

Cognitive Tutors Provide Aspects of Human Tutoring

Cognitive Tutors support learning by doing, an essential aspect of human tutoring. Learning by doing is the idea of putting students in performance situations whereby the objective concepts and skills can be applied and instruction can be provided in the context of or in response to student needs[1]. Cognitive Tutors accomplish two of the principal tasks characteristic of human tutoring: (1) monitoring the student's *performance* and providing context-specific instruction just when the individual student needs it, and (2) monitoring the student's *learning* and selecting problem-solving activities involving knowledge goals just within the individual student's reach.

This monitoring of students' performance and learning makes use of the cognitive model and two key algorithms, model tracing and knowledge tracing. In

model tracing, the cognitive tutor runs the cognitive model forward step-by-step along with the student to follow the student's individual path through complex problem spaces, providing just-in-time accuracy feedback and context-specific advice. In *knowledge tracing*, the tutor employs a simple Bayesian method of estimating the student's knowledge and employs this student model to select appropriate problems.

Chapter Overview

In the following section we describe Cognitive Tutors and their foundation in ACT-R theory. Extensive Cognitive Tutor research has served both to validate and modify the ACT-R cognitive architecture model (cf. Anderson & Lebiere, 1998) and we review six general principles of intelligent tutor design that were derived from this theoretical framework (Anderson et al., 1995). While ACT-R theory provides an important cognitive modeling framework, it does not prescribe course curriculum objectives and activities, it cannot precisely anticipate the prior knowledge that students bring with them to a course or a problem-solving activity, and it cannot prescribe scaffolding activities to help students develop a deep understanding of domain knowledge. In the final section of the chapter, we describe the learning science principles and methods that we have employed to address these instructional design questions.

==INSERT FIGURE 1 HERE==

Cognitive Tutor Algebra: A Brief Example

A screen shot of a unit in Cognitive Tutor Algebra is shown in Figure 1. Cognitive Tutors tend to have relatively rich graphical user interfaces that provide a workspace in which students can demonstrate a wide variety of problem solving

behavior. The workspace changes as students progress through units. The workspace in Figure 1 includes a problem scenario window in the upper left where students are presented with a problem situation, often with real facts or data, that they are expected to analyze and model using the tools in the workspace. The tools illustrated in Figure 1 are the Worksheet, Grapher, and Solver. In this unit, the Worksheet has automated features like a spreadsheet. Once students write the algebraic expression for the height "67+2.5T" given the time, then the worksheet computes a height value (e.g., 117) when a time value is entered (e.g., 20). In earlier units, the Worksheet does not have these automated features, but is more like a table representation on paper and students must demonstrate they can perform the steps on their own.. Similarly, the Grapher and Solver tools change as students advance through tutor units. Initially these behave much like blank pieces of paper where students do all the work. Later these tools begin to automate lower level skills, like plotting points or performing arithmetic and let students focus on acquiring higher-level concepts and skills, like deciding what the symbolic function to graph or what algebraic manipulation to perform. As students work, the Cognitive Tutor monitors their performance and may provide just-in-time feedback or on-demand solution-sensitive hints in the hint window. The Cognitive Tutor also monitors student learning and displays these results in the Skills chart, shown in the top center of Figure 1.

It is critical to consider the social context of use of any technology or educational innovation and Cognitive Tutors are no exception. We have tended to create complete Cognitive Tutor courses whereby we apply learning sciences theory to develop instructional materials, like consumable textbooks, in addition

to Cognitive Tutor software. Virtually all schools using our mathematics Cognitive Tutors also use the curriculum and text materials. The typical procedure is to spend 2 days a week in the computer lab using the Cognitive Tutor software and 3 days a week in the regular classroom using our text materials. In the classroom, learning is active, student-centered, and focused primarily on learning by doing. Teachers spend less time in whole-group lecture and more time facilitating individual and cooperative problem solving and learning. In the classroom, students often work together in collaborative groups to solve problems similar to those presented by the tutor. Teachers play a key role in helping students to make connections between the computer tools and paper and pencil techniques.

Learning Sciences Theory Behind Cognitive Tutors

Cognitive Tutors are based on the ACT-R theory of learning and performance (Anderson & Lebiere, 1998). The theory distinguishes between implicit performance knowledge, called "procedural knowledge" and explicit verbal knowledge and visual images, called "declarative knowledge". According to ACT-R, performance knowledge can only be learned *by doing*, not by listening or watching. In other words, it is induced from constructive experiences -- it cannot be directly placed in our heads. Such performance knowledge is represented in the notation of if-then production rules that associate internal goals and/or external perceptual cues with new internal goals and/or external actions. Examples of English versions of production rules are shown in Table 1.

==INSERT TABLE 1 HERE==

Production rules characterize how both advanced and beginning students think or reason in a domain. Students may acquire informal, heuristic, or incorrect patterns of thinking that are different than the concepts and rules that are normatively taught or presented in textbooks. Learning sciences researchers have identified "informal" or "intuitive" forms of thinking that students may learn implicitly and outside of school (cf. Lave, Murtaugh, de la Rocha, 1984; Resnick, 1987). Production rules can represent such thinking patterns as illustrated by production #1 in Table 1, which represents an informal alternative to the formal approach of using algebraic equations like "8x = 40" (cf., Koedinger & Nathan, 2004). Production rules can also represent heuristic methods for discovering approaches to solutions (Polya, 1957). Production #2 in Table 1 does not suggest any particular operation per se, but characterizes how a good problem solver may think through a plan of action before selecting a particular operation or theorem to apply. Non-traditional strategies can be represented in production rules, as illustrated by #3 in Table 1, which characterizes the use of a graphical rather than symbolic strategy for solving an equation.

The if-part of a production rule can help identify when the knowledge students acquire is not at the right level of generality. For instance, production #4 in Table 1 is too specific—it represents when students can combine like terms in an equation when coefficients are present (e.g., 2x+3x -> 5x) but not when a coefficient is missing (e.g., x – 0.2x). Alternatively, students sometimes acquire productions that are too general. Production #3 in Table 1 represents how students may learn to combine numbers by the operator between them (e.g., 2*3+4=x ->

6+4=x) without acquiring knowledge that prevents order of operations errors (e.g., x*3+4=10 -> x*7=10).

The Cognitive Model and Model Tracing in Cognitive Tutors

Developing Cognitive Tutor software involves the use of the ACT-R theory and empirical studies of learners to create a "cognitive model". A cognitive model uses a production system to represent the multiple strategies students might employ as well as their typical student misconceptions. To take a simplified example from an Algebra equation solving problem, Figure 2 depicts 3 productions that can apply in solving the equation 3(2X + 5) = 9. The production rule in Strategy 1 distributes (multiplies) “3” across the sum (2X + 5). The production rule in Strategy 2 divides both sides of the equation by “3.” The third rule is a “buggy” production that represents a misconception (cf., Matz, 1982) and fails to fully distribute the “3” across the sum (2X + 5).

==INSERT FIGURE 2 HERE==

By representing alternative strategies for the same goal, the Cognitive Tutor can follow different students down different problem solving paths of the students’ own choosing, using an algorithm called “model tracing.” Model tracing allows the Cognitive Tutor to trace each student’s problem-solving steps and provide individualized assistance that is just-in-time and sensitive to the students’ particular approach to a problem. When a student performs a step, it is compared against the alternative next steps that the cognitive model generates. There are three categories of response. For example, in Figure 2, if a student’s problem solving action matches either strategy 1 or strategy 2, the tutor highlights the step as correct and the student and tutor move on to the next step. Second, if the

student action, like “6x + 5 = 9”, is matched by a buggy production the tutor highlights the step as incorrect and presents a feedback message, like "You need to multiply 5 by 3 also." This message is generated from a template attached to the buggy rule, with the variables “c” and “a” getting context-specific values from the matching of the production rule to the current situation. Third, if the student performs a problem-solving action that does not match the action of any rule in the cognitive model, the tutor simply flags the action as an error, for instance, by making the text red and italicized.

At any time the student can request a hint (e.g., by clicking on the “?” button shown in figure 1). Again the tutor runs the model forward one step, selects one of the model productions that matches and presents advice text attached to the production. For instance, the hint associated with strategy 1 in Figure 2 will say “Distribute 3 across the parentheses” because the production variable “a” has the value 2 in this case.

Knowledge Tracing in Cognitive Tutors

ACT-R theory holds that knowledge is acquired gradually and the brain essentially keeps statistics on the frequency, recency, and utility of knowledge components including production rules (Anderson & Lebiere, 1998). The Knowledge Tracing algorithm in Cognitive Tutors monitors students’ gradual acquisition of production rules across problem solving activities. At each opportunity to apply a production rule in problem solving, the tutor updates its estimate of the probability the student knows the rule based on whether the student applies the rule correctly. Knowledge Tracing employs a Bayesian update and has been shown to predict students’ performance and posttest accuracy

(Corbett & Anderson, 1995). These probability estimates are displayed with "skill bars" in the computer tutor interface (see the lower right in Figure 1). The Cognitive Tutor uses these estimates to determine when a student is ready to move on to the next section of the curriculum, thus adapting the pacing of instruction to individual student needs. New problems are individually selected for students to provide more instruction and practice on the skills that have not yet been mastered (i.e., the ones for which the estimate is less than 95% that the student knows that skill).

Why Production Rule Cognitive Models are Powerful

A key feature of production rules is that they are *modular*, that is, they represent knowledge components that can be flexibly recombined. It does not matter how a student reached the state of "3(2x + 5) = 9" shown in Figure 2. It may have been the result, for instance, of translating a story problem into this equation or of simplifying a more complex equation (e.g., "3(2x + 5) + 10 = 19") into this form. In any case, the production rule model is always applied to the current state of the problem regardless of how the student reached the current state.

This modularity makes developing Cognitive Tutors more feasible as the production rules can be reused and recombined in different ways to follow students in a potentially infinite variety of problems within a course unit or even across courses (e.g., the equation solving cognitive model is used in the Geometry Cognitive Tutor as well as in Algebra). In addition to facilitating development, modularity is a key scientific claim of ACT-R that yields empirically testable predictions. For instance, knowledge will transfer from a learning activity to an

assessment activity to the extent that the kind and number of productions needed in the learning activity are also applicable in the assessment activity (Singley and Anderson, 1989).

Model and Knowledge Tracing Implement Features of Human Tutoring

Model and knowledge tracing algorithms implement key features of human tutoring and apprenticeship training (cf., Bloom, 1984; Collins, Brown & Newman, 1989; McArthur, Stasz, & Zmuidzinas,1990; Vygotsky, 1978). The tutor gives the student a task and monitors how well the student is performing the task. Model tracing is a form of such monitoring. When the student strays too far from the tutor's model of desired performance, the tutor may intervene and provide feedback. If the student is stuck, the tutor can provide hints or performance assistance based on his or her own domain expertise. The cognitive model provides the domain expertise or model of desired performance in a Cognitive Tutor. After the student finishes the task, the tutor selects a next task based on the tutor's sense of what the student knows and does not know. Knowledge tracing implements a method for determining, over time, what each student seems to knows and not know.

Principles and Methods for Cognitive Tutor Design

Cognitive Tutor Design Principles

In 1995, we published a report on the status of the lessons learned from Cognitive Tutor development (Anderson,et al., 1995). We described some general Cognitive Tutor design principles consistent with ACT-R and our research and development experience to that date. Here we review the status of the six most frequently used of those principles. Table 2 lists these six principles,

slightly rephrased based on our experiences since the 1995 paper. We will briefly describe these principles and then provide more extended examples of two of them.

==INSERT TABLE 2 HERE==

1. Represent student competence as a production set

The principle "represent student competence as a production set" suggests that the instructional designer guides design based on an analysis not of domain content per se, but of the way in which students think about the content. Acquiring competence in a domain is complex and we tend to be surprisingly unaware of the immense number of details and subtle decision capabilities we *implicitly* acquire on the way to expertise (Berry & Dienes, 1993). Complex tasks like reading become second nature to us with time and we forget or perhaps are never quite aware of the learning experiences and resulting knowledge changes that led to such competence. Skinner (1968) estimated that to perform at 4th grade level in math, a student must acquire about 25,000 “chunks” of knowledge (p. 17). Production rules provide a way to represent such chunks of knowledge and decision capabilities.

The *modularity* of production rules in a production set predicts that we can diagnose specific student weaknesses and focus instructional activities on improving these. The *context-specific* nature of production rules means that instruction cannot be effective if it does not connect knowledge with its contexts of use. Students need true problem solving experiences to learn the if-part of productions, the conditions for appropriate use of domain principles.

2. Provide instruction in a problem-solving context

A fundamental assumption of ACT-R is that people learn by doing as the brain generalizes from one's explicit and implicit interpretations or "encodings" of one's experiences. It is not the information or even the instructional activities students are given per se that matters, but how students experience and engage in such information and activities that determines what knowledge they construct from them. Thus, another principle in Anderson et al. (1995) was "provide instruction in the problem-solving context". This principle is consistent with the learning sciences finding that instruction should be *situated* in *authentic* tasks.

3. Communicate the goal structure underlying the problem solving

Among the formidable challenges facing novice problem solvers in complex problem solving is decomposing an initial problem statement into successive subgoals and keeping track of these subgoals (Singley, 1990). The underlying goal structure of a problem solution often remains hidden in traditional problem-solving representations. We have employed two methods for making the goal structure explicit. First, we develop interfaces that make the goal structure visible in the problem-solving interface (Collins et al., 1989). The most notable examples of this strategy are the geometry proof tutors (Koedinger & Anderson, 1993). A variety of studies have shown that explicit goal-structure scaffolding in the problem-solving interface can speed problem solving even in simple problems (Corbett & Trask, 2000) and result in better learning outcomes in more complex problem solving (Scheines and Sieg, 1994; Singley,1990). Second, the underlying goal structure of the problem can be communicated through help messages. Typically in model tracing tutors, the first level of help is

a description of the current goal in the context of the overall problem. Subsequent help messages advise on how to achieve the goal.

4. Promote a correct and general understanding of the problem-solving knowledge

In learning to problem solve, students construct production rules based on their own, perhaps idiosyncratic understanding or *encoding* of problem-solving activities and examples. For example novices have been shown to encode physics problems based on superficial features of the problems, rather than the underlying physical principles that apply (Chi, Feltovich and Glaser, 1981). In geometry problem solving, students notoriously will conclude that angles are equal in measure because they look equal rather than because structurally they must be (Aleven & Koedinger, 2002). As illustrated above, students can acquire overly general productions that generate errors outside of the situations very similar to those in which they were acquired and overly specific rules that fail to transfer as expected across problem solving contexts. We have successfully deployed different strategies to help students generate a more general understanding and illustrate one below.

5. Minimize working memory load that is extraneous to learning

It has been documented that errors in complex problem solving can stem from loss of information from working memory (Anderson & Jeffries, 1988) and that high working memory load or "cognitive load" can impede learning (Sweller, 1988). As a result, we have employed multiple tactics in cognitive tutors to reduce such load. Efforts to make the goal structure visible (principle 3) can help reduce working memory load. Another strategy is to simplify problem-solving

actions in the interface that are irrelevant to the current learning goals. For example, the equation solver in our mathematics tutors has an auto-arithmetic mode in which students only indicate the algebraic operation to perform in each solution step, but they are not required to perform the arithmetic (Ritter and Anderson, 1995).

Similarly, our programming cognitive tutors employ structure editors to reduce the working memory load of remembering surface syntax. Littman (1991) reports human tutor behavior that is similar to this strategy. Human tutors avoid interrupting students and disrupting their working memory state to point out relatively minor errors that have little consequence for the overall problem solution.

6. Provide immediate feedback on errors relative to the model of desired performance

Studies have shown that human tutors tend to provide immediate feedback after each problem-solving step (Merrill, Reiser, Ranney & Trafton, 1992), although the feedback may be minimal (Lepper, Aspinwall, Mumme & Chabay, 1990; Fox 1991) and provided only on “important” errors (Littman, 1991). But these studies don’t reveal the relative benefits of immediate feedback. In a study with the Lisp Cognitive Tutor immediate feedback led to significantly faster learning (Corbett & Anderson, 2001). Not only can immediate feedback make learning more efficient, but it can also be motivating for students (Schofield, 1995). Mathan and Koedinger (2003) demonstrate benefits of providing immediate feedback relative to an “intelligent novice” model of desired

performance that allows for certain student errors and so appears like delayed feedback when compared to an error-free expert model of desired performance.

Cognitive Tutor Meta-Design Principles

The strengths of ACT-R and the Cognitive Tutor principles are that they are general and can apply in multiple domains. However, these principles beg some higher-level curriculum design questions: What should students be learning? What problem-solving activities support that learning? What relevant knowledge do students bring with them? We need to design cognitive tutor activities that not only "work," but work well within the curricular and social context of course objectives, teacher practices, and classroom use. Here we abstract that experience in a set of Cognitive Tutor "Meta-Design" principles.

1. Design with Instructors and Classroom Use from the Start

An experienced classroom teacher plays many key roles in a Cognitive Tutor development project, contributing hard-won knowledge of the specific learning hurdles students face and tactics for helping students past those hurdles. An experienced teacher also plays essential roles in integrating Cognitive Tutor activities with other course activities. First, an experienced teacher will help guide the initial tutor development so that the tutor dovetails with other course activities. Second, an experienced teacher who is part of the design team is best positioned to take the technology into the classroom and provide informed observations on situations in which the classroom activities and tutor activities do not mesh.

2. Design the Full Course Experience

We encountered a high-level curriculum compatibility problem in the ANGLE Geometry Proof Project (Koedinger & Anderson, 1993). Between the start of the project and the first classroom piloting, the school district had adopted a new curriculum that de-emphasized proofs and thus, it became difficult to integrate the tutor into the curriculum. The project teacher who was intimately familiar with the tutor's curriculum objectives was more successful in integrating the tutor into the new curriculum than other teachers and obtained greater learning effects. In the aftermath of the ANGLE project, we have developed the full set of course activities in all our Cognitive Tutor math courses, including the course text and assignments for two related reasons. First, it spares the classroom teacher from having to figure out how to integrate the Cognitive Tutor activities into the course. Second, it enables us to develop a course that more fully emphasizes problem-based learning throughout**.**

3. All Design Phases should be Empirically-Based

The designer should collect student data to guide and test the application of principles, including (a) design experiments that guide initial development, (b) formative evaluations that analyze the successes and failures of problem-solving activities at a fine grain-size, and (c) summative evaluations that examine whether course-level curriculum objectives are being achieved. A spectrum of empirical research methods are available from lower cost, lower reliability to higher cost, higher reliability (e.g., Koedinger, 2002).

Design Research Examples

An important message of this chapter is that we not only need to make progress in better articulating theory and principles, but also in specifying associated empirical and analytic methods that better ensure these principles will be appropriately applied.

The following three sections provide extended examples of each of these three classes of empirical research. The first section describes the use of design studies to guide application of the fifth principle, reduce working memory load. The second section describes the use of design studies to guide application of the fourth principle, promote a general understanding. The third section describes summative evaluations of the Cognitive Tutor Algebra course.

Design Research Guides Reduction of Working Memory Load

In addition to the strategies discussed above to "minimize working memory load", we employed the strategy to design instruction that builds on students' prior knowledge (cf., Bransford, Brown, & Cocking, 1999). When instruction makes connections to what students already know, they need less cognitive load to process, understand, and integrate new knowledge into long-term memory.

While building on prior knowledge is a more specific strategy for minimizing working memory load, how do we know what prior knowledge students have? Sometimes theoretical analysis of domain content is used to predict prior knowledge under the assumption that smaller component tasks are more likely to tap prior knowledge than larger whole tasks (cf., Van Merrienboer, 1997). However, while smaller tasks typically involve fewer knowledge

components, they are not always simpler for students. It is not the surface form of tasks that determine how accessible they are to students. Instead, it is the internal mental representations that students acquire and use in task performance that determines what will be simple or not. Thus, to identify what prior knowledge students have and which tasks are most likely to tap prior knowledge, it is not sufficient to analyze the content domain. Instead it is critical to study how students actually perform on tasks – to see student thinking as it really is, not as a content analysis might assume it to be.

Consider the three problems shown below, a story problem, a word problem, and an equation, all with the same underlying quantitative structure and the same solution.

Story Problem: As a waiter, Ted gets $6 per hour. One night he made $66 in tips and earned a total of $81.90. How many hours did Ted work?

Word Problem: Starting with some number, if I multiply it by 6 and then add 66, I get 81.90. What number did I start with?

Equation: x * 6 + 66 = 81.90

Which would be most difficult for high school students in a first year algebra course? Nathan & Koedinger (2000) discussed results of surveys of mathematics teachers on a variation of this question. The survey respondents tended to predict that such story problems would be most difficult and such equations would be easiest. Typical justifications for this prediction include that the story problem requires more reading or that the way the story problem is solved is by translating to the equation.

In contrast, Koedinger & Nathan (2004) found that students perform best at the story and word problems (70% and 61% respectively) like these and worst at the analogous equations (42%). Clearly many students were not solving the story and word problems using equation solving. Instead they used alternative informal strategies like guess-and-test and "unwinding", working backwards from the result, inverting operations to find the unknown starting quantity. Students had difficulty in comprehending equations and, when they did succeed in comprehending, they often had difficulty in reliably executing the equation solving strategy.

This result is important within the algebra domain. It indicates that if we want to create instruction that builds on prior knowledge, we should make use of the fact that beginning algebra students have quantitative reasoning skills that can be tapped through verbal or situational contexts. Unlike many textbooks that teach equation solving prior to story problem solving (Nathan, Long, & Alibali, 2002), it may be better to use story problem situations and verbal descriptions first to help students informally understand quantitative relationships before moving to more abstract processing of formal representations.

This study illustrates why we advocate the mantra “the student is not like me”. We need empirical methods to see past our biases or “expert blind spot” to what students are really like.

Table 3 further illustrates why it is important to use empirical methods to determine when and how to employ a principle. Using problem situations to build on prior knowledge and reduce working memory load will not work if those problem situations are not familiar. In our development of Cognitive Tutor Math

6 (Koedinger, 2002), we used a Difficulty Factors Assessment to find which kinds of problem situations make problems easier for students and which kinds do not. Table 3 illustrates different content areas in middle school math where we compared concrete story problem situations with abstract context-free problems.

==INSERT TABLE 3 HERE==

Table 3 shows 6th graders' average percent correct on multiple pre-test items in each content area. In three of the areas, the problem situation consistently facilitates performance significantly above the abstract problem. These are decimal place value, decimal arithmetic, and fraction addition. In data analysis, the situation facilitated performance on a global interpretation task, but not on a local interpretation task. In the area of factors and multiples, the situation reduced performance.

Thus using situations to build on prior knowledge may not be effective for concepts and procedures related to factors and multiples unless situations can be found that are easier to understand than abstract problems. While one might still want to use such a problem situation as motivation for learning, given this data, it does not appear that such a situation will provide a student with an easier, less cognitively taxing access to understanding of the domain content.

Reflection Promotes General Understanding

One well-researched approach to promoting a correct and general encoding is called “self-explanation” whereby students explain to themselves steps in problem solutions (e.g., Chi, de Leeuw, Chiu, & Lavancher, 1994). Aleven and Koedinger (2002) implemented a version of self-explanation in Cognitive Tutor Geometry and experimented with its effectiveness. Figure 3

illustrates the "explanation by reference" approach employed whereby students provided explanations for problem solving steps by making reference to geometry rules or reasons in an on-line glossary. Students could either type the name of the rule or select it from the glossary. This form of explanation is different from the speech-based explanations in most prior experiments on self-explanation, but has the benefit of the explanations being machine readable without the need to implement natural language understanding.

Prior Difficulty Factors Assessments had indicated that students were better able to perform a problem-solving step, like determine that angle ARN in Figure 3 is equal to 43.5 degrees, than to explain that step by referring to the "Alternate Interior Angles" rule. One reason for this difference is that students' prior knowledge includes over-generalized production rules like "if an angle *looks equal* to another, then it is" that can provide correct answers to steps, but without an understanding of when such steps are justified and why. Such over-generalized productions may result from shallow encoding and learning. According to Aleven & Koedinger's ACT-R interpretation, self-explanation promotes more general encoding because students think more deliberately, with greater explicit reflection, about the verbal declarative representation of domain rules. This deliberate reflection helps identify key features of the domain rules and thus improves the accuracy of the implicit inductive process of "compiling" (a form of learning) production rules from examples and visual input. Indeed, Aleven and Koedinger (2002) found that students using the self-explanation version of Cognitive Tutor Geometry were not only better able to provide accurate explanations, but learned the domain rules with greater understanding such that

they could better transfer to novel problems and avoid shallow inferences, like the "looks equal" production illustrated above.

Summative Field Study Evaluations and Classroom Observations

We originally assessed Cognitive Tutor Algebra in experimental field studies in city schools in Pittsburgh and Milwaukee, replicated over 3 different school years. The assessments used in these field studies targeted both 1) higher order conceptual achievement as measured by performance assessments of problem solving and representation use and 2) basic skills achievement as measured by standardized test items, for instance, from the math SAT. In comparison with traditional algebra classes at the same and similar schools, we have found that students using Cognitive Tutor Algebra perform 15-25% better than control classes on standardized test items and 50-100% better on problem solving & representation use (Koedinger, et al., 1997; Corbett, Koedinger, & Hadley, 2001; also see http://www.carnegielearning.com/results/reports). More recent studies have continued to confirm the validity of the approach. For example, the Moore (Oklahoma) Independent School District conducted a within-teacher experiment (Morgan & Ritter,2002). Eight teachers at four junior high schools taught some of their classes using Cognitive Tutor Algebra I and others using their traditional textbook. On the ETS End-of-Course Algebra exam, students taking the Cognitive Tutor curriculum scored significantly higher than control students. Cognitive Tutor students also earned higher course grades and had more positive attitudes towards mathematics.

The deployment of educational technology in the classroom has an impact beyond learning outcomes. Following the observations of Schofield, Evans-

Rhodes, & Huber (1990) and Wertheimer (1990), we have also observed the impact of the use of Cognitive Tutor Algebra on changes in classroom social and motivational processes (Corbett et al., 2001). Visitors to these classrooms often comment on how engaged students are. Cognitive Tutor Algebra may enhance student motivation for a number of different reasons. First, authentic problem situations make mathematics more interesting, sensible, or relevant. Second, students on the average would rather be doing than listening, and the incremental achievement and feedback within Cognitive Tutor Algebra problems provide a video-game-like appeal. Third, the safety net provided by the tutor reduces potential for frustration and provides assistance on errors without social stigma. Finally, the longer-term achievement of mastering the mathematics is empowering.

Conclusions and Future Work

Human tutoring is an extremely effective and enjoyable way to learn. But, buying one computer per student is a lot more cost effective than hiring a teacher for every student. Before a computer program can function as a tutor, it has to be able to do several key things that human tutors can do: 1) use domain knowledge to solve problems and reason as we want students to do; 2) have knowledge of typical student, misconceptions, relevant prior informal knowledge, and learning trajectories; 3) follow student reasoning step by step and understand when and where students reveal a lack of knowledge or understanding; 4) provide appropriate scaffolding, feedback and assistance to students when they need it and in the context of that need; 5) adapt instruction to individual student needs based on an on-going assessment of those needs. The cognitive model, model tracing

and knowledge tracing algorithms of the Cognitive Tutor architecture provide these key behaviors of good tutoring.

Cognitive Tutors and ACT-R are one manifestation of the many advances in cognitive psychology and instructional design made by learning scientists. Cognitive Tutors implement a decidedly simple form of tutoring. More sophisticated tutoring strategies can be imagined (cf., Collins et al., 1989) and some of these strategies, such as natural language tutorial dialog, are being implemented in increasingly practical forms (e.g., Jordan, Rosé, & VanLehn, 2001; Wiemer-Hastings, Wiemer-Hastings, & Graesser, 1999). Experimental studies are testing whether such added sophistication leads to increased student learning (e.g., Aleven, Popescu, & Koedinger, 2003). There is also substantial research on whether even simpler forms of instruction, like worked examples of problem solutions, are as effective or more effective than more complex forms (e.g., Clark & Mayer, 2003).

Cognitive Tutor research is actively advancing along a number of dimensions that build off of the prior work demonstrating the value of Cognitive Tutors for delivering individual tutoring to aid the acquisition of cognitive skills. A major current research topic is tutoring meta-cognitive skills in addition to cognitive skills, including self-explanation (Aleven & Koedinger, 2002), error detection and correction (Mathan & Koedinger, 2003), and learning and help-seeking skills (Aleven, McLaren, Roll, & Koedinger, 2004). Cognitive Tutors are also being deployed to support required state testing and school accountability (http://assistment.org) and authoring tools are being created to speed Cognitive Tutor development (http://ctat.pact.cs.cmu.edu). Finally, Cognitive Tutors are

being employed as research platforms to allow rigorous experimental tests of learning principles “in vivo”, that is, in classrooms with real students and real courses. (http://learnlab.org).

Acknowledgements

The National Science Foundation (NSF), Department of Education, and DARPA supported Algebra Cognitive Tutor research. Carnegie Learning, Inc supported Cognitive Tutor Math 6. NSF’s Science of Learning Center is supporting the Pittsburgh Science of Learning Center and creation of LearnLab.

References

Aleven, V., & Koedinger, K.R. (2002). An effective meta-cognitive strategy: Learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science, 26*, 147-179.

Aleven, V., McLaren, B., Roll, I., & Koedinger, K. R. (2004). Toward tutoring help seeking. In Lester, Vicari, & Parguacu (Eds.) *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, 227-239. Berlin: Springer-Verlag

Aleven, V., Popescu, O. & Koedinger, K. R. (2003). A tutorial dialog system to support self-explanation: Evaluation and open questions. In U. Hoppe, F. Verdejo, & J. Kay (Eds.), *Artificial Intelligence in Education: Shaping the Future of Learning through Intelligent Technologies, Proceedings of AI-ED 2003* (pp. 39-46). Amsterdam, IOS Press.

Anderson, J. R., Boyle, C. F., & Reiser, B. J. (1985). Intelligent tutoring systems. Science, 228, 456-468.

Anderson, J.R. & Jeffries, R. (1988). Novice LISP errors: Undetected losses of information from working memory. Human-Computer Interaction, 1, 107-131.

Anderson, J. R., & Lebiere, C. (1998). *The Atomic Components of Thought.* Hillsdale, NJ: Erlbaum.

Berry, D. C. & Dienes, Z. (1993). *Implicit Learning: Theoretical and Empirical Issues*. Hove, UK: Erlbaum.

Bloom, B. S. (1984). The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher, 13*, 3-16.

Bransford, J. D., Brown, A. L, & Cocking, R. R. (1999). *How People Learn: Brain, Mind, Experience, and School*. Committee on Developments in the Science of Learning Commission on Behavioral and Social Sciences and Education. National Research Council. Washington, D.C.: National Academy Press.

Chi, M.T.H., Feltovich, P.J., & Glaser, R. (1981). Categorization and representation of physics problems by experts and novices. *Cognitive Science,* 5, 121-152.

Chi, M. T. H., de Leeuw, N., Chiu, M., & Lavancher, C. (1994). Eliciting self-explanations improves understanding. *Cognitive Science, 18*, 439–477.

Clark, R. C., & Mayer, R. E. (2003). *e-Learning and the Science of Instruction : Proven Guidelines for Consumers and Designers of Multimedia Learning*. San Francisco: Jossey-Bass.

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: Essays in honor of Robert Glaser* (pp. 453-494). Hillsdale, NJ: Lawrence Erlbaum Associates.

Corbett, A.T. (2001). Cognitive computer tutors: Solving the two-sigma problem. *User Modeling: Proceedings of the Eighth International Conference, UM 2001*, 137-147.

Corbett, A.T. & Anderson, J.R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4, 253-278.

Corbett, A.T. & Anderson, J.R. (2001). Locus of feedback control in computer-based tutoring: Impact on learning rate, achievement and attitudes. *Proceedings of ACM CHI 2001 Conference on Human Factors in Computing Systems*, 245-252.

Corbett, A. T., Koedinger, K. R., & Anderson, J. R. (1997). Intelligent tutoring systems. In M. G. Helander, T. K. Landauer, & P. V. Prabhu, (Eds.), *Handbook of human-computer interaction* (pp. 849–874). Amsterdam: Elsevier.

Corbett, A. T., Koedinger, K. R., & Hadley, W. H. (2001). Cognitive Tutors: From the research classroom to all classrooms. In Goodman, P. S. (Ed.), *Technology-enhanced learning: Opportunities for change* (pp. 235-263). Mahwah, NJ: Erlbaum.

Corbett, A.T., MacLaren, B., Kauffman, L., Wagner, A., Jones, E., & Koedinger, K.R. (2005). Evaluating a genetics cognitive tutor: Modeling and supporting a pedigree analysis task. Carnegie Mellon University Technical Report.

Corbett, A.T. & Trask, H. (2000). Instructional interventions in computer-based tutoring: Differential impact on learning time and accuracy. *Proceedings of ACM CHI 2000 Conference on Human Factors in Computing Systems*, 97-104.

Eberts, R. E. (1997). Computer-based instruction. In Helander, M. G., Landauer, T. K., & Prabhu, P. V. (Ed.s) *Handbook of Human-Computer Interaction*, (pp. 825-847). Amsterdam, The Netherlands: Elsevier Science B. V.

Fox, B. (1991). Cognitive and interactional aspects of correction in tutoring. In P. Goodyear (Ed.) *Teaching knowledge and intelligent tutoring*, 149-172. Norwood, NJ: Ablex Publishing.

Griffin, S., Case, R., & Siegler, R. S. (1994). Rightstart: Providing the central conceptual prerequisites for first formal learning of arithmetic to students at risk for school failure. In K. McGilly (Ed.), *Classroom Lessons* (pp. 25-49). Cambridge, MA: MIT Press.

Jordan, P. W.; Rosé, C. P.; and VanLehn, K (2001) Tools for Authoring Tutorial Dialogue Knowledge. In J. D. Moore, C. L. Redfield, & W. L. Johnson (Eds.), *Artificial Intelligence in Education: AI-ED in the Wired and Wireless Future, Proceedings of AI-ED 2001*. Amsterdam, IOS Press.

Koedinger, K. R. (2002). Toward evidence for instructional design principles: Examples from Cognitive Tutor Math 6. In *Proceedings of PME-NA XXXIII (The North American Chapter of the International Group for the Psychology of Mathematics Education).*

Koedinger, K.R. & Anderson, J.R. (1993). Effective use of intelligent software in high school math classrooms. In P. Brna, S. Ohlsson and H. Pain (Eds.) *Proceedings of AIED 93 World Conference on Artificial Intelligence in Education*, 241-248.

Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education, 8*, 30-43.

Koedinger, K. R. & Nathan, M. J. (2004). The real story behind story problems: Effects of representations on quantitative reasoning. *The Journal of the Learning Sciences, 13* (2), 129-164.

Kulik, C. C. & Kulik, J. A. (1991). Effectiveness of Computer-Based Instruction: An Updated Analysis. *Computers in Human Behavior, 7,* 75-95.

Lave, J., Murtaugh, M., & de la Rocha, O. (1984). The dialectic of arithmetic in grocery shopping. In B. Rogoff & J. Lave (Eds.), *Everyday Cognition* (pp. 67-94). Cambridge, MA: Harvard University Press.

Lepper, M.R., Aspinwall, L., Mumme, D. & Chabay, R.W. (1990). Self-perception and social perception processes in tutoring: Subtle social control strategies of expert tutors. In J. Olson & M. Zanna (Eds.) *Self inference processes: The sixth Ontario symposium in social psychology* (pp. 217-237). Hillsdale, NJ: Lawrence Erlbaum Associates.

Littman, D. (1991). Tutorial planning schemas. In P. Goodyear (Ed.) *Teaching knowledge and intelligent tutoring*, 107-122. Norwood, NJ: Ablex Publishing.

Mathan, S. & Koedinger, K. R. (2003). Recasting the feedback debate: Benefits of tutoring error detection and correction skills. In Hoppe, Verdejo, & Kay (Eds.), *Artificial Intelligence in Education, Proceedings of AI-ED 2003* (pp. 13-18). Amsterdam, IOS Press.

Matz, M. (1982). Towards a process model for high school algebra errors. In D. Sleeman and J. S. Brown (Eds.) *Intelligent tutoring systems* (pp. 25-50). New York: Academic Press.

McArthur, D., Stasz, C., & Zmuidzinas, M. (1990). Tutoring techniques in algebra. Cognition and Instruction, 7, 197-244.

Merrill, D.C., Reiser, B.J., Ranney, M. & Trafton, G.J. (1992). Effective tutoring techniques: A comparison of human tutors and intelligent tutoring systems. *The Journal of the Learning Sciences*, 2, 277-305.

Morgan, P., & Ritter, S. (2002). *An experimental study of the effects of Cognitive Tutor® Algebra I on student knowledge and attitude*. Pittsburgh, PA: Carnegie Learning Inc. Retrieved April 14, 2005 from http://www.carnegielearning.com/wwc/originalstudy.pdf

Nathan, M. J. & Koedinger, K. R. (2000). An investigation of teachers' beliefs of students' algebra development. *Cognition and Instruction, 18*(2), 207-235.

Nathan, M. J., Long, S. D., & Alibali, M. W. (2002). Symbol precedence in mathematics textbooks: A corpus analysis. *Discourse Processes, 33*, 1-21.

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

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

Resnick, L. B. (1987). Learning in school and out. *Educational Researcher, 16*(9), 13-20.

Ritter, S. and Anderson, J. R. (1995). Calculation and strategy in the equation solving tutor. In J.D. Moore & J.F. Lehman (Eds.), *Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society*. (pp. 413-418). Hillsdale, NJ: Erlbaum.

Scheines, R. & Seig, W. (1994). Computer environments for proof construction. *Interactive Learning Environments*, 4, 159-169.

Singley, M.K. (1990). The reification of goal structures in a calculus tutor: Effects on problem solving performance. *Interactive Learning Environments*, 1, 102-123.

Singley, M. K., & Anderson, J. R. (1989). *Transfer of Cognitive Skill*. Hillsdale, NJ: Erlbaum.

Skinner, B. F. (1968). *The technology of teaching*. New York: Appleton-Century-Crofts.

Sleeman, D. H., & Brown, J. S. (1982). *Intelligent Tutoring Systems*. New York, NY: Academic Press.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. Cognitive Science, 12, 257-285.

van Merrienboer, J. J. G. (1997). Training complex cognitive skills: A Four-Component Instructional Design Model for Technical Training. Englewood Cliffs, NJ: Educational Technology Publications.

Vygotsky, L. S. (1978). *Mind in Society*. Cambridge, MA: Harvard University Press.

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

Wertheimer, R. (1990). The geometry proof tutor: An "intelligent" computer-based tutor in the classroom. *Mathematics Teacher, 83*(4), 308-317.

Wiemer-Hastings, P., Wiemer-Hastings, K., & Graesser, A. C. (1999). Improving an intelligent tutor's comprehension of students with latent semantic analysis. In Lajoie, S. P. and Vivet, M. eds. *Artificial Intelligence in Education, Open Learning Environments: New Computational Technologies to Support Learning, Exploration, and Collaboration, Proceedings of AIED-99*, 535-542. Amsterdam, Netherlands: IOS Press.

---

[1] In this chapter we will often refer to such performance situations as "problems" or "problem-solving activities" though we believe that many of the ideas about tutoring expressed in this chapter are relevant to performances that are not usually described as problem solving, like writing an essay, making a scientific discovery, communicating in a foreign language.

Figures and Tables

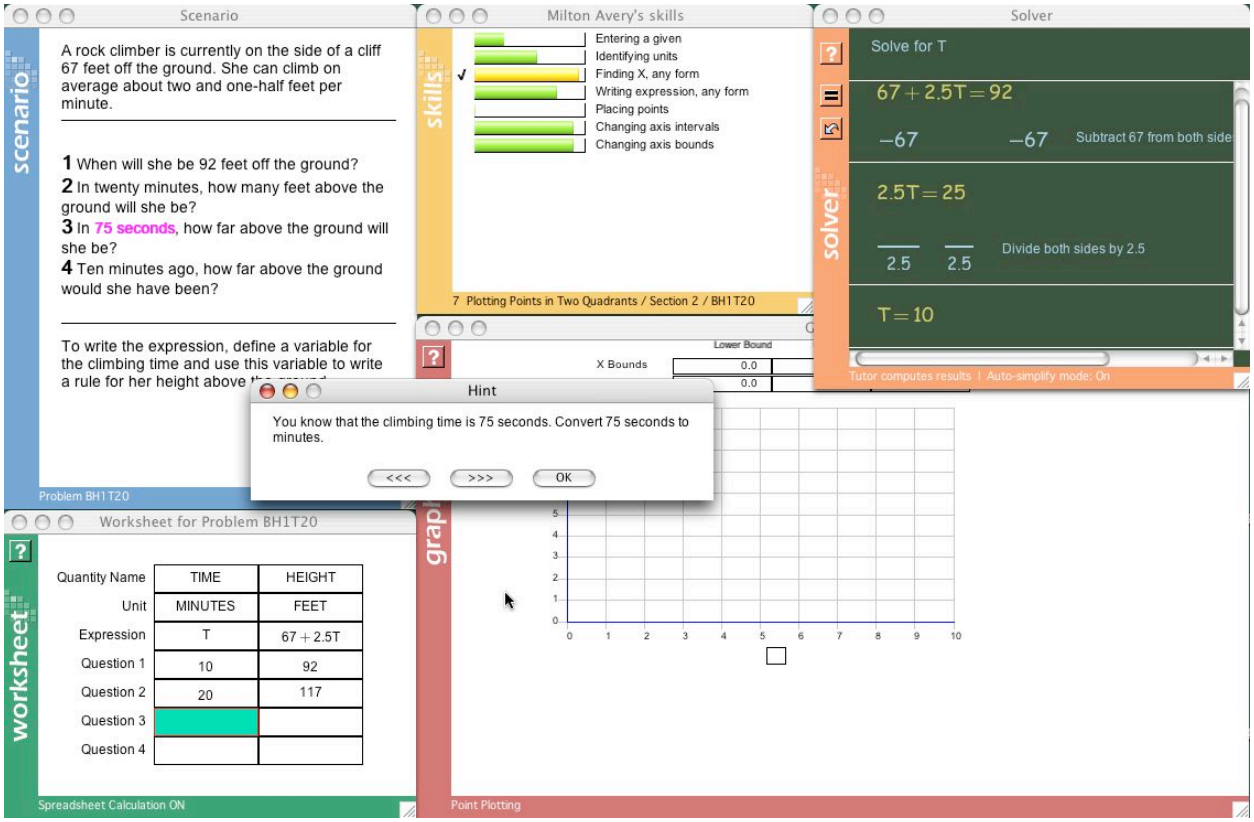

Figure 1. A screen shot of a problem solving activity within Cognitive Tutor Algebra. Students are presented a problem situation and use various tools, like the Worksheet, Grapher, and Solver shown here, to analyze and model the problem situation. As they work, "model tracing" is used to provide just-in-time feedback or on-demand solution-sensitive hints through the Messages window. The results of "knowledge tracing" are displayed in the skills chart in the top center.

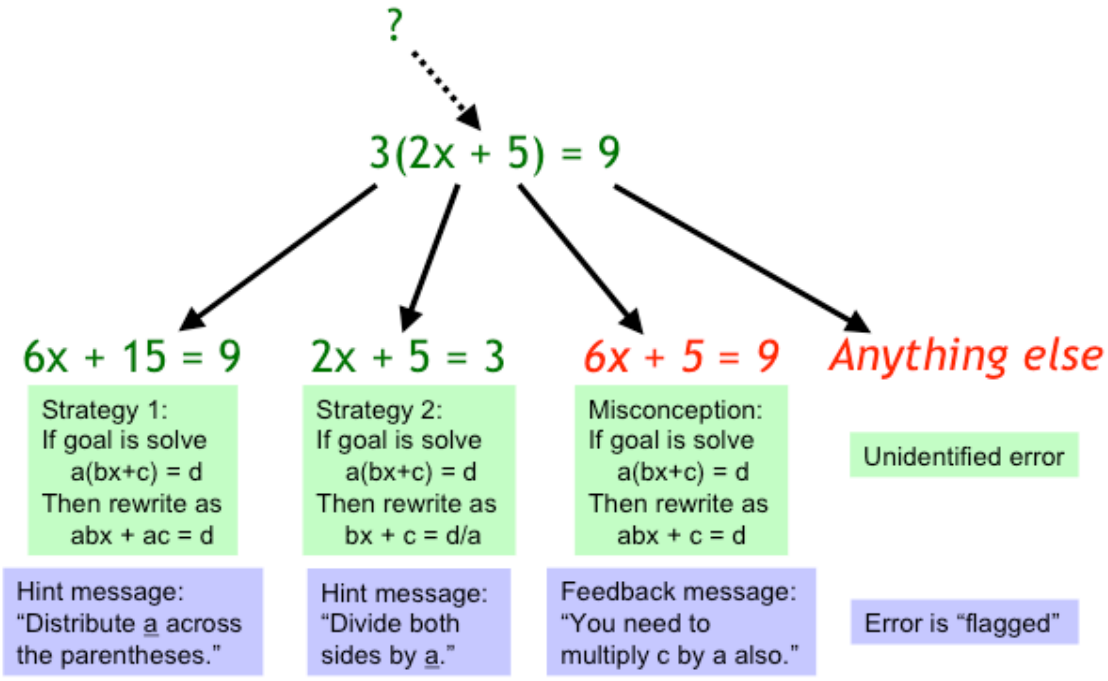

Figure 2. How model tracing uses production rules to trace different possible actions students may take. Here the student has reached the state "3(2x + 5) = 9". The "?" at the top indicates that these production rules work no matter how the student reached this state. The figure shows how the production rules apply to generate three possible next steps. Attached to the production rules are feedback messages for common errors or next-step hints that students can request if they are stuck.

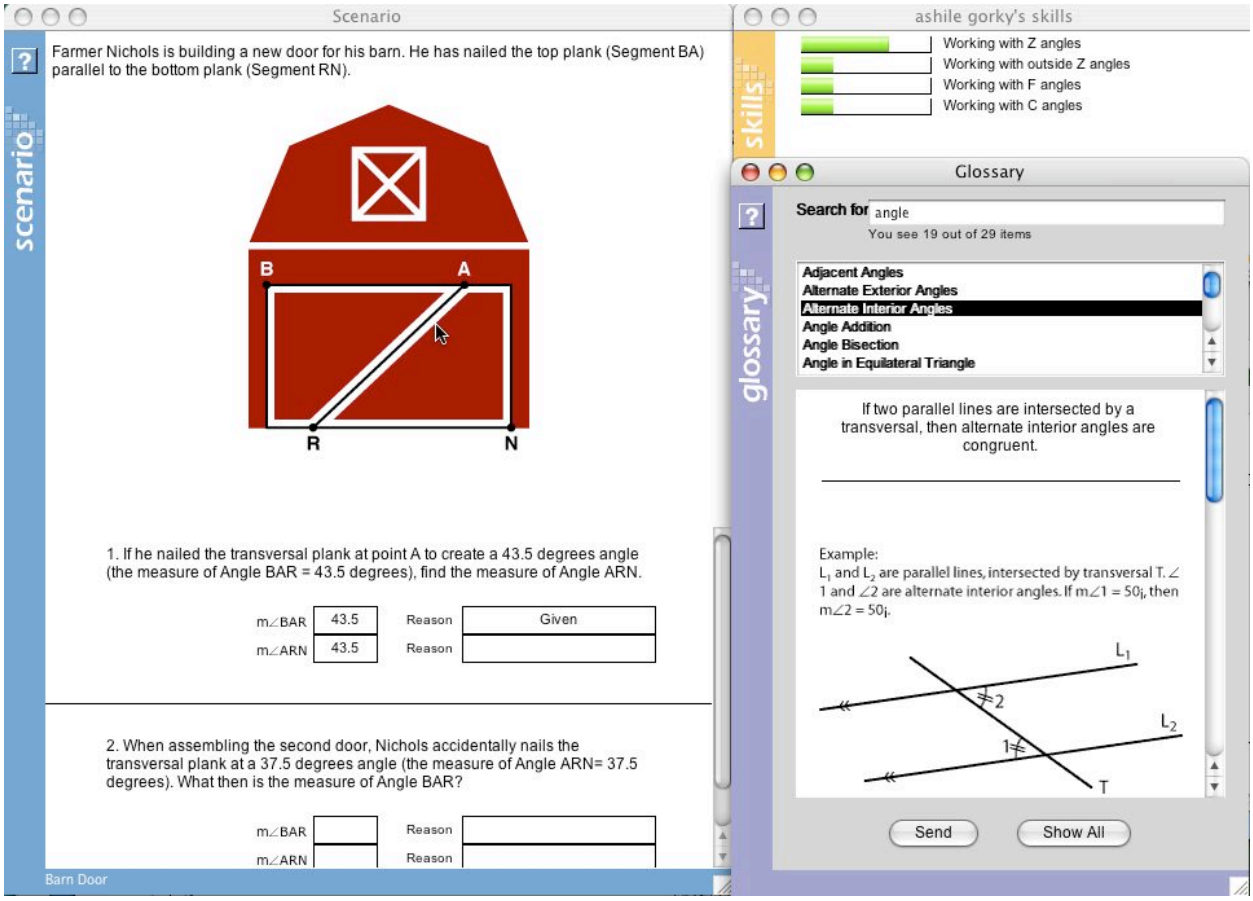

Figure 3. A screen image of Cognitive Tutor Geometry with support for self-explanation of solution steps.

Table 1. Example Production Rules

Production Rules in English	Example of its application
1. *Correct production possibly acquired implicitly* IF the goal is to find the value of quantity Q and Q divided by Num1 is Num2 THEN find Q by multiplying Num1 and Num2.	To solve "You have some money that you divide evenly among 8 people and each gets 40" find the original amount of money by multiplying 8 and 40.
2. *Correct production that does heuristic planning* IF the goal is to prove two triangles congruent and the triangles share a side THEN check for other corresponding sides or angles that may congruent.	Try to prove triangles ABC and DBC are congruent by checking whether any of the corresponding angles, like BCA and BCD, or any of the corresponding sides, like AB and DB, are congruent.
3. *Correct production for a non-traditional strategy* IF the goal is to solve an equation in X THEN graph the left and right sides of the equation and find the intersection point(s).	Solve equation sin x = x^2 by graphing both sin x and x^2 and finding where the lines cross.
4. *Correct but overly specific production* IF "ax + bx" appears in an expression and c = a + b THEN replace it with "cx"	Works for "*2x* + 3x" but not for "*x* + 3x"
5. *Incorrect, overly general production* IF "Num1 + Num2" appears in an expression THEN replace it with the sum	Leads to order of operations error: "x * *3 + 4*" is rewritten as "x * *7*"

Table 2. Six Instructional Design Principles for Cognitive Tutors

1. Represent student competence as a production set
2. Provide instruction in a problem-solving context
3. Communicate the goal structure underlying the problem solving
4. Promote a correct and general understanding of the problem-solving knowledge
5. Minimize working memory load that is extraneous to learning
6. Provide immediate feedback on errors relative to the model of desired performance

Table 3. Comparisons of Situational and Abstract Problems in Five Content Areas

	Decimal place value	Decimal arith	Fraction addition	Data interp-global
Situation	Show 5 different ways that you can give Ben $4.07. [A place value table was provided.]	You had $8.72. Your grandmother gave you $25 for your birthday. How much money do you have now?	Mrs. Jules bought each of her children a chocolate bar. Jarren ate 1/4 of a chocolate bar and Alicia ate 1/5 of a chocolate bar. How much of a chocolate bar did they eat altogether?	[2 scatterplots given] Do students sell more boxes of Candy Bars or Cookies as the months pass?
% correct	61%	65%	32%	62%
Abstract	List 5 different ways to show the amount 4.07. [Place value table given.]	Add: 8.72 + 25	Add: 1/4 + 1/5	[Scatterplots given] Are there more Moops per Zog in the Left graph or the Right graph?
% correct	20%	35%	22%	48%