Educ Psychol Rev (2007) 19:239–264
DOI 10.1007/s10648-007-9049-0

ORIGINAL ARTICLE

Exploring the Assistance Dilemma in Experiments with Cognitive Tutors

Kenneth R. Koedinger · Vincent Aleven

Published online: 21 July 2007
© Springer Science + Business Media, LLC 2007

Abstract Intelligent tutoring systems are highly interactive learning environments that have been shown to improve upon typical classroom instruction. Cognitive Tutors are a type of intelligent tutor based on cognitive psychology theory of problem solving and learning. Cognitive Tutors provide a rich problem-solving environment with tutorial guidance in the form of step-by-step feedback, specific messages in response to common errors, and on-demand instructional hints. They also select problems based on individual student performance. The learning benefits of these forms of interactivity are supported, to varying extents, by a growing number of results from experimental studies. As Cognitive Tutors have matured and are being applied in new subject-matter areas, they have been used as a research platform and, particularly, to explore interactive methods to support metacognition. We review experiments with Cognitive Tutors that have compared different forms of interactivity and we reinterpret their results as partial answers to the general question: How should learning environments balance information or assistance *giving* and *withholding* to achieve optimal student learning? How best to achieve this balance remains a fundamental open problem in instructional science. We call this problem the "assistance dilemma" and emphasize the need for further science to yield specific conditions and parameters that indicate when and to what extent to use information giving versus information withholding forms of interaction.

Keywords Cognitive tutors · Intelligent tutoring system · Self-explanation · Desirable difficulties · Cognitive load · Worked examples · Tutoring · Problem solving · Interaction · Cognitive psychology · Experimental studies · Cognitive theory

K. R. Koedinger (✉)
Human-Computer Interaction Institute and Psychology Department, Carnegie Mellon University, CMU, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA
e-mail: koedinger@cmu.edu

K. R. Koedinger · V. Aleven
Human-Computer Interaction Institute, Carnegie Mellon University, CMU, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA

V. Aleven
e-mail: aleven@cs.cmu.edu

In designing learning environments that effectively support student learning, one faces many choices with respect to the interactive and non-interactive features the system will provide. While instructional options like text and video are non-interactive, they will continue to have their place in learning environments (cf., Schwartz and Bransford 1998). Homework problems, learning-by-doing projects (e.g., Duch *et al.* 2001; Krajcik and Starr 2001) or self-explanation prompts (Chi *et al.* 1994) may seem to be more interactive forms of instruction than text and video, but they are not in and of themselves interactive. They may facilitate interactive forms of learning, if the student engages with them, but there is nothing about the instructional materials themselves that contributes to the interaction. They are not dynamically responsive to an individual student's approach to a particular example or problem, nor to the difficulties that an individual student may experience. The materials stay what they are.

Tutoring, on the other hand is a genuinely interactive form of instruction. Human one-on-one tutoring is highly effective, with experienced tutors achieving a two standard deviation improvement over classroom instruction (Bloom 1984). Interactive learning environments that mimic aspects of human tutors have been highly successful as well. For instance, Intelligent Tutoring Systems have been shown to be highly effective in improving students' learning in real classrooms (VanLehn 2006). Intelligent Tutors draw on artificial intelligence technology to provide interactive instruction that adapts to individual students' needs and, most typically, supports student practice in learning complex problem solving and reasoning. Among computer-based interactive learning environments, intelligent tutoring systems are typically found toward the high end of the interactivity spectrum. The interactive capabilities of these systems usually include step-by-step feedback and hints that are specific to the particular solution path that a student has chosen and to the particular step with which the student is experiencing difficulties. Other forms of interactivity may be available as well, such as keeping track of students' mastery of skills and concomitant individualized problem selection. Although we focus on intelligent tutoring systems, our analytical framework, introduced next, is likely to pertain to a broader range of instructional interventions including e-learning as well as non-technology approaches.

Framework: Our View on What Counts as Interactivity

As an organizing framework throughout this paper, we classify kinds of instructional techniques or events along two dimensions, summarized in Table 1. The first dimension (represented by the rows in Table 1) concerns whether the information presented to or requested of the student involves explicit verbal generalizations or whether it involves instances or examples of such generalizations or activities that engage the use of them without explicit expression of them. The second dimension (represented by the columns in Table 1) concerns the direction of communication *after* the system presents a learning task or learning materials to a student: whether or not the student responds to this initial presentation, and if so, whether the system provides feedback on the student's response. The first column in Table 1 shows *passive* instructional events, like textbook descriptions or worked examples, where the learner is presented information to process, but is not requested to produce anything.[1] The second column shows *active* instructional events, like asking students to solve problems or "self-explain" a worked example (Chi *et al.* 1989). Unlike passive instructional events, in active instructional events some information is

[1] While texts come with an at least implicit request to have students try to understand them, they do not explicitly request students to *produce* a written or spoken output. The student "learning event" that may result from a passive instructional event may be active (e.g., taking notes), but passive instructional events do not explicitly prompt for such activity.

Table 1 Examples of Instructional Events in Terms of (See the Columns) Whether Students Must Respond (Active) and Get Feedback (Interactive) and (See the Rows) Whether the Instruction Involves or Encourages Explicit Verbal Generalizations or Whether it Involves Implicit Instances of those Generalizations

	Non-interactive		Interactive
	Passive	Active	
Explicit, verbal generalization	1. Description	2. Self-explanation	3. Self-explanation with feedback
Implicit, instances	4. Example	5. Unguided practice	6. Tutored practice

withheld and the student is required to fill in that information, that is, give a response like a problem solution or self-explanation of an example. The third column shows *interactive* instructional events, like problem-solving practice with a tutor, that not only require a student response, but also provide feedback on students' responses, allow students to change responses, provide feedback on the changes, and so on. This iteration of response, feedback, and potential response change and further feedback (or "feedforward" in the form of hints on how to proceed with the problem) is a hallmark of interactive instruction. As in active instructional events, some information is initially withheld from the learner. Unlike active instructional events, in interactive instructional events additional information is later given, if the learner fails to construct it him or herself.

Whether an instructional event is passive, active, or interactive does not guarantee that a student will or will not interactively engage with it. Indeed, well-written texts can be quite engaging and *students* can interact with them, for instance, after reading a later sentence a new interpretation emerges that leads the student to reread a prior sentence to check (get feedback on) that interpretation. However, the focus of our framework is on first distinguishing differences in *instructional methods* themselves. The framework then sets the stage for discussing whether and how such differences may cause students to interactively engage or not.

The rows in Table 1 contrast more explicit verbal instructional events from more implicit, instance-based ones. Explicit verbal instructional events involve the expression, by the instructor or student, of generalizations or rules that describe or explain concepts and principles in the domain. Implicit or instance-based kinds of instructional events, like examples and practice activities, illustrate or demand the application of these general concepts or principles sometimes without explicitly stating them. Instruction usually involves combinations of cells in the rows and columns in Table 1 (that's why we call these cells instructional "events" rather than "methods"). For instance, traditional textbook instruction tends to combine passive reading of general explicit descriptions of concepts (cell 1, passive/explicit) followed by active practice with problem-solving instances (cell 5, active/implicit). New forms of instruction like self-explanation combine passive examples (cell 4, passive/implicit) followed by active explication of generalizations (cell 2, active/explicit). The "action-generalization" principle (Koedinger 2002) combines instance-based problem solving (cell 5) followed by active explication (cell 2).

The kinds of instructional events in Table 1 differ in terms of whether and how they give or withhold information or assistance. Instruction would not be instruction if it did not provide some information or assistance to students, but many lines of research and theory suggest the importance of practice, learning by doing, self-testing during study, or, more generally, requiring students to construct knowledge. These approaches involve withholding information from students so that they can exercise, test, or reason toward new knowledge on their own. For example, often textbooks provide explanations (cell 1), but in self-explanation (cell 2), the explanation is withheld and the student is asked to provide one. Table 2 summarizes our view on the benefits and costs of information giving versus information withholding.

Given the trade-offs between information giving and withholding, the instructional designer faces the *assistance dilemma*. To what degree should an interactive learning environment provide students with information relevant to their learning processes and what are the most opportune moments for doing so? And when is student learning supported more effectively by withholding information, either temporarily, until the student has had an opportunity to generate or synthesize the information for him or herself, or even permanently? For example, what is the appropriate balance between passive examples versus problem solving? Should students be given explanations or asked to generate them themselves? Should feedback in interactive systems be immediate (providing timely information) or delayed (withholding information to a later time)? Should instructional hints be more specific or more general? Should they provide informative descriptions of principles or examples that the student can use to infer those principles themselves? Should information or problems on the same knowledge be placed close together (providing the student with assistance in retrieving information from one problem to the next) or spaced more widely (withholding easy information carry-over from one problem to the next)? Under what circumstances is guided problem solving more effective than (pure or guided) discovery learning?

The assistance dilemma is related to Bjork's "desirable difficulties" and the notion that while assisting performance during instruction can sometimes improve learning, in some cases making performance more difficult during instruction improves learning (Schmidt and Bjork 1992). For instance, Paas and Van Merrienboer (1994) found, on one hand, that worked examples made performance easier during instruction *and* led to better learning and, on the other hand, that introducing variability in example and problem content made performance harder during instruction but also led to better learning. The explanation that worked examples reduce "extraneous" cognitive load that distracts from learning whereas variability induces "germane" cognitive load that enhances learning is a first step. However, this explanation begs the question of what forms of instruction yield extraneous versus germane load. It does not resolve the assistance dilemma.

Several authors advocate more information/assistance giving (Kirschner *et al.* 2006; Klahr and Nigam 2004), relative to more discovery oriented, situated, or constructivist approaches (cf., Anderson *et al.* 1996, 1998). While the experiments referenced in these papers represent progress toward addressing the assistance dilemma, this work is not clear

Table 2 The Assistance Dilemma: Finding the Balance Between Information or Assistance Giving and Withholding is a Fundamental Challenge in Designing Effective Instruction

	Benefit	Cost
Giving information or assistance[a]	Accuracy	Shallow processing
	Efficiency of communication	Lack of attention
	Thrill of (supported) success	May not engage long-term memory
		Stealing chance to shine
Withholding information or assistance	Generation effect	Cost of errors
	Forces attention	Floundering, confusion, wasted time
	Engages long-term memory	Frustration of failure
	Thrill of independent success	

[a] Since *information* giving vs withholding seems a better description for non-interactive forms of instruction, where as *assistance* giving vs withholding seems a better description for interactive forms of instruction, we use both terms

and precise enough about how much more information/assistance to provide and under what circumstances. They do not define clear boundaries for how much information giving is too much and when information withholding is more effective. For example, Kirschner *et al.* 2006 state, "a worked example constitutes the epitome of strongly guided instruction," but they surely do not mean to advocate that instruction should consist only of worked examples. Indeed, many studies on worked examples show benefits of interleaving worked examples with problem solving practice (assistance withholding) relative to all problem-solving practice (e.g., Trafton and Reiser 1993; Ward and Sweller 1990). The extreme no-dilemma position, that information giving is always better, would predict that all worked examples would be better than interleaving worked examples and problem solving. We do not know of such a direct comparison, though it seems unlikely to hold in general given results like those of Trafton and Reiser's (1993) interleaving effect, expertise reversal effect (Kalyuga *et al.* 2001), and generation effects (Slamecka and Graf 1978).

In this paper, we consider the assistance dilemma as it relates to Cognitive Tutors, a form of interactive learning environments that have been shown to be highly successful in improving students learning in a range of domains including high-school mathematics (Koedinger *et al.* 1997) and Lisp programming (Anderson *et al.* 1989). Cognitive Tutors grew out of an attempt to apply and test the ACT-R theory of cognition and rely on cognitive psychology research in their design and development (Anderson *et al.* 1995). Cognitive Tutors make an interesting case study for two reasons: First, their designers (like those of all interactive learning environments) have faced the assistance dilemma in various manifestations and have proposed some methods for resolving it. Such methods have some theoretical support, through application of the ACT-R theory of cognition and learning (Anderson 1993; Anderson and Lebiere 1998), as well as empirical support from a range of studies that evaluated both the effectiveness of the tutors as a whole and of individual tutor features. Second, Cognitive Tutors (and intelligent tutoring systems more generally) provide suitable testbeds for further research on the assistance dilemma. We are only beginning to understand how to resolve the dilemma. It is quite likely that there are better ways of resolving the dilemma than we have explored in Cognitive Tutors. Other student-related and system-related factors may need to be considered. Before we can establish a strong theory, a wider empirical base is needed. Cognitive Tutors provide an attractive platform for such studies, given their presence in many schools in the US and given the fact that they facilitate detailed logging of students' learning events.

Thus, our program in this paper is as follows: After a brief overview of the Cognitive Tutor technology, we re-cast some of the prior research involving Cognitive Tutors as investigating the balancing of information giving and information withholding, and consider whether this re-casting helps in formulating further hypotheses with respect to this issue. Finally, we address potential additions to Cognitive Tutors that may better address the assistance dilemma.

Brief Overview of Cognitive Tutors

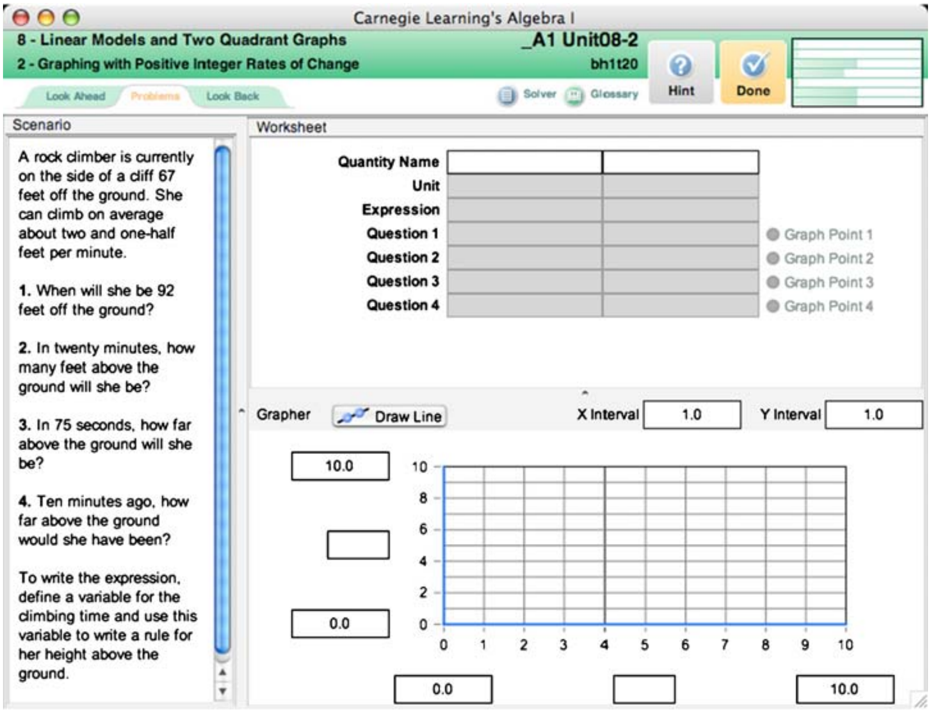
Cognitive Tutors are interactive learning environments that provide various kinds of assistance as students learn a complex cognitive skill through practice. Cognitive Tutors for high-school mathematics and college-level genetics have been successful in real educational settings, as detailed below. Cognitive Tutors for high-school math have also been successful in the market place, with the Cognitive Tutor Algebra curriculum being used in over 2,000 schools at the time of this writing (see http://carnegielearning.com). All Cognitive Tutors share a set of interactive elements, listed in Table 3.

Table 3 Main Elements of Interactivity in Cognitive Tutors

Key ways cognitive tutors achieve interactivity
1. Problem-solving environment, often with interactive tools
2. Tutorial guidance, in the form of
a. implicit yes–no feedback on correctness on a step-by-step basis
b. specific feedback messages for commonly-occurring errors
c. next-step hints (on demand or tutor initiated)
3. Adaptive problem selection based on student performance solving problems with the tutor

Each Cognitive Tutor provides the student with a rich problem-solving environment with a variety of representational tools and presents authentic problem scenarios for the student to solve. For example, the Algebra Cognitive Tutor, shown in Fig. 1, provides students with real-world scenarios that require algebraic reasoning, as well as various tools, such as a worksheet, a grapher, and a symbolic equation solver (which can be viewed by clicking on the Solver button in the upper right).

As students analyze the scenario, they enter intermediate steps into the tutor's graphical user interface as illustrated in Fig. 2. The tutor provides implicit feedback on the correctness of their steps, accepting correct steps without any fanfare and "flagging" erroneous steps



A rock climber is currently on the side of a cliff 67 feet off the ground. She can climb on average about two and one-half feet per minute.

1. When will she be 92 feet off the ground?

2. In twenty minutes, how many feet above the ground will she be?

3. In 75 seconds, how far above the ground will she be?

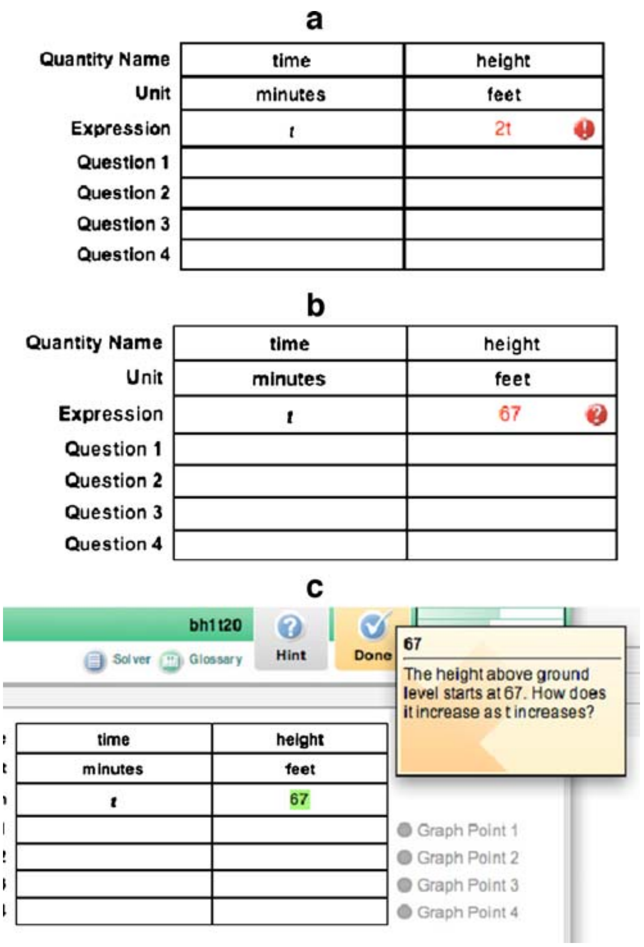
4. Ten minutes ago, how far above the ground would she have been?

To write the expression, define a variable for the climbing time and use this variable to write a rule for her height above the ground.

Fig. 1 The Algebra Cognitive Tutor presents students with authentic problem scenarios (*left*) and a problem-solving environment made up of representational tools including a table/spreadsheet (*top*), graphing tool (*bottom*), and a symbolic equation solver (available by clicking on the *Solver* button at the *top*). Instructional facilities include feedback on entries in these tools, hints on request (by clicking on the *Hint* button at *top*), an on-line glossary (*Glossary* at *top*), and feedback on learning progress (skill bar chart at *top*)

(see Fig. 2a). For certain errors that students commonly make, the tutor presents an error feedback message that explains why the step is wrong (see Fig. 2b, c).

In addition to error feedback messages, at any point in the problem scenario, students can request help from the tutor. The tutor presents hints that are specific to the solution strategy taken by the student. Typically, multiple levels of hints are available, as shown in Fig. 3, with each giving progressively more specific advice. The hints explain which problem-solving principle can be applied and how. The last hint in the sequence (sometimes referred to as the "bottom-out hint") often states what the next step should be (i.e., provides the answer—or something close to it, such as an arithmetic or algebraic expression that can be used to find the answer). For instance, the "bottom-out" hint shown in Fig. 3 is "Enter



Quantity Name	time	height
Unit	minutes	feet
Expression	t	2t
Question 1		
Question 2		
Question 3		
Question 4		

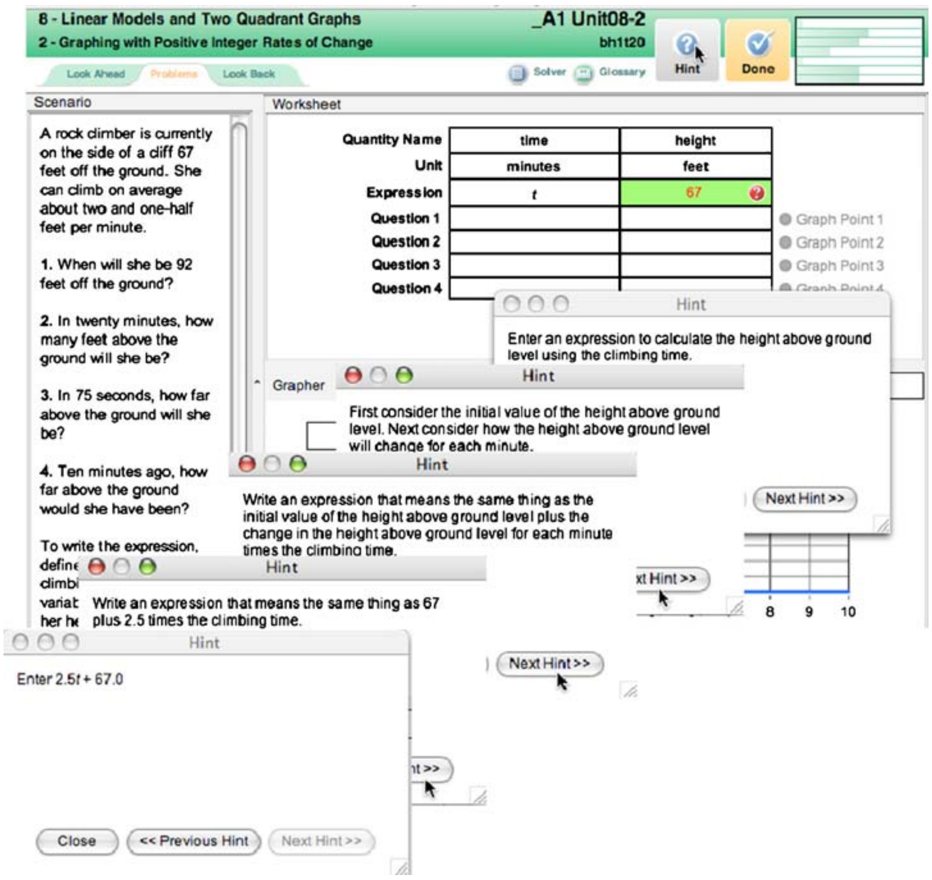
Quantity Name	time	height
Unit	minutes	feet
Expression	t	67
Question 1		
Question 2		
Question 3		
Question 4		

time	height
minutes	feet
t	67

67

The height above ground level starts at 67. How does it increase as t increases?

Fig. 2 Student–tutor interactions on the problem shown in Fig. 1. **a** The student has made correct entries in the Worksheet for the quantity names (*time*, *height*) and units (*minutes*, *feet*) for the relationship described in the scenario. The tutor accepts multiple possible correct answers (e.g., "time" could be "climbing time") and multiple possible orders (e.g., time and height columns could be swapped). The student has entered *t* to represent time and the incorrect expression *2t* for the height. The tutor flags this error by turning the entry *red* and displaying the *exclamation mark*. **b** Here the student tries *67* and it is flagged as an error this time with a *question mark*. **c** The student rolls the mouse over the *question mark* to display the error feedback message



A rock climber is currently on the side of a cliff 67 feet off the ground. She can climb on average about two and one-half feet per minute.

1. When will she be 92 feet off the ground?

2. In twenty minutes, how many feet above the ground will she be?

3. In 75 seconds, how far above the ground will she be?

4. Ten minutes ago, how far above the ground would she have been?

To write the expression, define climb variat her he

Quantity Name	time	height
Unit	minutes	feet
Expression	*t*	67
Question 1		
Question 2		
Question 3		
Question 4		

Hint

Enter an expression to calculate the height above ground level using the climbing time.

Next Hint >>

Hint

First consider the initial value of the height above ground level. Next consider how the height above ground level will change for each minute.

Hint

Write an expression that means the same thing as the initial value of the height above ground level plus the change in the height above ground level for each minute times the climbing time.

Next Hint >>

Hint

Write an expression that means the same thing as 67 plus 2.5 times the climbing time.

Next Hint >>

Hint

Enter 2.5*t* + 67.0

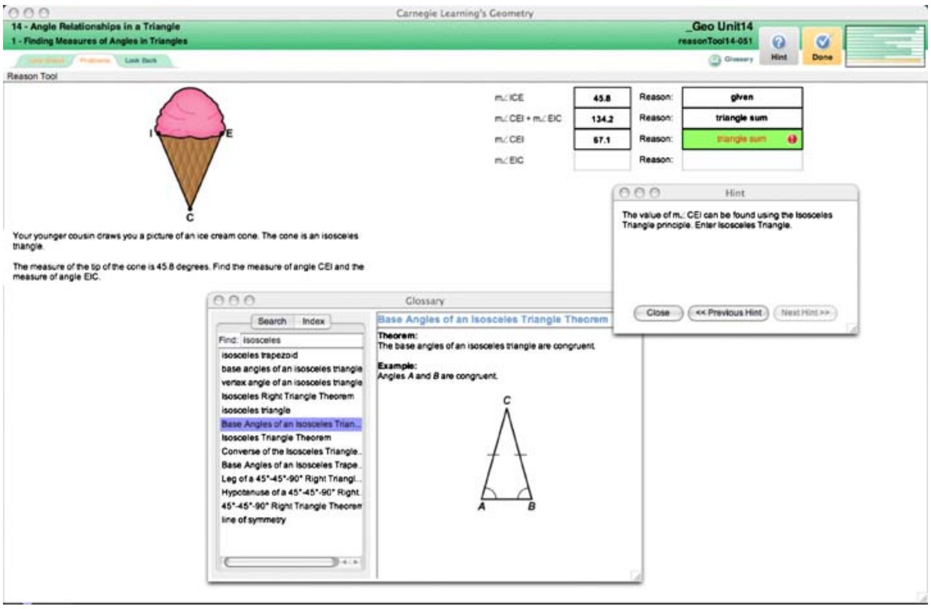
Close | << Previous Hint | Next Hint >>

Fig. 3 After the student error illustrated in Fig. 2b, the student requests a hint message by clicking on the *Hint* button in the *upper right*. The hint message "Enter an expression to calculate ..." appears. If the student needs more help, she can click on the *Next Hint* button. All five levels of hint are illustrated with the last "bottom-out" hint being "Enter 2.5*t*+67.0" (The tutor has only a single hint window and displays only one hint level at a time within this window. The multiple hint windows represent different content displayed in the hint window at different times.)

2.5t+67.0." This message also illustrates how hints are adapted to students' solutions—here using the variable *t* that the student had earlier entered as the expression for time. Had the student used a different name for the variable, the hint would have referred to that variable. Some Cognitive Tutors provide hints proactively when students make multiple errors on a step without requesting a hint.

In addition to on-demand hints, many Cognitive Tutors provide an on-line glossary, which contains definitions of terms as well as statements of important theorems and definitions, often illustrated with an example. Figure 4 shows an example of a glossary entry (see the bottom middle) in the Geometry Cognitive Tutor. The students can browse the glossary freely. One motivation for adding the glossary is that general reference resources like the glossary are part of *doing* math, not just of *learning* math.

Finally, Cognitive Tutors select problems for students on an individual basis. They keep track of students' knowledge growth over time in order to implement a mastery-learning approach. The system tracks individual "knowledge components," which include skills, like



14 - Angle Relationships in a Triangle
1 - Finding Measures of Angles in Triangles

_Geo Unit14
reasonTool14-051

m∠ICE	45.8	Reason:	given
m∠CEI + m∠EIC	134.2	Reason:	triangle sum
m∠CEI	67.1	Reason:	triangle sum
m∠EIC		Reason:	

Your younger cousin draws you a picture of an ice cream cone. The cone is an isosceles triangle.

The measure of the tip of the cone is 45.8 degrees. Find the measure of angle CEI and the measure of angle EIC.

The value of m∠CEI can be found using the Isosceles Triangle principle. Enter Isosceles Triangle.

Find: isosceles

isosceles trapezoid
base angles of an isosceles triangle
vertex angle of an isosceles triangle
Isosceles Right Triangle Theorem
isosceles triangle
Base Angles of an Isosceles Trian...
Isosceles Triangle Theorem
Converse of the Isosceles Triangle...
Base Angles of an Isosceles Trape...
Leg of a 45°-45°-90° Right Triangl...
Hypotenuse of a 45°-45°-90° Right...
45°-45°-90° Right Triangle Theorem
line of symmetry

Base Angles of an Isosceles Triangle Theorem

Theorem:
The base angles of an isosceles triangle are congruent.

Example:
Angles *A* and *B* are congruent.

Fig. 4 Support for self-explanation in the Geometry Cognitive Tutor. In problem scenarios such as the one shown at the *top left*, students compute angle measures and explain their answers, with feedback from the tutor, in the worksheet at the *top right*. To explain each step, they enter the name of a geometry theorem or definition that was applied. They can either type the name or select it from the tutor's glossary of geometry knowledge, shown at the *bottom*. In the example shown above, the student entered "triangle sum" as the reason for one of their steps. After the tutor's feedback indicates that the explanation is wrong (by means of an *exclamation mark* and *red* font), a hint from the tutor (shown at the student's request in the window just *below* the worksheet) reveals the correct reason

knowing how to express a linear function in algebraic symbols, and concepts, like the slope of a function. For each knowledge component targeted in the instruction, the tutor maintains an estimate of the probability that an individual student has learned that knowledge component, based on the student's performance on the assigned problems. The tutor uses the probability estimates to select problems on an individual basis. After a student completes the required problems within any given curriculum section, the tutor selects problems that involve knowledge components that the student has not mastered yet, meaning that its probability estimate is below a pre-defined threshold, typically set to 0.95. When all targeted knowledge components in the given curriculum section are mastered, the student is promoted to the next section. The tutor keeps the student informed about its assessment of their knowledge, displaying the estimates as a set of "skill bars" (see top right in Fig. 1). Knowledge components that are mastered are ticked off in the skill bars pane and displayed in gold. The skill bars provide students with an up-to-date measure of their progress through the tutor's problem set. Students can click on the small skill bar panel to see an enlarged version that contains a brief descriptive phrase for each knowledge component.

Cognitive Tutors are grounded in the ACT-R theory of cognition (Anderson 1993). They were created partly as an effort to test key tenets of this theory. One of those tenets is that a complex cognitive domain can be understood in terms of small knowledge components called production rules that are learned independently of each other. Thus, each Cognitive Tutor has a production rule model that explicitly represents the target competence that the tutor is meant to help students acquire. The production rules are fine-grained condition-action pairs that tie particular actions (such as writing out an intermediate result or final answer) or subgoals to particular higher-level goals and context features. A model for a complex domain like solving linear equations would typically contain several hundreds of production rules, depending somewhat on the grain size with which knowledge components are modeled (e.g., Koedinger and Corbett 2006).

The production rule model provides the domain intelligence, that is the tutoring system's knowledge of algebra or geometry. The model enables the tutor to solve the same class of problems that it asks students to solve. The tutoring intelligence comes from two algorithms called model tracing and knowledge tracing. *Model tracing* uses the model to interpret each student action and to follow students' different strategies as they work through problem scenarios. The results of model tracing are used to provide students with correctness and error feedback and to individualize instructional advice to each student's particular reasoning steps or chosen strategy. An algorithm called *knowledge tracing* is used to estimate how well an individual student has mastered each key production rule. The results of knowledge tracing are used to determine (a) the selection of problems relevant to individual student needs and (b) when the student has mastered the all the knowledge components, the concepts and skills, in a curriculum unit.

Evidence in support of ACT-R's notion of the production rule as the unit of procedural knowledge comes from research that involves Cognitive Tutors (Anderson 1993; Anderson *et al.* 1995). In particular, a production rule analysis was key in accounting for "learning curve" data generated from logs of students' interactions with the Cognitive Tutor for LISP programming. Without using production rules to guide analysis, these data do not yield a smooth learning curve, that is, there is not a systematic decline in student error rate on successive actions they perform in solving problems. However, when those actions are categorized in terms of the corresponding production rules, a smooth learning curve is revealed consistent with the overall improvement seen from pre-test to post-test. Error rate does not go down in the generic curve because students are not learning "programming" as a large-grain general skill, but in smaller production-rule-sized components. Later problems in a curriculum require new ideas or production rules and students initially struggle with these new production rules (error rate goes up). Students later show successive improvement (error rate goes down) on actions involving the same production rules. These patterns provide evidence that learning occurs at a grain size about the size of a production rule. Ohlsson and Mitrovic (2006) have also applied this approach of using smooth learning curves as a empirical method to zero in on the grain size of knowledge acquisition. They used "constraints" instead of production rules as their unit of analysis, but the same pattern emerges. Debates about knowledge representation aside, the key point is that the knowledge components that students acquire, whether production rules, constraints, skills, or concepts, are quite specific and differentiated.

Algorithms that support interactivity in cognitive tutors

The model-tracing algorithm evaluates the correctness of each student attempt at solving a step by comparing the student's step to the possible steps that the cognitive model would take in the same situation. If the action taken by the student is among these actions, the tutor provides implicit positive feedback, and the student is assumed to have used the production rules that were used by the model to produce the given action. If a student action corresponds to a production rule that models incorrect behavior, the tutor provides negative feedback and presents an error feedback message associated with the corresponding production rule (see Fig. 2b, c). If the student action does not correspond to any production, the tutor provides negative feedback without further explanation. The model-tracing algorithm is used in a similar manner to provide hints upon a student's request. When the student requests a hint, the tutor selects one of the productions that could apply to generate a next step at this point. A hint template attached to this production is filled in with problem-specific information and then presented to the student. Variants of this model-tracing technique are used in a number of other intelligent tutoring systems, for example, the Andes tutor for physics (VanLehn *et al.* 2005) and the SlideTutor for interpreting diagnostic images related to skin diseases (Crowley *et al.* 2005; Crowley and Medvedeva 2006).

A second major algorithm, *knowledge tracing,* maintains estimates of the probability that the student knows each knowledge component in the model, represented by a key production rule (Corbett and Anderson 1995). The knowledge tracer uses information provided by the model-tracing algorithm: when the student action is correct, the production rules involved in that correct action; when the student action is incorrect, those that *should* have been used. For each step in a problem, the estimates of the relevant knowledge components are updated (i.e., the production rules determined by the model-tracing algorithm), contingent upon whether the student performed the step correctly on her first attempt or whether she made an error or requested a hint. The updating procedure is based on a simple Bayesian formula, which assumes that the student is in one of two states with respect to a given production rule: the student either knows the rule or not. The Bayesian formula expresses the probability that the student knows the knowledge component as a function of three parameters, assumed to be fixed: (a) a probability that a student learns the knowledge component as a result of encountering it (once) in any given tutor problem, (b) a probability of guessing right even when the knowledge component is not mastered, and (c) a probability of not getting the step right even if the knowledge component is mastered. The knowledge-tracing procedure enables the tutor to determine when a student is ready to move on to the next curriculum unit and, before that, to select problems that give students more practice and instruction on un-mastered knowledge components.

Characterizing the interactive elements of cognitive tutors

It is informative to compare Cognitive Tutors to various alternatives, along the dimensions displayed in Table 1. We would place Cognitive Tutors squarely in the Interactive column—we stress that the table in spite of its "discrete" character really represents a continuum. With respect to the implicit/explicit dimension, some features may be considered implicit learning (e.g., yes/no feedback), others exemplify a more explicit instructional approach (e.g., principle-based hints). Along the interactivity dimension, it may be clear that Cognitive Tutors are significantly more interactive than alternatives such as having students solve textbook problems as homework, or having them solve problems with the help of typical computer-assisted instruction (Eberts 1997). A student who solves problems at the end of a textbook chapter receives feedback only after the teacher has graded the solutions, a day later. Since there is feedback, the instruction is Interactive, but it is low on the interactivity scale. Typical CAI systems are more interactive: they offer feedback at the end of each problem. In addition to indicating whether an overall problem solution is correct, they may provide short explanations with respect to certain anticipated wrong answers. Cognitive Tutors are even more interactive. Compared to these alternatives, the assistance provided by Cognitive Tutors is more frequent, often more detailed, and more explicit. Cognitive Tutors offer feedback on problem steps, not just final solutions. Further, they offer solution-specific step-by-step hints and adaptive problem selection. These differences in interactivity are sometimes analyzed in terms of nested (instructional) loops (VanLehn 2006).

Compared to a skilled human tutor, often thought to be the most effective form of instruction (Bloom 1984), Cognitive Tutors are close in their level of interactivity. They have much in common with the way human tutors support students as they work through problems (Merrill *et al.* 1992), even if human tutors are capable of more flexible dialogue with students and may have a wider range of instructional and motivational strategies than Cognitive Tutors do (Lepper and Malone 1987).

Related to the framework outlined in the introduction, it is interesting to consider how Cognitive Tutors balance the giving and withholding of information. At first blush, it may seem that Cognitive Tutors are heavy "information givers." They provide yes/no feedback after each attempt by a student at solving a step, sometimes accompanied by an error feedback message. They also provide detailed solution-specific hints for each step. However, Cognitive Tutors also withhold a considerable amount of information: First, these systems present problems, not worked-out examples. It is up to the student to generate the solution steps. Second, hints are given mainly at the student's request, and far less often, at the tutor's initiative. Within hint sequences, information is revealed gradually, with subsequent hint levels being displayed only when the student requests more information. Also, there is an incentive for students not to ask for hints before their first attempt at solving a step, namely, that such hint requests typically cause the tutor to revise its estimate of their knowledge component mastery in the downward direction. Third, even when yes/no feedback is provided, information is being withheld: the feedback does not give the answer, for example. This combination of features was thought to balance the benefits of information giving and withholding in a reasonable way—see related discussion by Anderson (1993) and a summary by Aleven *et al.* (2003).

The separate elements of this strategy are consistent to varying degrees with the ACT-R theory, which has served as both a guide to Cognitive Tutor design and as a target for insights from tutor experiments that have led to revisions in the theory. Learning by doing (i.e., by solving problems) is a key tenet of the ACT-R theory (Anderson 1993; Anderson and Lebiere 1998), which claims that production rules are acquired and strengthened through use in problem solving and reasoning. In designing the Cognitive Tutor feedback strategy, an overriding concern was to minimize floundering on the part of students. Under the ACT-R theory, learning occurs when production rules are applied successfully in the course of problem solving. Allowing students to spend extended time to pursue incorrect paths does not contribute to this process and thus can waste valuable learning time (see Anderson *et al.* 1995). To keep students focused on successful learning experiences, Cognitive Tutors provide feedback immediately whenever a student makes an error in attempting to apply a targeted knowledge component.

Since hints are given mostly at the student's request, Cognitive Tutors rely on students to find a good balance between the provision and withholding of explanatory information (as opposed to yes/no feedback on the correctness of their solution steps). It was thought that students would be in a better position than the tutor to judge when they could benefit from more explicit information than that contained in the tutor feedback. For example, after an error, it may be difficult for the tutor to judge whether the error is the result of a fundamental difficulty or is simply a slip. A student may be in a better position to make that judgment.

Empirical Support for Interactivity in Cognitive Tutors

Two strands of empirical evidence support the interactive features found in Cognitive Tutors: studies focused on the overall effectiveness of the tutors and the curricula of which they are part, and studies focused on the effect of individual elements of interactivity. Together, these studies make a strong case that Cognitive Tutors are highly effective and that many interactive elements contribute to their overall effectiveness. Other relevant research on related issues comes from the older computer-aided instruction tradition, which has yielded similar conclusions and open questions (Eberts 1997; Kluger and DeNisi 1996). We re-interpret the studies focused on individual interactive elements as investigating where the balance between information giving and information withholding should lie, within a context of problem-solving practice. By doing so we take a modest first step toward theory formation around this issue.

Overall effectiveness

Evaluations of the overall effectiveness of Cognitive Tutors show significant advantages over common learning environments that do not involve computer tutors, such as mathematics classroom instruction or a traditional programming environment for Lisp. An early study of the Geometry Proof Tutor (a pre-cursor of the Geometry Cognitive Tutor shown in Fig. 4) in classrooms, showed large learning gains, due to the tutor, and showed that students who had worked with this tutor scored about one standard deviation better than students taught by the same teacher who did not work with the tutor (Anderson *et al.* 1995). The benefit was not observed for students working with the tutor in pairs, indicating that the tutors support individual learning more effectively than collaborative learning. Further, early studies involving the Lisp Tutor showed 30–43% higher learning gains and 30–64% more efficient learning, compared to working in a standard Lisp programming environment (Anderson *et al.* 1995).

A number of studies have evaluated the effectiveness of the complete Cognitive Tutor Algebra course. This year-long course combines text materials and classroom activities, which students typically use for 3 days a week, and the Algebra Cognitive Tutor, which they typically use 2 days a week. In studies in Pittsburgh and Milwaukee, students in the Algebra Cognitive Tutor curriculum were compared to students in a standard algebra curriculum (Koedinger *et al.* 1997). Students in the Cognitive Tutor curriculum scored 15–25% higher on items taken from standardized tests and 50–100% higher on test items that involved problem solving and the use of representations. A number of subsequent studies, some of which were conducted by third parties, confirmed the advantages of the Cognitive Tutor curriculum (Morgan and Ritter 2002; Plano 2004; Sarkis 2004; Shneyderman 2001). In a number of these studies, the effect was particularly pronounced for special education students, non-native speakers of English, and low-income students (Plano 2004; Sarkis 2004). See also http://www.carnegielearning.com/approach_research_reports.cfm. Data collected in the Pittsburgh School District show that students taking a three-course sequence of Cognitive Tutor Algebra, Geometry, and Algebra II, did 30% better on TIMMS test items and 227% better on a task involving real-world problem solving than did students in a comparable school who took traditional courses. Finally, evaluations of the Cognitive Tutor curricula for middle-school math also indicate that students learn better with the Cognitive Tutor math curricula than with other curricula (Koedinger 2002).

The studies discussed above provide evidence of the overall effectiveness of Cognitive Tutors curricula, compared to other forms of instruction. They however do not show that it is the tutors' interactivity per se that causes the effect, given that there were a number of additional differences between the Cognitive Tutor curricula and the comparison curricula. In the next section we discuss a number of studies focused on individual elements of interactivity that do allow for tighter causal attributions.

Interactivity 1: immediate yes/no feedback

Although we have argued that good instruction should find an appropriate balance between the giving and withholding of information, there is not as yet a strong theoretical basis for predicting where the balance should lie—currently, finding the right balance is an empirical question. Let us consider where the balance should lie immediately following a student's attempt at solving a step in a tutor problem.

As mentioned, in designing the Cognitive Tutors' feedback strategy, an overriding concern was to minimize students' floundering in an attempt to detect and fix errors, for which the ACT-R theory predicts no benefit. Therefore, Cognitive Tutors inform students of the correctness of solution steps as soon as students enter them. One might wonder, however, whether it would not be better to allow potential advantages of information withholding to occur after students enter solution steps, for example by delaying the feedback until it is clear that the student is not going to repair any errors they may have made.

A study by Corbett and Anderson (1995), however, provided no support for the effectiveness of information withholding via simple forms of delayed feedback. Using the Lisp tutor, they compared four feedback conditions: immediate feedback (where feedback is given immediately following a student's attempt at solving a step, as in the regular Cognitive Tutor), flag feedback (where the tutor flags errors but provides no feedback messages until the student asks for them), on demand feedback (where errors are not flagged until the student requests feedback), and no feedback (where feedback is given only at the end of each programming exercise). They found that the three feedback conditions led to better and faster learning compared to the no feedback condition. There was no difference in the learning results between the three feedback conditions, but there was a difference in the amount of time spent to complete a fixed set of problems. The students in the immediate feedback condition did so the fastest, about three times faster than the students in the no feedback condition. Somewhat surprisingly, the students in the demand feedback condition did not request feedback very often. In 90% of the programming exercises, they did not ask for feedback until they had reached a preliminary solution.

This result provides strong experimental support for one of the key interactive features of Cognitive Tutors, namely, the immediate provision of yes/no feedback after student problem-solving steps. Withholding of this information was shown to be counterproductive. On the other hand, the study result does leave open the possibility that a different method of timing yes/no feedback is more successful. A study addressing this issue is described in a later section of the paper.

Interactivity 2: feedback content

If it is fruitful to provide yes/no feedback after problem-solving steps, how about providing even more information? For example, the tutor could provide explanatory feedback, in addition to yes/no feedback, in an effort to make the learning process more "explicit" or to reduce floundering. But what kinds of explanations are most helpful in this regard?

A study by McKendree (1990) varied the feedback content in the Geometry Proof Tutor along two dimensions, which were crossed in a 2×2 design: whether the feedback included goal information (i.e., pointed out the subgoal that the student should be pursuing, if they made a wrong choice) and whether it included condition violation information (i.e., pointed out an error in the way the student applied a chosen geometry theorem or definition). She found that feedback with explanatory content supports performance and learning better than yes/no feedback. The difference was statistically significant for the post-test error rate and marginally significant for the error rate during training. She found also that students who received explanatory feedback were more likely to correct their errors on subsequent attempts than did students who only received yes/no feedback. The advantages were strongest in the groups who received feedback on goals.

A likely interpretation of the result is that the goal statements in the feedback messages helped reduce floundering after errors. The better performance on the post-test suggests perhaps that they also led to a more explicit learning process. (See also Anderson *et al.* 1995, p. 191.) The result of the study suggests further that it is not necessary to be concerned that meaningful feedback after errors is difficult to provide if the tutor does not know the exact nature of the students' difficulties (see the discussion above). Apparently, when a good guess can be made about how to help students proceed with the current problem, it is an effective strategy to provide that information. Different information may be helpful in different domains. For example, in solving algebraic equations, it may be helpful to provide goal information, but not in a situation where the next goal to work on is obvious (e.g., in a worksheet-style interface). The broader conclusion however is that giving more information after problem-solving errors than just yes/no feedback is helpful, in particular if that information helps to reduce floundering or make learning more explicit.

Interactivity 3: hint content and timing

In contrast to feedback messages, which are given in reaction to a student's previous attempt at a problem-solving step, tutor hints provide information about the next step a student may perform. Two questions about the current Cognitive Tutor approach to providing hints are relevant: (1) are hints containing principle-based explanations effective in supporting learning beyond hints that simply provide the next step? and (2) is it better to provide hints on demand only (a kind of information withholding) or also for the tutor to proactively provide hints? Although these questions have not been fully answered yet by empirical studies, the tentative answer to the first question appears to be yes, and to the second question appears to be no.

Anderson *et al.* (1989) conducted a study in which they evaluated the effect of the tutor's mastery learning method and of explanatory content in both the tutor's hints and its feedback messages. They compared the regular Lisp Tutor, which provides explanatory content in its hints and in some of its error feedback messages, with a version that simply told students they were wrong when they made errors, or gave them the correct answer when they requested a hint. They found that explanatory messages help students learn faster, but not better. They speculated that the students in the no explanations condition, after seeing the answers provided by the system, were able to generate their own explanations of the answers, but that it took extra time to do so.

Together, these studies provide suggestive evidence (albeit not decisive evidence) that the content of on-demand hints can have an effect on students learning results. Aleven *et al.* (2003) provide a review of similar results including studies by Schworm and Renkl (2002) that showed that on-demand hints lead to better learning in a system for example studying.

Given that principle-based explanations are effective, how can we make sure that students get them when they need them? As mentioned, Cognitive Tutors give hints primarily at the student's request. Thus, it is the student (and not the system) who works toward achieving an effective balance between information received and information generated. It was thought that students would be better at doing so than computer tutors. The evidence is mounting, however, that students are not good at seeking assistance or information at the right time. The evidence stems both from middle-school to high-school students (e.g., Aleven and Koedinger 2000a; Aleven *et al.* 2006, 2003; Baker *et al.* 2004;

Koedinger and Anderson 1993). For example, log data from the Geometry tutor indicate that students frequently use bottom-out hints to obtain answers, without reading prior hints that explain why the answer is what it is. Other forms of poor help seeking were quite frequent as well. For example, even after multiple errors on a step, students often do not request help. These results contradict the notion that students may be better able than the system to decide when they can benefit from the tutor's help messages. It is hard for them to request help at the right time, just as it is hard for them, as we saw above, to request feedback at the right time.

There are a number of different ways of thinking about the design implications of these findings. The first is to redesign the tutor so that it achieves a better balance between withholding and providing problem-solving hints, for example by making it provide more information proactively after problem-solving errors. The second is to focus on helping students learn to create a better balance for themselves. For example, the system could provide remedial instruction to mitigate the negative effects of poor help-seeking decisions ("gaming" the system; e.g., Baker *et al.* 2004). Alternatively, one could extend the tutor so that it provides tutoring not just with respect to domain-specific knowledge components (e.g., algebra or geometry) but also with respect to students' help-seeking skills (Aleven *et al.* 2006; Baker *et al.* 2006; Roll *et al.* 2006). The aim of this kind of metacognitive instruction is for students to learn to balance, themselves, when to seek information and assistance versus when to try to think on their own. If that ability can be successfully acquired in a domain-general transferable way, it would enable students to do better in learning environments that do not provide a good balance between information-giving and information-withholding. This tough challenge for tutor designers is discussed further below.

Interactivity 4: knowledge component assessment and mastery learning

The third main form of interactivity in Cognitive Tutors (see Table 3) is the mastery learning method. It represents a form of assistance giving that involves choosing problems for students to solve as opposed to students choosing problems themselves (it is information giving at the metacognitive level). As mentioned, Cognitive Tutors select problems on an individual basis, focusing on knowledge components that the student has not mastered yet, until sufficient evidence has accumulated that the student masters all knowledge components targeted in a given curriculum section. Three studies provide evidence of the effectiveness of the tutor's mastery learning method, suggesting that information providing at the meta-cognitive level is a good thing to do.

Mastery learning is of course not new with Cognitive Tutors (e.g., Bloom 1984; Guskey 1987), although its implementation in Cognitive Tutors is different from most other implementations in that students' mastery is evaluated on an individual basis and with respect to detailed knowledge components (Anderson *et al.* 1995). Thus, the main question addressed in the studies of mastery learning in Cognitive Tutors is whether individual mastery learning focused on production rules as the units of learning is successful, and whether the tutor's particular implementation of this idea is adequate. Specifically, whether the model-tracing and knowledge-tracing algorithms working in tandem assess a student's skill accurately and whether the tutor's problem selection algorithm selects appropriately difficult problems.

Two early studies involving the Lisp Cognitive Tutor showed that the tutor's mastery learning method leads to improved learning (Anderson *et al.* 1989; Corbett and Anderson 1995). In these studies, a version of the tutor with the mastery learning approach was compared against a version that assigned a fixed set of tutor problems to all students, regardless of their performance. The students in the mastery condition had significantly higher learning gains, confirming that the tutor was successful in assigning extra problems that were not redundant with what the students knew already. A third study took into account the time that students spent. The study showed that the mastery approach leads to large learning gains at the cost of only very little extra time spent (Corbett 2001). An interesting finding from this study is that the students in the mastery learning condition solve many more problems, but spend almost no extra time.

The three studies described above provided ample evidence that the mastery-learning mechanism is an effective way of improving student learning, without great cost in terms of time spent. Thus, they indicate that information giving at the meta-cognitive level can be effective. As a practical matter, the mastery learning method is a valuable addition to Cognitive Tutor technology. It is used in both the Algebra and Geometry Cognitive Tutors (see Figs. 1 and 4). In addition to supporting individualized problem selection, the mastery learning mechanism helps provide students with a goal to work for, namely, to get their skill bars ticked off by the tutor. Also, the mastery-level criterion discourages a "gaming" strategy by which students repeatedly ask for hints until the next problem-solving step is revealed to them. Under the mastery-level criterion, this strategy yields short-term success only. It helps in getting through the problem at hand, but it will lead the tutor to assign more problems later on. Students typically are aware of that fact, although that does not always stop them from using this strategy. It remains an open question for future experimentation whether displaying the skill bars provides motivational and learning benefits above and beyond the benefits of knowledge tracing for problem selection and pacing.

Implications

The studies presented above provide strong evidence for the effectiveness of Cognitive Tutors over other forms of instruction, including typical classroom instruction. They also support the main interactive features listed in Table 3 (yes/no feedback, on-demand hints, and mastery learning). With respect to the information giving/withholding dimension, the results of the studies on yes/no feedback, feedback content, and hint timing consistently indicate that giving information after a problem-solving step is better than withholding it. This conclusion should not be interpreted as sweeping support for information giving in general, because it is important to recall that these strategies were evaluated in a context in which students were engaged in active problem solving, which is an important kind of information/assistance withholding. Instead, these techniques for tutored problem solving provide a particular approach to effective balancing of giving and withholding. The Corbett *et al.* study showed that immediate yes/no feedback is better than no feedback or delayed feedback. The McKendree study showed that explanatory error feedback is better than just yes/no feedback. Finally, the study on hint use confirms that on-demand hints are often not used as intended by students. Generalizing beyond the specifics of the studies, it seems implied that within a context of tutored problem solving, information should be withheld very sparingly and that subsequent research on improving tutored problem solving may be more successful if it focuses on methods to *give* more information rather than methods to *withhold* it.

Enhancing Cognitive Tutors: What Does and Does Not Work

Should worked examples be added to cognitive tutors?

Given numerous laboratory results showing benefits of alternating worked examples with problem solving over problem solving alone (Atkinson *et al.* 2000; Renkl 2002; Sweller and Cooper 1985; Trafton and Reiser 1993; Zhu and Simon 1987), we began to wonder whether Cognitive Tutors, which have students performing problem solving, might be enhanced by adding worked examples. Mathan (2003) began investigating the addition of worked examples within a Cognitive Tutor for Excel Programming.[2] Given the passive nature of worked examples (cell 4 in Table 1, passive/implicit), Mathan (2003) decided to create a more active form of worked example whereby students were told the steps to perform, but were required to answer questions about reasoning toward these steps and perform the steps themselves. These "walkthroughs," as Mathan (2003) called them, require activity on the students' part and included feedback. They thus belong in the interactive column of Table 1. He found increased learning due to walkthroughs in two of four comparisons and no difference in the other two. The results suggest benefits of adding such interactive worked examples to Cognitive Tutors in certain situations. However, they were not a necessary part of the conditions that achieved the best results in Mathan's (2003) studies (discussed further below).

In a more direct adaptation of successful laboratory results on worked examples, McLaren *et al.* (2006) inserted worked examples between tutored problems in a Cognitive Tutor for chemistry. In contrast to numerous prior studies, they found no benefit for the addition of these worked examples. Students in the problem-solving condition learned just as much as those in the interspersed worked-example and problem-solving condition. This result was replicated with both college and high school students and thus, appears to not be a consequence of an "expertise-reversal effect" whereby the benefits of examples fade and reverse as students develop expertise (Kalyuga *et al.* 2001). It appears the key difference is that in prior studies, the problem-solving or practice activities did not involve regular feedback (they are active, but not interactive), whereas in the study of McLaren *et al.* (2006) the problem-solving activity was tutored, that is interactive.

Tables 1 and 2 can be used to interpret the difference in the results. In the prior studies, combining worked examples and problem-solving practice (the passive/implicit and active/implicit cells 4 and 5 in Table 1) improves on problem-solving practice alone (cell 5, active/implicit) because the combination represents a better mix of information giving (examples) and withholding (practice). Problem-solving practice alone has too many of the costs of information withholding (Table 2)—without having a good sense for what are the correct domain knowledge components (operators, concepts, principles or strategies) students flounder and make too many errors. This interpretation is similar to the "extraneous cognitive load" explanation provided by others (Clark and Mayer 2003; Paas and Van Merrienboer 1994; Sweller 1988). However, what is extraneous in this interpretation is not the reasoning needed during problem solving per se, as posited by Sweller (1988), but the errors and non-productive search that occur due to lack of available information on correct domain knowledge.

In contrast, in the study of McLaren *et al.* (2006) study, worked examples are not added to *unsupported* problem-solving practice, but to *tutored* problem-solving practice. Because the tutored problem-solving practice group (interactive/implicit cell 6 in Table 1) gets feedback on their errors and can request hints if needed, they too have a mix of information/assistance giving and withholding. Information is withheld while students are successfully

[2] This effort was not the first to incorporate examples in Cognitive Tutors as the original LISP tutor had hypercard declarative instruction and examples interspersed with problem-solving practice in the tutor. Use of worked examples in the Cognitive Tutor Algebra course, both in text materials and in the tutor, were discouraged by our collaborating instructors (cf., Koedinger *et al.* 1997) because it was thought that urban students, if they processed the examples at all, would do so shallowly, which would impede deeper conceptual understanding that might better come from classroom discussion and collaborative projects.

engaging in problem solving, but the potential error/floundering cost of withholding is reduced because students are immediately cued when they make errors. Further, the benefits of information giving are present because accurate and timely information is provided, through hints at the students' request (which often follow an error). In the context of tutored practice as opposed to untutored practice, the information-giving benefits of worked examples may essentially be redundant. In essence, the tutor dynamically converts a problem-solving experience into an annotated worked example when the student is having enough trouble such that they request the final "bottom-out" level of hint that tells them what to do next.

To summarize, while untutored problem solving practice may suffer from not enough information giving for beginning learners, interactive tutoring of problem solving may provide sufficient information, in the form of immediate step-based feedback, just when students need it. Relative to tutored problem solving, the information giving in interleaved worked example study may be superfluous in beginning learning and may tip into too much information giving in later learning.

In contrast to the McLaren *et al.* (2006) result, Schwonke *et al.* (2007) found a benefit of adding worked examples to the Geometry Cognitive Tutor, but with a twist. To adjust information giving/withholding to learners' growing competence, they implemented a gradual transition from example study to problem solving in the form of "faded" worked examples (Renkl *et al.* 2004). Students first see an example, where answers to all problem steps are given, and then in subsequent examples the answers to steps are gradually taken away or "faded" as so as to convert examples into problems. Both example steps and problem-solving steps are interactive (or tutored). On the example steps, students are asked to explain the worked-out steps and receive feedback on their explanations (see the next section on self-explanation). Students in the faded-example condition learned more efficiently, taking significantly less instructional time to achieve better post-test outcomes on declarative knowledge and equal outcomes on procedural knowledge (Schwonke *et al.* 2007). These results highlight the possibility that in interactive instruction that involves interleaving examples and problems, it may be better for beginning learners to have information-giving examples come before information-withholding problems. And then transition to tutoring, where examples follow problems in the form of as-needed hints, as learners begin to develop greater independent competence on targeted knowledge components.

Exploring self-explanation in cognitive tutors

Perhaps one of the most important findings regarding learning and instruction in the past 20 years is the role of "self-explanation" in effective learning of complex reasoning and problem solving. Chi *et al.* (1989) found that poor learners of physics skim worked-examples in textbooks and make shallow analogies when solving homework problems whereas good learners try to explain to themselves the reasoning from one step to the next and then make deeper analogies during problem-solving practice. In terms of Table 1, while implicit example-based induction is powerful for learning (cell 4, passive/implicit), it can be enhanced by the more explicit rule-based reasoning behind self-explanation (cell 2, active/explicit). VanLehn *et al.* (1992) provided a computational model of good versus poor learners in terms of how good learners are more likely to try to fill the gap between steps in an example by chaining together existing, perhaps intuitive, knowledge to derive the result of the step.

Prompting students to self-explain has been shown in laboratory studies to enhance learning (Atkinson *et al.* 2003; Chi *et al.* 1994; Renkl *et al.* 1998; Siegler 2002). Self-explanation support has also been shown to be effective in classroom use (Aleven and Koedinger 2002) and has "gone to scale" as it is implemented as part of the Cognitive Tutor

Geometry course. Such benefits can be successfully implemented in a computer-interpretable form (explaining by reference to names of reasons in a glossary, which is essentially a long menu) that facilitates automatic feedback on the correctness of those explanations.

We can contrast self-explanation without feedback (cell 2 in Table 1, active/explicit) with self-explanation with feedback (cell 3, interactive/explicit). Most past studies have prompted for self-explanations without providing students with feedback on whether or not those self-explanations are correct and accurate. They typically involve the presence of an experimenter in a one-on-one interpersonal setting. This setting produces social demands for the student to comply with the request that they try to self-explain. Self-explanation without feedback may work in this lab setting because of the presence of the experimenter and the implicit "demand characteristics" on the student to make an effort.

In contrast, compliance may be reduced in a classroom or homework setting where students do not have an adult nearby. Indeed, we created a new self-explanation version of the Geometry Cognitive Tutor where students were prompted to type principle-based self-explanations (e.g., "the angles in a triangle sum to 180") instead of referencing a reason in glossary. In an initial pilot study in a high school (Aleven and Koedinger 2000b), students were not given feedback on their explanations and we found they only rarely made reasonable attempts at explanations (less than 10% of the time) and instead often provided inadequate explanations, and even more often gave off-task explanation responses like "because I said so."

These students appear to be missing the motivation, self-discipline, or meta-cognitive skills to seriously engage in self-explanation without feedback. For such students, interactive forms of instruction that provide feedback may be fairly critical to keeping them on task. It may be that for more motivated and better-prepared students, feedback on self-explanations is not necessary.

Indeed, the best students learn reasonably effectively (though perhaps not as efficiently as possible)[3] from passive forms of instruction like textbook descriptions and examples (cells 1 and 4 in Table 1). Can we help poor students become better learners through more direct forms of metacognitive instruction (cf., Schoenfeld 1983; White and Frederiksen 1998)? As described in the next section, we have been exploring this question in the context of Cognitive Tutors.

Other cognitive tutor studies of metacognitive support

Improving learning through error self-detection and correction Despite the successful demonstration of immediate feedback in the LISP tutor study described above, many have been critical of the notion of immediate feedback in tutors. To some, immediate feedback seems to play the assistance dilemma too much in the direction of assistance giving. In fact, some critics view Cognitive Tutors as being on the information/assistance-giving end of a simple dichotomy—a view that is oversimplified as indeed tutors that provide immediate feedback during problem-solving practice withhold more information than textbooks or worked-examples. Nevertheless, it is possible that immediate feedback is stealing from students an opportunity to learn to detect their own errors and to learn from them[4] (Schmidt and Bjork 1992).

[3] As described above, even excellent college students at Carnegie Mellon University benefited in time savings from the immediate feedback of the LISP tutor.

[4] In fact, immediate feedback tutors do allow students to learn from their errors—after all students are required to correct their errors and can and often do so without further assistance. However, they are not given the opportunity to learn from the *downstream consequences* of their errors.

Using the Cognitive Tutor for Excel described above, Mathan and Koedinger (2005) explored whether a particular form of delayed feedback can improve student learning. They reinterpreted the feedback debate as being more about the "model of desired performance" that is the goal of instruction than about the delay between student response and tutor feedback. In contrast to the goal of producing error-free experts, one might want to produce "intelligent novices" who may make initial errors, but are able to detect them and correct them. If immediate feedback is given relative to an intelligent novice model of desired performance (where certain initial errors are allowed, provided that the student catches them right away), it will appear delayed in comparison to immediate feedback relative to an expert model of desired performance (where all errors get immediate feedback). Students learned more from the intelligent novice tutor. Why? It was probably not by improving meta-cognitive "evaluative" skills (as Schmidt and Bjork call them). We would expect such skills to be acquired gradually over time, but a learning curve analysis showed that the benefit was already present after students' first opportunity to experience the feedback manipulation. In addition, improvement in evaluative skills would be consistent with a pattern of post-test results whereby the groups are equal on an immediate post-test, but separate on measures of robust learning, like long-term retention and transfer. In fact, the treatment was better on all measures, both immediate and robust learning measures.

An alternative interpretation is that through seeing the consequences of their errors, which are particularly apparent in Excel, students were able to explicitly reason about how their initial attempts led to errors and how they could be modified to achieve the desired outcome. This interpretation appears consistent with Siegler's (2002) results that having students do self-explanations not only of correct solutions, but also incorrect solutions improves their learning. He argues that learners must not only acquire and strengthen correct knowledge components, but also weaken and eliminate incorrect knowledge. In effect, the intelligent novice condition helped students rationalize, like the derivations in the self-explanation model by VanLehn *et al.* (1992), why their misconceptions do not work. It also helped them rationalize why critical relevant features must be included in correct knowledge components. Nathan (1998) also demonstrated benefits of providing feedback that helps students see meaningful downstream consequences of their errors and Ohlsson (1996) provided a detailed theory and computational model of learning from errors.

Cognitive Tutors for improving help-seeking skills and reducing 'gaming' Besides meta-cognitive support for self-explanation and error self-correction, we have explored providing meta-cognitive support for improving student help-seeking skills and reducing unproductive learning behaviors. This work was inspired by data mining of logged student–tutor interactions in which we noticed a high frequency of what appeared to be non-ideal student behaviors that were correlated with poor learning (Aleven and Koedinger 2000a). As mentioned above, despite the feelings of many advocates for greater student control in interactive systems, when given that control students do not always use interactive features as intended. On the one hand, students sometimes attempt to circumvent information withholding (i.e., avoid thinking) by engaging in fast and repeated guessing or asking for hints more often or faster than appropriate—Baker coined the phrase "gaming the system" to describe such behaviors (Baker *et al.* 2004). On the other hand, students sometimes avoid information giving by not seeking help when it is likely needed.

Early efforts to create Cognitive Tutors to improve help seeking and reduce gaming have resulted in some limited success. We developed a help-seeking tutor and integrated it into the Geometry Cognitive Tutor. This adjunct tutor provides feedback on students' help-seeking behavior, as they use the tutor to solve geometry problems. For example, the help-seeking tutor provides feedback when students engage in the maladaptive help-seeking behaviors described above. An initial study showed reductions in some poor help-seeking behaviors during instruction, but no consequent improvements in geometry learning (Roll *et al.* 2006). In a different effort, a Data Analysis Cognitive Tutor was enhanced with a machine-learning-based gaming detector (Baker et al. 2006). When the enhanced tutor detected gaming behavior, it responded by emotional displays of an animated agent and by presenting supplementary exercises on related material. Students with the gaming tutor showed reduced overall gaming behavior and the number of supplementary exercises received was associated with better learning. However, the difference in domain learning was not statistically reliable across the whole sample, perhaps because harmful gaming was present in a relatively small subset (about 10%) of the students.

It is still an open question as to why students engage in gaming behavior. It may be in part a rational effort by less knowledgeable students to compensate for what may be, from their perspective, premature information withholding. Some "gamers" may be hurrying to get an example of a correct step in order to have something to study because they are, as of yet, incapable of correctly generating the step themselves. For instance, Crowley and Medvedeva (2006) found that a subset of medical students using an intelligent tutor engage in gaming-like behavior during early problems in the curriculum and evidence greater independent success on later problems. These arguably established good learners may essentially be using the tutor's bottom-out hints to create worked-examples for themselves and may be choosing to engage in self-explanation rather than be explained to.

Conclusion

Instructional interaction should optimize student involvement, not maximize or minimize it. The potential benefits of withholding information or assistance are many, including allowing students to learn by doing, to construct knowledge, to benefit from generation effects, to reduce zoning out, to engage recall from long-term memory, and to provide knowledge checks that prevent them from thinking they know when they do not. The costs are also numerous. Students may get stuck, make mistakes and strengthen error pathways, find it too hard and cognitively disengage. We have provided a number of examples of how experiments within Cognitive Tutors have explored trade-offs between giving and withholding instructional assistance.

Cognitive Tutors initially withhold information about problem solutions and solution steps, and then interactively *add* information, only as needed, through yes/no feedback, explanatory hints, and dynamic problem selection. The reviewed studies provide support for this particular approach to balancing the giving and withholding of information and for its individual interactive elements. This result should not be construed as supporting information giving in general, or greater interactivity in general. We also do not mean to claim that Cognitive Tutors currently strike an ideal balance—additional studies show that faded examples (a subtle form of information giving) and feedback based on an intelligent novice model (a subtle form of information withholding) sometimes are improvements. It is likely that additional interactive elements will be found to be more effective than existing approaches.

Given that a key downside of information withholding is errors and floundering (if not complete failure), a rough criterion for deciding to give rather than withhold is when the task gets too difficult and thus the probability of error or unproductive thinking is high. What is the probability of error that is the ideal threshold point? That is a great question for future research.

In related work, Pavlik (2007) proposed an ideal error rate of about 5–25% to decide on the length of delay between practice trials (long delay between practice trials is a kind of information withholding). This estimate was based on ACT-R simulations and is supported in experiments comparing practice-scheduling algorithms where the derived expanded spacing schedule led to greater learning than alternatives based on prior theory and standard practice. To what extent and in what ways such an error rate threshold might generalize to more complex learning objectives than the fact associations explored by Pavlik is wide open.

The crux of the assistance dilemma is prescribing decision criteria (e.g., conditions and cut-off parameters) for when it is best to switch between information giving (more assistance) and information withholding (less assistance). This dilemma may be the fundamental open problem in learning and instructional science. Given the many different ways to provide information or assistance, there is not going to be one solution to the assistance dilemma. However, we believe it is critical to not only acknowledge the dilemma, as others have (e.g., Vygotsky 1978), but to strive toward characterizing qualitative conditions and quantitative threshold parameters that can aid instructional designers and instructors in making good decisions.

Further experimental studies and theoretical unification will be necessary to achieve this ambitious goal. The Pittsburgh Science of Learning Center (http://learnlab.org) is committed to supporting such an effort through both joint theory development and through providing an infrastructure, called LearnLab, to aid researchers in running tightly controlled experiments in real classroom settings and in doing microgenetic analyses of detailed logs of student interactions in these classrooms.

References

Aleven, V., & Koedinger, K. R. (2000a). Limitations of student control: Do students know when they need help? In G. Gauthier, C. Frasson, & K. VanLehn (Eds.), *Proceedings of the 5th International Conference on Intelligent Tutoring Systems*, *ITS 2000* (pp. 292–303). Berlin: Springer.

Aleven, V., & Koedinger, K. R. (2000b). The need for tutorial dialog to support self-explanation. In C. P. Rose & R. Freedman (Eds.), *Building dialogue systems for tutorial applications, papers of the 2000 AAAI Fall Symposium* (pp. 65–73). Technical Report FS-00-01. Menlo Park, CA: AAAI.

Aleven, V., & Koedinger, K. R. (2002). An effective metacognitive strategy: Learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science, 26*(2):147–179.

Aleven, V., McLaren, B., Roll, I., & Koedinger, K. (2006). Toward meta-cognitive tutoring: A model of help seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence and Education, 16*, 101–128.

Aleven, V., Stahl, E., Schworm, S., Fischer, F., & Wallace, R. M. (2003). Help seeking and help design in interactive learning environments. *Review of Educational Research, 73*(2), 277–320.

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

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

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.

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

Anderson, J. R., Reder, L. M., & Simon, H. A. (1996). Situated learning and education. *Educational Researcher, 25*(4), 5–11.

Anderson, J. R., Reder, L. M., & Simon, H. A. (1998). Radical constructivism and cognitive psychology. In D. Ravitch (Ed.), *Brookings papers on education policy 1998*. Washington, DC: Brookings Institute.

Atkinson, R. K., Derry, S. J., Renkl, A., & Wortham, D. (2000). Learning from examples: Instructional principles from the worked examples research. *Review of Educational Research, 70*(2), 181–214.

Atkinson, R. K., Renkl, A., & Merrill, M. M. (2003). Transitioning from studying examples to solving problems: Effects of self-explanation prompts and fading worked-out steps. *Journal of Educational Psychology, 95*(4), 774–783.

Baker, R., Corbett, A., Koedinger, K. R., Evenson, S., Roll, I., Wagner, A., *et al.* (2006). Adapting to when students game an intelligent tutoring system. In M. Ikeda, K. D. Ashley, & T.-W. Chan (Eds.), *Proceedings of the 8th International Conference on Intelligent Tutoring Systems* (pp. 392–401). Berlin: Springer.

Baker, R. S., Corbett, A. T., Koedinger, K. R., & Wagner, A. Z. (2004). Off-task behavior in the Cognitive Tutor classroom: When students "game the system." *Proceedings of ACM CHI 2004: Computer-Human Interaction* (pp. 383–390). New York: ACM.

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*, 4–16.

Chi, M. T. H., Bassok, M., Lewis, M. W., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science, 13*(2), 145–182.

Chi, M. T. H., de Leeuw, N., Chiu, M. H., & LaVancher, C. (1994). Eliciting self-explanations improves understanding. *Cognitive Science, 18*(3), 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.

Corbett, A. (2001). Cognitive computer tutors: solving the two-sigma problem. In M. Bauer, P. J. Gmytrasiewicz, & J. Vassileva (Eds.), *Proceedings of the 2001 International Conference on User Modeling* (pp. 137–147). Berlin: Springer.

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

Crowley, R. S., Legowski, E., Medvedeva, O., & Tseytlin, E. (2005). An ITS for medical classification problem-solving: Effects of tutoring and representations. In C. K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.), *Proceedings of the 12th International Conference on Artificial Intelligence*. Amsterdam, The Netherlands: AIED.

Crowley, R. S., & Medvedeva, O. (2006). An intelligent tutoring system for visual classification problem solving. *Artificial Intelligence in Medicine, 36*(1), 85–117.

Duch, B., Gron, S., & Allen, D. (2001). *The Power of Problem-Based Learning; A Practical "How To" For Teaching Undergraduate Courses in Any Discipline*. Sterling, Virginia: Stylus.

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

Guskey, T. R. (1987). The essential elements of mastery learning. *Journal of Classroom Interaction, 22*(2), 19–22.

Kalyuga, S., Chandler, P., Tuovinen, J., & Sweller, J. (2001). When problem solving is superior to studying worked examples. *Journal of Educational Psychology, 93*(3), 579–588.

Krajcik, J., & Starr, M. (2001). Learning science content in a project-based environment. In R. Tinker & J. S. Krajcik (Eds.), *Portable technologies: Science learning in context*. The Netherlands: Kluwer.

Kirschner, P. A., Sweller, J., & Clark, R. E. (2006). Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational Psychologist, 41*(2), 75–86.

Klahr, D., & Nigam, M. (2004). The equivalence of learning paths in early science instruction: Effects of direct instruction and discovery learning. *Psychological Science, 15*(10), 661–667.

Kluger, A. N., & DeNisi, A. (1996). The effects of feedback intervention on performance: A historical review, a meta-analysis and a preliminary feedback intervention theory. *Psychological Bulletin, 112*(2), 254–284.

Koedinger, K. R. (2002). Toward evidence for instructional design principles: Examples from Cognitive Tutor Math 6. In D. S. Mewborn, P. Sztajn, D. Y. White, H. G. Wiegel, R. L. Bryant, & K. Nooney (Eds.), *Proceedings of twenty-fourth annual meeting of the North American Chapter of the International Group for the Psychology of Mathematics Education Vol. 1* (pp. 21–49). Columbus, OH: ERIC Clearinghouse for Science, Mathematics, and Environmental Education.

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

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., & Corbett, A. T. (2006). Cognitive tutors: Technology bringing learning science to the classroom. In K. Sawyer (Ed.), *The Cambridge Handbook of the Learning Sciences*. Cambridge, MA: Cambridge University Press.

Lepper, M. R., & Malone, T. W. (1987). Intrinsic motivation and instructional effectiveness in computer-based education. In R. E. Snow & M. J. Farr (Eds.), *Aptitude, learning and instruction: Volume III Conative and affective process analyses*. Hillsdale, NJ: Erlbaum.

Mathan, S. A. (2003). *Recasting the feedback debate: Benefits of tutoring error detection and correction skills*. Unpublished dissertation, Carnegie Mellon University, Pittsburgh, PA.

Mathan, S. A., & Koedinger, K. R. (2005) Fostering the Intelligent Novice: Learning from errors with metacognitive tutoring. *Educational Psychologist, 40*(4), 257–265.

McKendree, J. E. (1990). Effective feedback content for tutoring complex skills. *Human Computer Interaction, 5*, 381–414.

McLaren, B. M., Lim, S., Gagnon, F., Yaron, D., & Koedinger, K. R. (2006). Studying the effects of personalized language and worked examples in the context of a web-based intelligent tutor. In M. Ikeda, K. D. Ashley, & T.-W. Chan (Eds.), *Proceedings of the 8th International Conference on Intelligent Tutoring Systems* (pp. 318–328). Berlin: Springer.

Merrill, D. C., Reiser, B. J., Ranney, M., & Trafton, J. G. (1992). Effective tutoring techniques: Comparison of human tutors and intelligent tutoring systems. *Journal of the Learning Sciences 2*(3), 277–305.

Morgan, P., & Ritter, S. (2002). An experimental study of the effects of Cognitive Tutor® Algebra I on student knowledge and attitude (Available from Carnegie Learning, Inc., 1200 Penn Avenue, Suite 150, Pittsburgh, PA 15222). Available at: http://www.carnegielearning.com.

Nathan, M. J. (1998). Knowledge and situational feedback in a learning environment for algebra story problem solving. *Interactive Learning Environments, 5*, 135–159.

Ohlsson, S. (1996). Learning from performance errors. *Psychological Review, 103*(2), 241–262.

Ohlsson, S., & Mitrovic, A. (2006). Constraint-based knowledge representation for individualized instruction. *Computer Science and Information Systems, 3*(1), 1–22.

Paas, F., & Van Merrienboer, J. (1994). Variability of worked examples and transfer of geometry problem-solving skills: A cognitive-load approach. *Journal of Educational Psychology, 86*, 122–133.

Pavlik, Jr., P. I. (2007). Timing is an order: Modeling order effects in the learning of information. In F. E. Ritter, J. Nerb, T. O'Shea & E. Lehtinen (Eds.), *In order to learn: How the sequences of topics affect learning*, (pp. 137–150). New York, NY: Oxford University Press.

Plano, G. S. (2004). *The Effects of the Cognitive Tutor Algebra on student attitudes and achievement in a 9th grade Algebra course*. Unpublished doctoral dissertation, Seton Hall University, South Orange, NJ.

Renkl, A. (2002). Learning from worked-out examples: Instructional explanations supplement self-explanations. *Learning & Instruction, 12*, 529–556.

Renkl, A., Atkinson, R. K., & Große, C. S. (2004) How fading worked solution steps works—A cognitive load perspective. *Instructional Science, 32*, 59–82.

Renkl, A., Stark, R., Gruber, H., & Mandl, H. (1998). Learning from worked-out examples: The effects of example variability and elicited self-explanations. *Contemporary Educational Psychology, 23*, 90–108.

Roll, I., Aleven, V., McLaren, B. M., Ryu, E., Baker, R., & Koedinger, K. R. (2006). The help tutor: Does metacognitive feedback improve students' help-seeking actions, skills and learning? In M. Ikeda, K. D. Ashley, & T.-W. Chan (Eds.), *Proceedings of the 8th International Conference on Intelligent Tutoring Systems* (pp. 360–369). Berlin: Springer.

Sarkis, H. (2004). *Cognitive Tutor Algebra 1 program evaluation, Miami-Dade County Public Schools*. Lighthouse Point, FL: The Reliability Group.

Schmidt, R. A., & Bjork, R. A. (1992). New conceptualizations of practice: Common principles in three paradigms suggest new concepts for training. *Psychological Science, 3*(4), 207–217.

Schoenfeld, A. (1983). Beyond the purely cognitive: Belief systems, social cognitions, and metacognitions as driving forces in intellectual performance. *Cognitive Science, 7*, 329–363.

Schwartz, D. L., & Bransford, J. D. (1998). A time for telling. *Cognition and Instruction, 16*(4), 475–522.

Schwonke, R., Wittwer, J., Aleven, V., Salden, R. J. C. M., Krieg, C., & Renkl, A. (2007). Can tutored problem solving benefit from faded worked-out examples? Paper presented at the European Cognitive Science Conference 2007, May 23–27. Delphi, Greece.

Schworm, S., & Renkl, A. (2002). Learning by solved example problems: Instructional explanations reduce self-explanation activity. In W. D. Gray & C. D. Schunn (Eds.), *Proceeding of the 24th Annual Conference of the Cognitive Science Society* (pp.816–821). Mahwah, NJ: Erlbaum.

Shneyderman, A. (2001). *Evaluation of the Cognitive Tutor Algebra I program, Miami-Dade County Public Schools, Office Of Evaluation And Research*. Retrieved from: http://oer.dadeschools.net/algebra.pdf#search=%22Shneyderman%20miami%20cognitive%20tutor%22.

Siegler, R. S. (2002). Microgenetic studies of self-explanation. In N. Garnott & J. Parziale (Eds.), *Microdevelopment: A process-oriented perspective for studying development and learning* (pp. 31–58). Cambridge, MA: Cambridge University Press.

Slamecka, N. J., & Graf, P. (1978). The generation effect: Delineation of a phenomenon. *Journal of Experimental Psychology: Human Learning and Memory, 4*, 592–604.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science, 12*(2), 257–285.

Sweller, J., & Cooper, G. A. (1985). The use of worked examples as a substitute for problem solving in learning algebra. *Cognition and Instruction, 2*, 59–89.

Trafton, J. G., & Reiser, B. J. (1993). The contributions of studying examples and solving problems to skill acquisition. In M. Polson (Ed.), *Proceedings of the Fifteenth annual conference of the Cognitive Science Society* (pp. 1017–1022). Hillsdale, N.J.: Erlbaum.

VanLehn, K. (2006). The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education, 16*, 227–265.

VanLehn, K., Jones, R. M., & Chi, M. T. H. (1992). A model of the self-explanation effect. *The Journal of the Learning Sciences, 2*, 1–59.

Vanlehn, K., Lynch, C., Schultz, K., Shapiro, J. A., Shelby, R. H., Taylor, L., et al. (2005). The Andes physics tutoring system: Lessons learned. *International Journal of Artificial Intelligence in Education, 15*(3), 147–204.

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

Ward, M., & Sweller, J. (1990). Structuring effective worked examples. *Cognition and Instruction, 7*, 1–39.

White, B. Y., & Frederiksen, J. R. (1998). Inquiry, modeling, and metacognition: Making science accessible to all students. *Cognition and Instruction, 16*(1), 3–118.

Zhu, X., & Simon, H. A. (1987). Learning mathematics from examples and by doing. *Cognition and Instruction, 4*(3), 137–166.