AI Magazine Volume 22 Number 4 (2001) (© AAAI)

Intelligent Tutoring Systems with Conversational Dialogue

Arthur C. Graesser, Kurt VanLehn, Carolyn P. Rosé, Pamela W. Jordan, and Derek Harter

■ Many of the intelligent tutoring systems that have been developed during the last 20 years have proven to be quite successful, particularly in the domains of mathematics, science, and technology. They produce significant learning gains beyond classroom environments. They are capable of engaging most students' attention and interest for hours. We have been working on a new generation of intelligent tutoring systems that hold mixed-initiative conversational dialogues with the learner. The tutoring systems present challenging problems and questions to the learner, the learner types in answers in English, and there is a lengthy multiturn dialogue as complete solutions or answers evolve. This article presents the tutoring systems that we have been developing. AUTOTUTOR is a conversational agent, with a talking head, that helps college students learn about computer literacy. ANDES, ATLAS, AND WHY2 help adults learn about physics. Instead of being mere information-delivery systems, our systems help students actively construct knowledge through conversations.

Intelligent tutoring systems (ITSs) are clearly one of the successful enterprises in AI. There is a long list of ITSs that have been tested on humans and have proven to facilitate learning. There are well-tested tutors of algebra, geometry, and computer languages (such as PACT [Koedinger et al. 1997]); physics (such as ANDES [Gertner and VanLehn 2000; VanLehn 1996]); and electronics (such as SHERLOCK [Lesgold et al. 1992]). These ITSs use a variety of computational modules that are familiar to those of us in the world of AI: production systems, Bayesian networks, schema templates, theorem proving, and explanatory reasoning. According to the current estimates, the arsenal of sophisticated computational modules inherited from AI produce learning gains of approximately .3 to 1.0 standard deviation units compared with students learning the same content in a classroom (Corbett et al. 1999).

The next generation of ITSs is expected to go one step further by adopting conversational interfaces. The tutor will speak to the student with an agent that has synthesized speech, facial expressions, and gestures, in addition to the normal business of having the computer display text, graphics, and animation. Animated conversational agents have now been developed to the point that they can be integrated with ITSs (Cassell and Thorisson, 1999; Johnson, Rickel, and Lester 2000; Lester et al. 1999). Learners will be able to type in their responses in English in addition to the conventional point and click. Recent developments in computational linguistics (Jurafsky and Martin 2000) have made it a realistic goal to have computers comprehend language, at least to an extent where the ITS can respond with something relevant and useful. Speech recognition would be highly desirable, of course, as long as it is also reliable.

At this point, we are uncertain whether the conversational interfaces will produce incremental gains in learning over and above the existing ITSs (Corbett et al. 1999). However, there are reasons for being optimistic. One reason is that human tutors produce impressive learning gains (between .4 and 2.3 standard deviation units over classroom teachers), even though the vast majority of tutors in a school's system have modest domain knowledge, have no training in pedagogical techniques, and rarely use the sophisticated tutoring strategies of ITSs (Cohen, Kulik, and Kulik 1982; Graesser, Person, and Magliano, 1995).

A second reason is that there are at least two success cases, namely, the AUTOTUTOR and

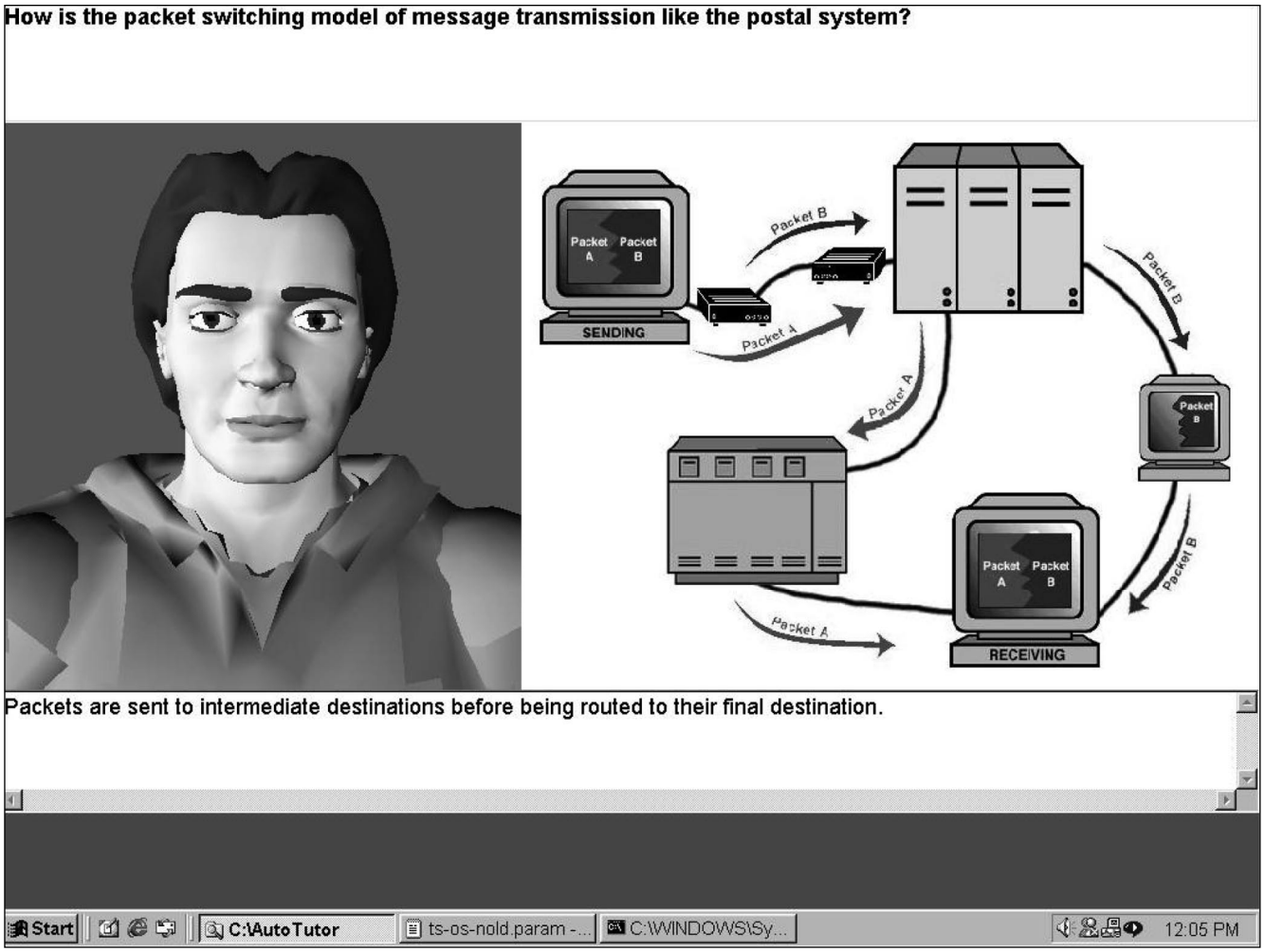


Figure 1. A Screen Shot of AUTOTUTOR.

ATLAS systems that we discuss in this article. AUTOTUTOR (Graesser et al. 1999) is a fully automated computer tutor that has tutored approximately 200 college students in an introductory course in computer literacy. An early version of AUTOTUTOR improved learning by .5 standard deviation units (that is, about a half a letter grade) when compared to a control condition where students reread yoked chapters in the book. ATLAS (VanLehn et al. 2000) is a computer tutor for college physics that focuses on improving students' conceptual knowledge. In a recent pilot evaluation, students who used ATLAS scored .9 standard deviation units higher than students who used a similar tutoring system that did not use natural language dialogues. Thus, it appears that there is something about conversational dialogue that plays an important role in learning. We believe that the most effective tutoring systems of the future will be a hybrid between normal conversational patterns and the ideal pedagogical strategies in the ITS enterprise.

This article describes some of the tutoring systems that we are developing to simulate conversational dialogue. We begin with AUTOTUTOR. Then we describe a series of physics tutors that vary from conventional ITS systems (the ANDES tutor) to agents that attempt to comprehend natural language and plan dialogue moves (ATLAS and WHY2).

AUTOTUTOR

The Tutoring Research Group (TRG) at the University of Memphis developed AUTOTUTOR to simulate the dialogue patterns of typical human tutors (Graesser et al. 1999; Person et al. 2001). AUTOTUTOR tries to comprehend student contributions and simulate dialogue moves of either normal (unskilled) tutors or sophisticated tutors. AUTOTUTOR is currently being developed for college students who are taking an introductory course in computer literacy. These students learn the fundamentals

of computer hardware, the operating system, and the internet.

Figure 1 is a screen shot that illustrates the interface of AUTOTUTOR. The left window has a talking head that acts as a dialogue partner with the learner. The talking head delivers AUTOTUTOR's dialogue moves with synthesized speech, intonation, facial expressions, nods, and gestures. The major question (or problem) that the learner is working on is both spoken by AUTOTUTOR and is printed at the top of the screen. The major questions are generated systematically from a curriculum script, a module that we discuss later. AUTOTUTOR's major questions are not the fill-in-the-blank, true-false, or multiple-choice questions that are so popular in the U.S. educational system. Instead, the questions invite lengthy explanations and deep reasoning (such as why, how, and what-if questions). The goal is to encourage students to articulate lengthier answers that exhibit deep reasoning rather than deliver short snippets of shallow knowledge. There is a continuous multiturn tutorial dialogue between AUTOTUTOR and the learner during the course of answering a deep-reasoning question. When considering both the learner and AUTOTUTOR, it typically takes 10 to 30 turns during the tutorial dialogue to answer a single question from the curriculum script. The learner types in his/her contributions during the exchange by keyboard, as reflected in the bottom window. For some topics, as in figure 1, there are graphic displays and animation, with components that AUTOTUTOR points to. AUTOTUTOR was designed to be a good conversational partner that comprehends, speaks, points, and displays emotions, all in a coordinated fashion.

An Example AUTOTUTOR-Learner Dialogue

Figure 2 shows a dialogue between a college student and AUTOTUTOR. Prior to this question, the student had been asked and attempted to answer 6 previous questions about the internet. The internet was the macrotopic, and students were tutored by answering several deep-reasoning questions about the internet. It should be noted that this is not a fabricated toy conversation. It is a bona fide dialogue from our corpus of approximately 200 AUTOTUTOR-student dialogues in a computer literacy course.

AUTOTUTOR begins this exchange by asking a how question in turn 1: What hardware do you need to take photos and send them over the internet? However, AUTOTUTOR doesn't merely pop the question out of the blue. It first presents a discourse marker that signals a change in topic (All right, let's go on.), presents a context to frame the question (You want to take photos and send them over the internet.), and then presents a discourse marker that signals the questions (Consider this problem.). Therefore, AUTOTUTOR monitors different levels of discourse structure and functions of dialogue moves. AUTOTUTOR inserts appropriate discourse markers that clarify these levels and functions to the learner. Without these discourse markers, learners are confused about what AUTOTUTOR is doing and what they are supposed to do next. A dialogue advancer network (DAN) has been designed to manage the conversational dialogue (Person et al. 2001). The DAN is a finite-

TUTOR-1: Alright, let's go on. You want to take photos and send them over the Internet. Consider this problem. What hardware do you need to take photos and send them over the Internet?
STUDENT-1: First of all you need to have access to the internet.
TUTOR-2: What else?
STUDENT-2: a camera
TUTOR-3: Yeah. For what type of connection do you need a network card?
STUDENT-3: network card
TUTOR-4: Okay. Now, if you have access to the Internet through a network card, then your connection is…
STUDENT-4: I don't know.
TUTOR-5: Is direct. Okay. A network card is needed if you have a direct connection to the Internet. Okay. How does the user get hooked up to the internet?
STUDENT-5: by a plug.
TUTOR-6: Once again, a network card is needed if you have a direct connection to the Internet. Okay, what does a modem do?
STUDENT-6: connects network card and internet
TUTOR-7: Okay. Try this. A modem is required when there is…
STUDENT-7: a connection
TUTOR-8: a dial up connection.
.
.
.
TUTOR-30: Let's review. To send your photos on the Internet, you need either a digital camera or a regular camera to take the photos. If you use a regular camera, you need a scanner to scan them onto a computer disk. If you have a direct connection to the Internet, then you need a network card. A modem is needed if you have a dial up connection.

Figure 2. An Example Tutorial Dialogue between AUTOTUTOR and a Learner.

state automaton that can handle different classes of information that learners type in. The DAN is augmented by production rules that are sensitive to learner ability and several parameters of the dialogue history.

How Does AutoTutor Handle the Student's Initial Answer to the Question?

After AutoTutor asks the question in the tutor-1 turn, the student gives an initial answer in the student-1 turn. The answer is very incomplete. A complete answer would include all the points in the summary at the final turn (tutor-30). What does AutoTutor do with this incomplete student contribution? AutoTutor doesn't simply grade the answer (for example, good, bad, incomplete, a quantitative score) as many conventional tutoring systems do. AutoTutor also stimulates a multiturn conversation that is designed to extract more information from the student and get the student to articulate pieces of the answer. Thus, instead of being an information-delivery system that bombards the student with a large volume of information, AutoTutor is a discourse prosthesis that attempts to get the student to do the talking and explores what the student knows. AutoTutor adopts the educational philosophy that students learn by actively constructing explanations and elaborations of the material (Chi et al. 1994; Conati and VanLehn 1999).

AutoTutor is a discourse prosthesis that attempts to get the student to do the talking and explores what the student knows.

How Does AutoTutor Get the Learner to Do the Talking?

AutoTutor has a number of dialogue moves to get the learner to do the talking. For starters, there are open-ended pumps that encourage the student to say more, such as What else? in the tutor-2 turn. *Pumps* are frequent dialogue moves after the student gives an initial answer, just as is the case with human tutors. The tutor pumps the learner for what the learner knows before drilling down to specific pieces of an answer. After the student is pumped for information, AutoTutor selects a piece of information to focus on. Both human tutors and AutoTutor have a set of expectations about what should be included in the answer. What they do is manage the multiturn dialogue to cover these expected answers. A complete answer to the example question in figure 2 would have four expectations, as listed here:

Expectation 1: You need a digital camera or regular camera to take the photos.

Expectation 2: If you use a regular camera, you need to scan the pictures onto the computer disk with a scanner.

Expectation 3: A network card is needed if you have a direct connection to the internet.

Expectation 4: A modem is needed if you have a dial-up connection.

AutoTutor decides which expectation to handle next and then selects dialogue moves that flesh out the expectation. The dialogue moves vary in directness and information content. The most indirect dialogue moves are hints, the most direct are assertions, and prompts are in between. Hints are often articulated in the form of questions, designed to lead the learner to construct the expected information. Assertions directly articulate the expected information. Prompts try to get the learner to produce a single word in the expectation. For example, the tutor turns 3, 4, 5, and 6 in figure 2 are all trying to get the learner to articulate expectation 3. Hints are in the tutor-3 turn (For what type of connection do you need a network card?) and the tutor-5 turn (How does the user get hooked up to the internet?). Prompts are in tutor-4 (If you have access to the internet through a network card, then your connection is…, with a hand gesture encouraging the learner to type in information). Assertions are in tutor-5 and tutor-6 (A network card is needed if you have a direct connection to the internet.). AutoTutor attempts to get the learner to articulate any given expectation *E* by going through two cycles of hint-prompt-assertion. Most students manage to articulate the expectation within the six dialogue moves (hint-prompt-assertion-hint-prompt-assertion). AutoTutor exits the six-move cycle as soon as the student has articulated the expected answer. Interestingly, sometimes students are unable to articulate an expectation even after AutoTutor spoke it in the previous turn. After expectation *E* is fleshed out, AutoTutor selects another expectation.

How Does AutoTutor Know Whether a Student Has Covered an Expectation?

AutoTutor does a surprisingly good job evaluating the quality of the answers that learners type in. AutoTutor attempts to "comprehend" the student input by segmenting the contributions into speech acts and matching the student's speech acts to the expectations. Latent semantic analysis (LSA) is used to compute these matches (Landauer, Foltz, and Laham 1998). When the tutor's expectation E is compared with the learner's speech act A, a cosine match score is computed that varies from 0 (no match) to 1.0 (perfect match). AutoTutor considers each combination of speech acts that the learner makes during the evolution of an

answer to a major question; the value of the highest cosine match is used when computing whether the student covers expectation E. LSA is a statistical, corpus-based method of representing knowledge. LSA provides the foundation for grading essays, even essays that are not well formed grammatically, semantically, and rhetorically. LSA-based essay graders can assign grades to essays as reliably as experts in composition (Landauer et al. 1998). Our research has revealed that AUTOTUTOR is almost as good as an expert in computer literacy in evaluating the quality of student answers in the tutorial dialogue (Graesser et al. 2000).

How Does AUTOTUTOR Select the Next Expectation to Cover?

AUTOTUTOR uses LSA in conjunction with various criteria when deciding which expectation to cover next. After each student turn, AUTOTUTOR updates the LSA score for each of the four expectations listed earlier. An expectation is considered covered if it meets or exceeds some threshold value (for example, .70 in our current tutor). One selection criterion uses the zone of proximal development to select the next expectation, which is the highest LSA score that is below threshold. A second criterion uses coherence, the expectation that has the highest LSA overlap with the previous expectation that was covered. Other criteria that are currently being implemented are preconditions and pivotal expectations. Ideally, AUTOTUTOR will decide to cover a new expectation in a fashion that both blends into the conversation and that advances the agenda in an optimal way. AUTOTUTOR generates a summary after all the expectations are covered (for example, the tutor-30 turn).

How Does AUTOTUTOR Give Feedback to the Student?

There are three levels of feedback. First, there is *backchannel feedback* that acknowledges the learner's input. AUTOTUTOR periodically nods and says uh-huh after learners type in important nouns but is not differentially sensitive to the correctness of the student's nouns. The backchannel feedback occurs online as the learner types in the words of the turn. Learners feel that they have an impact on AUTOTUTOR when they get feedback at this fine-grain level. Second, AUTOTUTOR gives *evaluative pedagogical feedback* on the learner's previous turn based on the LSA values of the learner's speech acts. The facial expressions and intonation convey different levels of feedback, such as negative (for example, not really while head shakes), neutral negative (okay with a skeptical look), neutral positive (okay at a moderate nod rate), and positive (right with a fast head nod). Third, there is *corrective feedback* that repairs bugs and misconceptions that learners articulate. Of course, these bugs and their corrections need to be anticipated ahead of time in AUTOTUTOR's curriculum script. This anticipation of content mimics human tutors. Most human tutors anticipate that learners will have a variety of particular bugs and misconceptions when they cover particular topics. An expert tutor often has canned routines for handling the particular errors that students make. AUTOTUTOR currently splices in correct information after these errors occur, as in turn tutor-8. Sometimes student errors are ignored, as in tutor-4 and tutor-7. These errors are ignored because AUTOTUTOR has not anticipated them by virtue of the content in the curriculum script. AUTOTUTOR evaluates student input by matching it to what it knows in the curriculum script, not by constructing a novel interpretation from whole cloth.

How Does AUTOTUTOR Handle Mixed-Initiative Dialogue?

We know from research on human tutoring that it is the tutor who controls the lion's share of the tutoring agenda (Graesser, Person, and Magliano 1995). Students rarely ask information-seeking questions and introduce new topics. However, when learners do take the initiative, AUTOTUTOR needs to be ready to handle these contributions. AUTOTUTOR does a moderately good job in managing mixed-initiative dialogue. AUTOTUTOR classifies the learner's speech acts into the following categories:

Assertion (RAM is a type of primary memory.)

WH-question (What does bus mean and other questions that begin with who, what, when, where, why, how, and so on.)

YES-NO question (Is the floppy disk working?)

Metacognitive comment (I don't understand.)

Metacommunicative act (Could you repeat that?)

Short response (okay, yes)

Obviously, AUTOTUTOR's dialogue moves on turn $N + 1$ need to be sensitive to the speech acts expressed by the learner in turn N. When the student asks a What does X mean? question, the tutor answers the question by giving a definition from a glossary. When the learner makes an assertion, the tutor evaluates the quality of the assertion and gives short evaluative feedback. When the learner asks, What did

Our research has revealed that AUTOTUTOR is almost as good as an expert in computer literacy in evaluating the quality of student answers in the tutorial dialogue ….

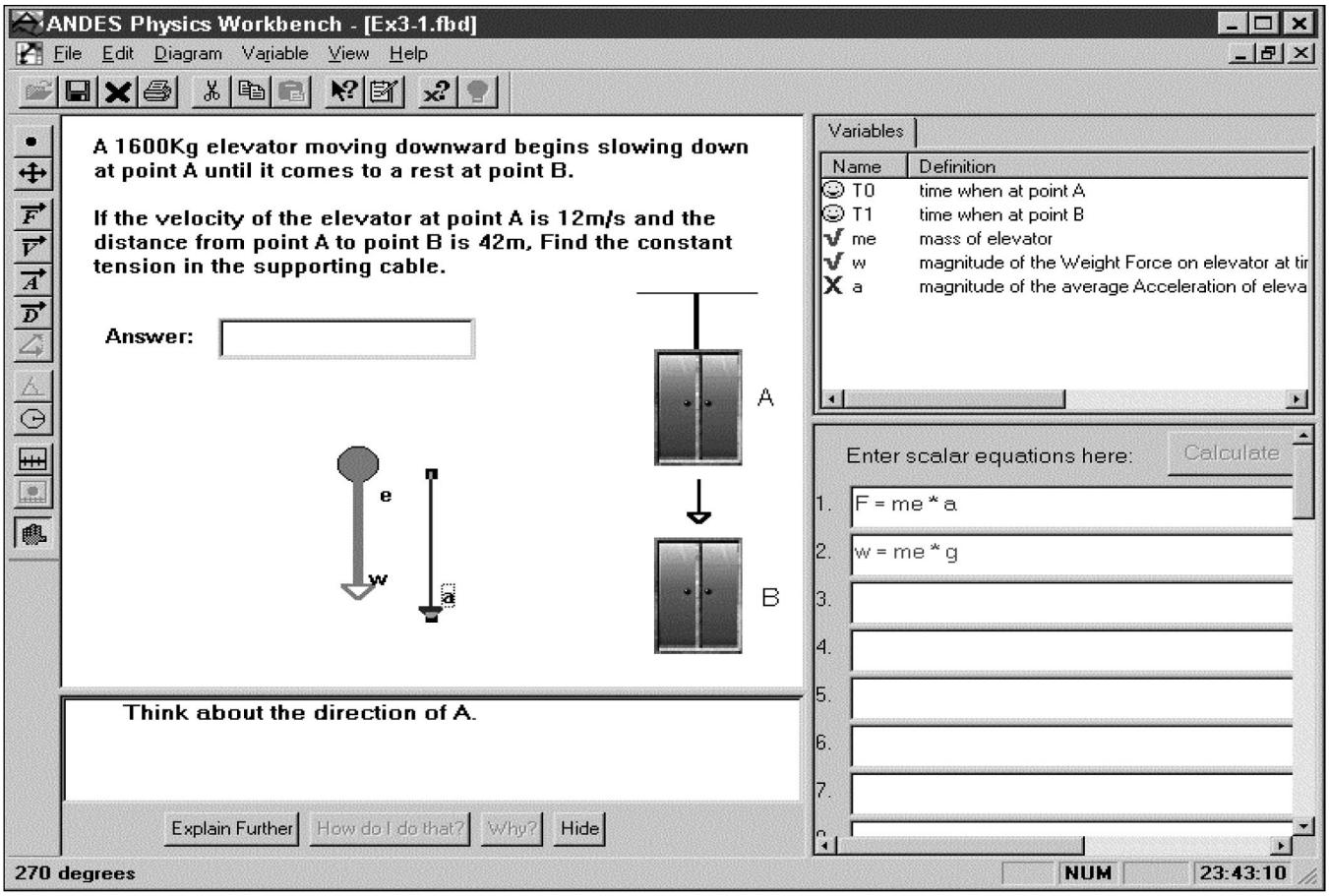


Figure 3. The ANDES Tutoring System.

you say? AUTOTUTOR repeats what it said in the last turn. The DAN manages the mixed-initiative dialogue.

The Curriculum Script

AUTOTUTOR has a curriculum script that organizes the content of the topics covered in the tutorial dialogue. There are 36 topics, one for each major question or problem that requires deep reasoning. Associated with each topic are a set of expectations, a set of hints and prompts for each expectation, a set of anticipated bugs-misconceptions and their corrections, and (optionally) pictures or animations. It is very easy for a lesson planner to create the content for these topics because they are English descriptions rather than structured code. Of course, pictures and animations would require appropriate media files. We are currently developing an authoring tool that makes it easy to create the curriculum scripts. Our ultimate goal is to make it very easy to create an AUTOTUTOR for a new knowledge domain. First, the developer creates an LSA space after identifying a corpus of electronic documents on the domain knowledge. The lesson planner creates a curriculum script with deep-reasoning questions and problems. The developer then computes LSA vectors on the content of the curriculum scripts. A glossary of important terms and their definitions is also prepared. After that, the built-in modules of AUTOTUTOR do all the rest. AUTOTUTOR is currently implemented in JAVA for PENTIUM computers, so there are no barriers to widespread use.

ANDES: A Physics Tutoring System That Does Not Use Natural Language

The goal of the second project is to use natural language–processing technology to improve an already successful intelligent tutoring system named ANDES (Gertner and VanLehn 2000; VanLehn 1996). ANDES is intended to be used as an adjunct to college and high-school physics courses to help students do their homework problems.

Figure 3 shows the ANDES screen. A physics problem is presented in the upper-left window. Students draw vectors below it, define variables

in the upper-right window, and enter equations in the lower-right window. When students enter a vector, variable, or equation, ANDES will color the entry green if it is correct and red if it is incorrect. This approach is called *immediate feedback* and is known to enhance learning from problem solving (Anderson et al. 1995).

To give immediate feedback, ANDES must understand the student's entries no matter how the student tries to solve the problem. ANDES uses a rule-based expert system to solve the problem in all correct ways. It gives negative feedback if the student's entry does not match one of the steps of one of the solutions from the expert model. For this reason, ANDES and similar tutoring systems are known as *model-tracing tutors*. They follow the student's reasoning by comparing it to a trace of the model's reasoning.

How Does ANDES Hint and Give Help?

Students can ask ANDES for help by either clicking on the menu item What do I do next? or by selecting a red entry and clicking on the menu item What's wrong with that? ANDES uses a Bayesian network to help it determine which step in the expert's solution to give the student help on (Gertner, Conati, and VanLehn 1998). It prints in the lower-left window a short message, such as the one shown in figure 3. The message is only a hint about what is wrong or what to do next. Often a mere hint suffices, and the students are able to correct their difficulty and move on. However, if the hint fails, then the student can ask for help again. ANDES generates a second hint that is more specific than the first. If the student continues to ask for help, ANDES's last hint will essentially tell the student what to do next. This technique of giving help is based on human-authored hint sequences. Each hint is represented as a template. It is filled in with text that is specific to the situation where help was requested. Such hint sequences are often used in intelligent tutoring systems and are known to enhance learning from problem solving (McKendree 1990).

During evaluations in the fall of 2000 at the U.S. Naval Academy, students using ANDES scored about a letter grade (0.92 standard deviation units) higher on the midterm exam than students in a control group (Shelby et al. 2002). Log file data indicate that students are using the help and hint facilities as expected. Questionaire data indicate that many of them prefer doing their homework on ANDES to doing it with paper and pencil.

Other intelligent tutoring systems use similar model tracing, immediate feedback, and hint sequences techniques, and many have been shown to be effective (for example, Anderson et al. [1995]; McKendree, Radlinski, and Atwood [1992]; Reiser et al. [2002]). A new company, Carnegie Learning,[1] is producing such tutors for use in high-school mathematics classes. As of fall 2000, approximately 10 percent of the algebra I classes in the United States will be using one of the Carnegie Learning tutors. Clearly, this AI technology is rapidly maturing.

Criticisms of ANDES and Other Similar Tutoring Systems

The pedagogy of immediate feedback and hint sequences has sometimes been criticized for failing to encourage deep learning. The following four criticisms are occasionally raised by colleagues:

First, if students don't reflect on the tutor's hints but merely keep guessing until they find an action that gets positive feedback, they can learn to do the right thing for the wrong reasons, and the tutor will never detect the shallow learning (Aleven, Koedinger, and Cross 1999).

Second, the tutor does not ask students to explain their actions, so students might not learn the domain's language. Educators have recently advocated that students learn to "talk science." Talking science is allegedly part of a deep understanding of the science. It also facilitates writing scientifically, working collaboratively in groups, and participating in the culture of science.

Talking science is allegedly part of a deep understanding of the science.

Third, to understand the students' thinking, the user interface of such systems requires students to display many of the details of their reasoning. This design doesn't promote stepping back to see the "basic approach" one has used to solve a problem. Even students who have received high grades in a physics course can seldom describe their basic approaches to solving a problem (Chi, Feltovich, and Glaser 1981).

Fourth, when students learn quantitative skills, such as algebra or physics problem solving, they are usually not encouraged to see their work from a qualitative, semantic perspective. As a consequence, they fail to induce versions of the skills that can be used to solve qualitative problems and check quantitative ones for reasonableness. Even physics students with high grades often score poorly on tests of qualitative physics (Halloun and Hestenes 1985).

Many of these objections can be made to just about any form of instruction. Even expert tutors and teachers have difficulty getting students to learn deeply. Therefore, these criticisms of intelligent tutoring systems should only encourage us to improve them, not reject them.

There are two common themes in this list of four criticisms. First, all four involve integrat-

1. Tutor: As you know, you can think of an acceleration vector as changing a velocity vector by pulling on its tip. In this case, the elevator's velocity vector points straight down and you drew its acceleration vector pointing down too. What does your acceleration vector do to the velocity vector?
2. Student: It stretches it downward.
3. Tutor: Does that make it longer?
4. Student: Yes.
5. Tutor: Which means the elevator is speeding up, right?
6. Student: Yeah, but its not really speeding up. It's slowing down.
7. Tutor: Right. So maybe your acceleration vector's direction is wrong. What direction should it be?
8. Student: Up.
9. Tutor: Right. So what have you learned about the direction of acceleration when an object is moving in a straight line and slowing down?
10. Student: It's in the opposite direction from the velocity.
11. Tutor: Yes! That's exactly right. Good job.

Figure 4. A Hypothetical Dialogue between ATLAS and a Student.

ing problem-solving knowledge with other knowledge, namely (1) principles or rationales; (2) domain language; (3) abstract, basic approaches; and (4) qualitative rules of inference. Second, the kinds of instructional activity that are currently used to tap these other kinds of knowledge make critical use of natural language. Although one can invent graphic or formal notations to teach these kinds of knowledge on a computer, they might be more confusing to the students and instructors than the knowledge that they are trying to convey. Moreover, students and instructors are likely to resist learning a new formalism, even a graphic one, if they will only use it temporarily.

ATLAS: A Natural Language Enhancement for Model-Tracing Tutors

We believe that tutoring systems must use natural language if they are to become more effective at encouraging deep learning. Therefore, we have begun building ATLAS, a module that can be added to ANDES or other similar model-tracing tutoring systems to conduct natural language dialogues and thereby promote deep learning. ATLAS uses natural language-processing technology that was originally developed for CIRCSIM tutor (Freedman and Evens 1996), the BASIC ELECTRICITY AND ELECTRONICS (BEE) tutor (Rosé, Di Eugenio, and Moore 1999), and the COCONUT model of collaborative dialogue (Di Eugenio et al. 2000). A number of natural language–understanding authoring tools have been developed, including the LC-FLEX parser (Rosé and Lavie 2001).

Currently, ATLAS plays only a small role in the student's total problem-solving process. Most of the time, the students interact with ANDES just as they ordinarily do. However, if ATLAS notices an opportunity to promote deep learning, it takes control of the interaction and begins a natural language dialogue. Although ATLAS can ask students to make ANDES actions as part of the dialogue (for example, it might have the student draw a single vector), most of the dialogue is conducted in a scrolling text window, which replaces the hint window shown in the lower left of figure 3. When ATLAS has finished leading the student through a line of reasoning, it signs off and lets the student return to solving the problem with ANDES.

The ATLAS dialogues are called *knowledge construction dialogues* (KCDs) because they are designed to encourage students to infer or construct the target knowledge. For example, ANDES might simply tell the student that when an object moving in a straight line is slowing down, its acceleration is in the opposite direction to it velocity. ATLAS will instead try to draw the knowledge out of the student with a dialogue such as the one shown in figure 4, where the student derived the target principle from a deeper one. KCDs are intended to provide deeper knowledge by connecting principles, relating them to commonsense knowledge, and giving the student practice in talking about them.

Knowledge Construction Dialogues to Teach Principles

To date, ATLAS conducts just one kind of KCD, namely, those that teach domain principles. Currently, we are concentrating on only a small portion of physics, so only 55 principles are covered. Even so, building so many KCDs was daunting enough that we built tools to help us. With these tools, we were able to build knowledge sources for our KCDs in only three person-months.

The primary design concept is to represent KCDs as recursive finite-state networks. States correspond to tutor utterances (usually questions), and arcs correspond to student responses. A few arcs are special in that they either call a subdialogue or return from one. Such recursive finite- state networks are often used in spo-

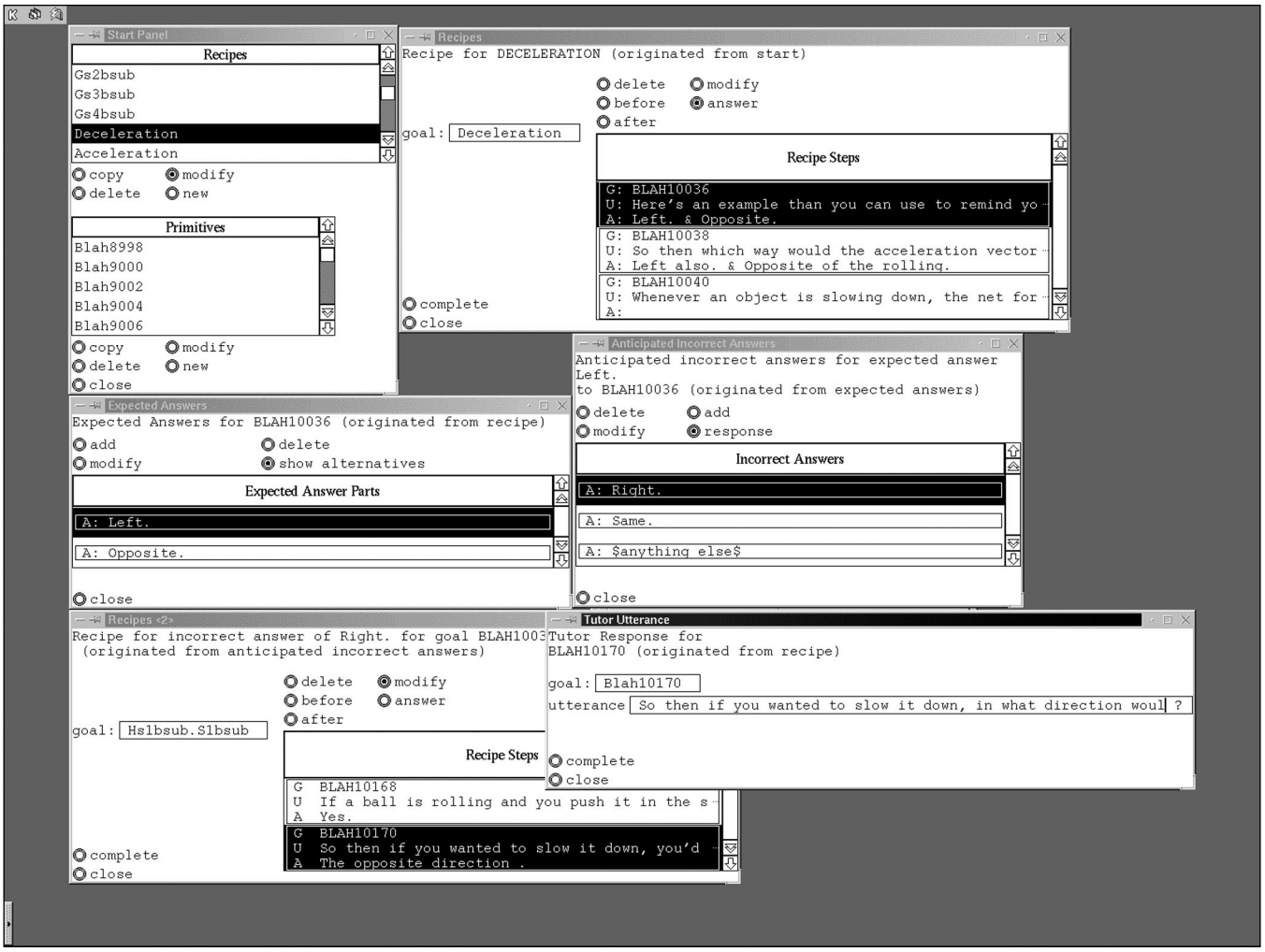


Figure 5. The Knowledge Construction Dialogue Editor.

ken language dialogue systems (Jurafsky and Martin 2000), so it makes sense to start with them and see where they break down.

Our primary tool is the *KCD editor* (figure 5). In the upper-left window, the author selects a topic, which is deceleration in this case. This selection causes a shorthand form of the recipes (discourse plans) to appear in the upper-right window. Selecting a tutor-student interaction brings up windows for seeing the tutor's contribution (as with the lower-right window) and the student's expected answers (middle windows). The left-middle window is for correct answers. As in AUTOTUTOR, the student's expected answer is represented as a set of expectations (left and opposite in this case). The right-middle window is for incorrect answers. When one of these is selected, a subdialogue for handling it is displayed in the lower-left window. Notice that the author enters natural language text for the tutor contribution, the expectations, and almost everything else.

In a limited sense, the KCDs are intended to be better than naturally occurring dialogues. Just as most text expresses its ideas more clearly than informal oral expositions, the KCD is intended to express its ideas more clearly than the oral tutorial dialogues that human tutors generate. Thus, we need a way for expert physicists, tutors, and educators to critique the KCDs and suggest improvements. Because the underlying finite-state network can be complex, it is not useful to merely print it out and let experts pencil in comments. The second tool facilitates critiquing KCDs by allowing expert physicists and psychologists to navigate around the network and enter comments on individual states and arcs (figure 6). It presents a dialogue in the left column and allows the user to enter comments in the right column. Because there are many expected responses for each tutorial contribution, the user can select a response from a pull-down menu, causing the whole dialogue to adjust, opening up new boxes for the user's

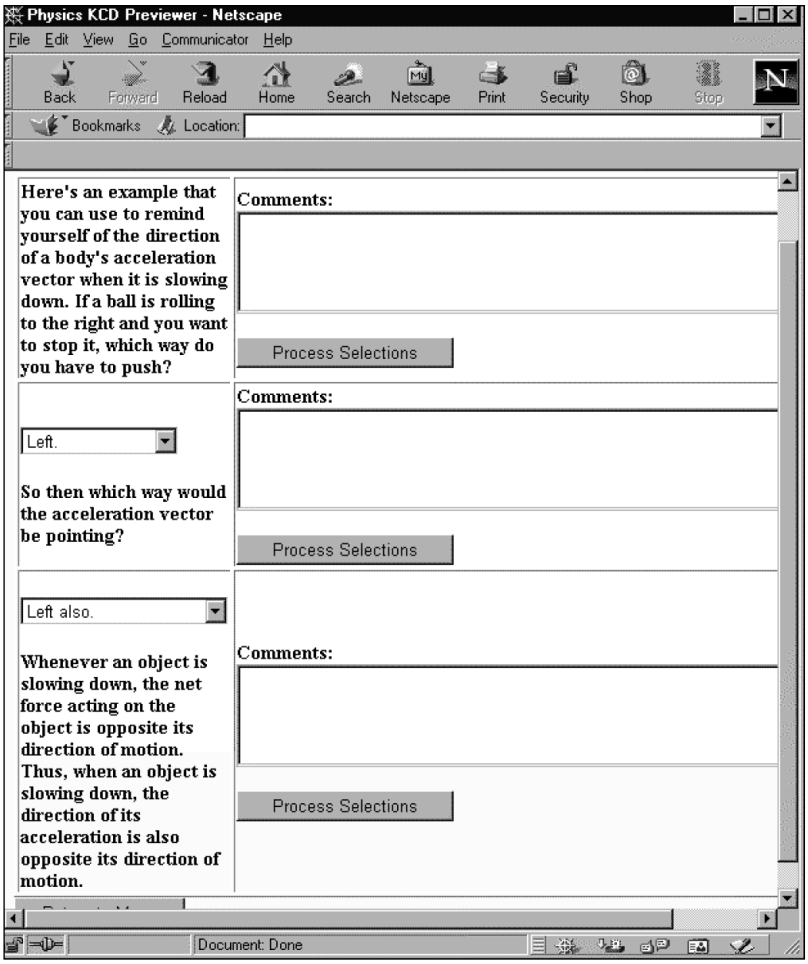


Figure 6. The Knowledge Construction Dialogue Commenter.

comments. This tool runs in a web browser, so experts can use it remotely.

At any point in its development, a KCD can be compiled into executable code. The code is interpreted by a reactive planning engine (Freedman 1999). The engine does not simply follow the finite-state network. Instead, it has rudimentary (but growing) capabilities for treating the network as a plan for the conversation that it will adapt as necessary. For example, in the conversation in figure 4, suppose the student said at line 2, "The acceleration makes the velocity vector longer, so the elevator should be going faster." The reactive planner should recognize that the student has skipped ahead in the conversation plan, so it should have ATLAS say line 7 instead of line 3.

Although the authors only see and edit natural language text, we cannot expect students to type in exactly the responses that the authors enter. The compiler uses CARMEL (Rosé 2000) to translate the expected student responses into semantic structures. Thus, it should recognize the expected responses even if they are not expressed with the same words and syntax as the author-provided versions.

Current Status and Future Directions

The initial version of ATLAS was pilot tested in the fall of 2000. Five students used ATLAS, and five students used ANDES without its hint sequences instead of the ATLAS KCDs. Despite the small number of subjects, the ATLAS students scored significantly higher than the ANDES students on a conceptual posttest. Surprisingly, the effect was large: The ATLAS students gained about .9 standard deviation units more than the ANDES students. Moreover, they scored about the same as the ANDES students on a quantitative posttest, suggesting that the improvements were limited to the material taught by ATLAS, as expected.

Five issues dominate ATLAS's future development. First, writing an effective KCD and debugging it with real students is an inherently labor-intensive process. We will continue building tools to expedite the process. Second, the conventional wisdom is that a recursive finite-state network does not provide sufficient flexibility for managing complex dialogues. Although a reactive planner interprets ATLAS's networks, we do not currently make use of all its power. Thus, a second important direction is to determine how much of this additional power is necessary for conducting more effective tutorial dialogues with students. Third, the current version of ATLAS does not make use of the full power offered by the CARMEL core-understanding component. Thus, another related direction is determining how sophisticated an analysis of student input is necessary for the system to determine how best to proceed with its dialogue with the student. Fourth, we have deliberately left the ANDES system's two major knowledge sources alone, so ANDES is still responsible for solving physics problems and deciding which hint sequence is appropriate. Thus, KCDs are used mostly to replace hint sequences. We are not sure if this simple design will allow pedagogically useful dialogues or whether we will need to port some of ANDES's knowledge to ATLAS. Fifth, we plan to extend ATLAS's capabilities to additional types of knowledge-construction dialogues, such as goal-scaffolding dialogues.

WHY2: Tutoring Qualitative Explanations

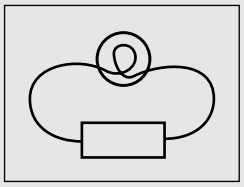
All tutoring systems have students perform a task, and they help the students do it. Some tutoring systems, such as ANDES and ATLAS, have the student solve problems. Other tutoring systems, such as AUTOTUTOR, ask the student deep questions and help them formulate a correct, complete answer. Recent work with human tutors (for example, Chi et al. [2001]) suggests that a good activity for teaching is to have students explain physical systems qualitatively. Although it is possible to have them express their explanations in formal or graphic languages (for example, CYCLEPAD [Forbus et al. 1998]), we believe that they will learn more if they can express their explanations in natural language. Thus, the goal of the WHY2 project is to coach students as they explain physical systems in natural language.

WHY2 is intended to be a successor of one of the first intelligent tutoring systems in the literature, the WHY system. WHY was envisioned and partially implemented by Albert Stevens and Alan Collins (Stevens and Collins 1977). They studied experts helping students articulate such explanations and tried to embed their tutorial strategies in the WHY system. Stevens and Collins discovered that students had a great many misconceptions about nature. These misconceptions would only surface when students expressed their ideas qualitatively because they could solve textbook quantitative problems correctly (Halloun and Hestenes 1985). Since this time, considerable effort has been expended by physics educators to discover, catalog, and invent remedies for student misconceptions. The remedies are usually intended for classroom or laboratories and have had only moderate success (Hake 1998). By adapting them to the tutorial setting and embedding the tutorial strategies uncovered by Collins, Stevens, and others, WHY2 might be much more successful.

The basic idea of WHY2 is to ask the student to type in an explanation for a simple physical situation, such as the battery-bulb circuit shown in figure 7. WHY2 analyzes the student's explanation (line 1 in Figure 7) to see if the student has any misconceptions. If it detects a misconception, it invokes a knowledge construction dialogue (KCD), such as the one shown in lines 2 through 9. During this dialogue, further misunderstandings may arise, which can cause another KCD to be selected and applied (see lines 10 onward).

WHY2 is a joint project involving both the AUTOTUTOR and ATLAS groups. It began recently and is still in the design stages. A corpus of explanations from students has been collected and is being analyzed to see what kinds of misconceptions and language the students are using. Our plan is to use a combination of the LSA technology from AUTOTUTOR and the semantic composition technology from ATLAS.

Question: A battery is connected to a bulb by two wires. The bulb lights. Why?



1. Student: There are two currents which come out, one current positive and one negative. They combine in the bulb to produce light and heat.
2. Tutor: I see. Which current is composed of moving electrons?
3. Student: I'm not sure. Both?
4. Tutor: Right, both are. An electrical current is just moving electrons. Which direction do the electrons move in your two currents?
5. Student: I think the electrons move out from the battery to the bulb. The two currents meet there and produce energy.
6. Tutor: Are the electrons actually destroyed in the bulb?
7. Student: I'm not sure.
8. Tutor: Actually, electrons are not destroyed or lost in an electrical current. Instead, they go out one side of the battery, go through the bulb and go back in the other side of the battery. Thus, your two currents are actually the same flow, only the electrons are leaving the battery for one and returning to the battery for the other. Given everything you know about batteries and bulbs, does this make sense?
9. Student: But I thought the battery dies when it runs out of electrons.
10. Tutor: Batteries die when they run out of the chemical energy that they use to pump electrons around the circuit. The battery is like a water pump that produces a powerful current, and the bulb is like a paddle wheel that is driven by the water current. When the water pump runs out of energy, the current slows to a stop. As this happens, does the number of electrons in the battery change?
11. Student: Maybe a little. But it stays about the same.
12. Tutor: Exactly right! I think you understand electrical current better now. See if you can write a better explanation.

Figure 7. A Hypothetical Dialogue between a Student and WHY2.

The KCDs of ATLAS will be generalized to incorporate elements of the DANs of AUTOTUTOR.

Our dialogue technology can be stressed by the complexity of the language and discourse we anticipate from the students. However, if we can make it work, the pedagogical payoffs will be enormous. Repairing the qualitative misconceptions of physics is a difficult and fundamentally important problem.

Conclusions

We discussed three projects that have several similarities. AUTOTUTOR, ATLAS, and WHY2 all endorse the idea that students learn best if they construct knowledge themselves. Thus, their dialogues try to elicit knowledge from the student by asking leading questions. They only tell the student the knowledge as a last resort. All three projects manage dialogues by using finite-state networks. Because we anticipate building hundreds of such networks, the projects are building tools to let domain authors enter these dialogues in natural language. All three projects use robust natural language–understanding techniques—LSA for AUTOTUTOR, CARMEL for ATLAS, and a combination of the two for WHY2. All three projects began by analyzing data from human tutors and are using evaluations with human students throughout their design cycle.

Although the three tutoring systems have the common objective of helping students perform activities, the specific tasks and knowledge domains are rather different. AUTOTUTOR's students are answering deep questions about computer technology, ATLAS's students are solving quantitative problems, and WHY2's students are explaining physical systems qualitatively. We might ultimately discover that the conversational patterns need to be different for these different domains and tasks. That is, dialogue styles might need to be distinctively tailored to particular classes of knowledge domains. A generic dialogue style might prove to be unsatisfactory. Whatever discoveries emerge, we suspect they will support one basic claim: Conversational dialogue substantially improves learning.

Acknowledgments

The AUTOTUTOR research was supported by grants from the National Science Foundation (SBR 9720314) and the Office of Naval Research (N00014-00-1-0600). The ANDES research was supported by grant N00014-96-1-0260 from the Cognitive Sciences Division of the Office of Naval Research. The ATLAS research is supported by grant 9720359 from the LIS program of the National Science Foundation. The WHY2 research is supported by grant N00014-00-1-0600 from the Cognitive Sciences Division of the Office of Naval Research.

Note

1. www.carnegielearning.com.

References

Aleven, V.; Koedinger, K. R.; and Cross, K. 1999. Tutoring Answer Explanation Fosters Learning with Understanding. In *Artificial Intelligence in Education,* eds. S. P. Lajoie and M. Vivet, 199–206. Amsterdam: IOS.

Anderson, J. R.; Corbett, A. T.; Koedinger, K. R.; and Pelletier, R. 1995. Cognitive Tutors: Lessons Learned. *The Journal of the Learning Sciences* 4(2): 167–207.

Cassell, J., and Thorisson, K. R. 1999. The Power of a Nod and a Glance: Envelope versus Emotional Feedback in Animated Conversational Agents. *Applied Artificial Intelligence* 13(3): 519–538.

Chi, M. T. H.; Feltovich, P.; and Glaser, R. 1981. Categorization and Representation of Physics Problems by Experts and Novices. *Cognitive Science* 5(2): 121–152.

Chi, M. T. H.; de Leeuw, N.; Chiu, M.; and LaVancher, C. 1994. Eliciting Self-Explanations Improves Understanding. *Cognitive Science* 18(3): 439–477.

Chi, M. T. H.; Siler, S.; Jeong, H.; Yamauchi, T.; and Hausmann, R. G. 2001. Learning from Tutoring: A Student-Centered versus a Tutor-Centered Approach. *Cognitive Science*. Forthcoming.

Cohen, P. A.; Kulik, J. A.; and Kulik, C. C. 1982. Educational Outcomes of Tutoring: A Meta-Analysis of Findings. *American Educational Research Journal* 19(2): 237–248.

Conati, C., and VanLehn, K. 1999. Teaching Metacognitive Skills: Implementation and Evaluation of a Tutoring System to Guide Self-Explanation While Learning from Examples. In *Artificial Intelligence in Education,* eds. S. P. Lajoie and M. Vivet, 297–304. Amsterdam: IOS.

Corbett, A.; Anderson, J.; Graesser, A.; Koedinger, K.; and VanLehn, K. 1999. Third Generation Computer Tutors: Learn from or Ignore Human Tutors? In Proceedings of the 1999 Conference of Computer-Human Interaction, 85–86. New York: Association of Computing Machinery.

Di Eugenio, B.; Jordan, P. W.; Thomason, R. H.; and Moore, J. D. 2000. The Agreement Process: An Empirical Investigation of Human-Human Computer-Mediated Dialogues. *International Journal of Human-Computer Studies* 53(6): 1017–1076.

Forbus, K. D.; Everett, J. O.; Ureel, L.; Brokowski, M.; Baher, J.; and Kuehne, S. E. 1998. Distributed Coaching for an Intelligent Learning Environment. Paper presented at the AAAWorkshop on Quantitative Reasoning, 26–29 May, Cape Cod, Massachusetts.

Freedman, R. 1999. ATLAS: A Plan Manager for Mixed-Initiative, Multimodal Dialogue. Paper presented at the 1999 AAAI Workshop on Mixed-Initiative Intelligence, 19 July, Orlando, Florida.

Freedman, R., and Evens, M. W. 1996. Generating and Revising Hierarchical Multi-Turn Text Plans in an ITS. In *Intelligent Tutoring Systems: Proceedings of the 1996 Conference,* eds. C. Frasson, G. Gauthier, and A. Lesgold, 632–640. Berlin: Springer.

Gertner, A.; Conati, C.; and VanLehn, K. 1998. Procedural Help in ANDES: Generating Hints Using a Bayesian Network Student Model. In Proceedings of the Fifteenth National Conference on Artificial Intelligence, 106–111. Menlo Park, Calif.: American Association for Artificial Intelligence.

Gertner, A. S., and VanLehn, K. 2000. ANDES: A Coached Problem-Solving Environment for Physics. In *Intelligent Tutoring Systems: Fifth International Conference, ITS 2000,* eds. G. Gautheier, C. Frasson, and K. VanLehn, 133–142. New York: Springer.

Graesser, A. C.; Person, N. K.; and Magliano, J. P. 1995. Collaborative Dialogue Patterns in Naturalistic One-on-One Tutoring. *Applied Cognitive Psychology* 9(4): 495–522.

Graesser, A. C.; Wiemer-Hastings, K.; Wiemer-Hastings, P.; Kreuz, R.; and the Tutoring Research Group 1999. AUTOTUTOR: A Simulation of a Human Tutor. *Journal of Cognitive Systems Researc*h 1(1): 35–51.

Graesser, A. C.; Wiemer-Hastings, P.; Wiemer-Hastings, K.; Harter, D.; Person, N.; and the Tutoring Research Group. 2000. Using Latent Semantic Analysis to Evaluate the Contributions of Students in AUTOTUTOR. *Interactive Learning Environments* 8(2): 129–148.

Hake, R. R. 1998. Interactive-Engagement versus Traditional Methods: A Six-Thousand Student Survey of Mechanics Test Data for Introductory Physics Students.

American Journal of Physics 66(4): 64–74.

Halloun, I. A., and Hestenes, D. 1985. Common Sense Concepts about Motion. *American Journal of Physics* 53(11): 1056–1065.

Hestenes, D.; Wells, M.; and Swackhamer, G. 1992. Force Concept Inventory. *The Physics Teacher* 30(3): 141–158.

Johnson, W. L.; Rickel, J. W.; and Lester, J. C. 2000. Animated Pedagogical Agents: Face-to-Face Interaction in Interactive Learning Environments. *International Journal of Artificial Intelligence in Education* 11(1): 47–78.

Jurafsky, D., and Martin, J. E. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Upper Saddle River, N.J.: Prentice Hall.

Koedinger, K. R.; Anderson, J. R.; Hadley, W. H.; and Mark, M. A. 1997. Intelligent Tutoring Goes to School in the Big City. *Journal of Artificial Intelligence in Education* 8(1): 30–43.

Kulik, J. A., and Kulik, C. L. C. 1988. Timing of Feedback and Verbal Learning. *Review of Educational Research* 58(1): 79–97.

Landauer, T. K.; Foltz, P. W.; and Laham, D. 1998. An Introduction to Latent Semantic Analysis. *Discourse Processes* 25(2–3): 259–284.

Lesgold, A.; Lajoie, S.; Bunzo, M.; and Eggan, G. 1992. SHERLOCK: A Coached Practice Environment for an Electronics Troubleshooting Job. In *Computer-Assisted Instruction and Intelligent Tutoring Systems,* eds. J. H. Larkin and R. W. Chabay, 201–238. Hillsdale, N.J.: Lawrence Erlbaum.

Lester, J. C.; Voerman, J. L.; Townes, S. G.; and Callaway, C. B. 1999. Deictic Believability: Coordinating Gesture, Locomotion, and Speech in Life-Like Pedagogical Agents. *Applied Artificial Intelligence* 13(4–5): 383–414.

McKendree, J. 1990. Effective Feedback Content for Tutoring Complex Skills. *Human-Computer Interaction* 5:381–413.

McKendree, J.; Radlinski, B.; and Atwood, M. E. 1992. The GRACE Tutor: A Qualified Success. In *Intelligent Tutoring Systems: Second International Conference,* eds. C. Frasson, G. Gautheir, and G. I. McCalla, 677–684. Berlin: Springer-Verlag.

Person, N. K.; Graesser, A. C.; Kreuz, R. J.; Pomeroy, V.; and the Tutoring Research Group. 2001. Simulating Human Tutor Dialogue Moves in AUTOTUTOR. *International Journal of Artificial Intelligence in Education*. Forthcoming.

Reiser, B. J.; Copen, W. A.; Ranney, M.; Hamid, A.; and Kimberg, D. Y. 2002. Cognitive and Motivational Consequences of Tutoring and Discovery Learning. *Cognition and Instruction.* Forthcoming.

Rosé, C. P. 2000. A Framework for Robust Semantic Interpretation. In *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics,* 311–318. San Francisco, Calif.: Morgan Kaufmann.

Rosé, C. P., and Lavie, A. 2001. Balancing Robustness and Efficiency in Unification-Augmented Context-Free Parsers for Large Practical Applications. In *Robustness in Language and Speech Technology,* eds. J. C. Junqua and G. V. Noord, 239–269. Amsterdam: Kluwer Academic.

Rosé, C. P.; DiEugenio, B.; and Moore, J. 1999. A Dialogue-Based Tutoring System for Basic Electricity and Electronics. In *Artificial Intelligence in Education,* eds. S. P. Lajoie and M. Vivet, 759–761. Amsterdam: IOS.

Shelby, R. N.; Schulze, K. G.; Treacy, D. J.; Wintersgill, M. C.; VanLehn, K.; and Weinstein, A. 2001. The Assessment of ANDES Tutor. Forthcoming.

Stevens, A., and Collins, A. 1977. The Goal Structure of a Socratic Tutor. In Proceedings of the National ACM Conference, 256–263. New York: Association of Computing Machinery.

VanLehn, K. 1996. Conceptual and Metalearning during Coached Problem Solving. In *Proceedings of the Third Intelligent Tutoring Systems Conference,* eds. C. Frasson, G. Gauthier, and A. Lesgold, 29–47. Berlin: Springer-Verlag.

VanLehn, K.; Freedman, R.; Jordan, P.; Murray, C.; Osan, R.; Ringenberg, M.; Rosé, C. P.; Schulze, K.; Shelby, R.; Treacy, D.; Weinstein, A.; and Wintersgill, M. 2000. Fading and Deepening: The Next Steps for ANDES and Other Model-Tracing Tutors. In *Intelligent Tutoring Systems: Fifth International Conference, ITS 2000,* eds. G. Gauthier, C. Frasson, and K. VanLehn, 474–483. Berlin: Springer-Verlag.

Arthur Graesser is a professor of psychology and computer science at the University of Memphis, codirector of the Institute for Intelligent Systems, and director of the Center for Applied Psychological Research. He has conducted research on tutorial dialogue in intelligent tutoring systems and is current editor of the journal *Discourse Processes*. His e-mail address is a-graesser@memphis.edu.

Kurt VanLehn is a professor of computer science and intelligent systems at the University of Pittsburgh, director of the Center for Interdisciplinary Research on Constructive Learning Environments, and a senior scientist at the Learning Research and Development Center. His main interests are applications of AI to tutoring and assessment. He is a senior editor for the journal *Cognitive Science.* His e-mail address is vanlehn@cs.pitt.edu.

Carolyn Rosé is a research associate at the Learning Research and Development Center at the University of Pittsburgh. Her main research focus is on developing robust language understanding technology and authoring tools to facilitate the rapid development of dialogue interfaces for tutoring systems. Her e-mail address is rosecp@pitt.edu.

Pamela Jordan is a research associate at the Learning Research and Development Center at the University of Pittsburgh. Her main interests are in computer-mediated natural language dialogue, the analysis of these dialogues to identify effective communication strategies, and the creation of dialogue agents to test strategies. Her e-mail address is pjordan@pitt.edu.

Derek Harter is a Ph.D. candidate in computer science at the University of Memphis. He holds a B.S. in computer science from Purdue University and an M.S. in computer science and AI from Johns Hopkins University. His main interests are in dynamic and embodied models of cognition and neurologically inspired models of action selection for autonomous agents. His e-mail address is dharter@memphism.edu.