Teaching Tactics in AutoTutor

Arthur C. Graesser[1], Natalie Person[2], Derek Harter[1] and the Tutoring Research Group

[1]Department of Psychology, University of Memphis, Memphis, TN 38152-6400
a-graesser@memphis.edu, dharter@memphis.edu

[2]Department of Psychology, Rhodes College, 2000 N. Parkway, Memphis, TN 38112
person@rhodes.edu

ABSTRACT: The Tutoring Research Group at the University of Memphis has developed a computer tutor (called AutoTutor) that simulates the discourse patterns and pedagogical strategies of a typical human tutor. The discourse patterns and pedagogical strategies were based on a previous project that dissected 100 hours of naturalistic tutoring sessions. AutoTutor is currently targeted for college students in introductory computer literacy courses, who learn the fundamentals of hardware, operating systems, and the Internet. Instead of merely being an information delivery system, AutoTutor serves as a discourse prosthesis or collaborative scaffold that assists the student in actively constructing knowledge. Evaluations of AutoTutor-1 have shown that the tutoring system improves learning and memory of the lessons by .5 standard deviation units, compared with a control condition in which college students reread chapters. This paper contrasts the teaching tactics of AutoTutor-1 and AutoTutor-2. In AutoTutor-1, a piece of information is considered covered if it mentioned by either the computer tutor or the student in a shared discourse space. In AutoTutor-2, a piece of information is considered covered only if it is articulated by the student.

1 Background

The Tutoring Research Group (TRG) at the University of Memphis has been developing a computer tutor, called AutoTutor, that simulates a typical human tutor [17, 20, 41, 48]. AutoTutor attempts to comprehend student contributions and to simulate dialog moves of human tutors. AutoTutor-1 attempts to simulate the dialog moves of normal (unskilled) tutors, whereas AutoTutor-2 will incorporate more sophisticated tutoring strategies. AutoTutor is currently being developed for college students who take an introductory course in computer literacy. These students learn the fundamentals of computer hardware, the operating system, and the Internet.

2 An Example Tutorial Dialog between AutoTutor and a Student

A brief snapshot of AutoTutor in action should concretize the nature of AutoTutor. AutoTutor works by having a conversation with the learner. AutoTutor appears as a talking head that acts as a dialog partner with the learner. The talking head delivers AutoTutor's dialog moves with synthesized speech, intonation, facial expressions, and gestures. The major question or problem that is being worked on is both spoken by AutoTutor and is printed at the top of the screen. The major questions/problems are generated systematically from a curriculum script, a module that will be discussed later. AutoTutor's major questions and problems are not the fill-in-the blank, true/false, or multiple-choice questions that are so popular in the US educational system. Instead, the questions and problems invite lengthy explanations and deep reasoning (e.g., answers to *why*, *how*, *what-if* questions). The goal is to encourage students to articulate lengthier answers that exhibit deep reasoning, rather than to recite short snippets of shallow knowledge. There is a continuous multi-turn tutorial dialog between AutoTutor and the learner during the course of answering a question (or solving a problem). When considering both the learner and AutoTutor, it typically takes 10 to 20 turns during the tutorial dialog that occurs when a single question from the curriculum script is answered. The learner types in his/her contributions during the exchange by keyboard. For some topics, there are graphical displays and animation, with components that AutoTutor points to. We want AutoTutor to be a good conversation partner that comprehends, speaks, points, and displays emotions, all in a coordinated fashion.

The exchange below is one of the shorter tutorial dialogs between a college student and AutoTutor. This student was an above average student in the computer literacy class.

01 AUTOTUTOR: When you turn on the computer, how is the operating system first activated and loaded into RAM?
02 STUDENT: When the computer is turned on a file is automatically booted up.
03 AUTOTUTOR: Anything else?
04 STUDENT: This file contains the operating system.
05 AUTOTUTOR: What else?
06 STUDENT: The operating system is then read into RAM and activated.
07 AUTOTUTOR: Right. The CPU executes instructions in ROM, which help the operating system to start.
08 STUDENT: After being read from ROM the OS is activated.
09 AUTOTUTOR: Right. Let's review what we've just gone over. The operating system must first be stored on the hard disk. When you turn on the computer, the CPU executes instructions stored in ROM. These instructions help boot the operating system. The operating system is then loaded from the hard disk into RAM.

AutoTutor first asks a how-question in turn 1. Then the student provides answer information in turns 2, 4, and 6. These answers are encouraged by AutoTutor's pumps in turns 3 and 5 (i.e., "Anything else?", "What else?"). AutoTutor delivers these pumps in an effort to encourage active learning and question answering on the part of the student. 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 explore what the student knows. That is, a central educational philosophy behind AutoTutor is that a key feature of effective learning lies in assisting students in actively constructing subjective explanations and elaborations of the material [7,13], as students answer questions and solve problems that require deep reasoning. At the same time, however, the students need to answer enough questions and to solve enough problems for them to understand the constraints of the domain knowledge. It would not be good for the student to flounder unproductively for a long time, so AutoTutor sometimes needs to bring the student back on track by supplying cues and clues that lead to the evolution of a complete answer to the question. The student had forgotten the role of ROM in launching the operating system, so AutoTutor brings up ROM in turn 7. The student builds on this suggestion in turn 8. At that point, the important pieces of a good complete answer have been covered, so AutoTutor summarizes the answer in turn 9. AutoTutor periodically gives positive immediate feedback after the student contributions (i.e., "right."). This feedback is not only motivating, but creates the impression that AutoTutor is listening to what the student is communicating. These characteristics of a tutorial exchange are quite similar to discourse patterns in normal tutoring between humans [18, 40].

3 Performance of AutoTutor-1

After we developed AutoTutor-1, performance evaluations were made on a number of levels. Some of these evaluations are summarized below.

Gains in Learning and Memory. AutoTutor-1 was tested on 64 students in a computer literacy course at the University of Memphis. Our evaluations of gains in learning and memory were very promising. AutoTutor provided an effect size increment of approximately .5 standard deviations units when compared to a control condition where students reread yoked chapters in the book. This increment in learning gains was found for questions that tap both deep and shallow learning.

Conversational Smoothness and Pedagogical Quality of AutoTutor's Speech Acts. We evaluated AutoTutor on the conversational smoothness and pedagogical quality of its dialog moves in the turn-by-turn tutorial dialog [42]. When experts rate the quality of AutoTutor's dialog moves, the mean ratings are positive (i.e., more good than bad), but there clearly is room to improve in the naturalness and pedagogical effectiveness of its dialog. One of the goals of AutoTutor-2 is to improve the quality of the tutorial dialog.

AutoTutor's Evaluation of the Quality of Student Contributions. We have evaluated AutoTutor on how well it computes the quality of contributions that students type in during the tutorial dialog. AutoTutor attempts to "comprehend" the student input by segmenting the contributions into speech acts and matching the content to ideal answers to questions through latent semantic analysis (LSA)[26]. LSA is a statistical, corpus-based method of representing knowledge that provides the foundation for grading essays, which are frequently not well formed grammatically, semantically, and rhetorically; LSA-based essay graders can assign grades to assays as reliably as experts in composition [14]. Our research revealed that AutoTutor is almost as good as an expert in computer literacy in evaluating the quality of student answers to questions and the quality of contributions in the tutorial dialog [21, 49].

4 Tutorial Dialog with Unskilled Tutors and Ideal Tutors

It is beyond the scope of this section to describe the methods and results of the research that Graesser and Person conducted on naturalistic tutoring sessions. It suffices to say that we videotaped, transcribed, and analyzed nearly 100 hours of naturalistic tutoring sessions during the last decade. After analyzing this rich corpus, we discovered what tutors do versus do not do during most tutoring sessions. Our discoveries were enlightening and often counterintuitive. Whatever tutors do is extremely effective when considering learning gains. Human-to-human tutoring enhances learning by .4 to 2.3 standard deviation units compared to classroom controls and other suitable controls [10,3]. Human tutors are extremely effective even though over 90% percent of the tutors in actual school systems are untrained in tutoring skills and have moderate domain knowledge. They are peer tutors, cross-age tutors, or paraprofessionals, but rarely accomplished professionals.

Whereas AutoTutor-1 simulated the dialog moves of normal human tutors, AutoTutor-2 was designed to advance the project to a new level by incorporating ideal tutoring strategies. Our anatomy of normal tutoring sessions revealed that normal unskilled tutors do not use most of the ideal tutoring strategies that have been identified in education and the intelligent tutoring system enterprise. These strategies include the Socratic method [11], modeling-scaffolding-fading [12], reciprocal training [39], anchored situated learning [4], error identification and correction [2, 47, 32], frontier learning, building on prerequisites [16], and sophisticated motivational techniques [31]. Detailed discourse analyses have been performed on small samples of accomplished tutors in an attempt to identify sophisticated tutoring strategies [15, 22, 35, 36, 44]. However, we discovered that the vast majority of these sophisticated tutoring strategies were virtually nonexistent in the unskilled tutoring sessions that we videotaped and analyzed [19, 40]. Tutors clearly need to be trained how to use the sophisticated tutoring skills because they do not routinely emerge in naturalistic tutoring with untrained tutors. We believe that the most effective computer tutor will be a hybrid between naturalistic tutorial dialog and ideal pedagogical strategies.

5 The Mechanisms of AutoTutor-1 and AutoTutor-2

This section briefly mentions the components and mechanisms of AutoTutor-1, along with the enhancements that are currently being developed in AutoTutor-2.

5.1 Curriculum Scripts with Example Problems, Deep Questions, Graphics, and Animation

A curriculum script is a loosely ordered set of skills, concepts, example problems, and question-answer units. Most human tutors follow a script-like macrostructure, but briefly deviate from the structure when the student manifests difficulties, misconceptions, and errors. The content of the curriculum script in tutoring (compared with classrooms) has more deep reasoning questions (e.g., why, how, what-if, what-if-not), more problems to solve, and more examples [19, 40]. AutoTutor has a curriculum script that organizes the topics (i.e., content) of the tutorial dialog. The script includes didactic descriptions, tutor-

posed questions, example problems, figures, and diagrams (along with anticipated good responses to each topic).

Verbal Content. Each topic in the curriculum script is represented as a structured set of words, sentences, or paragraphs in a free text format. Associated with each topic (problem or question) is a focal question, a set of basic noun-like concepts, a set of ideal good answer aspects (each being roughly a sentence of 10-20 words), different forms of expressing or eliciting each ideal answer aspect (i.e., a hint, prompt, versus assertion), a set of anticipated bad answers (i.e., bugs, misconceptions), a correction for each bad answer, a summary of the answer or solution, and a set of anticipated subquestions and answers. All of this content had to be hand-crafted by the research team in AutoTutor-1.

Appendix A shows a portion of one of the curriculum scripts on the topic of operating systems. This curriculum script was developed for AutoTutor-2. The difficulty level of this question is moderate, as opposed to easy or difficult. Context information is presented prior to the focal question: "How does the operating system of a typical computer process !several jobs simultaneously, with only !one CPU?" The explanation point symbol (!) is part of the mark-up language to the talking head; it designates that the word should be stressed in the synthesized speech. The ideal answer to this question is a lengthy answer that has 5 good answer aspects; Appendix A includes information about two of the 5 good answer aspects, but not the other three good answer aspects. For each good answer aspect (signified as pgood), there is a verbal description. The student's input, entered by keyboard, is constantly compared to each of the 5 good answer aspects as the dialog evolves, turn by turn. LSA is used to assess the match between student contributions and each good answer aspect.

Very often, a student cannot articulate a particular good answer aspect. AutoTutor needs to provide some scaffolding to extract this information from student or to get the student to articulate this knowledge. Elaborations, hints, and prompts provide this scaffolding. Elaborations succinctly assert the desired information (signified by pelab). Hints are questions (signified by phint) that lead the student to the desired information (signified by phintc). Prompts are assertions that that leave out the last word (signified by pprompt); when these get articulated, there is an intonation that strongly encourages or signals the student to fill in the desired word (signified by ppromptk). After the student types in the response to the prompt, AutoTutor presents the correct missing word or expressions (signified by ppromptc). It should be noted that there are several prompts and hints for each good answer aspect. AutoTutor attempts to extract the desired information from the student by many different cues and clues.

The curriculum script also has a number of bad answers that capture common bugs and misconceptions. When a student expresses a contribution that matches one of the bad answers (signified by bad and bbad), then AutoTutor corrects the error by splicing in a correction (signified by splice).

In the future we plan on developing software modules that either automatically generates or assists the instructor in generating the hints, prompts, assertions for each good answer aspect, using recent advances in computational linguistics. Similarly, the corrections need to be generated for each bad answer. We also plan on tuning and increasing the number of production rules that assign the correct intonation patterns to questions, answers, hints, prompts, assertions, corrections, and summaries.

Graphics and Animation. AutoTutor-1 and AutoTutor-2 have graphic displays and a small number of animation clips that coordinate graphic and animated displays, speech synthesis, facial expressions, communicative gestures, and pointing. Extensive use of multimedia can have its advantages. But there are also liabilities from "feature bloat" to the extent that a multimedia show splits the attention of the student [46]. The coordination of the visual and auditory information was guided by recent research on multimedia, education, and cognitive science to the extent that research is available. For example, narrative information (i.e., what AutoTutor says) needs to be sequenced simultaneously with visual information [33], text and pictures must be in spatial and temporal contiguity [33], and it is not a good policy to present lengthier narrative messages in both a text and an auditory modality simultaneously [24].

5.2 Natural Language Extraction and Speech Act Classification.

AutoTutor needs to classify the speech acts of student contributions in order to flexibly respond to what the student types in. Speech act classification requires analysis of the language and punctuation that is entered in a student's turn. The field of computational linguistics has made noticeable progress in automating many components of language analysis that lie within the span of a sentence and short discourse segments, such as identifying the correct sense of words with multiple senses, parsing sentence syntax, and extracting information that is relevant to slots in conceptual templates [1, 29]. Unfortunately, most of the work in computational linguistics has used programming languages (i.e., LISP and Prolog) that are not readily integrated with AutoTutor's Java. We did use some of the some of the lexicons and computational architectures that have been developed in cognitive science, however. AutoTutor segments each student's turn into words and assigns each word into one of 17 syntactic classes with near perfect accuracy [38]. Then AutoTutor segments the categorized string of words and punctuation marks within a learner's turn into speech act units, relying on punctuation to perform this segmentation. Then a neural network and a frozen expression catalogue are used to assign each speech act into one of the following speech act categories: Assertion, WH-question, YES/NO question, Metacognitive comments (I don't understand), Metacommunicative acts (Could you repeat that?), and Short Response. The syntactic tags of the first 3 words of a speech act are very diagnostic of the speech act category.

5.3 Latent Semantic Analysis

The fact that world knowledge is inextricably bound to the process of comprehending language and discourse is widely acknowledged, but researchers in computational linguistics and artificial intelligence have not had a satisfactory approach to handling the deep abyss of world knowledge. The traditional approach to representing world knowledge in artificial intelligence has been structured representations, such as semantic networks and conceptual graphs [28, 30]. However, it takes too long to perform the knowledge engineering that is needed to build an intelligent tutoring system of reasonable scope. Latent semantic analysis (LSA) has recently been proposed as a statistical representation of a large body of world knowledge [26, 27]. LSA capitalizes on the fact that particular words appear in particular texts (called "documents"). Each word, sentence, or text ends up being a weighted vector on the K dimensions. The "match" (i.e., similarity in meaning, conceptual relatedness) between two words, sentences, or texts is computed as a geometric cosine (or dot product) between the two vectors, with values ranging from 0 to 1. The match between two language strings can be high even though there are few if any words in common between the two strings. LSA goes well beyond simple string matches because the meaning of a language string is partly determined by the company (other words) that each word keeps.

As mentioned earlier, AutoTutor-1 successfully used LSA as the backbone for representing computer literacy. The quality of student answers was successfully computed from the assertions expressed during the student's turns. A student with high ability had (a) a high mean LSA between the assertions and good answer aspects and (b) a low mean LSA match with bad answers. AutoTutor-2 will continue to use LSA for representing world knowledge, but will also use structured representations that are needed for some of the natural language processing modules.

5.4 Selection of Dialog Moves

AutoTutor-1 selects dialog moves by using fuzzy production rules and algorithms that will not be elaborated here. There are different categories of dialog moves: main questions, short feedback (i.e., positive, neutral, negative), pumps ("uh huh", "tell me more"), prompts ("The primary memories of the CPU are ROM and _____"), hints, assertions (elaborations), corrections, and summaries. The selection and sequencing of the categories are sensitive to various parameters that are induced from the tutorial dialog. Fuzzy production rules are tuned to (a) the quality of the student's assertions in the preceding turn, as computed by LSA, (b) global parameters that refer to the ability, verbosity, and initiative of the student, and (c) the extent to which the good answer aspects of the topic had been covered.

The selection of the next good answer aspect to cover was determined by the zone of proximal development in AutoTutor-1. AutoTutor-1 keeps track of the extent to which each aspect (A_i) has been covered as the

dialog evolves for a topic. The coverage metric varies from 0 to 1 and gets updated as each Assertion is produced by the tutor or learner. LSA is used to compute the extent to which the various Assertions cover the particular aspects associated with a topic. If some threshold (t) is met or exceeded, then the aspect A_i is considered covered. AutoTutor-1 selects, as the next aspect to cover, that aspect that has the highest subthreshold coverage score. Therefore, AutoTutor builds on the fringes of what is known in the discourse space between the student and AutoTutor. A topic is finished when all of the aspects have coverage values that meet or exceed the threshold t. AutoTutor-2 will use two additional factors when computing what aspect to cover next. First, AutoTutor-2 will enhance discourse coherence by selecting the next aspect (A_i) that is most similar to the previous aspect that was covered. Second, AutoTutor-2 will select pivotal aspects that have a high family resemblance to the remaining uncovered aspects; that is, there will be an attempt to select an aspect that drags in the content of the remaining aspects to be covered. Whereas AutoTutor-1 capitalized on the zone of proximal development exclusively, AutoTutor-2 will also consider conversational coherence and pivotal ideas when selecting the next good answer aspect to cover.

AutoTutor-2 will incorporate tactics that attempt to get the student to articulate the good answer aspect that is selected. Whereas AutoTutor-1 scored aspect (A_i) as covered if it was expressed by either the tutor or the student, AutoTutor-2 counts only what the student says when evaluating coverage; so if aspect (A_i) was not expressed by the student, it is not covered at all. This forces the student to articulate the explanations in their entirety, an extreme form of constructivism. AutoTutor-2 will use progressive specificity when formulating dialog moves that flesh out a particular aspect (A_i); there will be 2 or more cycles of hinting, prompting, and then asserting until the student articulates (A_i).

5.5 Talking Head with Gestures

Researchers have recently developed computer-generated animated talking heads that have facial features synchronized with speech and in some cases appropriate gestures [6, 9, 23]. Ideally, the computer would control the eyes, eyebrows, mouth, lips, teeth, tongue, cheekbones, and other parts of the face in a fashion that is meshed appropriately with the language and emotions of the speaker. A talking head is an important feature of AutoTutor because it concretely grounds the conversation between the tutor and learner. A talking head also provides a separate channel of cues for providing mixed feedback to the learner. When a learner's contribution is incorrect or vague, for example, the speech is often positive and polite whereas the face has a puzzled expression; this conflicting message that satisfies both pedagogical and politeness constraints would be preferable to a threatening speech message that says, "That's wrong" or "I'm having trouble understanding you." The nonverbal facial cues are known to be an important form of backchannel feedback during tutoring [15, 19, 43], as well as other contexts of conversation [8]. Similarly, pitch, pause, duration, amplitude, and intonation contours are among the intonation cues that signal backchannel feedback, affect, and emphasis [5]. AutoTutor-1's dialog moves are delivered by a talking head that synchronizes synthesized speech, facial expressions, and sometimes gestures. Microsoft Agent is currently being used as the talking head with synthesized speech, with parameters of the facial expressions and intonation being generated by fuzzy production rules [34]. Unfortunately, the grain size of the intonation and facial parameters of Microsoft Agent is too crude to handle the subtle facial expressions that we desire. Also, the sequential constraints of microcomputers make it difficult to handle the synchronization of components in parallel. For example, the current version of Microsoft Agent does not allow AutoTutor to point and to display facial expressions at the same time that it produces synthesized speech. Therefore, AutoTutor-2 will use Java 3D plus a neurofuzzy controller [25] that allows lower-level programming for new behaviors on the fly, after interpolating from a sample of pre-scripted prototype behaviors during its development. The animated agents in the MIT Media Labs [6] normally require several powerful computers to provide synchronization of speech, intonation, face, and gesture, whereas AutoTutor-2 will be delivered on a Pentium and the web.

6 Conclusions

The success of AutoTutor rests on the fundamental premise that discourse patterns provide an important class of teaching tactics and strategies. A computer tutor can be viewed as a dialogue partner that assists the learner in exploring his or her own knowledge and that exposes the students to the constraints of the problem and the fragments of the good answer. In essence, AutoTutor is a discourse prosthesis that

scaffolds the student to new levels of mastery through conversation. The pedagogical philosophy is similar to the Intelligent Collaborative Learning System [45], except that the latter system manages discourse in groups of learners. We believe that one important key to learning lies in getting the student to say the right thing and the right time.

References

1. Allen, J. (1995). Natural language understanding. Redwood City, CA: Benjamin/Cummings.
2. Anderson, J. R., Corbett, A. T., Koedinger, K. R., Pelletier, R. (1995). Cognitive tutors: Lessons learned. The Journal of the Learning Sciences, 4, 167-207.
3. 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.
4. Bransford, J. D., Goldman, S. R., Vye, N. J. (1991). Making a difference in people's ability to think: Reflections on a decade of work and some hopes for the future. In R. J. Sternberg, L. Okagaki (Eds.), Influences on children (pp. 147-180). Hillsdale, NJ: Erlbaum.
5. Brennan, S. E., Williams, M. (1995). The feeling of another's knowing: Prosody and filled pauses as cues to listeners about the metacognitive states of speakers. Journal of Memory and Language, 34, 383- 398.
6. Cassell, J., Thorisson, K.R. (1999). The power of a nod and a glance: Envelope vs. emotional feedback in animated conversational agents. Applied Artificial Intelligence, 13, 519-538.
7. Chi, M. T. H., de Leeuw, N., Chiu, M., LaVancher, C. (1994). Eliciting self-explanations improves understanding. Cognitive Science, 18, 439-477.
8. Clark, H.H. (1996). Using language. Cambridge: Cambridge University Press.
9. Cohen, M.M., Massaro, D.W. (1994). Development and experimentation with synthetic visible speech. Behavior Research Methods, Instruments, and Computers, 26, 260-265.
10. Cohen, P. A., Kulik, J. A., Kulik, C. C. (1982). Educational outcomes of tutoring: A meta-analysis of findings. American Educational Research Journal, 19, 237-248.
11. Collins, A. (1985). Teaching reasoning skills. In S.F. Chipman, J.W. Segal, R. Glaser (Eds), Thinking and learning skills (vol. 2, pp 579-586). Hillsdale, NJ: Erlbaum.
12. Collins, A., Brown, J. S., Newman, S. E. (1989). Cognitive apprenticeship: Teaching the craft of reading, writing, and mathematics. In L. B. Resnick (Ed.), Knowing, learning, and instruction: Essays in honor of Robert Glaser (pp. 453-494). Hillsdale, NJ: Erlbaum.
13. Conati, C., VanLehn, K. (1999). Teaching metacognitive skills: Implemention and evaluation of a tutoring system to guide self-explanation while learning from examples. In S.P. Lajoie and M. Vivet, Artificial Intelligence in Education (pp. 297-304). Amsterdam: IOS Press.
14. Foltz, P.W. (1996). Latent semantic analysis for text-based research. Behavior Research Methods, Instruments, and Computers, 28, 197-202.
15. Fox, B. (1993). The human tutorial dialog project. Hillsdale, NJ: Erlbaum.
16. Gagné, R. M. (1977). The conditions of learning (3rd ed.). New York: Holdt, Rinehart, Winston.
17. Graesser, A.C., Franklin, S., Wiemer-Hastings, P. TRG (1998). Simulating smooth tutorial dialog with pedagogical value. Proceedings of the American Association for Artificial Intelligence (pp. 163-167). Menlo Park, CA: AAAI Press.
18. Graesser, A.C., Person, N.K. (1994). Question asking during tutoring. American Educational Research Journal, 31, 104-137.
19. Graesser, A.C., Person, N.K., Magliano, J.P. (1995). Collaborative dialog patterns in naturalistic one-on-one tutoring. Applied Cognitive Psychology, 9, 359-387.
20. Graesser, A.C., Wiemer-Hastings, K., Wiemer-Hastings, P., Kreuz, R., TRG (1999). AutoTutor: A simulation of a human tutor. Journal of Cognitive Systems Research, 1, 35-51.
21. Graesser, A.C., Wiemer-Hastings, P., Wiemer-Hastings, K., Harter, D., Person, N., TRG (2000, in press). Using latent semantic analysis to evaluate the contributions of students in AutoTutor. Interactive Learning Environments.
22. Hume, G. D., Michael, J.A., Rovick, A., Evens, M. W. (1996). Hinting as a tactic in one-on-one tutoring. The Journal of the Learning Sciences, 5, 23-47.
23. Johnson, W. L., Rickel, J. W., Lester, J.C. (in press). Animated pedagogical agents: Face-to-face interaction in interactive learning environments. International Journal of Artificial Intelligence in Education.
24. Kalyuga, S., Chandler, P., Sweller, J. (1999). Managing split-attention and redundancy in multimedia intsruction. Applied Cognitive Psychology, 13, 351-371.
25. Kosko, B. (1992). Neural networks and fuzzy systems. New York: Prentice Hall.

26. Landauer, T.K., Dumais, S.T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. Psychological Review.
27. Landauer, T.K., Foltz, P.W., Laham, D. (1998). An introduction to latent semantic analysis. Discourse Processes, 25, 259-284.
28. Lehmann, F. (1992)(Eds.). Semantic networks in artificial intelligence. New York: Pergamon.
29. Lehnert, W. (1997). Information extraction: What have we learned? Discourse Processes, 23, 441-470.
30. Lenat, D.B. (1995). CYC: A large-scale investiment in knowledge infrastructure. Communications of the ACM, 38, 33-38.
31. Lepper, M. R., Woolverton, M., Mumme, D.L., Gurtner, J.L. (1991). Motivational techniques of expert human tutors: Lessons for the design of computer-based tutors. In S.P. Lajoie, S.J. Derry (Eds.), Computers as cognitive tools (pp. 75-105). Hillsdale, NJ: Erblaum.
32. Lesgold, A., Lajoie, S., Bunzo, M., Eggan, G. (1992). SHERLOCK: A coached practice environment for an electronics troubleshooting job. In J. H. Larkin, R. W. Chabay (Eds.), Computer-assisted instruction and intelligent tutoring systems (pp. 201-238). Hillsdale, NJ: Erlbaum.
33. Mayer, R. E., Moreno, R. (1998). A split attention effect in multimedia learning: Evidence for dual processing systems in working memory, Journal of Educational Psychology, 90, 312-320.
34. McCauley, L., Gholson, B., Hu, X., Graesser, A.C., and the Tutoring Research Group (1998). Delivering smooth tutorial dialogue using a talking head. Proceedings of the Workshop on Embodied Conversation Characters (pp. 31-38). Tahoe City, CA: AAAI and ACM.
35. Merrill, D. C., Reiser, B. J., Ranney, M., Trafton, J. G. (1992). Effective tutoring techniques: A comparison of human tutors and intelligent tutoring systems. The Journal of the Learning Sciences, 2, 277-305.
36. Moore, J.D. (1995). Participating in explanatory dialogues. Cambridge, MA: MIT Press.
37. Moreno, R., Mayer, R. E. (1999). Cognitve principles of multimedia learning: The role of modality and contingency. Journal of Educational Psychology, 91, 358-368.
38. Olde, B.A., Hoeffner, J., Chipman, P., Graesser, A.C., and the Tutoring Research Group (1999). A connectionist model for part of speech tagging. Proceedings of the American Association for Artificial Intelligence (pp. 172-176). Menlo Park, CA: AAAI Press.
39. Palinscar, A. S., Brown, A. (1984). Reciprocal teaching of comprehension-fostering and comprehension-monitoring activities. Cognition Instruction, 1, 117-175.
40. Person, N.K, Graesser, A.C. (1999). Evolution of discourse in cross-age tutoring. In A.M. O'Donnell and A. King (Eds.), Cognitive perspectives on peer learning (pp. 69-86). Mahwah, NJ: Erlbaum.
41. Person, N. K., Graesser, A. C., the Tutoring Research Group (in press). Designing AutoTutor to be an effective conversational partner. Proceedings for the 4th International Conference of the Learning Sciences. Ann Arbor, MI.
42. Person, N.K., Graesser, A.C., Kreuz, R.J., Pomeroy, V., TRG (in press). Simulating human tutor dialog moves in AutoTutor. Journal of Artificial Intelligence in Education.
43. Person, N. K., Kreuz, R. J., Zwaan, R., Graesser, A. C. (1995). Pragmatics and pedagogy: Conversational rules and politeness strategies may inhibit effective tutoring. Cognition and Instruction, 13, 161-188.
44. Putnam, R.T. (1987). Structuring and adjusting content for students: A study of live and simulated tutoring of addition. American Educational Research Journal, 24, 13-48.
45. Soller, A., Linton, F., Goodman, B., Lesgold, A. (1999). Toward intelligent analysis and support of collaborative learning interaction. In S.P. Lajoie and M. Vivet, Artificial Intelligence in Education (pp. 75-82). Amsterdam: IOS Press.
46. Sweller, J., Chandler, P. (1994). Why some material is difficult to learn. Cognition and Instruction, 4, 295-312.
47. VanLehn, K. (1990). Mind bugs: The origins of procedural misconceptions. Cambridge, MA: MIT Press.
48. Wiemer-Hastings, P., Graesser, A.C., Harter, D., and the Tutoring Research Group (1998). The foundations and architecture of AutoTutor. Proceedings of the 4th International Conference on Intelligent Tutoring Systems (pp. 334-343). Berlin, Germany: Springer-Verlag.
49. Wiemer-Hastings, P., Wiemer-Hastings, K., Graesser, A. (1999). Improving an intelligent tutor's comprehension of students with Latent Semantic Analysis. In S.P. Lajoie and M. Vivet, Artificial Intelligence in Education (pp. 535-542). Amsterdam: IOS Press.

Appendix A: An Example Curriculum Script for One Question and Answer

\topic_Operating_System
\problem_solution-8 \moderate

\info-8 !Large, !multi-user !computers often work on several jobs !simultaneously. This is known as !concurrent processing. Computers with state-of-the-art !parallel !processing use multiple CPUs to process !several jobs simultaneously. However, the typical computer today has only !one CPU. So here's your !question.

\question-8 How does the operating system of a typical computer process !several jobs simultaneously, with only !one CPU?

\ideal-8 The operating system helps the computer to work on several jobs simultaneously by rapidly switching back and forth between jobs. By rapidly switch back and forth between jobs, the operating system takes advantage of idle time on one job by working on another job. Timesharing computers use concurrent processing whenever multiple users are connected to the system. A timesharing computer moves from terminal to terminal, checking for input and processing each user's data in turn. Concurrent processing is common in personal computer operating systems that allow multitasking. Multitasking allows the computer user to issue a command that initiates a process in one application while the user works with other applications.

\pgood-8-1 The operating system helps the computer to work on several jobs simultaneously by rapidly switching back and forth between jobs.
\pelab-8-1 The operating system switches rapidly !back and !forth between !jobs.
\phint-8-1-1 How can the operating system take advantage of !idle !time on the job?
\phintc-8-1-1 The operating system switches between jobs.
\phint-8-1-2 How does the operating system manage different programs?
\phintc-8-1-2 It switches rapidly back and forth.
\phint-8-1-3 Why would the operating system switch rapidly between jobs?
\phintc-8-1-3 The computer can work on several jobs simultaneously.
\pprompt-8-1-1 The operating system switches rapidly between
\ppromptc-8-1-1 Between !jobs.
\ppromptk-8-1-1 jobs.
\pprompt-8-1-2 Instead of being !sequential , the computer works on several jobs
\ppromptc-8-1-2 Several jobs !simultaneously.
\ppromptk-8-1-2 simultaneously, concurrently, at the same time.
\pprompt-8-1-3 Several jobs are run at the same !time by having the operating system
\ppromptc-8-1-3 The operating system !switch between jobs.
\ppromptk-8-1-3 switch, alternate.

\pgood-8-2 When there is idle time on one process or job, the operating system takes advantage of this idle time by working on another job.
\pelab-8-2 The operating system takes advantage of idle time on !one job by working on !another job.
\phint-8-2-1 How does the operating system avoid idle time on a job?
\phintc-8-2-1 When there is idle time on one process or job, the operating system works on another job.
\phint-8-2-2 How can the !operating system take advantage of !idle !time on a job?
\phintc-8-2-2 The operating system works on another job.
\phint-8-2-3 What computer component allows the computer to work on !several jobs at the same !time?
\phintc-8-2-3 The operating system.
\pprompt-8-2-1 When there is idle time on !one job, the operating system can work on another
\ppromptc-8-2-1 On another !job.
\ppromptk-8-2-1 job.
\pprompt-8-2-2 The operating system can work on !another job when the !first job encounters
\ppromptc-8-2-2 Encounters !idle !time.
\ppromptk-8-2-2 idle time.
\pprompt-8-2-3 A job faces idle time when there is no progress on any of its
\ppromptc-8-2-3 Any of its !processes.
\ppromptk-8-2-3 processes.

THERE ARE 3 ADDITIONAL GOOD ANSWER ASPECTS

\bad-8-1 The operating system completes one job first and then works on another.
\bbad-8-1 The operating system does one job at a time.
\splice-8-1 The operating system can work on !several jobs at !once.

\bad-8-5 If I give my command and someone else gives their command, then my command will be carried out first.
\bbad-8-5 The command entered first is done first.
\splice-8-5 The operating system responds to !both commands !concurrently.

THERE ARE 5 ADDITIONAL BAD ANSWERS

\summary-8 The operating system !rapidly switches !back and !forth between !jobs. When there is idle time on !one job, the operating system switches to !another job. multi !tasking allows the computer to work concurrently on !one command while processing !other commands of a single user. !Timesharing allows several !users to use an operating system !simultaneously.