Simulating Human Tutor Dialog Moves in AutoTutor

Natalie K. Person, Arthur C. Graesser, Roger J. Kreuz, Victoria Pomeroy

International Journal of Artificial Intelligence in Education (2001), 12, to appear

Simulating Human Tutor Dialog Moves in AutoTutor

NATALIE K. PERSON[1], ARTHUR C. GRAESSER[2], ROGER J. KREUZ[2], VICTORIA POMEROY[2], and the TUTORING RESEARCH GROUP[2]

[1]*Department of Psychology, Rhodes College, 2000 N. Parkway, Memphis, TN 38112*
Person@rhodes.edu

[2]*Department of Psychology, University of Memphis, Memphis, TN 38152*
a-graesser@memphis.edu, kreuzrj@memphis.edu, vpomeroy@memphis.edu
http://mnemosyne.csl.psyc.memphis.edu/home/graesser/
www.psyc.memphis.edu/faculty/kreuz/kreuz.htm
http://mnemosyne.csl.psyc.memphis.edu/home/pomeroyv/

ABSTRACT

This purpose of this paper is to show how prevalent features of successful human tutoring interactions can be integrated into a pedagogical agent, AutoTutor. AutoTutor is a fully automated computer tutor that responds to learner input by simulating the dialog moves of effective, normal human tutors. AutoTutor's delivery of dialog moves is organized within a 5-step framework that is unique to normal human tutoring interactions. We assessed AutoTutor's performance as an effective tutor and conversational partner during tutoring sessions with virtual students of varying ability levels. Results from three evaluation cycles indicate the following: (1) AutoTutor is capable of delivering pedagogically effective dialog moves that mimic the dialog move choices of human tutors, and (2) AutoTutor is a reasonably effective conversational partner.

INTRODUCTION AND BACKGROUND

Over the last decade a number of researchers have attempted to uncover the mechanisms of human tutoring that are responsible for student learning gains. Many of the informative findings have been reported in studies that have systematically analyzed the collaborative discourse that occurs between tutors and students (Fox, 1993; Graesser & Person, 1994; Graesser, Person, & Magliano, 1995; Hume, Michael, Rovick, & Evens, 1996; McArthur, Stasz, & Zmuidzinas, 1990; Merrill, Reiser, Ranney, & Trafton, 1992; Moore, 1995; Person & Graesser, 1999; Person, Graesser, Magliano, & Kreuz, 1994; Person, Kreuz, Zwaan, & Graesser, 1995; Putnam, 1987). For example, we have learned that the tutorial session is predominately controlled by the tutor. That is, tutors, not students, typically determine *when* and *what* topics will be covered in the session. Further, we know that human tutors rarely employ sophisticated or "ideal" tutoring models that are often incorporated into intelligent tutoring systems. Instead, human tutors are more likely to rely on localized strategies that are embedded within conversational turns. Although many findings such as these have illuminated the tutoring process, they present formidable challenges for designers of intelligent tutoring systems. After all, building a knowledgeable conversational partner is no small feat. However, if designers of future tutoring systems wish to capitalize on the knowledge gained from human tutoring studies, the next generation of tutoring systems will incorporate pedagogical agents that engage in learning dialogs with students. The purpose of this paper is twofold. First, we will describe how prevalent features of successful human tutoring interactions can be incorporated into a pedagogical agent, AutoTutor. Second, we will provide data from several preliminary performance evaluations in which AutoTutor interacts with virtual students of varying ability levels.

AutoTutor is a fully automated computer tutor that is currently being developed by the Tutoring Research Group (TRG). AutoTutor is a working system that attempts to comprehend students' natural language contributions and then respond to the student input by simulating the dialogue moves of human tutors. AutoTutor differs from other natural language tutors in several ways. First, AutoTutor does not restrict the natural language input of the student like other systems (e.g., Adele (Shaw, Johnson, & Ganeshan, 1999); the Ymir agents (Cassell & Thórisson, 1999); Cirscim-Tutor (Hume, Michael, Rovick, & Evens, 1996; Zhou et al., 1999); Atlas (Freedman, 1999); and Basic Electricity and Electronics (Moore, 1995; Rose, Di Eugenio, & Moore, 1999)). These systems tend to limit student input to a small subset of judiciously worded speech acts. Second, AutoTutor does not allow the user to substitute natural language contributions with GUI menu options like those in the Atlas and Adele systems. The third difference involves the open-world nature of AutoTutor's content domain (i.e., computer literacy). The previously mentioned tutoring systems are relatively more closed-world in nature, and therefore, constrain the scope of student contributions.

The current version of AutoTutor simulates the tutorial dialog moves of normal, untrained tutors; however, plans for subsequent versions include the integration of more sophisticated ideal tutoring strategies. AutoTutor is currently designed to assist college students learn about topics covered in an introductory computer literacy course. In a typical tutoring session with AutoTutor, students will learn the fundamentals of computer hardware, the operating system, and the Internet.

A Brief Sketch of AutoTutor

AutoTutor is an animated pedagogical agent that serves as a conversational partner with the student. AutoTutor's interface is comprised of four features: a two-dimensional, talking head, a text box for typed student input, a text box that displays the problem/question being discussed, and a graphics box that displays pictures and animations that are related to the topic at hand. AutoTutor begins the session by introducing himself and then presents the student with a question or problem that is selected from a curriculum script. The question/problem remains in a text box at the top of the screen until AutoTutor moves on to the next topic. For some questions and problems, there are graphical displays and animations that appear in a specially designated box on the screen. Once AutoTutor has presented the student with a problem or question, a multi-turn tutorial dialog occurs between AutoTutor and the learner. All student contributions are typed into the keyboard and appear in a text box at the bottom of the screen. AutoTutor responds to each student contribution with one or a combination of pedagogically appropriate dialog moves. These dialog moves are conveyed via synthesized speech, appropriate intonation, facial expressions, and gestures and do not appear in text form on the screen. In the future, we hope to have AutoTutor handle speech recognition, so students can speak their contributions. However, current speech recognition packages require time-consuming training that is not optimal for systems that interact with multiple users.

The various modules that enable AutoTutor to interact with the learner will be described in subsequent sections of the paper. For now, however, it is important to note that our initial goals for building AutoTutor have been achieved. That is, we have designed a computer tutor that participates in a conversation with the learner while simulating the dialog moves of normal human tutors.

WHY SIMULATE NORMAL HUMAN TUTORS?

It has been well documented that normal, untrained human tutors are effective. Effect sizes ranging between .5 and 2.3 have been reported in studies where student learning gains were measured (Bloom, 1984; Cohen, Kulik, & Kulik, 1982). For quite a while, these rather large effect sizes were somewhat puzzling. That is, normal tutors typically do not have expert domain knowledge nor do they have knowledge about sophisticated tutoring strategies. In order to gain a better understanding of the primary mechanisms that are responsible for student learning

gains, a handful of researchers have systematically analyzed the dialogue that occurs between normal, untrained tutors and students (Graesser & Person, 1994; Graesser et al., 1995; Person & Graesser, 1999; Person et al., 1994; Person et al., 1995). Graesser, Person, and colleagues analyzed over 100 hours of tutoring interactions and identified two prominent features of human tutoring dialogs: (1) a five-step dialog frame that is unique to tutoring interactions, and (2) a set of tutor-initiated dialog moves that serve specific pedagogical functions. We believe these two features are responsible for the positive learning outcomes that occur in typical tutoring settings, and further, these features can be implemented in a tutoring system more easily than the sophisticated methods and strategies that have been advocated by other educational researchers and ITS developers.

Five-step Dialog Frame

The structure of human tutorial dialogs differs from learning dialogs that often occur in classrooms. Mehan (1979) and others have reported a 3-step pattern that is prevalent in classroom interactions. This pattern is often referred to as IRE, which stands for Initiation (a question or claim articulated by the teacher), Response (an answer or comment provided by the student), and Evaluation (teacher evaluates the student contribution). In tutoring, however, the dialog is managed by a 5-step dialog frame (Graesser & Person, 1994; Graesser et al., 1995). The five steps in this frame are presented below.

Step 1: Tutor asks question (or presents problem).
Step 2: Learner answers question (or begins to solve problem).
Step 3: Tutor gives short immediate feedback on the quality of the answer (or solution).
Step 4: Tutor and learner collaboratively improve the quality of the answer.
Step 5: Tutor assesses learner's understanding of the answer.

This 5-step dialog frame in tutoring is a significant augmentation over the 3-step dialog frame in classrooms. We believe that the advantage of tutoring over classroom settings lies primarily in Step 4. Typically, Step 4 is a lengthy multi-turn dialog in which the tutor and student collaboratively contribute to the explanation that answers the question or solves the problem.

At a macro-level, the dialog that occurs between AutoTutor and the learner conforms to Steps 1 through 4 of the 5-step frame. For example, at the beginning of each new topic, AutoTutor presents the learner with a problem or asks the learner a question (Step 1). The learner then attempts to solve the problem or answer the question (Step 2). Next, AutoTutor provides some type of short, evaluative feedback (Step 3). During Step 4, AutoTutor employs a variety of dialog moves (see next section) that encourage learner participation. Thus, instead of being an information delivery system that bombards the learner with a large volume of information, AutoTutor is a *discourse prosthesis* that attempts to get the learner talking about his or her own knowledge. From a pedagogical standpoint, Step 4 promotes active student learning. Other researchers have similarly proposed that the process of actively constructing explanations, elaborations, and mental models of the material is critical for learning, and usually is more effective than merely presenting information to learners (Chi, Bassok, Lewis, Reinmann, & Glaser, 1989; Chi et al., 1994; Moore, 1995; Pressley, Wood, Woloshyn, Martin, King, & Menk, 1992; Webb et al., 1996).

The decision to eliminate Step 5 from AutoTutor's design was empirically motivated. During this step, tutors frequently ask global, comprehension-gauging questions (e.g., "Do you understand?"). Past research indicates that students' answers to these questions tend to be somewhat paradoxical. For example, good students are more likely to say, "No, I don't understand," than poor students (Chi et al., 1989; Person et al., 1994). Given that students' answers to these questions are often unreliable, we chose not to incorporate Step 5 in AutoTutor's dialog structure.

Dialogue Moves of Normal Human Tutors

In our analyses tutoring dialogs, we found that normal human tutors rarely use sophisticated tutoring strategies that have been advocated by educational researchers and designers of intelligent tutoring systems. These strategies include the Socratic method (Collins, 1985), modeling-scaffolding-fading (Collins, Brown, & Newman, 1989), reciprocal training (Palincsar & Brown, 1984), anchored learning (Bransford, Goldman, & Vye, 1991), error diagnosis and correction (Anderson, Corbett, Koedinger, & Pelletier, 1995; VanLehn, 1990; Lesgold et al., 1992), frontier learning, building on prerequisites (Gagne, 1977), and sophisticated motivational techniques (Lepper, Aspinwall, Mumme, & Chabay, 1990). Although detailed discourse analyses have been performed on samples of these sophisticated tutoring strategies (Fox, 1993; Hume et al., 1996; McArthur et al. 1990; Merrill et al., 1992; Putnam, 1987), such strategies were noticeably absent in the untrained tutoring sessions that we analyzed (For a detailed description on how the human tutoring transcripts were analyzed see Graesser & Person, 1994; Graesser et al., 1995; Person & Graesser, 1999; Person et al., 1994).

We found that normal human tutors prefer dialog moves that are carefully tailored to the previous student contribution. More specifically, human tutors choose dialog moves that are sensitive to the quality and quantity of the preceding student turn. The tutor dialog move categories that we identified in human tutoring sessions are provided below.

(1) Positive immediate feedback. "That's right" "Yeah"
(2) Neutral immediate feedback. "Okay" "Uh-huh"
(3) Negative immediate feedback. "Not quite" "No"
(4) Pumping for more information. "Uh-huh" "What else"
(5) Prompting for specific information. "The primary memories of the CPU are ROM and _____"
(6) Hinting. "The hard disk can be used for storage" or "What about the hard disk?"
(7) Elaborating. "CD ROM is another storage medium."
(8) Splicing in/correcting content after a student error.
(9) Summarizing. "So to recap," <succinct recap of answer to question>

Like human tutors, AutoTutor simulates one or a combination of these dialog moves after each student contribution. The conditions under which particular dialog moves are generated will be discussed in the dialog move generator section.

ARCHITECHTURE OF AUTOTUTOR

AutoTutor is an amalgamation of classical symbolic architectures (e.g., those with propositional representations, conceptual structures, and production rules) and architectures that have multiple soft constraints (e.g., neural networks, fuzzy production systems). AutoTutor's major modules include an animated agent, a curriculum script, language analyzers, latent semantic analysis (LSA), and a dialog move generator. All but one of these modules have been discussed rather extensively in previous publications (see Foltz, 1996; Graesser, Franklin, Wiemer-Hastings, & the TRG, 1998; Graesser, Wiemer-Hastings, Wiemer-Hastings, Harter, Person, & the TRG, in press; Hu, Graesser, and the TRG, 1998; Landauer & Dumais, 1997; McCauley, Gholson, Hu, Graesser, & the TRG, 1998; Wiemer-Hastings, Graesser, Harter, & the TRG, 1998). The exception is the dialog move generator. A thorough description of the dialog move generator will follow brief descriptions of the other modules.

AutoTutor's Major Modules

Animated Agent

AutoTutor was created in Microsoft Agent. He is a two-dimensional embodied agent who remains on the screen seated behind a table throughout the entire tutoring session (we are in the process of integrating a 3-dimensional agent). AutoTutor communicates with the learner via synthesized speech, facial expressions, and rudimentary pointing gestures. Each of these communication parameters can be adjusted to maximize AutoTutor's overall effectiveness as a tutor and conversational partner. Although a great deal more could be said about the workings of the animated agent, these mechanisms have been described elsewhere (see McCauley, Gholson, Hu, Graesser, and the TRG, 1998; Person, Klettke, Link, Kreuz, & the TRG, 1999) and are simply beyond the scope of this paper.

Curriculum script

Tutoring sessions with AutoTutor are guided by curriculum scripts. Curriculum scripts are well-defined, loosely structured lesson plans that include important concepts, questions, cases, and problems that teachers and tutors wish to cover in a particular lesson (Graesser & Person, 1994; Graesser et al. 1995; McArthur et al., 1990; Putnam, 1987). AutoTutor's curriculum script includes 37 computer literacy questions and/or problems: one introductory question (i.e., "What are the parts and uses of a computer?") that allows the student to acclimate to the synthesized voice, and 36 topic related questions/problems. AutoTutor's curriculum script currently contains knowledge for three macrotopics: hardware, the operating system, and the Internet. The ordering of the information in the three macrotopics is similar to that presented in the computer literacy course and the textbook (Beekman, 1997).

There are 12 topics within each of the 3 macrotopics (36 total). The 36 topics contain didactic descriptions, tutor-posed questions, cases, problems, figures, and diagrams (along with anticipated good and bad responses for each question/problem). Within each set of 12 topics, 3 levels of difficulty are crossed with 4 topic formats. The three levels of difficulty (easy, medium, difficult) map onto taxonomies of cognitive difficulty and question difficulty (Bloom, 1956; Graesser & Person, 1994; Wakefield, 1996). The four topic formats are: (1) Deep-reasoning Question, (2) Didactic-information + Question, (3) Graphic-display + Question, and (4) Problem + Question.

The curriculum script also includes 36 *Ideal Answers* that correspond to each of the 36 topics. An Ideal Answer consists of a set of N good answers or aspects, $\{A_1, A_2,\ldots A_N\}$, which were determined by experts in area of computer literacy. The numbers of aspects for the 36 topics ranged from 3 to 9. All of the aspects for a given topic need to be covered in the tutorial dialog before AutoTutor will proceed to the next topic. The quality of any given learner contribution is determined by matching the learner contribution to each aspect and all possible combinations of aspects in a particular Ideal Answer. LSA (next section) performs these pattern-matching operations.

Additional information contained in the curriculum script includes: (1) anticipated bad answers for each of the 36 topics, (2) corrective splices (i.e., correct answers) for each anticipated bad answer, and (3) numerous dialog moves (i.e., elaborations, hints, prompts, prompt responses, and summaries) that are related to the aspects in the Ideal Answers. It should be noted that all of the content in the curriculum script is written in English, as opposed to computer code. Therefore, a teacher or other individual who is not an expert programmer can easily author the curriculum script.

Language analyzers

AutoTutor contains several language analyzers that operate on the words that the learner types into the keyboard during a particular conversational turn. These analyzers include: (1) a word and punctuation segmenter, (2) a syntactic class identifier, and (3) a speech act classifier. After

the learner constructs a message and hits the Enter key, the message is broken down into individual words and punctuation marks. The syntactic class identifier then matches each word to the appropriate entry in a large lexicon (approximately 10,000 words) and identifies *all possible* syntactic classes and user frequencies in the English language. For example, "program" is either a noun, verb, or adjective. A neural network then assigns the *correct* syntactic class to word (W), taking into consideration the syntactic classes of the preceding word (W-1) and the subsequent word (W+1). AutoTutor is capable of segmenting the learner input into a sequence of words and punctuation marks with 99%+ accuracy, of assigning alternative syntactic classes to words with 97% accuracy, and of assigning the correct syntactic class to a word (based on context) with 93% accuracy (Olde, Hoeffner, Chipman, Graesser, & the TRG, 1999).

The speech act classifier is a neural network that segments and classifies the learner's input into one of five speech act categories. The five categories are: Assertion, WH-question, YES/NO question, Directive, and Short Response. AutoTutor currently classifies 89% of the speech acts correctly. The Assertions are most relevant to our present implementation of LSA. Namely, LSA is used to assess the *quality* of a learner contribution once it has been classified as an Assertion. The LSA quality assessments play a critical role in determining the type of feedback and dialog move AutoTutor will generate next. AutoTutor uses different strategies for dealing with the other speech act categories: WH-question, YES/NO question, Directive, and Short Response. These strategies, which are needed for a smooth mixed-initiative dialog, will not be addressed in the present article.

Latent semantic analysis (LSA)

AutoTutor's knowledge about computer literacy is represented by Latent Semantic Analysis (LSA) (Foltz, 1996; Foltz, Britt, & Perfetti, 1996; Landauer & Dumais, 1997; Landauer, Foltz, & Laham, 1998). LSA is a statistical technique that compresses a large corpus of texts into K dimensions (usually 100 to 500). For AutoTutor, we performed LSA on 2.3 MB of texts. The texts included AutoTutor's curriculum script, two computer literacy textbooks, and 30 articles that discuss hardware, operating systems, and the Internet. We evaluated LSA's performance for dimension values ranging from 100 to 500; we adopted 200 for our current version of AutoTutor. The dimensions in LSA serve as orthogonal factors that are used to compute a conceptual relatedness score (a geometric cosine between 0 and 1) between any two "bags" of words. A "bag" must contain at least one word; however, there is no upper limit on the number of words a bag may contain. Thus, LSA computes a conceptual relatedness value between any two bags of words in which each bag contains one or more words.

In tutoring sessions, there are several parameters that must be constantly monitored by the tutor. These parameters include: (1) the quality of the learner's current Assertion, (2) how much of the topic being discussed has been covered, and (3) the learner's overall ability level for the topic material. AutoTutor is able to monitor these parameters by comparing various combinations of learner and tutor dialog contributions to specified conceptual bags. To assess the quality of a learner Assertion, LSA matches the learner Assertion against two separate conceptual bags, a bag that contains good answers versus a bag that contains the bad answers. The higher of the two LSA values is considered the best conceptual match, and therefore, determines how AutoTutor construes the learner's Assertion. For the domain of computer literacy, we have found our application of LSA to be quite accurate and economical in evaluating the quality of learner Assertions (Graesser, et al, in press; Wiemer-Hastings, Wiemer-Hastings, Graesser, and the TRG, 1999).

LSA also computes values for two additional parameters, topic coverage and student ability. The LSA topic coverage value is an index that reflects how much of the Ideal Answer has been covered in the tutoring dialog for a particular topic (e.g., why computers need peripherals). The topic coverage value considers the previous contributions of both the tutor *and* the learner against a conceptual bag that contains the Ideal Answer. The LSA student ability value is simply an index that reflects the student's ability level within a particular topic. Thus, only the previous student contributions are matched against the Ideal Answer bag. A set

of production rules that dictate AutoTutor's next move are based on predetermined values of the three LSA parameters described in this section, quality of learner contribution, topic coverage, and student ability. These production rules are outlined in the next section.

Dialog Move Generator

AutoTutor is designed to simulate the dialog moves of effective, normal human tutors. Ideally, we want AutoTutor to produce dialog moves that have pedagogical value, that are sensitive to the learner's abilities, and that fit the conversational context. AutoTutor currently has a repertoire of 12 dialog moves that are controlled by the dialog move generator (descriptions of these moves are provided in a previous section). They are pump, positive pump, hint, splice, prompt, elaborate, and summarize and five forms of immediate short-feedback (positive, positive-neutral, neutral, negative-neutral, and negative). These 12 dialog moves are generated in response to learner contributions that are classified as Assertions by the Speech Act Classifier. The learner Assertions receive special treatment for two reasons: (1) learner assertions are more diagnostic of student ability than are student questions (Person et al., 1995), and (2) learner assertions are more prevalent in tutoring dialogs than other speech acts, particularly questions (Graesser & Person, 1994). AutoTutor is equipped with mechanisms to handle the other speech act categories (WH- question, YES/NO question, Directive, and Short-Response). For example, in the case of WH- questions (e.g., "What does X mean?"), X is matched to the entries in a glossary and AutoTutor produces the definition if there is a high match. These other mechanisms are beyond the scope of this paper, and therefore, will not be addressed.

AutoTutor's dialog move generator is governed by 15 fuzzy production rules that primarily exploit data provided by the LSA module. Each fuzzy production rule specifies the parameter values in which a particular dialog move should be initiated. Thus, AutoTutor adopts a traditional production rule architecture except that the parameter values are evaluated by fuzzy matches (Kosko, 1992). The dialog move production rules are tuned to the following four parameters: (a) the quality of the learner's Assertions in the preceding turn, (b) the learner's ability level for the topic, (c) the extent to which the topic has been covered, and (d) student verbosity. The first three parameter values are computed by LSA, whereas the fourth (student verbosity) is simply a measure of how much (rather than how well) the student is contributing to the tutoring topic. AutoTutor's dialog move production rules are provided below.

PUMP

(1) IF [topic coverage = LOW or MEDIUM after learner's first Assertion] THEN [select PUMP]

(2) IF [match with good answer bag = MEDIUM or HIGH & topic coverage = LOW or MEDIUM] THEN [select PUMP]

POSITIVE PUMP

(3) IF [topic coverage = HIGH after learner's first Assertion] THEN [select POSITIVE PUMP]

SPLICE

(4) IF [student ability = LOW or MEDIUM & student verbosity = LOW or MEDIUM & topic coverage = LOW or MEDIUM & match with bad answer bag = HIGH] THEN [select SPLICE]

PROMPT

(5) IF [student verbosity = LOW & topic coverage = LOW or MEDIUM] THEN [select PROMPT]

HINT

(6) IF [student ability = MEDIUM or HIGH & match with good answer bag = LOW] THEN [select HINT]

(7) IF [student ability = LOW & student verbosity = HIGH & match with good answer bag = LOW] THEN [select HINT]

SUMMARY

(8) IF [topic coverage = HIGH or number of turns = HIGH] THEN [select SUMMARY]

ELABORATIONS

(9) IF [topic coverage = MEDIUM or SOMEWHAT HIGH] THEN [select ELABORATE]

POSITIVE FEEDBACK

(10) IF [match with good answer bag = HIGH or VERY HIGH] THEN [select POSITIVE FEEDBACK]

NEGATIVE FEEDBACK

(11) IF [match with bad answer bag = HIGH or VERY HIGH & topic coverage = MEDIUM or HIGH) THEN [select NEGATIVE FEEDBACK]

NEUTRAL FEEDBACK

(12) IF [match with good answer bag = MEDIUM or SOMEWHAT HIGH] THEN [select POSITIVE NEUTRAL FEEDBACK]

(13) IF [match with bad answer bag = SOMEWHAT HIGH] THEN [select NEGATIVE NEUTRAL FEEDBACK]

(14) IF [match with bad answer bag = HIGH or VERY HIGH & topic coverage = LOW) THEN [select NEGATIVE NEUTRAL FEEDBACK]

(15) IF [match with good answer bag = LOW or MEDIUM] THEN [select NEUTRAL FEEDBACK]

[NOTE: These are the dialog move production rules that currently exist in AutoTutor. They were revised over the course of three evaluation cycles.]

In order to make sense of these production rules, the somewhat generic LSA values (e.g., LOW, MEDIUM, HIGH) need further elaboration. Recall that the LSA values are geometric cosines that range between 0 and 1, where higher values indicate a greater conceptual match. The generic values specified in the production rules (e.g., LOW, MEDIUM, HIGH) correspond to a range of arbitrarily determined LSA values. In AutoTutor, a HIGH value typically corresponds to LSA values that range between .5 and 1.0, whereas the MEDIUM value typically corresponds to values between .25 and .75. The overlap between the LSA values (e.g., MEDIUM and HIGH) is an inherent feature of the fuzzy logic (see Kosko, 1992). AutoTutor's dialog moves were evaluated in three different cycles. The range of LSA values associated with

each of the LSA parameters were slightly adjusted after each evaluation cycle to maximize AutoTutor's performance.

Some of the dialog moves have more than one production rule. This is the case because some dialog moves serve more than one pedagogical function. Consider the two Hint production rules. In Rule (6), a medium or high ability student has wandered off track and produced an Assertion that is low in quality. This Hint rule is designed to steer the student back on the right course. In Rule (7), a verbose, low ability student has produced an Assertion that is low in quality. This particular rules gives the verbose student a second chance to produce a high quality contribution before the tutor intervenes with the correct information. Thus, this Hint rule places the onus on the student to produce a high quality Assertion rather than on the tutor (which would be a pedagogically inferior strategy).

EVALUATING AUTOTUTOR'S PERFORMANCE

We assessed AutoTutor's performance as an effective tutor and conversational partner in three evaluation cycles. The purpose of the evaluation cycles was to identify and correct particular dialog move problems before AutoTutor's debut with human learners. Several virtual students were created to emulate human students of varying ability and verbosity levels. The use of virtual (or synthetic) students to test tutoring systems is not uncommon and has been advocated by other researchers (Ur & VanLehn, 1995; VanLehn, Ohlsson, & Nason, 1994). Experts in language and pedagogy rated the pedagogical effectiveness and the conversational appropriateness of AutoTutor's dialog moves in the tutoring sessions with the virtual students. After each evaluation cycle, the curriculum script, the fuzzy production rules, and the LSA parameter thresholds were revised to enhance AutoTutor's overall performance.

The Virtual Students

To evaluate AutoTutor's effectiveness during the development phases, we created different classes of virtual students. Each of the virtual students differed in terms of ability level and/or discourse style. To create the virtual students, the 36 topic questions in the curriculum script were presented to approximately 100 human students enrolled in a computer literacy course. Knowledgeable judges then rated the quality of the students' answers to each of the 36 topic questions. The following virtual students were created for each of the 36 topics in the curriculum script:

1. Good verbose student. The first 5 turns of this virtual student had 2 or 3 Assertions that human experts had rated as good Assertions from the human sample. The student is regarded as verbose because the student has 2 or 3 Assertions within one turn, which is more than the average number of Assertions per turn in human tutoring.

2. Good succinct student. The first 5 turns of this virtual student had 1 Assertion that human experts had rated as a good Assertion.

3. Vague student. The first 5 turns of this virtual student had an Assertion that had been rated as vague (neither good nor bad) by the human experts.

4. Erroneous student. The first 5 turns of this virtual student had an Assertion that contained a misconception or bug according to human experts.

5. Mute student. The first 5 turns of this virtual student had semantically depleted content, such as "Well", "Okay", "I see", and "Uh".

6. Good coherent student. The first 5 turns of this virtual student had 1 Assertion that had been rated as good. However, unlike the other two Good virtual students, all of the Assertions in the first 5 turns for a particular topic were provided by *one* human student.

7. Monte Carlo student. The first 5 turns of this virtual student were generated in a Monte Carlo fashion to simulate the variability of student Assertion quality that typically occurs human tutoring sessions. That is, all classes of Assertions (e.g., good and vague) were represented.

For AutoTutor to be an effective tutor, he must be able to: (1) discriminate learner ability and Assertion quality, and (2) respond with one or a combination of appropriate dialog moves. Graesser et al. (in press) reported that the LSA parameters are sensitive to learner ability and Assertion quality. The purpose of the three evaluation cycles reported in this article was to see whether AutoTutor generates pedagogically effective dialog moves that are sensitive to such differences.

.

The Judges and the Quality Dimensions

Four judges were selected to rate the quality of AutoTutor's dialog moves on two holistic dimensions, pedagogical effectiveness (PE) and conversational appropriateness (CA). Two judges were assigned to each dimension. The two judges who rated PE were quite knowledgeable of the pedagogical strategies that are frequently employed by normal human tutors. For each AutoTutor dialog move, the PE judges considered: (1) whether the dialog was pedagogically effective, and (2) whether the dialog move was reasonable for a *normal* human tutor. The two judges who rated CA had extensive of knowledgeable of conversational discourse. The CA judges considered several factors relevant to conversation in their holistic rating of each AutoTutor dialog move. These factors included politeness norms along with the Gricean maxims of quality, quantity, relevance, and manner (Brown & Levinson, 1987; Grice, 1975, 1978). Both PE and CA were rated on a six-point scale, where 1 reflected a very low quality rating and 6 reflected a very high quality rating. Interjudge reliability measures were computed for both pairs of judges. Results indicated significant reliability between judges for both dimensions (Cronbach's alpha = .94 for PE and .89 for CA).

The Three Evaluation Cycles

Cycle 1

Tutoring transcripts were created for five of the virtual students described above, Good Verbose, Good Succinct, Vague, Mute, and Erroneous. The Good Coherent and Monte Carlo students were not created until the second evaluation cycle. Given that this was AutoTutor's first time to interact with learners, we did not view Cycle 1 as a full-fledged evaluation of AutoTutor's tutoring and conversational skills. Each transcript was rather lengthy (approximately 25 pages); and we were not particularly confident that our knowledge of normal human tutors was accurately represented in the fuzzy production rules. Thus, the PE and CA judges were not required to provide ratings for every AutoTutor dialog move.

The two sets of judges provided PE and CA ratings for AutoTutor's third turn in each of the 36 curriculum script topics. The mean Pedagogical Effective ratings are reported for each of the virtual students in Table 1, whereas the Conversational Appropriateness ratings are reported in Table 2. The results from the Cycle 1 ratings indicated two things. First, AutoTutor's performance ratings were inversely related to the substantive Assertions of the virtual students. That is, AutoTutor performed best with students who said very little, namely, the Vague and Mute students. Second, AutoTutor's overall performance as a tutor and conversational partner could withstand considerable improvement.

Table 1.
Means for Pedagogical Effectiveness Quality Ratings

	Cycle 3			Cycle 2			Cycle 1		
Virtual Student Type	*M*	*SD*	*n*	*M*	*SD*	*n*	*M*	*SD*	*n*
Good Verbose	4.52	1.41	210				2.52	1.49	36
Good Succinct	4.63	1.42	283				3.55	1.78	36
Vague	4.12	1.52	338				3.90	1.65	36
Erroneous	3.30	1.59	301				3.53	1.69	36
Mute	3.75	1.47	592				3.63	1.62	36
Good Coherent	5.26	1.26	286	4.40	1.64	307			
Monte Carlo	4.03	1.77	273	4.09	1.67	298			
Overall	**4.23**	**1.47**	**2283**	**4.25**	**1.66**	**605**	**3.46**	**1.66**	**180**

After reviewing the Cycle 1 data, a few minor changes were made to the dialog move production rules and the LSA parameter thresholds. We were somewhat hesitant, however, to make substantial changes to the rules and LSA parameters since only one dialog move in each topic was evaluated. In addition, several of the dialog moves had very low frequencies and some did not occur at all (i.e., negative feedback). We decided to forego any major adjustments to AutoTutor until we collected more representative data.

Cycle 2

Two new virtual students were created for the Cycle 2 evaluation, a Good Coherent Student and a Monte Carlo Student. These two students were designed to be more representative of the student contributions that occur in human tutoring sessions. The Good Coherent Student was created to emulate a good student who provides relatively coherent contributions that "hang together" over several conversational turns. The Coherent student differs from the other two good students (i.e., Good Verbose and Good Succinct) in that the first 5 Assertions in a topic are provided by the same student. The Monte Carlo student was designed to reflect the variability of student Assertion quality that typically occurs human tutoring sessions. All classes of Assertions (e.g., good, bad, and vague) were generated for the Monte Carlo student.

The second evaluation cycle differed from Cycle 1 in that the judges provided PE and CA ratings for *every* AutoTutor dialog move (605 total) in the Good Coherent and Monte Carlo transcripts. Results from this evaluation are reported in Tables 1 and 2. The overall means for both PE (4.25) and CA (4.97) indicate that AutoTutor's performance improved considerably between Cycle 1 and Cycle 2. However, it is unclear whether the improvement was due to the slight changes that were made to the production rules and LSA parameters after Cycle 1 or whether it could be attributed to the more representative sample of dialog moves.

After the Cycle 2 evaluation, AutoTutor underwent a number of substantial changes. First, the content in the curriculum script was revised. The introduction sections for the 36 topics were rewritten so that AutoTutor's sentences were shorter and more conversational. In addition, all of AutoTutor's dialog moves were assigned discourse markers and rewritten to sound more conversational. Second, changes were made to the dialog move production rules. A Positive Pump production rule was added and the LSA values in other rules were adjusted. For example,

when we examined the mean ratings and frequencies for each dialog move category, we noticed that AutoTutor was generating too many Pumps and rarely generated Negative Feedback, Neutral Negative Feedback, Neutral Positive Feedback, or Splices (even when it would have been pedagogically advantageous to do so). We adjusted the LSA values in particular production rules with hopes that the changes would be manifested in the next evaluation cycle ratings.

Table 2.
Means for Conversational Appropriateness Quality Ratings

	Cycle 3			Cycle 2			Cycle 1		
Virtual Student Type	*M*	*SD*	*n*	*M*	*SD*	*n*	*M*	*SD*	*n*
Good Verbose	4.77	1.52	210				2.59	1.24	36
Good Succinct	4.48	1.17	283				3.65	1.30	36
Vague	4.47	1.24	338				4.02	1.38	36
Erroneous	3.97	1.59	301				3.78	1.36	36
Mute	4.52	1.20	592				4.92	1.30	36
Good Coherent	4.86	1.29	286	4.91	1.22	307			
Monte Carlo	4.39	1.41	273	5.04	1.35	298			
Overall	**4.49**	**1.34**	**2283**	**4.97**	**1.28**	**605**	**3.79**	**1.45**	**180**

Cycle 3

All seven virtual students were included in the Cycle 3 evaluation. The two sets of judges assigned PE and CA ratings to every AutoTutor dialog move (2283 total moves) in the seven tutoring transcripts. The results are reported in Tables 1 and 2. The Overall Means are not particularly encouraging in that AutoTutor's performance decreased on the CA dimension and remained roughly the same for the PE dimension. However, the results were somewhat promising when the cycle means for the different virtual students were considered. For example, AutoTutor showed substantial improvement between Cycle 1 and Cycle 3 for the Good Verbose and Good Succinct students. In the majority of other cycle comparisons, the virtual student means reflected marginal improvement.

There are certainly a number of plausible explanations for why the changes made to AutoTutor after Cycle 2 did not result in higher performance ratings on the PE and CA dimensions. We suspect, however, that the primary problem had more to do with the virtual students than with AutoTutor. Anecdotal reports from the judges indicated that they were more frustrated with the virtual student Assertions than with AutoTutor's dialog move generation. Recall that the Assertions for each of the virtual students (except the Good Coherent student) were provided by different students enrolled in the computer literacy course. Hence, the learner Assertions for the virtual student were often redundant and/or lacked global coherence.

CONCLUDING REMARKS

We have tried to highlight the discourse features of AutoTutor that distinguish this system from other animated pedagogical agents. Of course, we recognize that AutoTutor still needs to hone some of his tutoring and conversational skills; however, we are certain that he embodies many of the pedagogical and discourse features of the next generation of computer tutors. Unlike the majority of other pedagogical agents, AutoTutor's understanding of natural language input is not restricted to a subset of possible student speech acts (e.g., the "Why?", "Hint", "Show" options in Adele (Shaw, Johnson, Ganeshan, 1999), the follow-on questions in the PPP Persona (André, Rist, & Müeller, 1998), or the limited set of utterances that are understood by Gandalf (Cassell & Thórisson, 1999). AutoTutor not only understands the natural language input provided by learners, but also responds with dialog moves that approach those of effective human tutors.

We have recently begun a fourth evaluation cycle in which AutoTutor interacts with human learners. This evaluation cycle will close the chapter on this version of AutoTutor. Our next project, AutoTutor2, is currently being developed. Some of the features of AutoTutor2 include: a back-channel feedback module, a three-dimensional agent that is capable of displaying complex emotions, and curriculum script topics that are organized in conceptual knowledge structures that will enable AutoTutor2 to make more sophisticated dialog move choices.

ACKNOWLEDGEMENTS

This research was funded by the National Science Foundation (SBR 9720314) in a grant awarded to the Tutoring Research Group (Art Graesser, Principal Investigator). The following members of the Tutoring Research Group at the University of Memphis conducted research on this project: Ashraf Anwar, Myles Bogner, Tim Brogdon, Patrick Chipman, Scotty Craig, Rachel DiPaolo, Stan Franklin, Max Garzon, Barry Gholson, Art Graesser, Doug Hacker, Peggy Halde, Derek Harter, Jim Hoeffner, Xiangen Hu, Jeff Janover, Bianca Klettke, Roger Kreuz, Kristen Link, William Marks, Johanna Marineau, Lee McCaulley, Brent Olde, Natalie Person, Victoria Pomeroy, Matt Weeks, Shannon Whitten, Katja Wiemer-Hastings, Peter Wiemer-Hastings, Shonijie Yang, Holly Yetman, and Zhaohua Zhang. Additional thanks to Eric Mathews. Requests for copies should be sent to Natalie Person, Department of Psychology, Rhodes College, 2000 N. Parkway, Memphis, TN 38112, person@rhodes.edu.

REFERENCES

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.

André, E., Rist, T., & Müller, J. (1998). Integrating reactive and scripted behaviors in a life-like presentation agent. *Proceedings of the Second International Conference on Autonomous Agents* (pp. 261-268). Minneapolis-St.Paul, MN.

Beekman, G. (1997). *Computer confluence*. New York: Benjamin/Cummings.

Bloom, B. S. (1956). *Taxonomy of educational objectives: The classification of educational goals. Handbook I: Cognitive domain*. New York: McKay.

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.

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.

Brown, P., & Levinson, S. C. (1987). *Politeness: Some universals in language use*. Cambridge: Cambridge University Press.

Cassell, J., & Thórisson, 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.

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

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

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.

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.

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.

Foltz, P.W. (1996). Latent semantic analysis for text-based research. *Behavior Research Methods, Instruments, and Computers, 28,* 197-202.

Foltz, P. W., Britt, M. A., & Perfetti, C. A. (1996). Reasoning from multiple texts: An automatic analysis of readers' situation models. *Proceedings of the 18th Annual Conference of the Cognitive Science Society* (pp. 110-115). Mahwah, NJ: Erlbaum.

Fox, B. (1993). *The human tutorial dialog project*. Hillsdale, NJ: Erlbaum.

Freedman, R. (1999). Atlas: A plan manager for mixed-initiative, multimodal dialogue. AAAI '99 Workshop on Mixed-Initiative Intelligence, Orlando.

Gagné, R. M. (1977). *The conditions of learning* (3rd ed.). New York: Holdt, Rinehart, & Winston.

Graesser, A. C., & Person, N.K. (1994). Question asking during tutoring. *American Educational Research Journal, 31,* 104-137.

Graesser, A.C., Franklin, S., & Wiemer-Hastings, P. and the Tutoring Research Group (1998). Simulating smooth tutorial dialog with pedagogical value. *Proceedings of the American Association for Artificial Intelligence* (pp. 163-167). Menlo Park, CA: AAAI Press.

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.

Graesser, A.C., Wiemer-Hastings, P., Wiemer-Hastings, K., Harter, D., Person, N., and the Tutoring Research Group (in press). Using latent semantic analysis to evaluate the contributions of students in AutoTutor. *Interactive Learning Environment.s*

Grice, H. P. (1975). Logic and conversation. In P. Cole & J. Morgan, (Eds.), *Syntax and semantics, vol. 3: Speech acts* (pp. 41-58). New York: Academic Press.

Grice, H. P. (1978). Further notes on logic and conversation. In P. Cole (Ed.*), Syntax and semantics, vol. 9; Pragmatics* (pp. 113-127).

Hu, X., Graesser, A. C., and the Tutoring Research Group (1998). Using WordNet and latent semantic analysis to evaluate the conversational contributions of learners in the tutorial dialog. *Proceedings of the International Conference on Computers in Education*, Vol. 2, (pp. 337-341). Beijing, China: Springer.

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.

Kosko, B. (1992). *Neural networks and fuzzy systems.* New York: Prentice Hall.

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.*

Landauer, T. K., Foltz, P. W., Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes, 25,* 259-284.

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

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.

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

McCauley, L., Gholson, B., Hu, X., Graesser, A. C., and the Tutoring Research Group (1998). Delivering smooth tutorial dialog using a talking head. *Proceedings of the Workshop on Embodied Conversation Characters* (pp. 31-38). Tahoe City, CA: AAAI and ACM.

Mehan, H. (1979). *Learning lessons: Social organization in the classroom.* Cambridge, MA: Harvard University Press.

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.

Moore, J.D. (1995). Participating in explanatory dialogues. Cambridge, MA: MIT Press.

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.

Palinscar, A. S., & Brown, A. (1984). Reciprocal teaching of comprehension-fostering and comprehension-monitoring activities. *Cognition & Instruction, 1,* 117-175.

Person, N. K., Graesser, A. C., Magliano, J. P., & Kreuz, R. J. (1994). Inferring what the student knows in one-to-one tutoring: The role of student questions and answers. *Learning and Individual Differences, 6,* 205-29.

Person, N. K., Klettke, B., Link, K., Kreuz, R. J., and the Tutoring Research Group (1999). The integration of affective responses into AutoTutor. *Proceeding of the International Workshop on Affect in Interactions* (pp. 167-178). Siena, Italy.

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.

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.

Pressley, M., Wood, E., Woloshyn, V. E., Martin, V., King, A., & Menk, D. (1992). Encouraging mindful use of prior knowledge: Attempting to construct explanatory answers facilitates learning. *Educational Psychologist, 27,* 91-110.

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.

Rose, C. P., Di Eugenio, B., & Moore, J. D. (1999). A dialogue based tutoring system for basic electricity and electronics. In S. P. Lajoie & M. Vivet (Eds.), *Artificial Intelligence in Education (Proceedings of AI-ED '99, Le Mans)* (pp. 759–761). Amsterdam: IOS Press

Shaw, E., Johnson, W.L., & Ganeshan, R. (1999). Pedagogical agents on the web. *Proceedings of the Ninth International Conference on Artificial Intelligence*. IOS Press.

Ur, S. & VanLehn, K. (1995) Steps: A simulated, tutorable physics student. *Journal of Artificial Intelligence in Education, 6*(4), pp. 405-437.

VanLehn, K. (1990). *Mind bugs: The origins of procedural misconceptions.* Cambridge, MA: MIT Press.

VanLehn, K., Ohlsson, S. & Nason, R. (1994). Applications of simulated students: An exploration. *Journal of Artificial Intelligence in Education, 5*(2), pp. 135-175.

Wakefield, J. F. (1996). *Educational psychology: Learning to be a problem solver*. Boston: Houghton Mifflin.

Webb, N. M., Troper, J. D., & Fall, R. (1995). Constructive activity and learning in collaborative small groups. *Journal of Educational Psychology, 87,* 406-423.

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.

Wiemer-Hastings, P., Wiemer-Hastings, K., and Graesser, A. (1999). Improving an intelligent tutor's comprehension of students with Latent Semantic Analysis. *Artificial Intelligence in Education* (pp. 535-542). Amsterdam: IOS Press.

Zhou, Y., Freedman, R., Glass, M., Michael, J. A., Rovick, A. A., Evens, M. W. (1999a). Delivering hints in a dialogue-based intelligent tutoring system. In Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI '99), Orlando, FL.

APPENDIX

Curriculum Script
Macrotopic: Computer Hardware
Topic: Adding additional RAM increases computer's overall performance
Conceptual Difficulty Level: Medium
Topic Format: Problem + Question

Topic Introduction, Problem + Question (delivered by AutoTutor)
Suppose that you want to get a business administration minor, and that you have to work with a statistics program named BusinessStat. The instructions to install the program say that you need a 486 50 computer

with 32 megabytes of RAM. You have a 486 50 computer, but only 8 megabytes of RAM. You decide that you want to upgrade your computer so you can run BusinessStat. Consider this problem. How will you upgrade your computer? How will your computer's performance for running other programs be affected?

Ideal Answer
To run BusinessStat, you need to add RAM to your computer or increase virtual memory. Adding RAM or increasing virtual memory will increase the computer's overall performance. Adding memory allows larger programs to be executed. Adding memory allows larger amounts of data to be manipulated. Adding memory also allows several programs to run simultaneously. The CPU can execute only one instruction of a program at a time and uses RAM and virtual memory to store the rest of the program and data until the CPU is going to use them. With greater amounts of memory, the CPU can store large programs and greater amounts of data.

Ideal Good Answer Aspects
1. To run BusinessStat, you need to add RAM to your computer or increase virtual memory.
2. Adding RAM or increasing virtual memory will increase the computer's overall performance.
3. Adding RAM memory allows larger programs to be executed.
4. Additional memory allows more data to be manipulated.
5. Additional memory also allows several programs to run simultaneously or at the same time.
6. The CPU can execute only one instruction of a program at a time and uses RAM and virtual memory to store the rest of the program and data until the CPU is going to use them.
7. With greater amounts of memory, the CPU can store larger programs and greater amounts of data.

Other possible Good Answers that were included in the LSA text compression. (These answers were supplied by human students enrolled the computer literacy course and were rated as "good answers" by experts).
1. To run BusinessStat at home, I need to add more RAM or virtual memory. This will most likely increase my computer's performance so programs which take up more memory can be run.
2. To run BusinessStat, I need to add more RAM or virtual memory. Adding RAM or virtual memory will most likely increase my computer's performance. I'll now be able to run programs that take up more memory or run additional smaller memory usage programs at the same time.
3. To run BusinessStat at home, I need to add more RAM or virtual memory. Adding RAM or virtual memory will most likely increase my computer's performance. Large amounts of RAM or virtual memory allow programs which take up a lot of memory to run. Adding memory allows a large amount of data to be dealt with at one time. This is true as the CPU uses RAM as temporary storage. Normally, the computer can perform more tasks the more the CPU can store in RAM or virtual memory. For example, several programs can be executed at once.
4. In order to make BusinessStat run, one has to add more RAM or virtual memory. More RAM will help the CPU to run larger programs. Adding memory will increase the computer's overall performance.
5. RAM or virtual memory has to be added to the computer in order to make BusinessStat run. Adding memory will lead to a greater amount of data and programs that can be stored.
6. BusinessStat will run if RAM or virtual memory is added to the computer. More RAM or virtual memory will lead to an increased performance of the computer. Larger programs will be able to run. A greater number of smaller programs will be able to run simultaneously. The CPU can store more programs and data.
7. You will buy more RAM that will increase your computer's speed.
8. The solution would be to purchase more RAM. Adding RAM will make your computer faster and able to run many more programs in the future.
9. You will need to add more RAM to your memory, which will increase the speed of future programs as well as allow you to use programs that require more RAM.
10. I suppose you would have upgraded your available RAM, therefore, your computer will run faster with future programs you might be using.
11. RAM can easily and inexpensively be increased by adding SIMMS or DIMS to the current memory. An increase in RAM results in an increase in speed while running software.
12. Adding more RAM will increase the computer's ability to store and use the information found in BusinessStat and other programs that demand higher amounts of available RAM.
13. I would add more RAM to my computer, and it will allow me to run BusinessStat and increase the performance of my computer because the computer will not have to run in virtual memory.

14. If the requirement is 486/25 and your machine has 486/50, you can run this program. Since it also stated 32 MB of RAM required, all you have to do is to increase your memory by adding a SIMM's chip. Otherwise, the computer will run a little slow when you running this software.
15. I will have to buy more RAM. This solution will enable me to run other programs in the future that also require a higher amount of memory.

Bad Answers

1. With less RAM, you can still store the program on a floppy disk.
2. For more storage space in RAM, remove old files or programs.
3. The computer is fast enough to run the program.
4. The less memory, the slower the execution of the programs.
5. Adding long term storage will allow the program to run.
6. The speed of the computer will allow the program to run.
7. You need a new computer with more memory to run the program.
8. More memory only affects BusinessStat.

Content-related Dialog Moves that are generated by AutoTutor

SPLICES for Bad Answers

1. Storing the program on a floppy disk will not help you run the program.
2. If you add RAM or virtual memory, you will not have to dump current programs.
3. A fast computer chip will not solve your need to have a larger immediate memory.
4. Adding RAM will not affect the processing speed of your computer. But it will increase the overall performance of executing the programs.
5. A zip drive will add long term storage capacity, but not immediate memory capacity.
6. You do not have enough RAM to use BusinessStat.
7. You can add more RAM and still use your old computer.
8. Adding more RAM will improve the overall performance of your computer on all programs.

ELABORATIONS

1. You can run BusinessStat if you add RAM.
2. More RAM increases the computer's overall performance.
3. With additional RAM, you can handle larger programs.
4. More data can be manipulated if there is more RAM.
5. A larger RAM can allow you to run several programs at the same time.
6. The CPU runs one instruction at a time, and RAM stores the rest.
7. If you add RAM, the CPU can handle larger programs and more data.

HINTS

1. Consider your needs to run BusinessStat.
2. What about the general effects of adding RAM or virtual memory?
3. What about the size of the programs you need to run?
4. What about the amount of data that can be manipulated?
5. What about concurrent processing of several programs?
6. What about RAM and the instruction set in the CPU?
7. The capability of the CPU must somehow be influenced by RAM.

PROMPTS and PROMPT RESPONSES generated by AutoTutor

1. To run BusinessStat, you need to add _____? RESPONSE: To add RAM.
2. By adding RAM, the overall performance of your computer will _____? RESPONSE: Will increase or improve.
3. When you add more RAM, the size of your programs can be _____? RESPONSE: Can be larger.
4. When you add more RAM, more of your data can be actively _____? RESPONSE: Can be actively manipulated or processed.
5. Adding RAM allows you to run several programs _____? RESPONSE: Run several programs at the same time.
6. How many instructions can be run at a time in the CPU? RESPONSE: Only one instruction.
7. If you add RAM, the CPU can store more data and larger _____? RESPONSE: And larger programs.

SUMMARY
You need to increase RAM to run BusinessStat. The additional RAM allows you to run larger programs, and manipulate larger amounts of data. With enough RAM, you can run several programs at once.

Possible WH- definitional questions + AutoTutor's canned Answers

1. If WH- question contains: 486/25, ANSWER: 486 refers to the name of an Intel chip common to IBM compatible computers in the early to mid 1990s. 25 refers to 25 megahertz, the particular speed of this computer's chip.
2. If WH- question contains: 486/50, ANSWER: 486 refers to the name of an Intel chip common to IBM compatible computers in the early to mid 1990s. 50 refers to 50 megahertz, the particular speed of this computer's chip.
3. If WH- question contains: MB, ANSWER: MB refers to megabytes. Megabytes is a way of stating the size of computer memory.
4. If WH- question contains: RAM, ANSWER: RAM stands for Random Access Memory. Random Access Memory is one of two types of memory found in a computer system, the other being ROM. RAM is both readable and writeable by the CPU. RAM's contents disappear when a program stops or the computer is turned off.
5. If WH- question contains: Virtual memory, ANSWER: Virtual memory is space set aside on the hard disk by the operating system which appears as RAM to the CPU.