ELSEVIER

Computers & Education 42 (2004) 375–388

COMPUTERS & EDUCATION

www.elsevier.com/locate/compedu

A framework of synthesizing tutoring conversation capability with web-based distance education courseware

Ki-Sang Song[a,*], Xiangen Hu[b], Andrew Olney[c], Arthur C. Graesser[b], Tutoring Research Group[b]

[a]*Department of Computer Education, Korea National University of Education, Chungbuk, 363-791, South Korea*
[b]*Department of Psychology, University of Memphis, Memphis, TN 38152, USA*
[c]*Department of Computer Science, University of Memphis, Memphis, TN 38152, USA*

Received 22 April 2003; accepted 2 September 2003

Abstract

Whereas existing learning environments on the Web lack high level interactivity, we have developed a human tutor-like tutorial conversation system for the Web that enhances educational courseware through mixed-initiative dialog with natural language processing. The conversational tutoring agent is composed of an animated tutor, a Latent Semantic Analysis (LSA) module, a database with curriculum scripts, and a dialog manager. As in the case of human tutors, the meaning of learner's contributions in natural language are compared with the content of expected answers to questions or problems specified in curriculum scripts. LSA is used to evaluate the conceptual matches between learner input and tutor expectations, whereas the dialog manager determines how the tutor adaptively responds to the learner by selecting content from the curriculum script. The integration of available courseware with the tutorial dialog system guarantees the reusability of existing Web tutorials with minimal effort in the modification of the curriculum script and LSA module. This development thereby simplifies the change into more valuable Web based training courseware.

Keywords: Distance education and telelearning; Distributed learning environments; Intelligent tutoring systems; Interactive learning environments; Multimedia/hypermedia system

* Corresponding author.

0360-1315/$ - see front matter © 2003 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compedu.2003.09.002

1. Introduction

Using advanced network technologies, the connection of learners with distributed learning resources in Web-based distance education is easier than traditional distance education environments. The World Wide Web (WWW) and Web browsers provide a more user-friendly and user-centered environment for Internet users, as well as easy creation of multimedia content that includes graphics, text, video clips, animation, and JAVA applets. Individuals and organizations can create home pages independently and can link to other sites to share information.

WWW applications have also provided educators in distance teaching and learning areas with exciting new opportunities that presumably improve the quality teaching and that are convenient and cost-effective for both learners and education providers. Convenient ways to distribute learning resources outside of classes include the use of information from the Web that had been used in the classroom and the creation of a classroom homepage that covers information about the class, the syllabus, lecture notes, exercises and references

In Web-based distance education, the teacher and students are separated spatially, but nevertheless connected in a virtual community through the WWW. These resulting freedoms and constraints make it important for researchers and teachers to carefully consider what course materials and environments promote effective learning. The materials, which range from lecture notes to deliberately designed courseware, typically serve as an online supplement to the classroom. Unfortunately, however, many of the Web-based educational offerings provide poor learning opportunities. They are too often merely the ''translation'' of books and lectures into an electronic format (Bork, 1996) or analogous to simply ''turning the pages'' in a textbook (Schank, 1998). The reason that learning modules have not achieved their full educational potential is that mere delivery of information to a user does not guarantee learning.

The Web and multimedia authoring tools that are commercially available use HTML, DHTML or XML techniques that support easy creation of Web contents. These advances allow with more interactivity, but they still lack the sophistication required to build intelligent tutors (Murray, 1999). For example, current Web-based tutorials with embedded Macromedia Flash animation and Shockwave multimedia are fashionable, but such an impressive display does not necessarily have a positive impact on learning. Moreover, the information organization style in Web courseware has been changed from the old frame structure to a user controllable hypermedia structure. Therefore, the lack of adaptation and intelligence to meet individualized learner needs and characteristics is forcing researchers to consider different approaches in developing courseware that facilitates the learning process.

More than a dozen ITS (Intelligent Tutoring System) authoring systems have been developed to provide intelligence to tutoring materials (Murray, 1999). Some of these authoring systems have facilities to integrate hypermedia with intelligent tutoring systems (Brusilovsky, Schwartz, & Weber, 1996; Kiyama, Ishiuchi, Ikeda, Tsujimoto, & Fukuhara, 1997). Most of the developed ITS authoring tools focus on the implementation of generic modules of an ITS, such as a Domain model, a Tutoring model, and a Student model, but are far beyond being applied for general use.

The Domain model of an ITS contains the knowledge to be taught by the ITS, normally in the form of conceptual modules that have been popular in artificial intelligence and cognitive science. These modules include production systems, semantic networks, structured case scenarios, and procedural representations. One challenge in applying AI techniques to ITS's is the assessment of

student contributions when they respond to problems and questions. In particular, using natural language processing to communicate with learner is substantially more difficult than with conventional techniques and interfaces. Moreover, authoring an ITS requires knowledge of production rule systems, cognitive modelling of students, system design and implementation. These activities are very time consuming.

Nevertheless, we are convinced that educators who practice their professional teaching may not require all the components of a generic ITS in order to create Web-based courses. It is absolutely crucial that the learning technologies needed to develop courseware should not create an obstacle for those in the educational field. Therefore, the creation of Web courseware that is better than the mere delivery of information and that fully uses the advantage of available Web editors requires a different approach than a conventional ITS authoring system.

Foremost among the frustrations experienced by students in distance education is the lack of prompt feedback from the instructor to the student. Since Web-based education is based on multimedia-based asynchronous courses, prompt unambiguous feedback is much more difficult than in face-to-face conditions (Hara & Kling, 2000). Also, instant feedback requires that the instructor is ready to give feedback at any time, which is an unrealistic demand.

With these considerations in mind, we introduce a framework for augmenting Web-based courseware with the capability of conversational tutoring. Conversational tutoring is expected to facilitate the learning process by providing quick, responsive, intelligent feedback, but without a human instructor. The questions, feedback, and other dialog moves of the computer tutor are similar to what a human tutor does. That is, we develop a human-like conversational tutoring agent that effectively communicates with the student through an interface with mixed-initiative dialog and natural language processing. Our system can also be easily integrated with existing Web courseware or educational materials, thereby allowing reusability of existing courseware.

The rest of this paper has four sections. We describe some issues of facilitating the learning process in distance education courseware in Section 2. The proposed system architecture is described in Section 3, whereas some of its operational issues are in Section 4. Finally we end with some conclusions in Section 5.

2. Issues for facilitating learning processes in distance education courseware

Courseware on the Web is mostly created using HTML pages with multimedia add-ons, such as animation, audio, video clips and Java applets. These pages afford some interactivity between learner and courseware. By changing parameters and manipulating buttons in the pages, a learner may initiate learning process and construct knowledge under a dynamic simulation environment. However, most learners are prone to explore unproductively when left to their own control unless there is a more thoughtful design of the interactivity, especially for a distance education environment. In addition to adding multimedia components to the pages, it is necessary to sustain the learner's attention through absorbing presentations, facilities for student-initiated questions, and challenges for the learner to think independently and deeply. Without these, many learners will not be sufficiently involved in the learning process. In interactive courseware design, comments and questions about material just presented may prevent the courseware from becoming monotonous to the learner. The selection of comments and questions is critical. Instead of questions

which require only memorizing and reciting answers, questions that invite deep reasoning and that link to what students already know or have learned are preferred. That is, the scaffolding of knowledge with appropriate questions allows learning through discovery; this promotes understanding and retention because new knowledge is linked to existing knowledge (Thibodeau, 1997).

Giving appropriate questions and positive feedback according to individualized learner characteristics requires that mentor roles be implemented in the courseware. Implementing these features in typical tutorial programs is not possible by simply using commercially available Web authoring tools. The processes of judging the quality of the user responses or providing effective feedback through hints or remediation requires specialized types of tutoring. Such characteristics are discussed below.

2.1. Communication mode

The most effective form of instruction known is provided by human tutors (Cohen, Kulik, & Kulik, 1982). It has also been proposed that the primary difference between human tutors and tutoring systems is their communication mode: human tutors use natural language (Graesser, Person, & Magliano, 1995). Although some Web tutorials print natural language and sometimes speak it, they do not let the student type or speak in natural language. Since student generation of explanations is known to increase learning (Chi, Siler, Jeong, Yamauchi, & Hausmann, 2001), current Web-based tutorial systems would benefit from this style of interactivity.

2.2. Initiatives in conversation

The initiation of conversation between Web tutorials and learners is believed to be an important feature of conversational learning environments. Most tutorial systems generate questions and get short-answer responses from the user. However, in the SCHOLAR system (Carbonell, 1970), CIRCSIM-Tutor (Evens et al., 2001), ATLAS (Freedman, 1999), and AutoTutor (Graesser, Person, Harter, & TRG, 2001), mixed-initiative dialogue is provided to enhance user interactivity. Mixed-initiative dialogue allows the student to ''take initiative'' by selecting problems and asking questions, thereby allowing the student more control in their knowledge construction.

2.3. Flexibility in question and answers

Another issue addresses the canned presentation of information, canned problems, and canned answers during the learning session. Providing different feedback with respect to the question's context makes the user an active participant in the learning process. Keeping learners in the learning process necessitates that the system be as flexible as possible. Ideally, a different response from the tutor will occur when the student expresses the same information on different occasions.

2.4. Real-time interactivity

The retention rate of the trainees in distance education has been raised from about 20 percent (using ordinary classroom methods) to about 75 by introducing real-time interactivity in distance

education (Millbank, 1994). Enhancement of independent learning materials through the use of interactive communication technologies and teacher mediation are also vital to the completion/success rate of distance education. Just as a human tutor can provide real-time interactivity, we propose implementing the same process in independent learning courses.

3. Embodying conversational tutoring capability in Web courseware

A well-structured Web-based distance education course needs to provide coaching at critical times and the scaffolding of support due to the separation of instructor and learner. Web courseware are ideally designed with following instructional components: (1) motivating the learner, (2) explaining what is to be learned, (3) helping the learner recall previous knowledge, (4) providing instructional material, (5) providing guidance and feedback, (6) testing comprehension, and (7) providing enrichment or remediation (Dick & Reiser, 1989). The coaching function is crucial for the learner's active involvement in the last three components and would benefit from some form of face-to-face interactivity, such as the human tutor characteristics described in Section 2. Therefore, facilitating the learning process in Web-based distance education would benefit from a tutoring dialog system that mimics human tutor behaviors and strategies in the learning process.

3.1. Architecture of synthesizing Web courseware with tutoring conversation capability

Unlike other ITSs that have well-structured domain knowledge and an ideal tutoring model, our system's purpose is to fortify existing Web educational contents with a human-like tutoring capability. Moreover, in order to support any educational material on the Web, from well-designed materials to lecture notes, we need modular and reusable tutoring conversation architecture.

The conversational tutoring agent server in Fig. 1 was designed as a reusable structure. It has the basic functions of managing conversation, including a natural language processing capability, an assessment of learner's knowledge, and a Graphic User Interface (GUI). With these

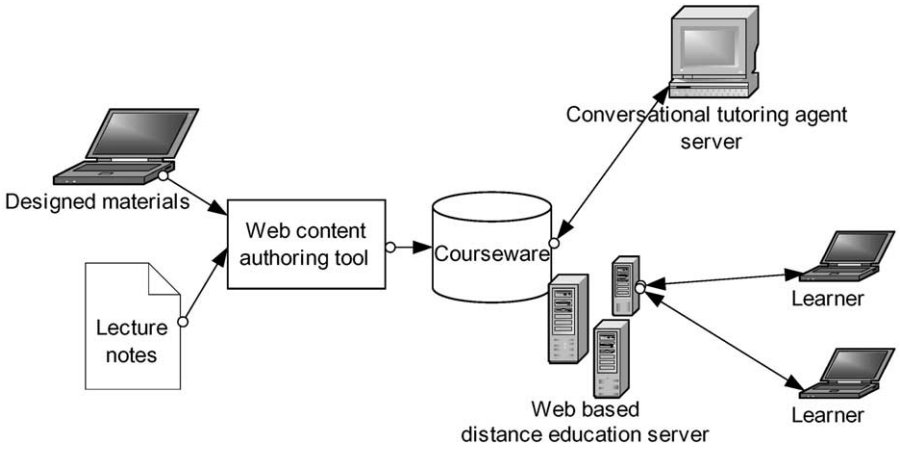

Fig. 1. Web-based distance education system with conversational tutoring agent.

functionalities, existing courseware may be equipped with providing guidance and feedback, testing comprehension, and providing enrichment or remediation functions for remote learners.

The conversational tutoring agent is activated when the learner clicks links that provide guidance and feedback, that test comprehension, and that provide enrichment or remediation conversation. The lecture can include such links anywhere they are needed in the courseware. There are several advantages of this modular scheme, including flexibility of courseware design, reusable tutoring conversation content, and ease in updating new information. After implementing any specific courseware, an instructor may evaluate the effectiveness of the courseware and improve it by rearranging the materials and conversation links.

Given the same subject matter, style, and planning strategies, Web courseware developed by instructors can be tailored, to some extent, to differences in analytical methods and pedagogical views of the instructors. Even though differences may exist between designed Web tutorials, we can assume that some of the subject matter content is independent of designer or instructor. Therefore, instead of providing unique tutoring conversation support for each of the teachers' Web tutorials, our system would allow sharable subject matter contents. The sharable content must of course be in a well-formed format, such as a list, a hierarchical composition, or a relational database. The conversational tutoring agent server would need to adopt a sharable knowledge assessment scheme and incorporate the necessary constraints in the database of curriculum scripts that the agent interacts with. Within a school, teachers of the same grade can cooperate to decide what curriculum scripts are needed for each subject, and thereby minimize the cost of operating a conversational tutoring agent server.

3.2. Conversational tutoring agent server overview

Our conversational tutoring agent server is based on AutoTutor (Graesser, Person, Harter, & TRG, 2001; Graesser, VanLehn, Rose, Jordan, & Harter, 2001), an animated pedagogical agent that serves as a conversational partner with the student. AutoTutor is a working system that responds to students' natural language contributions by simulating the dialog moves of normal human tutors (but not necessarily expert human tutors, which are much more difficult to simulate). Empirical tests of AutoTutor has shown that the tutoring system improves learning of computer literacy and conceptual physics by nearly a letter grade compared with reading a textbook an equivalent amount of time (Graesser et al., 2003).

The creation of AutoTutor was inspired by studies that have systematically analyzed the collaborative discourse that occurs between human tutors and students (Graesser et al., 1995; Person, Kreuz, Zwaan, & Graesser, 1995; Putnam, 1987). One reoccurring finding in several of these studies is that human tutors rarely adhere to "ideal" tutoring models that are often integrated into intelligent tutoring systems. Instead, human tutors tend to rely on pedagogically effective strategies that are embedded within the conversational turns of the tutorial dialogue. The major components of conversational tutoring agent structure are presented in the Fig. 2.

3.2.1. Animated tutor

An animated talking head component of the conversational agent is always waiting to receive a text and a request from the server to utter words or sentences, along with some appropriate gestures and facial expressions. The talking head speaks the piece of text passed in the request,

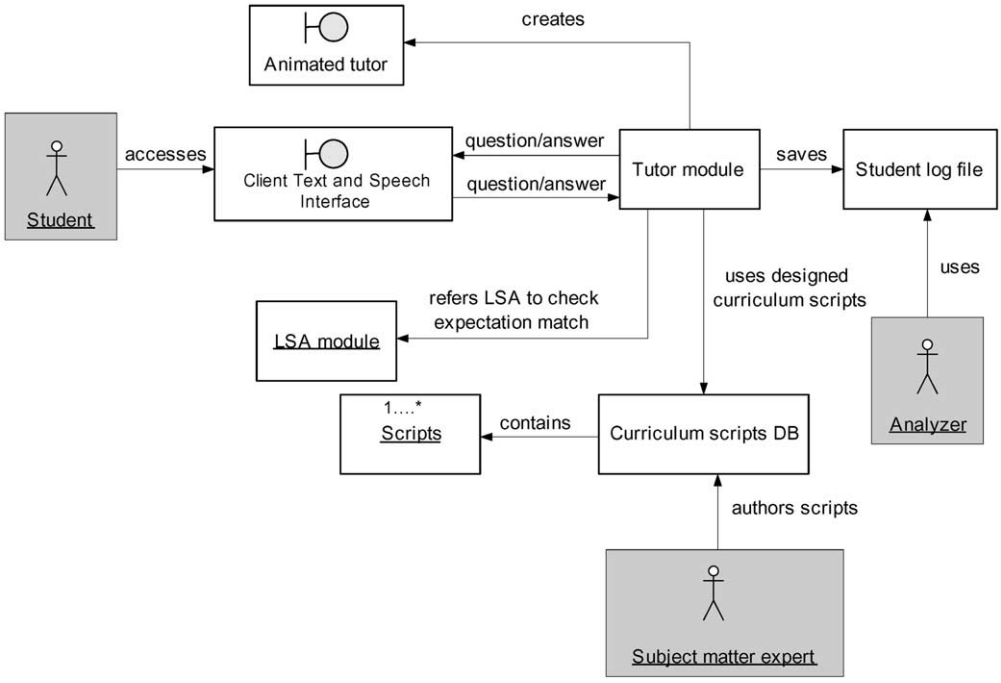

Fig. 2. The conversational tutoring agent server diagram.

sometimes by adding correct intonation to the utterance. The animated tutor easily gets the learner's attention.

3.2.2. LSA module

Our conversational tutoring agent uses Latent Semantic Analysis (LSA) to evaluate the quality of the student contributions that are typed by the student in natural language. LSA is a mathematical/statistical technique for extracting and representing the similarity in meaning of words, sentences, and passages by an analysis of large bodies of text (called a corpus). A large corpus of texts, as large as an encyclopedia, is first converted to a word-by-document matrix that scores the number times each word occurs in each document. LSA then uses singular value decomposition (SVD), a general form of factor analysis, to condense the very large word-by-document matrix into a much smaller space with typically 100–500 dimensions (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990; Landauer, Foltz, & Laham, 1998).

With a compressed K-dimensional space, the relevance or similarity between any two sets of words (e.g. $\boldsymbol{X}$ and $\boldsymbol{Y}$) can be evaluated. In most LSA applications, a cosine match between the vectors in the K-dimensional space, ranging from -1 to 1, is used to compute the similarity of $\boldsymbol{X}$ to $\boldsymbol{Y}$ (or relevance of $\boldsymbol{X}$ to $\boldsymbol{Y}$). For example, in AutoTutor and the present system, set $\boldsymbol{X}$ is the assertions expressed by the learner to a particular question during tutoring process. The LSA module calculates the cosine similarity score between the student assertions and the entries in the curriculum script (set $\boldsymbol{Y}$) that the instructor expects from the learner for that question. The computed cosine values that assess the similarities between assertions and expected answers are used

to evaluate the quality of what the student expresses. Such similarities subsequently help determine the next action of the tutor in tutoring process.

In order to coordinate with existing Web-based distance education courseware, the instructor needs to build up LSA spaces using related curriculum scripts and texts that are electronically available. The LSA spaces can be easily updated with new information and thereby provides a convenient way of maintaining the subject matter content in addition to monitoring learners' performance during the course of tutoring.

The LSA module in the conversational tutoring agent includes the LSA engine that processes SVD and the LSA space created for a specific subject matter. The module also includes support objects that might do some preprocessing on the student contribution before it is analyzed by LSA (such as syntactically parsing the student contribution). The LSA module therefore expects to receive a raw student contribution (either text strings or output from the parser) and produces a set of assessments of each student contribution. The LSA module may also need some information about the current context in order to calculate some of its assessments (see Graesser, Person et al., 2001). Using the LSA module, the students' contributions to questions posed by the computer tutor are evaluated, just as a human tutor would evaluate the students' responses.

3.2.3. Curriculum script database

The curriculum script database includes scripts with snippets of information that are dynamically selected during the tutorial process and uttered by the conversational tutoring agent. The curriculum script is a set of questions or problems. Associated with each question or problem is: (a) the focal question or problem, (b) didactic information and support media that serves as the context of question or problem, (c) a good answer, segregated into one or more sentences of information (called expectations), (d) hints, prompts, and sub-questions that would potentially elicit each sentence from the learner, (e) potential misconceptions and corrections, (f) succinct summaries of an ideal answer, and (g) other affiliated content. The selection of a particular question (or problem) to tutor next is dependent on the dialog history and the ability of the students. For example, more difficult questions are selected for students with higher ability. The dialog management component of the system decides what expectations, corrections, hints, sub-questions, and other dialog moves to generate, depending on student ability parameters and what was covered in the dialog history. The curriculum script database is a flat file or a hierarchically structured database that can be shared with some other learning management systems.

The curriculum script for specific Web courseware needs to be authored by the lesson planner in order to systematically create the core concepts, questions, problems, and expected coverage of presented material. Authoring tools for AutoTutor and other learning environments have been created for this purpose (Hu, Mathews, Graesser, & Susarla, 2002; Murray, 1999; Susarla, Adcock, VanEck, Moreno, & Graesser, 2003). After the student independently studies the existing Web courseware, a mixed-initiative question and answering process leads students to construct knowledge and to receive snippets of information that the tutor selects from the curriculum script database that was designed to cover the subject matter. The quality of student responses is determined by the LSA module that compares student contributions with information expected by the tutor, so we need to predefine what kind of answers is expected for a given question in advance. The information in the curriculum scripts is expressed in a plain text format (in natural

language), so the instructor does not need to know how to master technical editing software. Each expected answer statement can be entered and updated easily.

Expected answer statement types are categorized into ''ideal answers'', and ''bad answers'', as authored by the instructor. In AutoTutor, there is a cycle of ''hint–prompt–assertion'' utterances that are produced by the tutor for each expectation, until the student articulates the expectation. The discussion is summarized by the tutor with a ''summary'' statement after all of the expectations are covered and when any misconception expressed by the student is corrected.

3.2.4. Natural language processing

3.2.4.1. Text to speech synthesizer. The conversational tutoring agent has a wrapper for synthesizing speech. The text to speech synthesizer (TTS) expects to receive requests to speak text and causes the text to be spoken on some audio device. It provides an interface to an external SAPI 4 or SAPI 5 speech synthesis system.

3.2.4.2. Text input and output user interface. Students can interact with the conversational tutoring agent via text input and output in natural language expressions. AutoTutor provides the mix-initiative dialog by recognizing text input from the learner by virtue of matches to content in the curriculum script, and by producing dialog moves that are responsive to the learner and that facilitate coverage of the material. . In addition to the speech, a display of the conversation history helps students to focus on the current topics in a coherent fashion.

The student contributions are classified into Assertion, WH-question (e.g. ''What is the Gravity?''), Yes/No question, Metacognitive comment (e.g. ''I'm lost''), Metacommunicative act (e.g. ''Could you repeat that question?'') and Short response (''Oh'', ''yes''). Based on the classification of input text, AutoTutor initiates an appropriate reaction and strategically advances the conversation.

3.2.5. Dialog manager

The Dialog manager controls the system's interaction between the student and the conversational tutoring agent. Based on a student contribution, the dialog move generator will decide the type of dialog move to produce next, such as feedback to the student (positive, neutral negative), hints, pumps for more information (e.g., what else?), prompts for specific information, corrections, and summaries. In AutoTutor, this dialog management module is generic in the sense that it is used for all subject matters that are tutored.

In the dialog manager, we use a Dialog Advancer Network (DAN) to manage the mixed-initiative dialogue (Person, Graesser, Harter, Mathews & the Tutoring Research Group, 2000). The DAN is a finite state automaton that handles the different classes of information typed by the learners. The DAN taps words and phrases in different classes of discourse markers in order to clarify the tutor's intention behind the dialog moves and to coherently string together the dialog moves within one turn of the tutor.

Fig. 3 shows the mechanisms of the DAN. After the main question or problem is asked by the tutor, the tutor first interprets the initial answer expressed by the student, and then selects one of the good answer expectations to focus on. AutoTutor initiates the ''Hint–Prompt–Assertion'' cycle twice in an attempt to get the student to articulate the current expectation. After the second round, if the learner has not given the expected answer yet, it will assert the expectation and

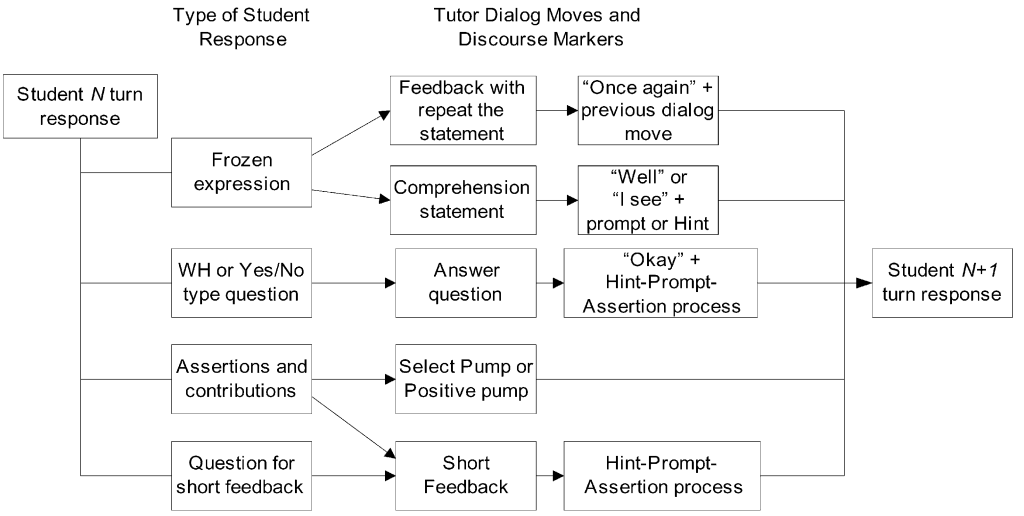

Fig. 3. The structure of Dialog Advancer Network.

proceed to the next expectation that has not yet been covered in the dialog. Once again, the expectations are dynamically selected on the basis of student ability and dialog history, as opposed to being covered in a rigid sequential order.

AutoTutor selects an expectation that gradually builds on what the student already knows, following what is called frontier learning (Sleeman & Brown, 1982) or the zone of proximal development (Vygotsky, 1978). AutoTutor also corrects misconceptions that arise during the course of the dialog, as long as the misconception was anticipated by the lesson planner and incorporated in the curriculum script.

3.3. *Tutoring session scheme in Web-based educational courseware*

The tutoring conversation session begins when a learner links to the "tutoring session" by following a hyperlink as shown in Fig. 4. At that point, the conversational tutoring agent appears and introduces himself to the student and describes the problems that will be covered in the session, which are related to the courseware just studied. After the introduction, the conversational tutoring agent asks the learner a very general question about the subject of the tutoring session. The student then types a response that is assessed by the LSA module and a mixed-initiative dialog is launched.

In real time, the LSA module computes values for a number of factors that dictate what dialogue move the conversational tutoring agent will supply next. Some of the LSA factors include student ability, topic coverage, good answer coverage, bad answer coverage, and student initiative. The conversational tutoring agent's current repertoire of dialogue moves simulate the dialogue moves that are frequently employed by human tutors (Graesser et al., 1995; Person, Graesser, Kreuz, Pomeroy, & TRG, 2001). The content of the conversational tutoring agent's dialog moves is specified a priori in the curriculum script, but snippets of information within the script are dynamically selected during the course of tutoring. That is, the conversational tutoring

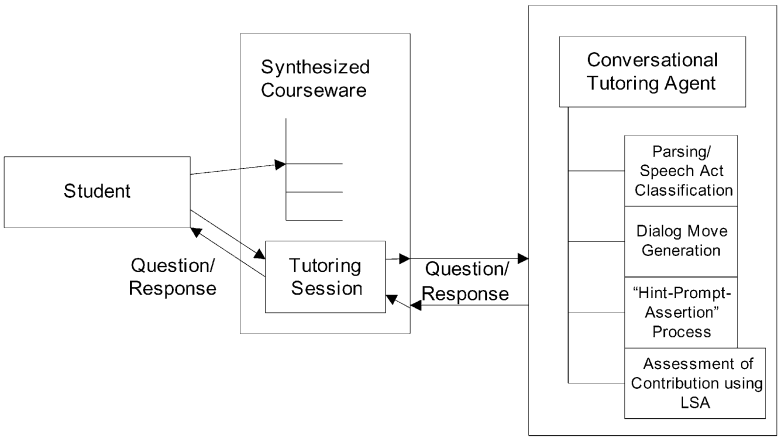

Fig. 4. The tutoring process with conversational tutoring agent.

agent does not generate the content of its dialog moves on the fly, but rather selects each dialog move from a large set of alternative moves in the script that is related to the tutoring topic being discussed. The conversational tutoring agent can respond to virtually any natural language student contribution in a conversationally (and pedagogically) appropriate way (Person et al., 2001; Person, Erkel, Graesser, & TRG, 2002).

4. Operational issues

The conversational tutoring agent was designed to be reusable for other knowledge domains that do not require mathematical precision and formal specification. Synthesizing the proposed tutor with existing Web courseware requires changing three modules: (1) a glossary of terms and definitions for the subject matter, (2) an LSA space for conceptual knowledge of given subject matter and (3) a curriculum script with deep reasoning questions and associated answers for the subjects (as discussed earlier).

Changing the glossary or definition process is relatively easy. Definitions of important words from text books need to be included in order to give AutoTutor the possibility of accurately answering definitional questions, such as "What does X mean?" Analyzing HTML pages of courseware and extracting available definitions of key terms can serve as the source of glossary of terms if an electronic glossary is not available.

Because the bulk of the tutor's comprehension mechanisms lies in LSA, the LSA space needs to be trained with an adequate corpus of texts that is relevant to the knowledge domain, such as textbooks, chapters, and technical articles. After the corpus is prepared in an electronic form, we first set up the parameters of LSA, such as the number of dimensions and size of document units (e.g., sentences, paragraphs, sections, articles). The training of the LSA space takes less than an hour. However, cleaning up the corpus, including the removal of pictures, adds additional time.

For curriculum script creation, only one particular format, the plain text format, is allowed. While an instructor authors the curriculum scripts, questions need to be defined, ideal answers

need to be formulated, and hints, prompts and pumps need to be included. It is very important to devise deep-reasoning questions for any given subject to retain the knowledge and also transfer the acquired knowledge into the next level. Most of these statements can be easily generated from teaching experiences, and therefore, the educational professionals can use their valuable teaching expertise to create and modify such Web based courseware. An excellent authoring tool has been developed for AutoTutor, called the ASAT (AutoTutor Script Authoring Tool, Susarla et al., 2003). ASAT is so easy to use that novices in computer technology can create a curriculum script in less than an hour.

Modifying the curriculum script for specific courseware is less time-consuming than most intelligent tutoring systems because the format of the entries is described in English rather than structured code (e.g., Lisp, Prolog). Because this process does not require special programming skills, several lesson planners can simultaneously work on the transition without help from sophisticated programming experts. Such simplicity allows for the easy updating the curriculum scripts reflecting the classroom situation every year. If our system is deployed in K-12 classes, teachers in the same institution and grade can cooperate to create possible expected answers and thus minimize the efforts to enhance and maintain conversational Web-based educational courseware.

The separation of the tutor server from Web courseware server gives maximum flexibility to develop Web courseware. It is also possible to integrate existing Web courseware that has already been developed with the traditional Computer-based Training approach. To the extent that this content is reusable, the effort of developing intelligent courseware for Web-based distance education can be minimized. One conversational tutoring agent server in an institution would be enough to serve a large number of students individually, successfully and extensively.

5. Conclusions

Even though many ITS systems have been developed, and some of them address porting ITS to WWW, none of have tried to reuse existing Web courseware to make them better than "page turning" course materials. We believe it is possible to facilitate the learning process in distance education by adding intelligence to courseware.

In this paper, we introduce a framework that integrates AutoTutor with existing Web-based distance education courseware. This approach enhances the interactivity in the courseware by synthesizing human tutoring with a conversational agent. The simulated human conversational tutoring agent is composed of an animated tutor, an LSA module, a curriculum script database, and dialog manager. To simulate a human tutor, natural language processing techniques are applied to the learner interface, whereas the LSA technology provides a metric for assessing the quality of learners' contributions to answering questions. The proposed conversational tutoring agent furnishes the distinctive dialog patterns of human tutors to traditional Web courseware and thereby attempts to improve Web educational courseware. The cost for developing such a system is substantially reduced by reusing the basic components of AutoTutor, by the use of an authoring tool that is easy to use for the development of curriculum scripts, and by the speedy development of LSA spaces from a large corpus of texts. The tutor can be developed in months, days, or even hours, rather than several years.

Acknowledgements

The Tutoring Research Group (TRG) is an interdisciplinary research team comprised of approximately 35 researchers from psychology, computer science, physics, and education (Visit http://www.autotutor.org). This research conducted by the authors and the TRG was supported by grants from the National Science Foundation (SBR 9720314 and REC 0106965) and the Department of Defense Multidisciplinary University Research Initiative (MURI) administered by the Office of Naval Research under grant N00014-00-1-0600. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of ONR or NSF.

References

Bork, A. (1996). Advantages of computer based learning. *Journal of Structured Learning*, *9*, 63–76.

Brusilovsky, P., Schwarz, E., & Weber, G. (1996) ELM-ART: an intelligent tutoring system on World Wide Web. In *Intelligent tutoring systems. Third International Conference, ITS'96* (pp. 261–269). Vol. 1086 of Lectures Notes of Computer Science, Springer-Verlag.

Carbonell, J. R. (1970). AI in CAI: an artificial intelligence approach to computer-assisted Instruction. *IEEE Transactions on Man-Machine Systems*, *11*, 190–202.

Chi, M. T. H., Siler, S., Jeong, H., Yamauchi, T., & Hausmann, R. G. (2001). Learning from human tutoring. *Cognitive Science*, *25*, 471–533.

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.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic indexing. *Journal of the American Society for Information Science*, *41*, 391–407.

Dick, W., & Reiser, R. (1989). *Planning effective instruction*. Englewood Cliffs, NJ: Prentice Hall.

Evens, M. W., Brandle, S., Chang R.U., Freedman, R., Glass M., Lee, Y. H., Shim, L. S., Woo, C. C., Zhang, Y., Zhou, Y., Michael, J. A., & Rovick, A. A. (2001). CIRCSIM-tutor: an intelligent tutoring system using natural language dialogue. In *Twelfth Midwest AI and Cognitive Science Conference, MAICS 2001* (pp. 16–23). Oxford, OH.

Freedman, R. (1999). Atlas: a plan manager for mixed-initiative, multimodal dialogue. In *AAAI-99 Workshop on Mixed-Initiative Intelligence* (pp. 1–8) Orlando, FL.

Graesser, A., Person, N., & Harter, D. (2001). TRG Teaching tactics and dialog in AutoTutor. *International Journal of Artificial Intelligence in Education*, *12*, 257–279.

Graesser, A. C., Person, N. K., & Magliano, J. P. (1995). Collaborative dialogue patterns in naturalistic one-to-one tutoring. *Applied Cognitive Psychology*, *9*, 1–28.

Graesser, A. C., VanLehn, K., Rose, C., Jordan, P., & Harter, D. (2001). Intelligent tutoring systems with conversational dialogue. *AI Magazine*, *22*, 39–51.

Graesser, A. C., Jackson, G. T., Mathews, E. C., Mitchell, H. H., Olney, A., Ventura, M., Chipman, P., Franceschetti, D., Hu, X., Louwerse, M. M., & Person, N. K. (2003). TRG Why/AutoTutor: a test of learning gains from a physics tutor with natural language dialog. In R. Alterman, & D. Hirsh (Eds.), *Proceedings of the 25th Annual Conference of the Cognitive Science Society* (pp. 1–5). Boston, MA: Cognitive Science Society.

Hara, N., & Kling, R. (2000). Students' distress in a Web-based distance education course, Information. *Communication & Society*, *3*, 557–579.

Hu, X., Mathews, E., Graesser, A. C., & Susarla, S. (2002). EBOOK.EXE: A desktop authoring tool for HURAA. In M. Reeves, & T. C. Reeves (Eds.), *Proceedings for E-Learning 2002: World Conference on E-Learning in Corporate, Government, Healthcare, & Higher Education* (pp. 471–476.

Kiyama, M., Ishiuchi, S., Ikeda, K., Tsujimoto, M., & Fukuhara, Y. (1997). Authoring methods for the Web-based intelligent CAI system CALAT and its application to telecommunications service. In: *Proceedings of AAAI-97*. Providence, RI.

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

Millbank, G. (1994). *Writing multimedia training with integrated simulation*. Paper presented at the Writers' Retreat on Interactive Technology and Equipment. Vancouver, BC: The University of British Columbia Continuing Studies.

Murray, T. (1999). Authoring Intelligent Tutoring Systems: An analysis of the sate of the art. *International Journal of Artificial Intelligence in Education*, *10*, 98–129.

Person, N. K., Erkel, M., & Graesser, A. C. (2002). TRG AutoTutor passes the bystander Turing test. In M. Driscoll, & T. C. Reeves (Eds.), *Proceedings for E-Learning 2002: World Conference on E-Learning in Corporate, Government, Healthcare, & Higher Education* (pp. 778–782). Montreal, Canada: AACE.

Person, N. K., Graesser, A. C., Kreuz, R. J., & Pomeroy, V. (2001). TRG Simulating human tutor dialog moves in AutoTutor. *International Journal of Artificial Intelligence in Education*, *12*, 23–39.

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.

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.

Schank, R. (1998). Horses for courses. *Communications of the ACM*, *41*(7), 23–25.

Sleeman, D., Brown, J. S. (Eds.). (1982). *Intelligent tutoring systems*. New York: Academic Press.

Susarla, S., Adcock, A., Van Eck, R., Moreno, K., & Graesser, A. C. (2003). Development and evaluation of a lesson authoring tool for AutoTutor. In V. Aleven, U. Hoppe, J. Kay, R. Mizoguchi, H. Pain, F. Verdejo, & K. Yacef (Eds.), *AIED2003 Supplemental Proceedings* (pp. 378–387). Sydney, Australia: University of Sydney School of Information Technologies.

Thibodeau, P. (1997). Design standards for visual elements and interactivity for courseware. *The Journal Online*, 84–86.

Vygotsky, L. S. (1978). *Mind and society: the development of higher mental processes*. Cambridge, MA: Harvard University Press.