An Adaptive Tutoring Machine Based on Web Learning Assessment

Timothy K. Shih, Dept. of Computer Software, Univ. of Aizu, Japan*
Shi-Kuo Chang, Dept. of Computer Science, Univ. of Pittsburgh, USA
Ching-Sheng Wang, Dep. of Computer Science and Information Engineering, Tamkang University
Jianhua Ma and Runhe Huang, Dept. of Computer Software, Univ. of Aizu, Japan
E-mail: TSHIH@CS.TKU.EDU.TW

ABSTRACT

Student performance is difficult to measure in distance learning. In this paper, we discuss a system which keeps track of the interaction behavior of each student while one is visiting a distance learning Web document. The system also uses a dynamic finite state machine to generate new Web documents based on the interaction behavior. The contribution of such a mechanism benefits both teachers in understanding their instruction achievement and students in realizing their learning progress.

Key words: WWW, Assessment, Adaptive Tutoring, Distance Learning, Virtual University

1. INTRODUCTION

Web-based distance learning is a trend of instruction delivery. One of the most difficult challenges of such a learning mechanism is the assessment of students' learning criteria. It is hard to judge the behavior of a student since the instructor is separated spatially and temporally from the students. However, it is possible to rely on some Web-based tools to keep track of a student's course attendance, as well as the navigation behavior of that student. In addition, the navigation behavior of an individual can be compared to those of others. Analysis can be conducted. And interactive tutorial can be generated to assist the student of poor score. This paper proposes such a mechanism, as well as its supporting system run on Windows browsers.

The objective of this research is to investigate the co-relation between the navigation behavior of students and their learning results, based on the navigation sequences and objects the students traversed, the courseware, the quiz, test, and exam w.r.t. a particular course, and the student assessment results. The contributions of the developed agent software tool have two purposes:

- The tool helps an instructor to analyze the co-relation between student Web navigation behavior and student learning curve.
- The tool provides a guideline for automatic intelligent tutoring.

The fundamental concepts that we use in the development of such an intelligent assessment technology base on four co-relations: the co-relation between Web course navigation sequences and courseware, the co-relation between courseware and a collection of questions, the co-relation between questions and different types of exams and assignments (which are obtained from the collection of questions), and the co-relation between the tests or homework and the students' grades. To compute the first co-relation, a control daemon running on a standard Web browser is required to collect the navigation sequences and objects from a student's participation in a Web course. The sequence and the courseware are compared and analyzed. On the other hand, when the instructor designs a course and the associated questions, the co-relation between them should be specified. Course documents and questions are stored as HTML files, which can be parsed easily by a HTML parser to obtain the co-relation (which is a set of URLs). Exams are built by an instructor with the help from an exam builder. Additionally, exams are distributed to students via a student assessment tool. The answers from students are collected and checked by an automatic tool or semi-automatically by a grader. The co-relation between exams and test results are thus stored. Based on these four co-relations, it is useful to deduce the fifth co-relation between the navigation sequence of a student and the testing results (i.e., credits). This co-relation, even it is difficult to analyze and compute, is used in our purposed model as the contribution of our research.

The proposed system is implemented based on Web technologies. Since a Web server only maintains the physical connection to a particular client within the duration while the client is downloading Web data, it is impossible for a Web server to keep track of all interaction messages of each student. Therefore, a mobile agent program is required to run on each client site. The agent program collects the interaction messages, such as the duration of traversal of a particular page, how many times an object is visited, as well as the sequence of navigation. After an interaction session is closed (i.e., change to another URL), the interaction messages are sent to the Web server database by the mobile agent. In order to make the assessment model useful, the analysis of student performances based on these interaction messages are given and intelligent tutorials are constructed. The machine which realizes these tutorials is discussed in section 2. Related work is given in section 3. A short conclusion is given in section 4.

2. THE PROGRESSIVE LEARNING STATE MACHINE

The results of assessment feedback computation has many purposes. One of the goals is to assert the feedback statistic data into a *Progressive Learning State Machine*, which helps the adaptive tutoring tool to construct intelligent tutorial. A tutorial is a navigation sequence which also contains some hints from an instructor. These hints could be interactive objects. A state machine can be used to control a tutorial, providing that student interactions are the input to the machine while the corresponding output w.r.t. a particular input are designed by the instructor. Thus, the construction of such a tutorial state machine is supervised by an instructor. A progressive learning state machine is a dynamic state machine. The statistic feedback from the assessment computation is analyzed by a behavior analysis tool and asserted to the dynamic state machine. Such an assertion results in an update of the state machine transitions, input, output, and the machine size. Each student has his/her own state machine. With the supervision from the instructor, individual behavior reasoning are applied to each student.

Since the distance-learning environment is Web-savvy, it is natural to consider the implementation of such a state machine based on Web technology. A Web document is a deterministic finite state machine, which is defined as:

$$Web\_FSM == (S, I, O, \delta, \omega)$$
$$S == \mathrm{P}_1 (URL \times DocPage)$$

The five components of *Web_FSM* include a non-empty finite set of *states* (including the starting state s_0), which is denoted as S. Each state, in terms of Web technology, is a navigation snapshot on a Web browser, which can be treated as a document page status, or a document page w.r.t. a particular time slot. A time slot is the duration between the activations of two distinct hyperlinks. Therefore, a snapshot can be identified by a hyperlink. Of course, the navigation content (i.e., the document page w.r.t. a hyperlink) is part of the state. Each state has a set of valid input and produces a set of output. The union of all possible input is denoted by:

$$I == \mathrm{I}_{ENUM} \cup \{ null \}$$

where I_{ENUM} represents the enumeration of all possible Web traversal input signal, and *null* is a special symbol which denotes no input signal at all. The union of all possible output can be defined by:

$$O == O_{ENUM} \cup \{ null, \varepsilon \}$$

where O_{ENUM} is a set of all possible enumeration of output, *null* means no output, and ε represents the output of an error message. The Web finite state machine also contains the next state function δ and the output function ω. These functions are defined as usual.

$$\delta == S \times I \rightarrow S$$
$$\omega == S \times I \rightarrow O$$

A Progressive Learning State Machine (PLSM) is a deterministic finite state machine with respect to a particular time slot. But, a PLSM may have different next step function and output function from time to time. As a conceptual model, a Web course (or any WWW document) is a state machine. The purpose of our research is to construct a tutorial, which is also a WWW document, from the Web course. The tutorial is a PLSM. Initially, the PLSM contains all states of the Web course (i.e., all snapshots), and all possible input and output. But, the δ and the ω functions are empty. The PLSM (or the tutorial) does not need to allow the same traversal as the Web course. Instead, it is constructed according to the analysis of navigation behavior. A PLSM is altered by a set of state machine *construction intelligence* (i.e., *CI*), which is extracted from the analysis.

$$CI == (S \times I \rightarrow S) \times (S \times I \rightarrow O)$$

Each element in *CI* has two parts. The first part is a domain to range value pair of the next step function. And the second part is a value pair of the output function. The changes of these two functions in a PLSM makes the deterministic finite state machine dynamic. Thus,

$$PLSM_i \times CI \rightarrow PLSM_{i+i}$$

is a construction function of PLSMs from time slot *i* to time slot *i+1*.

The construction of PLSM is based on several factors: the Web document content, how they are organized, how they are presented, and the interaction of an individual student. In the following subsections, we discuss these concepts, with a mechanism for PLSM construction.

2.1 Web Document Structure versus Web Knowledge Structure

The translation extracts information from the construction intelligence and build two types of links to construct a PLSM. The first is the *original hyperlinks* which were designed by the instructor in a Web course. This strategy connects the multimedia objects which may cost a student's misunderstanding of the course material. The second is the *new hyperlinks* created which links from one document page to another. This strategy makes a new tutorial sequence, which may be related to the order of questions in an exam. These two types of links can be used by the PLSM construction intelligence, with a understanding of what should be extracted from a Web course document.

When an instructor designs the Web course, the document is organized in a structure -- the ***Web document structure***. It represents the logical connection of various Web pieces. The Web course material may or may not be used by the same instructor. It is also possible that several instructors use the same course material in a virtual university. But, different instructor may use different navigation sequence to explain the material. The sequence reflects the individual knowledge of the instructor to the understanding of the material. Therefore, the navigation sequence is a ***Web knowledge structure*** of the Web document w.r.t. an instructor. According to the current Web technique, the knowledge structure is a subset of the document structure. Most likely, the knowledge structure is a proper subset. This knowledge structure is useful in the construction of a PLSM, which is a tutorial for a particular student. To traverse the tutorial, an interactive mechanism is used by the student. The interaction can be controlled by a visual language. One of the focuses of this research is to investigate the languages of building a PLSM construction intelligence. The technique used involves two level of languages of a Web document construction. A Web document is designed in a definition language (e.g., HTML). As a portion of the definition language, the input and output instructions define the interaction semantics of a Web document. If we consider the interaction as a running visual program, each Web document has an interaction language, which precisely defines the visual interaction of the Web document. If the interaction language of a Web document can be precisely defined, it is possible to generate a deterministic finite state machine for the Web document. The purpose of the construction intelligence is to define the interaction language of the generated tutorial. Since the process of tutorial generation is dynamic, thus, the semantics of the interaction language is dynamic. That is, the navigation options of the tutorial is changeable.

2.2 The Construction Intelligence

One of the factors which influents the generation of PLSM is the interaction of a student. We defined two co-relations. The first co-relation specifies the relation between a sequence of Web pages (i.e. navigation trail) visited and a test result. The second keeps the relation between a set of multimedia objects (i.e., navigation set) in those Web pages which are visited, and a test result. Whether the co-relation is specified based on navigation trail or navigation set, each pair element of the co-relation is a mapping from the content of a Web object to one of the following conditions:

- The content is related, the credit is obtained (represented by a ``Y")
- The content is related, but the credit is not obtained (represented by an ``N")
- T\he content is not related to the question (represented by a ``U")

The above two types of co-relations can be represented as a two dimensional array. The value of the array are either ``Y", ``N", or ``U". Note that, each student in a class has a unique array. Therefore, individual tutorials can be generated for each student. In section 2, we defined construction intelligence as

$$CI == (\delta \times \omega)$$

which is one of the input parameters of the construction function of PLSMs. Construction intelligent aims to change the next step function (i.e., δ) and the output function (i.e., ω). As the input set I of a Web document finite state machine (i.e., *Web_FSM*) is restricted to mouse clicks and keyboard input, the interaction language of the PLSMs is deduced from the construction intelligence, which includes the two functions (i.e., δ, and ω). Each instruction in the interaction language accepts an input action, performs a state transition, and produces an output as the side effect of the interaction.

The strategy of PLSM construction mechanism firstly checks at the missing items (i.e., with a ``N" representation) in the co-relation array. For each of such item, the corresponding Web object is selected. However, these objects are not enough to construct a good tutorial. The construction mechanism also needs to consider Web knowledge structure. As a presentation navigation sequence by an instructor, the structure represents a dependency relation among Web document pieces. A sub-machine is constructed for each Web object w.r.t. a missing item. Each sub-machine maintains the *original hyperlinks*, which connect other pieces based on the dependency relation (the knowledge structure). The *new hyperlinks* are used to connect these sub-machines. The order of this new connection is based on the order of the missing items in an exam.

The strategy discussed is the first step to construct a PLSM from the Web course document. But, as the generated tutorial is traversed, the PLSM should be able to consider other factors.

2.2.1 The Construction Intelligence with Relevance Feedback

We discuss a mechanism to enforce students to pay attention to the course material. The pop-up quiz is part of a Web course. When a tutorial is constructed from the Web course, these quizzes are used again. The student is encouraged to pass these quizzes in order to proceed with the next page of the tutorial. Moreover, since each course section is associated with some questions, while a student is traversing a tutorial, the PLSM constructor selects some questions, which are dynamic sub-machines constructed on-the-fly, to test the effectiveness of the tutorial. The sub-machine is constructed if the student fails to answer pop-up quizzes upto a number of times. Or, the construction is performed at the end of each session of the tutorial.

2.2.2 The Construction Intelligence with Norm-referenced Evaluation

The construction intelligence discussed in the previous sections is based on the Web knowledge structure which is deduced from the navigation of the instructor. However, it is possible to look at the Web traverse sequence from the perspective of students. If some students have a high average, their navigation sequences can be worthy. The learning performance of a student is compared with the average of the class. The average can be classified into three types:

- **High Average:** the average of the front 50 percent of students
- **Middle Average:** the average of all students
- **Low Average:** the average of the last 50 percent of students

The construction intelligence collects the union of all Web objects in the navigation of those students with a score higher than the high average. If a student has a score lower than the low average, a tutorial is generated for him/her based on the collection, with the original hyperlinks of the Web document included.

As a consequence, the construction intelligence based on the comparison of performance averages reflects to norm-referenced evaluation. On the other hand, the construction intelligence based on the instructor's navigation reflects to criterion-referenced evaluation.

3. RELATED WORKS

A number of researchers have developed domain specific presentations using artificial intelligence techniques. For example, COMET (Coordinated Multimedia Explanation Testbed) [4] uses a knowledge base and AI techniques to generate coordinated, interactive explanations with text and graphics that illustrate how to repair a military radio receiver-transmitter. WIP [1] is able to generate knowledge-based presentations that explain to a user how to use an espresso machine. A Piano tutor described in [3] is able to use coordinated media, video, voice, and graphics display, to teach beginners how to play the piano. An intelligent presentation system based on logic programming techniques is proposed in [2]. The system generates multimedia presentations based on user's interaction. An intelligent tutorial tool based on the Byzantium model is presented in [5].

4. CONCLUSIONS

The proposed system relies on the instructor to prepare for course materials such as slides and tests in advance. While a student is accessing the materials, the interaction between the student and a Web server is recorded and analyzed by the system. The system generates reports for the instructor and the student to find out which part of the lecture should be reviewed. Thus, the system helps both the instructor and the students. The system is implemented on Windows 98/NT. It is possible to extend our research using a neural network approach to keep the interaction behavior of good leaning examples. The approach can be used in the generation of intelligent tutorials.

REFERENCES

[1] E. Andre. "WIP: The Automatic Synthesis of Multimodal Presentations". In Mark T. Maybury, editor, In Intelligent Multimedia Interfaces}, pages 75--93. American Association for Artificial Intelligence, 1993.

[2] C. M. Chung, T. K. Shih, J. Huang, Y. Wang and T. Kuo, "An Object-Oriented Approach and System for Intelligent Multimedia Presentation Designs". In proceedings of the ICMCS'95 conference, pages 278--281, 1995.

[3] R. B. Dannenberg and R. L. Joseph. "Human Computer Interaction in the Piano Tutor". In M. M. Blattner and R. B. Dannenberg, editors, In Multimedia Interface Design, pages 65--78. 1992.

[4] S. K. Feiner and K. R. McKeown. "Automating the Generation of Coordinated Multimedia Explanations". In Mark T. Maybury, editor, In Intelligent Multimedia Interfaces, pages 117--138. 1993.

[5] A. Patel, et. al. "Intelligent tutoring tools-a problem solving framework for learning and assessment". Frontiers in Education Conference, 1996. FIE '96. 26th Annual Conference., Proceedings of Volume: 1 , 1996 , Page(s): 140 - 144 vol.1