A Web-based Intelligent Tutoring System for Computer Programming

C.J. Butz, S. Hua, R.B. Maguire
Department of Computer Science
University of Regina
Regina, Saskatchewan, Canada S4S 0A2
Email: {butz, huash111, rbm}@cs.uregina.ca

Abstract

Web Intelligence is a direction for scientific research that explores practical applications of Artificial Intelligence to the next generation of Web-empowered systems. In this paper, we present a Web-based intelligent tutoring system for computer programming. The decision making process conducted in our intelligent system is guided by Bayesian networks, which are a formal framework for uncertainty management in Artificial Intelligence based on probability theory. Whereas many tutoring systems are static HTML Web pages of a class textbook or lecture notes, our intelligent system can help a student navigate through the online course materials, recommend learning goals, and generate appropriate reading sequences.

1 Introduction

Web-based learning systems are increasingly popular due to their appeal over traditional paper-based textbooks. Web courseware is easily accessible and offers greater flexibility, that is, students can control their own pace of study. Unlike printed textbooks, Web-based tutoring systems can incorporate multi-media such as audio and video to make a point. However, since many current Web-based tutoring systems are static HTML Web pages, they suffer from two major shortcomings, namely, they are neither interactive nor adaptive [3].

Web Intelligence is a direction for scientific research that explores practical applications of Artificial Intelligence to the next generation of Web-empowered systems [16]. For instance, Yao and Yao [19] argue that a system should be robust enough to deal with various types of users. In the context of Web-based tutoring systems, Liu et al. [9] developed an intelligent system for assisting a user in solving a problem. Obviously, this involves creating systems that can make decisions based on uncertain or incomplete information. One formal framework for uncertainty management is *Bayesian networks* [11, 17, 18], which utilize probability theory as a formal framework for uncertainty management in Artificial Intelligence. Web intelligence researchers have applied Bayesian networks to many tasks, including student monitoring [7, 9], e-commerce [5, 12], and multi-agents [8, 15].

In this paper, we put forth a Web-based intelligent tutoring system, called BITS, for computer programming. The decision making process conducted in our intelligent system is guided by a Bayesian network. Similar to [7, 9], BITS can assist a student in navigation through the online materials. Unlike [7, 9], however, BITS can recommend learning goals and generate appropriate learning sequences. For example, a student may want to learn "File I/O" without having to learn every concept discussed in the previous materials. BITS can determine the minimum prerequisite knowledge needed in order to understand "File I/O" and display the links for these concepts in the correct learning sequence. BITS has been implemented and will be used in the summer 2004 session of CS110, the initial computer programming course at the University of Regina. As empirical studies have shown that individual one-on-one tutoring is the most effective mode of teaching and learning [2], BITS serves as intelligent software for implementing computer-assisted one-on-one tutoring.

The rest of this paper is organized as follows. In Section 2, we briefly review intelligent tutoring systems and Bayesian networks. In Section 3, we describe how to use a Bayesian network in BITS for modelling and inference. BITS's capability for adaptive guidance is discussed in Section 4. In Section 5, we describe the features that allow BITS to be accessed via the Web. Related works are discussed in Section 6. The conclusion is presented in Section 7.

2 Background Knowledge

In this section, we briefly review intelligent tutoring systems and Bayesian networks.

2.1 Intelligent Tutoring Systems

Computers have been used in education for over 35 years [1]. Traditional *Computer-Assisted Instruction* (CAI) presents instructional materials in a rigid tree structure to guide the student from one content page to another depending on his/her answers. This approach is restrictive in that it does not consider the diversity of students' knowledge states and their particular needs (c.f. [19]). Moreover, CAI systems are not adaptive and are unable to provide individualized attention that a human instructor can provide [3].

An *Intelligent Tutoring System* (ITS) is a computer-based program that presents educational materials in a flexible and personalized way [3, 7]. These systems can be used in the normal educational process, in distant learning courses, either operating on stand-alone computers or as applications that deliver knowledge over the Internet. As noted by Shute and Psotka [13], an ITS must be able to achieve three main tasks:

(i) accurately diagnose a student's knowledge level using principles, rather than preprogrammed responses;

(ii) decide what to do next and adapt instruction accordingly;

(iii) provide feedback.

This kind of diagnosis and adaptation, which is usually accomplished using Artificial Intelligence techniques, is what distinguishes an ITS from CAI. Empirical studies have shown that individual one-on-one tutoring is the most effective mode of teaching and learning, and ITSs uniquely offer a technology to implement computer-assisted one-on-one tutoring [2].

2.2 Bayesian networks

Let $U = \{a_1, a_2, \ldots, a_n\}$ denote a finite set of discrete random variables. Each variable a_i is associated with a finite domain $dom(a_i)$. Let V be the Cartesian product of the variable domains, namely,

$$V = dom(a_1) \times dom(a_2) \times \ldots \times dom(a_n).$$

A *joint probability distribution* is a function p on V such that the following two conditions hold:

(i) $0 \leq p(v) \leq 1.0$, for each configuration $v \in V$,

(ii) $\sum_{v \in V} p(v) = 1.0$.

Clearly, it may be impractical to obtain the joint distribution on U directly: for example, one would have to specify 2^n entries for a distribution over n binary variables.

A *Bayesian network* [11] is a pair $\mathcal{B} = (D, C)$. In this pair, D is a *directed acyclic graph* (DAG) on a set U of variables and $C = \{p(a_i|P_i) \mid a_i \in D\}$ is the corresponding set of *conditional probability distributions* (CPDs), where P_i denotes the *parent set* of variable a_i in the DAG D. A CPD $p(a_i|P_i)$ has the property that for each configuration (instantiation) of the variables in P_i, the sum of the probabilities of a_i is 1.0. Based on the *probabilistic conditional independencies* [17, 18] encoded in the DAG, the product of the CPDs is a unique joint probability distribution on U, namely,

$$p(a_1, a_2, \cdots, a_n) = \prod_{i=1}^{n} p(a_i|P_i).$$

Thus, Bayesian networks provide a semantic modelling tool which facilitates the acquisition of probabilistic knowledge.

3 BITS

In this section, we introduce a *Bayesian intelligent tutoring system*, called BITS, for computer programming.

3.1 Modelling the Problem Domain

There are two tasks involved in helping a student navigate in a personalized Web-based learning environment. Firstly, the structure of the problem domain must be modelled. Secondly, student knowledge regarding each concept in the problem domain must be tracked. Bayesian networks can help us meet both of these objectives.

To simplify the task of developing an intelligent tutoring system, we restrict the scope of the problem. Only elementary topics are covered, namely, those typically found in a first course on programming. That is, concepts such as variables, assignments, and control structures are included, but more sophisticated topics like pointers and inheritance are not. For our purposes, we identified a set of concepts that are taught in CS110, the initial computer programming course at the University of Regina. Each concept is represented by a node in the graph. We add a directed edge from one concept (node) to another, if knowledge of the former is a prerequisite for understanding the latter. The DAG can be constructed manually with the aid of the course textbook and it encodes the proper sequence for learning all the concepts in the problem domain.

Example 1 *Consider the following instance of the "For Loop" construct in C++:*

```
for(i=1; i<=10; i++).
```

To understand the "For Loop" construct, one must first understand the concepts of "Variable Assignment" `(i=1)`*, "Relational Operators"* `(i<=10)`*, and "Increment/Decrement Operators"* `(i++)`*. These prerequisite relationships can be modelled as the DAG depicted in Figure 1.*

Naturally, Figure 1 depicts a small portion of the entire DAG implemented in BITS. The entire DAG implemented in BITS consists of 29 nodes and 43 edges.

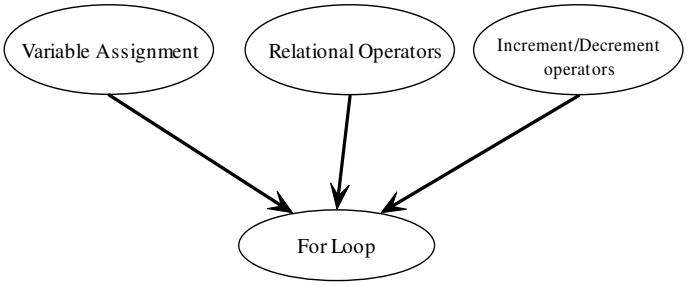

Figure 1. Modelling the prerequisite concepts of the "For Loop" construct.

The next task in the construction of the Bayesian network is to specify a CPD for each node given its parents.

Example 2 *Recall the node a_i="For Loop" with parent set P_i={"Variable Assignment," "Relational Operators," "Increment/Decrement operators"} depicted in Figure 1. A CPD p(For Loop|Variable Assignment, Relational Operators, Increment/Decrement Operators) is shown in Figure 2.*

Parent Nodes			For Loop	
Variable Assignment	Relational Operators	Incre/Decrement Operators	*known*	*not known*
known	*known*	*known*	0.75	0.25
		not known	0.39	0.61
	not known	*known*	0.50	0.50
		not known	0.22	0.78
not known	*known*	*known*	0.50	0.50
		not known	0.29	0.71
	not known	*known*	0.40	0.60
		not known	0.15	0.85

Figure 2. The CPD corresponding to the "For Loop" node in Figure 1.

All CPDs for the DAG were obtained from the results of previous CS110 final exams. We first identified the concept being tested for each question. If the student answered the question correctly, then we considered the concept *known*. Similarly, if the student answered the question incorrectly, then we considered the concept *unknown* (*not known*). The probability of each concept being known, namely, $p(a_i = known)$, can then be determined. Moreover, we can also compute $p(a_i = known, P_i = known)$, i.e., the probability that the student correctly answers both the concept a_i and the prerequisite concepts P_i. From $p(a_i = known, P_i = known)$, the desired CPD $p(a_i = known|P_i = known)$ can be obtained. Thereby, we can calculate every CPD for the entire Bayesian network.

3.2 Personalized Learning

It has been argued [19] that systems should provide personalized environments. In this sub-section, we show how BITS adapts to the individual user. We begin by motivating the discussion.

Brusilovsky [3] states that several systems detect the fact that the student reads information to update the estimate of her knowledge. Some systems also include reading time or the sequence of read pages to enhance this estimation. However, we believe the disadvantage of this approach is that it is difficult to measure whether a student really understands the knowledge by "visiting" Web pages of lecture notes.

In BITS, there are two methods to obtain evidence for updating the Bayesian network:

(a) A student's direct reply to a BITS query if this student knows a particular concept.

(b) Sample quiz result for the corresponding concept to determine whether or not a student has understood a particular concept.

We believe this approach is a more reliable way for estimation.

After the student finishes reading the displayed lecture notes, she provides feedback to BITS. More specifically, she selects one of the following three choices:

- I understand this concept,
- I don't understand this concept,
- I'm not sure (quiz me),

as illustrated in the bottom right corner of Figure 3. The first two answers fall under case (a) for obtaining evidence, while the last answer falls into case (b).

In case (a), the Bayesian network can be immediately updated to reflect the student's knowledge or lack thereof. In case (b), BITS will retrieve the appropriate quiz from the database and present it to the user. For instance, if the student indicates that she is not sure whether she understands the concept "File I/O," then BITS displays the quiz on "File I/O" in Figure 4. BITS will then compare the student's answer with the appropriate solution key stored in the database. The student is informed whether the answer is correct or not, and the Bayesian network is updated accordingly. If the answer is incorrect, the correct answer is displayed. In the next section, we turn our attention to using the updated Bayesian network for adaptive guidance.

Figure 3. A screenshot of BITS displaying the lecture notes and querying whether this concept is understood for the concept "File I/O."

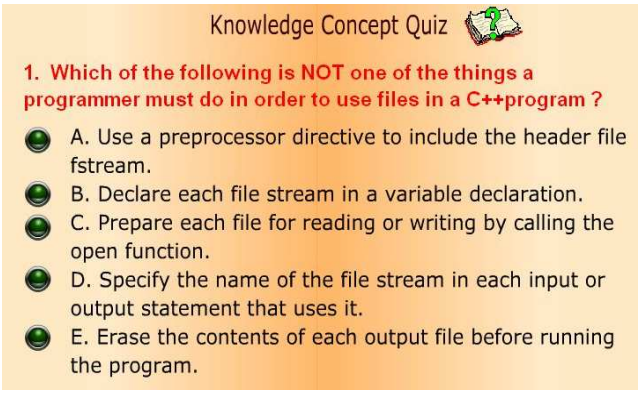

Figure 4. One question on a sample quiz for the concept "File I/O" in Figure 3.

4 Adaptive Guidance

Using the state of the Bayesian network regarding the knowledge of the student, BITS can offer tailored pedagogical options to support the individual student. In this section, we describe three kinds of adaptive guidance that BITS can provide, namely, *navigation support*, *prerequisite recommendations* for problem solving, and *generating a learning sequence* to study a particular concept [3].

4.1 Navigation Support

The navigation menu is used to navigate through the concepts under consideration.

In order to help the student browse the materials, BITS marks each concept with an appropriate traffic light. These traffic lights are computed dynamically from the Bayesian network and indicate the student's knowledge regarding these topics.

Each concept is marked as belonging to one of the following three categories:

(i) *already known*,
(ii) *ready to learn*, and
(iii) *not ready to learn*.

A concept is considered "already known," if the Bayesian network indicates the probability $p(concept = known|evidence)$ is greater than or equal to 0.70, where evidence is the student's knowledge on previous concepts obtained indirectly from quiz results or directly by the student replying to a query from BITS (see Section 3.2). It should be noted that the choice of 0.70 to indicate a concept is known is subjective. A concept is marked "ready to learn," if the probability $p(concept = known|evidence)$ is less than 0.70 and all of the parent concepts are "already known." Finally, a concept is labelled "not ready to learn," if at least one parent concept is not "already known." Traffic lights are employed as follows: yellow (already known), green (ready to learn), and red (not ready to learn).

When BITS is first started, the concepts are marked with traffic lights based on the initial probabilities obtained from the Bayesian network. The opening screen-shot of BITS is depicted in Figure 5. The navigation menu appears on the left, while a brief introduction to BITS is shown on the right. A student can *preview* a "ready to learn" concept by highlighting it. By doing so, BITS will display a brief description of this topic and why it is important.

Example 3 Recall the entry page for BITS in Figure 5. A student can *preview* the topic "File I/O" by highlighting it. A brief overview of "File I/O" is shown as in Figure *6.*

If the student selects to study this topic, she can press the start learning button. If this topic belongs to a ready to learn concept, the lecture notes for this topic are retrieved from the database and displayed for the user.

Example 4 *If the student selects the "ready to learn" concept "File I/O" in the navigation menu, then BITS displays the lecture notes shown in Figure 3.*

4.2 Prerequisite Recommendations

After reading the lecture notes of a "ready to learn" concept (see Section 4.1), a student may indicate that she does not understand the concept, either directly by answering a

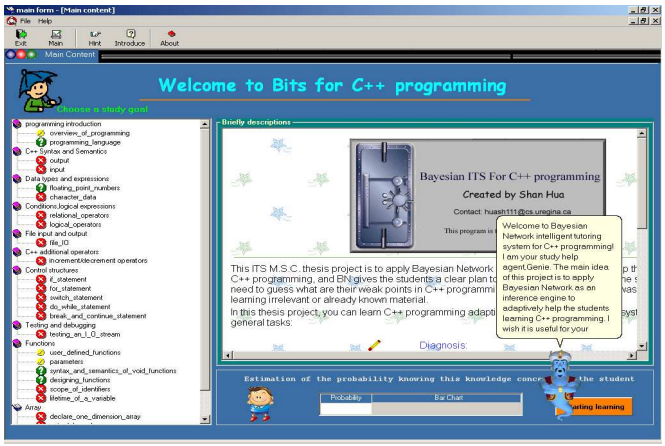

Figure 5. Entry page of BITS with navigation menu (left frame): green means "ready to learn;" yellow means "already known;" red means "not ready to learn," and a brief introduction to BITS (right frame).

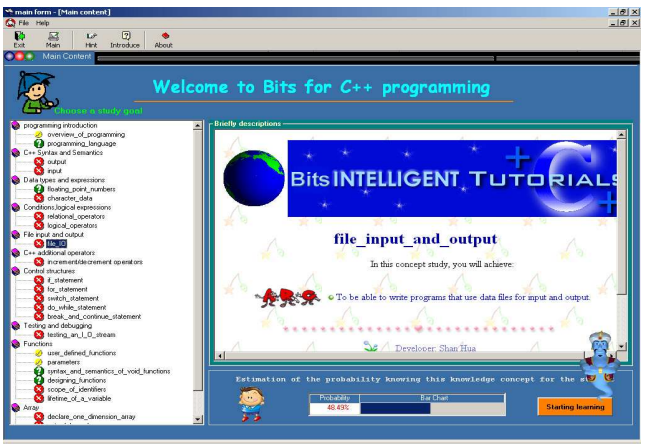

Figure 6. A student can preview the concept "File I/O" (right frame) by highlight it in the navigation menu (left frame).

query or indirectly by an incorrect answer for the corresponding quiz (see Section 3.2).

In these situations, BITS is designed to present links to the prerequisite concepts of this topic, namely, the links to each concept in the parent set of the variables in the Bayesian network. Instead of repeating the problem concept over and over, our approach is useful as it provides the flexibility to revisit the prerequisite concepts to confirm they are indeed understood. Our rationale is that a student may believe that a prerequisite concept is understood when in fact it is not.

Example 5 *Suppose that after reading the lecture notes for the concept "For Loop," the student indicates that she has not understood. BITS will then determine the parent set of the "For Loop" node, i.e., "Variable Assignment," "Relational Operators," "Increment/Decrement Operators" as shown in Figure 1. Finally, BITS will display the links to the lecture notes for these three concepts.*

4.3 Generating Learning Sequences

A student may want to learn a particular topic without learning every single topic previously mentioned. For example, a student may want to learn "File I/O" for an impending exam or assignment deadline. The student would then like to learn the minimum set of concepts in order to understand the chosen concept.

BITS meets this need by generating learning sequences. The student is allowed to select a "not ready to learn" concept in the navigation menu. In this situation, BITS will display a learning sequence for the chosen topic. In other words, all unknown ancestral concepts in the Bayesian network will be shown to the student in a proper sequence for learning.

Example 6 *Suppose the student selects the not ready to learn concept "File I/O" in the navigation menu of Figure 5. Then BITS displays the ancestral concepts in order, grouping by known and unknown, namely, "overview of programming" marked by known, and "programming language," "output," "input" marked by unknown, as depicted in Figure 7. Thereby, the student needs to learn "programming language," "output" and "input" first.*

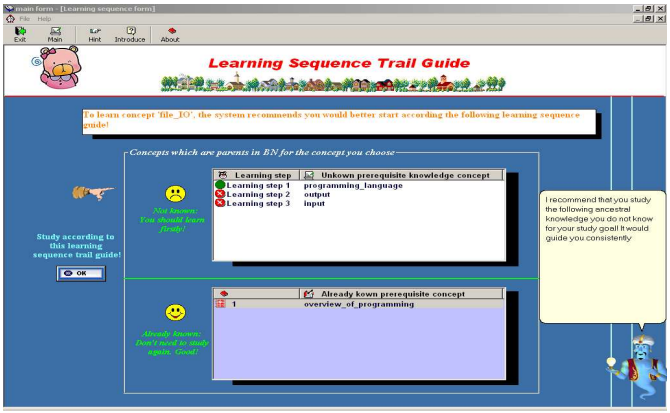

Figure 7. BITS generates a learning sequence for the "not ready to learn" concept "File I/O" in Figure 5.

5 BITS via the Web

All of the course material, including lecture notes, examples and quizzes, is stored in hypermedia format. In this section, we describe the features that allow BITS to be accessed via the Web.

Each quiz consists of interactive Flash multimedia files together with XML documents. More specifically, Flash multimedia files are used to format the questions displayed, while XML documents are used to describe the quiz contents, store solution keys and validate the student's answer.

Example 7 *Recall the quiz for the concept "File I/O" in Figure 4. The XML document for this quiz is shown Figure 8.*

The correct answer for the question is stated in the XML attribute answer. The correct answer for the question in Figure 4 is E, as depicted near the top of Figure 8. This facility allows BITS to provide dynamic validation of the input answer and to proceed with appropriate action.

```
<MainElement>
  <Question answer="E"> Which of the following is NOT one of
   the things a programmer must do in order to use files in a C++
    program?
     <choices>
       <Items> Use a preprocessor directive to include the header
               file fstream.</Items>
       <Items> Declare each file stream in a variable declaration.
       </Items>
       <Items> Prepare each file for reading or writing by calling
               the open function.</Items>
       <Items> Specify the name of the file stream in each input
               or output statement that uses it. </Items>
       <Items> Erase the contents of each output file before running
               the program.</Items>
     </choices>
  </Question>
</MainElement>
```

Figure 8. The XML file for the question on the concept "File I/O" in Figure 4.

BITS uses HTML Web pages to represent the online instructional material. In some cases, multimedia examples using animated Flash files are utilized to illustrate various abstract concepts of C++. The lecture notes and quiz questions are displayed using a Web browser embedded in BITS. BITS also provides the ability to access other C++ programming sites on the Web.

Example 8 *As illustrated in the bottom left corner of Figure 3, BITS allows the user to access C++ sites found on the Web.*

Finally, it is worth mentioning that BITS includes an animated study agent [7] "genie." The goal of the animated study agent is to convey appropriate emotion and encouragement to the student. The feedback given by the student agent is in the form of voice animation and dialog boxes. By providing useful and informative feedback, BITS provides a positive environment for learning.

Example 9 In the bottom right corner of Figure 3, the study agent *informs* the student that she has chosen to learn the concept "File I/O," while the study agent *recommends* that the student first learn three prerequisite concepts in Figure *7.*

6 Related Works

In this section, we briefly contrast BITS with some related works.

Villano [14] first suggested applying Bayesian networks in intelligent tutoring systems. However, Martin and Vanlehn [10] explicitly state that Villano's assessments cannot communicate precisely what a student does not know and cannot identify the components of knowledge that must be taught. BITS, on the other hand, uses yellow traffic lights to indicate known concepts, green traffic lights to indicate ready to learn concepts, and red traffic lights to indicate concepts that the student is not ready to learn.

The *assessment* system proposed in [10] focuses solely on assessing what a student knows. Our *intelligent tutoring system* not only assesses what a student knows, but, in addition, assists the student to navigate the unknown concepts.

Conati et al. [4] developed an intelligent tutoring system for physics. The primary objective of that system, however, is to help the student learn how to *problem solve*. Our purpose, on the other hand, is quite different; it is to help the student *navigate* the course material. Although problem solving is an integral part of computer programming, it is outside the focus of BITS.

It is worth mentioning that Jameson [6] reviewed several frameworks for managing uncertainty in intelligent tutoring systems, including Bayesian networks, the Dempster-Shafer theory of evidence, and fuzzy logic. As Pearl [11] has shown that Bayesian neworks have certain advantages over the other two frameworks, we decided to use Bayesian networks for uncertainty management in BITS.

7 Conclusion

Web Intelligence explores the practical applications of Artificial Intelligence to the next generation of Web-empowered systems [16]. In this paper, we have proposed a Web-based intelligent tutoring system for computer programming by utilizing Bayesian networks, a proven

framework for uncertainty management in Artificial Intelligence [11, 17, 18]. Unlike many traditional tutoring systems which are not interactive nor adaptive [3], our system is intelligent. It can help a student navigate the online course material using traffic lights (see Section 4.1). It can recommend learning goals when a particular concept is not understood (see Section 4.2). Finally, when a student wants to learn a particular concept without learning all of the previous concepts, BITS can present the minimum prerequisite knowledge needed in order to understand the desired concept in the proper learning sequence (see Section 4.3). Our intelligent system has been implemented and will be used in the summer offering of CS110, which is the initial computer programming course at the University of Regina. Empirical studies have shown that individual one-on-one tutoring is the most effective mode of teaching and learning, and intelligent tutoring systems uniquely offer a technology to implement computer-assisted one-on-one tutoring [2]. The work here, together with [5, 7, 8, 9, 15], explicitly demonstrates the practical usefulness of Bayesian networks for Web Intelligence.

References

[1] J. Beck, M. Stern, and E. Haugsjaa, "Applications of AI in education," *ACM Crossroads*, pp. 11-15, 1996.

[2] B. Bloom, "The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring," *Educational Researcher*, 13(6): pp. 4-16, 1984.

[3] P. Brusilovsky, "Adaptive and intelligent technologies for Web-based education," *Special Issue on Intelligent Systems and Teleteaching*, 4: pp. 19-25, 1999.

[4] C. Conati, A. Gertner, and K. VanLehn, "Using Bayesian networks to manage uncertainty in student modeling," *User Modeling and User-Adapted Interaction*, 12(4): pp. 371-417, 2002.

[5] J. Ji, L. Zheng, and C. Liu, "The intelligent electronic shopping system based on Bayesian customer modeling," *First Asia-Pacific Conference on Web Intelligence*, pp. 574-578, 2001.

[6] A. Jameson, "Numerical uncertainty management in user and student modeling: An overview of systems and issues," *User Modeling and user-Adapted Interaction*, pp. 193-251, 1995.

[7] W.L. Johnson, "Pedagogical agents for Web-based learning," *First Asia-Pacific Conference on Web Intelligence*, pp. 43, 2001.

[8] S. Lee, C. Sung, and S. Cho, "An effective conversational agent with user modeling based on Bayesian network," *First Asia-Pacific Conference on Web Intelligence*, pp. 428-432, 2001.

[9] C. Liu, L. Zheng, J. Ji, C. Yang, J. Li, and W. Yang, "Electronic homework on the WWW," *First Asia-Pacific Conference on Web Intelligence*, pp. 540-547, 2001.

[10] J. Martin and K. Vanlehn, "Student assessment using Bayesian nets," *International Journal of Human-Computer Studies*, 42: pp. 575-591, 1995.

[11] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, 1988.

[12] V. Robles, P. Lfarrañaga, E. Menasalvas, M.S. Pérez, and V. Herves, "Improvement of Naive Bayes collaborative filtering using interval estimation," *2nd Annual Asia-Pacific Conference on Web Intelligence*, pp. 168-174, 2003.

[13] V.J. Shute and J. Psotka, "Intelligent tutoring systems: past, present, and future," *Handbook of Research on Educational Communications and Technology*, Macmillan, New York, pp. 570-600, 1996.

[14] M. Villano, "Probabilistic student models: Bayesian belief networks and knowledge space theory," *Proceedings of 2nd International Conference on Intelligence Tutoring System*, pp. 491-498, 1992.

[15] Y. Wang and J. Vassileva, "Bayesian Network-based trust model," *2nd Annual Asia-Pacific Conference on Web Intelligence*, pp. 372-378, 2003.

[16] Web Intelligence Consortium, April 1, 2004, http://wi-consortium.org/

[17] S.K.M. Wong and C.J. Butz, "Constructing the dependency structure of a multi-agent probabilistic network," *IEEE Transactions on Knowledge and Data Engineering*, 13(3): pp. 395-415, 2001.

[18] S.K.M. Wong, C.J. Butz, and D. Wu, "On the implication problem for probabilistic conditional independency," *IEEE Transactions on Systems, Man, and Cybernetics*, Part A: Systems and Humans, 30(6): pp. 785-805, 2000.

[19] J.T. Yao and Y.Y. Yao, "Web-based support systems," *Proceedings of the WI/IAT Workshop on Applications, Products and Services of Web-based Support Systems*, pp. 1-5, 2003.