Prototyping a Student Model for Educational Games

YoungMee Choi*, MoonWon Choo*, and SeongAh Chin*

Abstract: When a pedagogical agent system aims to provide students with interactive help, it needs to know what knowledge the student has and what goals the student is currently trying to achieve. That is, it must do both assessment and plan recognition. These modeling tasks involve a high level of uncertainty when students are allowed to follow various lines of reasoning and are not required to show all their reasoning explicitly. In this paper, the student model for interactive edutainment applications is proposed. This model is based on Bayesian Networks to expose constructs and parameters of rules and propositions pertaining to game and problem solving activities. This student model could be utilized as the emotion generation model for student and agent as well.

Keywords: edutainment, pedagogical agents, emotion generation, student modeling, Intelligent Tutoring Systems, dynamic Bayesian networks.

1. Introduction

With technology rapidly developing in graphics, sound, real-time video and audio, electronic games have become more and more entertaining and enjoyable for kids, as well as adults. Among all the kinds of games, there is a special category – educational games – which has one goal beyond just entertainment, and that is education. Since the 1970s, various educational games have emerged and some of them claimed to have educational effectiveness. However, very few formal evaluations have been conducted to evaluate the actual pedagogical values of these games [2]. At the same time, educational games have been receiving criticism and resistance from both teachers and academics in terms of their effectiveness in education. While educational games are usually successful in increasing student engagement, they often fail in triggering learning. One of the major problems in educational games is derived from ignorance of the personal differences among users. Some researchers found that while boys often enjoy aggression, violence, competition, fast-action, and speed in games, girls enjoy the opportunity to socially interact with others [3]. These different personal interests, plus different knowledge status and learning abilities, often lead to different playing patterns, which result in different needs for individuals who interact with educational games. Also, students may develop game skills without learning the underlying instruc-tional domain; moreover, some students do not access available help even if they have problems playing the game [1]. All of these issues dramatically reduce the educational effects of educational games. A possible solution to these pro-blems is to devise educational games that can provide proactive help tailored to the specific needs of each individual student.

In this paper, we describe our work in making a mathematical educational game, Yut-nori, more effective through an animated pedagogical agent that can provide individualized support to student learning. Yut-nori is a very easy and popular board game in Korea. Since even elementary students can understand and learn the game strategy in a short time, we adopt this game as the frame-work for educational entertainment application suggested here. Currently, individualized support is based on both some simple heuristics and a probabilistic student model. The model tracks a student's behavior during game playing, and uses this to assess the evolution of his/her knowledge as the interaction proceeds. The content of the paper is arranged as follows: Chapter 2 describes the gaming situation; Chapter 3 describes the student model and its variables; Chapter 4 discusses the issues; Chapter 5 presents the conclusions and discusses future work.

2. Gaming Situation

Yut-nori is one of the traditional folk games played in Korea. It has four sticks, each with a flat side and a round side, just like the head and a tail of a coin and game board called Yut-pan. Each player tosses four sticks at a time onto a mat and counts the number of sticks that fall on the flat side or the round side. As a result, five different com-binations are possible when a player tosses four sticks at a time. Each combination has a special name and specifies the movement of markers on the game board called Yut-pan. Each player is given four playing markers called Mals, which can be coins, buttons, poker chips, etc. and takes turns until one player brings all of the four Mals back to home first.

Manuscript received September 30, 2005; accepted November 8, 2005.
* Division of Multimedia, Sungkyul University, Anyang, Korea ({choiym, mchoo,solideo}@sungkyul.edu)

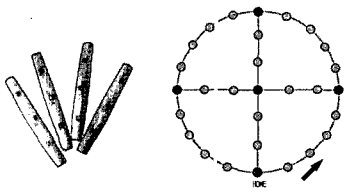

Fig. 1. The four sticks and the game board called Yut-pan (double circled spot represents Home, and dark spots are intersections for taking shortcuts)

In this research, instead of throwing four sticks, each player receives three numbers randomly generated, which must be used in an arithmetic expression where the operations such as addition, subtraction, multiplication, and division as well as parentheses are used. The expression is imposed the constraint that no operator or number can be used more than once and the result will be in the range of 1 to 5. If the four numbers are given, the level of difficulty will raised. In this paper, only three numbers are given. The value of the expression is the numbers of spaces the Mal is moved along the board. The object of the game is to be the first player to land over the Home spot. To make the student's task more complicated than just making the biggest number, there are several kinds of special moves. The intersections occur every five spots around the board and at the center spot. If you land on one of these, you can advance to the spots along the shortcut. Also, if you land on the spot your opponent is occupying, he is bumped back to the starting point which is on the outside of the board.

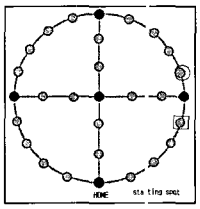

Fig. 2. The game board showing the Mals' moves

Fig. 2 shows a board situation that illustrates some of the complexities of tutoring even in this simple game. Suppose that the student is at spot 4 (surrounded with square) and his opponent is at 6 (surrounded with a circle) and with his numbers (2, 2, 2). If the student makes the expression 2+2/2, resulting in a move of 3, then his current position is at spot 8. In this situation, the student could have made alternative moves: he could have moved 1 (its expression may be 2-(2/2)), landed on the intersection and taken the shortcut; he could have moved 2 (its expression may be 2-2+2) and bumped his opponent.

3. Student Model

The student model's goal is to generate an assessment of students' knowledge on the evaluation of basic arithmetic expression as students play the game in order to allow the pedagogical agent to provide tailored help that stimulates student learning. To generate its assessments, the student model keeps track of the student's behaviors during the game, since such game behaviors are often a direct result of the student's knowledge, or lack thereof.

Modeling students' knowledge in educational games involves a high level of uncertainty [6]. The student model only has access to information such as student moves and tools access, but not to the intermediate mental states that are the causes of the students' actions. It is common for a student to intuitively manage solving some mathematical questions successfully without necessarily understanding the math principles behind it. Thus, analyzing student performance in Yut-nori does not necessarily give an unambiguous insight into the real state of the student's knowledge. A solution to this problem could be to insert in the game more explicit tests of math knowledge. However this would endanger the high level of motivation that an educational game usually inspires exactly because it does not remind students of traditional pedagogical activities. Thus, both Yut-nori and our agent are designed to interrupt game playing as little as possible, making the interpretations of student actions highly ambiguous.

3.1 Variables in the student model

In this paper, a Bayesian Network is used to model the user, since it can generate as accurate an assessment as possible by leveraging any existing information on the user and explicitly express the uncertainty of the user's behavioral predictions when limited information is available. Several random variables are introduced in the Bayesian Network to represent students' behaviors and knowledge.

Expression Nodes E_x: for each arithmetic expression, the student model for that basic arithmetic operation includes a node E_x that models a student's ability to express X to get an optimal move. Each node E_x has two states: Mastered and Unmastered. The Mastered state denotes that the student mastered the expression of X down to its basic math operations. Unmastered denotes that the student does not know how to Fig. out the expression X using the basic math operations, as well as parenthesis.

Parenthesis Nodes P_x: for each arithmetic expression, the student model for that basic arithmetic operation includes a node P_x that models a student's ability to express E_x with parenthesis. This node has two states: Mastered and Unmastered. The Mastered state denotes that the student mastered the usage of parenthesis for E_x to get the legal moves. Unmastered denotes that the student does not know how to Fig. out the expression E_x using parenthesis properly.

Optimal Move Nodes O_x: this node models a student's ability to select E_x to get an optimal move. This node E_x has two states: Mastered and Unmastered. The Mastered state denotes that the student mastered the expression of E_x to select the optimal move. Unmastered denotes that the student does not know how to Fig. out the expression E_x to select the optimal move.

Bump Nodes B_x: for each arithmetic expression E_x, this node models a student's ability to use Bump strategy to win the game. The node B_x has two states: Mastered and Unmastered. The Mastered state denotes that the student mastered the Bump strategy and Unmastered denotes that the student does not know how to use his arithmetic result for bumping the opponent Mal.

Shortcut Nodes S_x: for each arithmetic expression E_x, this node models a student's ability to use Shortcut strategy to win the game. The node S_x has two states: Mastered and Unmastered. The Mastered state denotes that the student mastered the Shortcut strategy and Unmastered denotes that the student does not know how to use his arithmetic result for taking the shortcut toward the Home spot.

Distance Nodes D_x: for each arithmetic expression E_x, this node models a student's ability to use normal Distance strategy to win the game. The node D_x has two states: Mastered and Unmastered. The Mastered state denotes that the student mastered the Distance strategy and Unmastered denotes that the student does not know how to use his arithmetic result for moving his Mal to create a distance from the opponent's spot.

Click Nodes C_x: each node C_x models a student's action of clicking the spot on the game board to move there. Each node C_x has two states: Correct and Wrong. Correct denotes that the student has clicked on a correct spot, i.e. a number from the arithmetic expression which is composed with the given three random numbers. This number may represent the optimal move. Wrong denotes a wrong move, which could represent the illegal numbers which could not make a legal move. The legal moves are selected as a positive integer from 1 to 5. These nodes are evidence nodes, which are only introduced in the model when the corresponding actions occur and are immediately set to either one of their two possible values.

Hint Node H_x: each node H_x denotes a student's action of using the hint window on the expression X. A node H_x has two states – Yes and No. Yes denotes that the student has used the hint window to see all possible arithmetic expressions given three numbers.

3.2 Assumptions underlying the model structure

Before going into detail about how the nodes described above are structured into the student models, we list a set of assumptions that we use to define the structure. Fig. 3 shows the basic dependencies among the nodes which encode the assumptions to be mentioned below. Knowing the arithmetic expression given numbers to calculate the desired move influences the probability of knowing the usage of parenthesis and optimal moves. In particular, our model assumes that if a student knows a particular math expression, he probably knows the game strategies to win the game. We represent the dependencies between nodes as Fig. 3, even if there may be other alternative representations.

If there are multiple parent nodes for a particular PA or

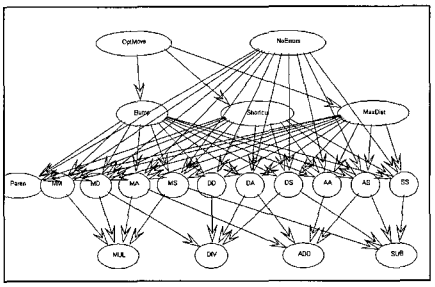

Fig. 3. The dependency between nodes

O node, its conditional probability table is defined in the following way: assume that P or O node has n parent nodes, E_1, E_2,..., E_n. For each assignment of the parent node values, if there are m parent nodes ($0 \le m \le n$) which have the state Unmastered, then the corresponding probability in the conditional probability table for P or O node to be Mastered is calculated using the following equation, which is a little bit modified suggested in [6]:

$$P(X=\text{Mastered}) = p - \{(p-(1-p)/n\} * m.$$

In this equation, p is designed by hand to denote a high probability as "Mastered". The value of p is determined by considering the student's prior knowledge about the relevant knowledge domain. This equation generates the following CPTs:

1. If all the parent nodes are mastered (i.e., m=0), the probability of mastering X is p.
2. If all the parent nodes are Unmastered (i.e., m=n), the probability of mastering X is $1-p$.
3. If $0<m<n$, the probability of X being Mastered is between $1-p$ and p, and it decreases proportionally with the number of Unmastered parent nodes.

More direct evidence on student math knowledge is provided by student actions in the game, such as clicking on a spot on the game board to move or using Hint on some resultant number. These actions are dynamically added as evidence nodes to the student model representing the current game state, and affect the probabilities of knowing the arithmetic expressions and game tactics.

Clicking on a number which is the optimal spot, which means that a student calculates the math expression correctly with the optimal gaming strategies, increases the probability that he knows the relevant math expression well, although this action could also be the result of a lucky guess or of remembering previous moving patterns. A wrong click decreases the probability that the student knows the basic math principles, although it could also be due to an error of distraction.

When a student uses the Hint on E_x at time t-1, and then correctly (incorrectly) moves to the optimal spot at time t, the move provides evidence that the student learned (did not learn) the correct move on E_x at time t-1. Thus, this action provides evidence that the student knows (does not know) how to composite and interpret the expression.

3.3 Updating the student model

The student model for a particular student's interaction with the game must capture the unfolding of this interaction over time, and the corresponding evolution of the student's math knowledge. Traditionally, Dynamic Bayesian Networks (DBN) [4] are extensions of BNs specifically designed to model worlds that change over time. DBN keeps track of variables whose values change over time by representing multiple copies of these variables, one for each time slice [5], and by adding links that represent the temporal dependencies among those variables.

However, it often becomes impractical to maintain in a DBN all the relevant time slices. The rollup mechanism allows maintaining only two time slices to represent the temporal dependencies in a particular domain: the network at slice t-1 is removed after the network for slice t is established. The prior probability of each root node E_x in t is set to the posterior probability of E_x in slice t-1. An example of DBN is shown in Fig. 4, where node H denotes evidence of a student's action at time t-1, while nodes E, P, and O represent the knowledge nodes in the network. Because student knowledge can evolve with interaction, knowledge nodes in time slice t must depend on knowledge nodes in t-1. This can greatly increase the complexity of the corresponding probability tables. Consequently, the updating of the networks can become quite time consuming.

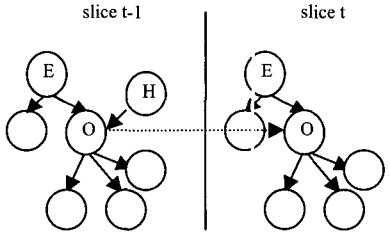

Fig. 4. Two slices of an example DBN.

Thus, an alternative approach to dynamically updating the student model is used here. The following procedure shows how we update the model after an action H occurs [6]:

After action H occurs, let us suppose with value true(=T), a new evidence node H is added to the network. After the network is updated, and before a new action node is taken in, the evidence node H is removed in slice t+1. The CPT for the node O is changed according to the value of H in slice t as follows:

P(O=T|H=T, E=T)=P(O=T|E=T)+d,
P(O=T|H=T, E=F)=P(O=T|E=F)+d,

where d is the weight the action H brings to the assessment of the corresponding knowledge nodes, and it can be either positive or negative, depending on the type of action. If O represents the arithmetic expression nodes E_x, the increment weight d_m is determined as follows.

$$d_m = \frac{1}{n}\sum_{i=1}^{N} p_i - p_m ,$$

where p_m is the probability of the node O and p_i is the probabilities of the occurrences of number i which are computed by applying the expression O.

The probability of knowledge nodes O not directly affected by Click node doesn't remain unchanged, because we could be assumed to slowly forget the unfired knowledge. The S-shaped membership function is utilized to guess the weight to be applied to decay the unfired knowledge.

$$f(x) = \frac{1}{1+e^{-\alpha x}}$$

α is the coefficient for controlling the slope. The range of x can be scaled as 0 to s. The scaling weight is calculated by $\beta = f(P(\cdot \mid M = F)/s)$, where s is scale factor for $f(x)$ and d is the given weight mentioned above. Hence, a node O may be decayed as the following simple rule.

If P(M=F|OM=T) > θ , then P(· |M=F) = P(|M=F) + β d,

θ is the given threshold value for making a decision about decay and is used to determine the timing for Help event.

4. Simulation and Future Works

This paper presented research on using an intelligent pedagogical agent in the Yut-nori educational game to enhance student learning. The agent's actions are based on both some simple strategies and the assessments from a probabilistic student model. This system is currently in the prototyping phase and has to solve lots of issues involved in teaching strategies in game environments. Fig. 5 shows the prototyping interface and the simulation result. The simulation is set up for the test of plausibility of overall performance effectiveness. The main goal of simulation is to show that a pedagogical agent that provides hints to students enhances student learning in the game environment. The 200 simulation sessions are set up and the related parameters are determined, such as d=0.01, θ=0.05, α =1.0, s=20. The Fig. 3 shows that the resultant transitions of several CPTs after 200 simulation sessions reflect relatively correct assessment of the suggested student model. Fig. (a) shows that three CPTs, P(OptimalMove=T|Paren=T), P(OptimalMove=T|Paren=T, Bump=T), and P(OptimalMove=T|Paren=T, Bump=T,MD =T) give the meaningful results that if a student masters the Bump strategy and the usage of multiplication and division operators (MD), he may master the optimal moves in the game board. We set up the parameters for OptimalMove

node to increase the true value monotonically. From this Fig., we can not tell that the Bump strategy node gives more evidence for the optimal move, but we can see the overall conditional values correctly generated, which can promise a highly plausible structure for future usage.

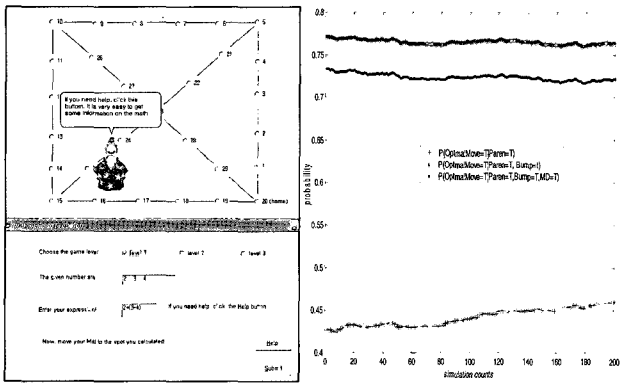

Fig. 5. Screen shot of Yut-nori Interface and the result of simu- lation: +:P(OptimalMove=T|Paren=T), *: P(Optimal Move=T|Paren=T, Bump=T), -: P(OptimalMove= T| Paren=T, Bump=T,MD=T)

The student model is based on Bayesian Networks. To design the precise model, we have to consult with elementary school math teachers in order to reflect the proper knowledge structure of the targeted domain knowledge of our game, which is omitted here. Basing the student model on Bayesian Networks provided us with a sound approach for handling the large amount of uncertainty involved in assessing student knowledge from the interaction with the game. Also, the difficulty levels of game and problem should be counted before presenting hints and to control the game levels. We also have to refine CPTs in the student models. Currently, the numbers and weights in the CPTs are assigned using our subjective estimates. We need to refine these to get more accurate assessments from the empirical model.

Reference

[1] C. Conati, and J. F. Lehman, (1993). "EFH-Soar: Modeling Education in Highly Interactive Micro-worlds". *In Lecture Notes on Artificial Intelligence. Advances in Artificial intelligence*, AI-IA '93 Springer Verlag, Berlin.

[2] J. M. Randel, B.A. Morris, C.D. Wetzel, and B. V. Whitehill, (1992). "The effectiveness of games for educational purposes: *A review of recent research.*" *Simulation & Gaming 23*, 3, 261-276.

[3] R. Burton & J.S. Brown, "An Investigation of computer coaching for informal learning activities," *Intelligent Tutoring Systems, Academic Press. INC.*, 1982.

[4] S. Russell, and P. Norvig, *Artificial Intelligence: A Modern Approach*, by Prentice-Hall, Inc. 1995.

[5] T. Dean, and K. Kanazawa, "A model for reasoning about persistence and causation", *Journal of Computational Intelligence*, Vol.5, 142-150, 1989.

[6] Xiaohong Zhao, "Adaptive Support for Student Learning in Educational Games," *Thesis for master's degree*, The Univ. of British Columbia, 2002.

YoungMee Choi

She received a Ph.D. in Computer Science from Aju University, Suwon, Korea. Since 1994, she has been with the Division of Multimedia, Sungkyul University, Anyang, Korea. Her main research interests include Artificial Intelligence, Software Agents and Multimedia Contents.

MoonWon Choo

He received a Ph.D. in Computer Science from Stevens Institute of Technology, New Jersey, USA. Since 1997, he has been with the Division of Multimedia, Sungkyul University, Anyang, Korea. His main research interests include Computer Vision, Image Processing and User Modeling.

SeongAh Chin

He received a Ph.D. in Computer Science from Stevens Institute of Technology, New Jersey, USA. Since 2001, he has been with the Division of Multimedia, Sungkyul University, Anyang, Korea. His main research interests include Computer Vision, Virtual Reality and Document Analysis.