Artificially Intelligent Adaptive Tutoring System

William Whiteley
Stanford School of Engineering
Stanford, CA 94305
bill.whiteley@gmail.com

Abstract

This paper presents an adaptive tutoring system based on a Markov Decision Process model. Applying this model to a tutoring system is a unique concept in computer based training. After a detailed derivation of the model, results from experiments are presented that show the Markov Decision Process model is an effective tool for adding intelligence to computer based tutoring.

1 Introduction

Intelligent Tutoring Systems are computer based learning systems designed to enhance the learning rate and retention of students. Modern tutoring systems of this type use software algorithms that adapt to the student's estimated knowledge of the material to deliver customized instruction. With these adaptations the tutoring system is able to more effectively guide the student through learning problems.

Computer based teaching systems have a long history dating back to the 1970's starting with the SCHOLAR [1] program in 1970. However, it was not until the late 1980's and 1990's that widespread research and realistic deployments were realized. Some well known intelligent tutoring systems from this time period include the PUMP Algebra Tutor [2], ANDES [3] physics tutor, and the computer literacy software AutoTutor [4]. These types of intelligent tutoring systems were shown to produce learning gains of approximately 0.3 to 1.0 standard deviation units compared with students learning the same content in a classroom [4].

Throughout much of the previous research in this area, the pedagogical side of computer based tutoring systems was often the primary focus. There are notable exceptions of published research that utilize neural networks, but more often the intelligent tutor systems utilized naïve Bayesian statistics [2] to model a student's knowledge and relatively simple heuristics to choose what content to deliver.

In contrast, the project described in this paper utilizes a Markov Decision Process (MDP) to not only model and learn the student's knowledge of a subject but also simultaneously choose the optimal mode of instruction. The MDP framework described is very flexible and contains a number of parameters available to modify the behavior of the MDP to mimic various pedagogical techniques. However, this project will defer the implementation of a specific teaching method and instead focus on the fundamental framework that allows the use of an MDP to implement an adaptive tutoring system.

2 Artificially Intelligent Adaptive Tutoring Framework

The MDP framework described in this paper is designed to tutor a student in a given subject area by instructing them and providing exercises on the subject's supporting concepts. The MDP decides at each step in the process which concept area to cover and whether to use a teaching or questioning mode. This section of the paper contains a detailed description of the MDP used in this project, but first, for completeness, a very brief explanation of Markov Decision Processes is given.

2.2 Markov Decision Process (MDP)

A Markov Decision Process is defined by a tuple (S,A,P_{sa}, γ, R), where S is a set of possible states; A is the set of possible actions; P_{sa} is the state transition probabilities; γ is the discount factor and is between 0 and 1; R is the reward function. On executing action *a* in state *s* the probability of transitioning to state *s'* is given by $P_{sa}(s')$ and the reward is given by $R(s')$. A policy π maps actions to each state in the MDP. The value of any given state under the policy π is the expected sum of discounted rewards obtained by following the policy from the current state forward. The primary goal in MDPs is to find an optimal policy that maximizes the value function in each state of the MDP.

The value function for a state *s* is defined in terms of the well known Bellman equations:

$$V^{\pi}(s) = R(s) + \gamma \sum_{s' \in s} P_{s\pi(s)}(s') V^{\pi}(s')$$

Therefore to find the optimal value function V^* and consequently the optimal policy π^* we solve the following equation for each state:

$$V^*(s) = V^{\pi^*}(s) = R(s) + \max_{a \in A} \gamma \sum_{s' \in s} P_{s\pi(s)}(s') V^*(s')$$

There are a number of methods used to solve this non-linear equation including dynamic programming, policy iteration and value iteration. This paper will utilize the commonly used value iteration method and will provide a brief explanation of that technique next.

2.3 Value Iteration

Given a MDP (S,A,P_{sa}, γ, R), the value iteration algorithm is used to find the optimal value function and is as follows:

1. For each state *s*, initialize $V(s)$ to some starting value
2. Repeat until convergence

{

For each state *s*, update $V(s) = R(s) + \max_{a \in A} \gamma \sum_{s' \in s} P_{s\pi(s)}(s') V(s')$

}

The updates to the value function for each state can be done all at once in a synchronous update or each time the value function is calculated for a given state in an asynchronous manner. Either method will cause the value functions to converge to the optimal value.

2.1 Knowledge Models

The knowledge model used in this project is based on two fundamental assumptions. The first assumption is that one way to show mastery of a subject is the ability to answer random questions from the given subject area. For example, if a student is able to answer five out of ten randomly drawn questions then it is likely that they have a moderate understanding of the material. The second assumption is that when subjects are taught, they are normally broken down into a set of supporting concepts. Figure one is an example of a simple knowledge model of Arithmetic, which is also the subject and model used later in the experimental portion of this paper.

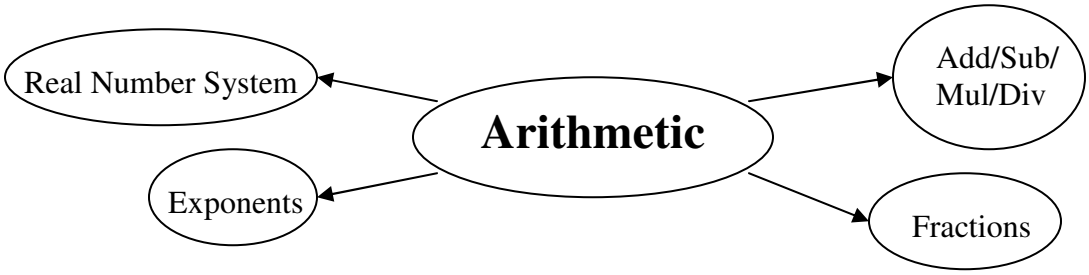

Figure 1 – Example Tutoring Knowledge Model of Math Subject

2.4 Adaptive Tutoring Markov Decision Process

The Tutoring MDP used in this project is designed to teach a student a single subject area consisting of multiple supporting concepts. Through reinforcement learning it derives a student knowledge model for each of the supporting concept areas and uses that model to determine what concept area to spend time on and whether to teach or quiz the student.

2.4.1 States

The tutoring MDP contains a tuple of states {*L, C, IC*} for each concept area. The state labeled *L* is the learning state for a given concept. The *C* state is reached after answering a question correctly and the *IC* state represents answering a question incorrectly. The set of all states for a subject area MDP is given by S: $\{L_1, C_1, IC_1, \ldots, L_n, C_n, IC_n\}$ for *n* supporting concepts.

2.4.2 Actions

The action set for the MDP utilized in this project contains two actions for each subject area: Teach (T) and Question (Q). Thus for a given subject, the set of actions is given by A:$\{T_1, Q_1, \ldots T_n, Q_n\}$ for *n* supporting concepts. Figure two shown below illustrates the actions for a single concept. However, it is important to understand that every action in the action set is possible in all states and the S_0 node in the figure below could represent any state in the MDP.

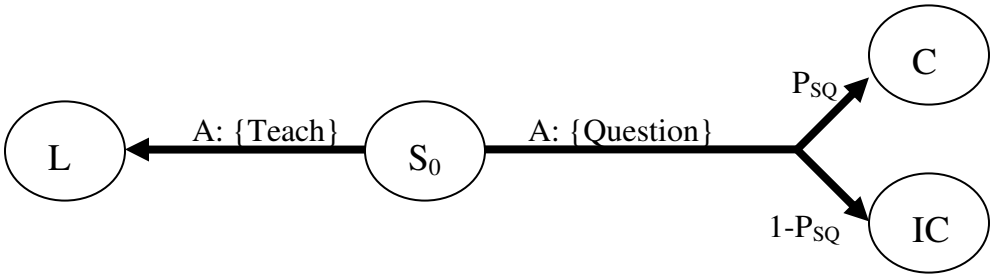

Figure 2 - Graphical Representation of Tutor MDP for a Single Concept

2.4.3 Transition Probabilities

The set of transition probabilities P_{SQ} contains two probabilities for each concept area. The first transition, which is deterministic, is the probability of transitioning to the learn state given the teach action and is equal to one. The second transition is the probability P_{SQ} of answering a random question from the given concept area correctly. Hence, given the *Question (Q)* action for a subject area, the prior probability of transitioning to the *Correct (C)* state is P_{SQ} and the prior probability of transitioning to the *Incorrect (IC)* state is 1- P_{SQ}. This transition probability is used to estimate a student's knowledge of a given concept and its use is illustrated in figure two above.

Initially the transition probabilities P_{SQ} for each of the concept areas are unknown and hence must be learned by the intelligent tutor and furthermore they are strictly dynamic in nature and evolve as the student gains (or possibly

loses) knowledge. This probability is calculated using the simple technique of counting the actual transitions as they occur with one minor exception. When the current state is a learn state, then the P_{SQ} for the learn state's concept area is boosted temporarily while in that learn state. This reflects the reality that immediately after learning a concept, a student is more likely to answer questions correctly in that concept area. This is reflected in the behavior of the MDP and makes the sequence of teach and then quiz in the same concept area very likely.

$$P_{S^iQ} = \begin{cases} \dfrac{\#\,questions\quad answered\quad correctly}{\#\,questions\quad asked} * B & if\left\{Cur\_State = L_{S^i}\right\} \\ \dfrac{\#\,questions\quad answered\quad correctly}{\#\,questions\quad asked} & otherwise \end{cases}$$

Where B is a constant > 1

The initialization of the transition probabilities is a key parameter of the model and significantly impacts the student experience. Starting with high probabilities assumes a high level of student knowledge and conversely low probabilities assume student ignorance. In addition the number of pseudo counts used to initialize the transition probabilities is another key parameter of the model and impacts how fast or slowly the estimate of the student model can change (how long it takes to convince the tutor that you are smart or ignorant).

2.4.4 Discount Factor

The discount factor is used in a MDP to enforce the principal that immediate rewards are more valuable than future rewards. This encourages the tutor agent to act greedily when choosing actions. The value of the discount factor ranges from zero to one. Various values were examined for the tutor MDP and a value of 0.65 was chosen, which allowed rapid convergence and scaled the value functions to values near zero.

2.4.5 Reward Function

The reward function is structured to provide a large positive reward for correct answer states, small positive rewards for learning states, and a large fixed negative reward for incorrect answer states. However, the positive rewards for learning and correct answers are necessarily diminished as a student's knowledge increases. In essence this presumes that a correct answer to a question which a student finds difficult is more valuable than a question which the student answers easily. This dynamic allows the tutor to transition from concept areas where the student's knowledge is high and many questions have been answered to more difficult areas. To accomplish this, the reward function for a correct answer is proportional to the probability ($1\text{-}P_{SQ}$) of transitioning to the *incorrect answer* state and uses the following equation where K is a fixed constant:

$$R(s) = \begin{cases} K * (1 - P_{SQ}) & s \in C \\ K * (1 - P_{SQ}) * .2 & s \in L \\ -1 & s \in IC \end{cases}$$

3 Bill's Intelligent Tutoring System (BITS)

BITS uses the MDP model derived in the previous section to tutor students in arithmetic using the supporting concepts of real numbers, addition / subtraction / multiplication / division, fractions and exponents (see figure one). In the teaching mode, BITS displays the rules used to solve problems from the particular concept area and then shows two worked examples. In the quiz mode, BITS displays a multiple choice math problem and, after receiving an answer from the student, shows the correct answer.

In this project two students were first given a separate 40 question baseline test of the arithmetic concepts with no teaching to assess their prior knowledge of the concept areas and provide a measuring stick to judge the tutoring algorithm. Then on a separate occasion each student was engaged in a fifty action BITS tutoring session. The results of this session are shown in the following section.

Given the unwillingness of my drafted students to continuously be subjected to math problems, the key parameters for BITS were derived from running simulations with imaginary students. Ultimately the transition probabilities were initialized to 0.5 and the number of pseudo counts used for initialization was two. These parameters allow the potential for significant progress across the four concept areas in only fifty actions. In a more realistic application where the tutoring spans many sessions, these initial parameters would likely be chosen a little differently.

4 BITS Experimental Results

The BITS experimental results for a single student's tutor session are shown in the two figures below. Figure three shows the transition probabilities for each of the concept areas over the tutor session along with each area's baseline results. The figure shows how the MDP quickly converges to high estimates of student knowledge in areas where the student showed a high baseline of understanding. It also correctly learns the area where the student was a priori weak in understanding. Another interesting observation from figure three is that for each of the concept areas that showed a relatively weak baseline, the student ultimately demonstrated learned knowledge above the initial baseline. Figure four shows the number of actions dedicated to each of the concept areas. This figure clearly shows that the tutor spent more time on the areas of weak understanding and less time on areas of strong understanding.

Overall the BITS MDP performed well with both of the test students given a relatively simple subject. In future work more features could be added to enhance the pedagogical effectiveness of the model. For example a parameter could be used to induce a weak sequencing of the topics to make it more likely to teach concepts in a particular order. Also questions could be sub-divided into multiple difficulty levels to increase the granularity of the estimate of the student's knowledge. There are many other interesting additions possible for the MDP tutoring model.

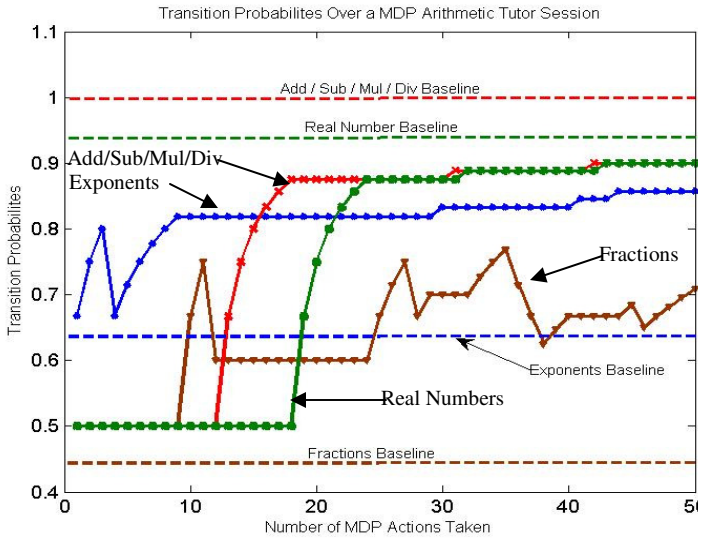

Figure 3 - Learning Results from MDP tutor

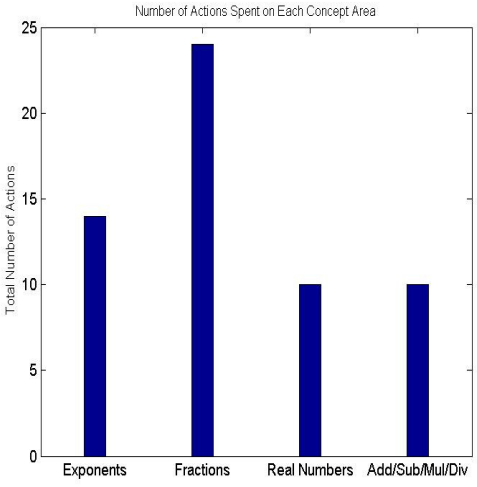

Figure 4- MDP tutor focuses on knowledge areas that need improvement

8 References

[1] Carbonell (1970). Al in CA!: An artificial intelligence approach to computer-assisted instruction. IEEE Transactions on Man-Machine Systems, 11,190-202.

[2] A.T. Corbett, J.R. Anderson, K.R. Koedinger. Intelligent Tutoring Systems. *Handbook of Human-Computer Interaction, Second, Completely Revised Edition,* Chapter 37, 1997.

[3] K. VanLehn, C. Lynch, K. Schulze. The Andes Physics Tutoring System: Lessons Learned. International Journal of Artificial Intelligence in Education, 15(3), 2005.

[4] A.C. Grasser, P. Chipman, B.C. Hayes, A. Olney. Auto Tutor: an intelligent tutoring system with mixed-initiative dialogue. Education, IEE Transactions on, Volume 48, Issue 4, 2005.