Extracting Student Models for Intelligent Tutoring Systems

John C. Stamper[1], Tiffany Barnes[1], Marvin Croy[2]

Department of Computer Science[1], Department of Philosophy[2]
University of North Carolina at Charlotte
9201 University City Blvd., Charlotte, NC 28223
{jcstampe, tbarnes2, mjcroy}@uncc.edu

Introduction

Intelligent Tutoring Systems (ITSs) that adapt to an individual student's needs have been shown to be extremely effective, showing significant improvement in achievement over non-adaptive instruction (Murray 1999). The most successful of these systems require the construction of complex cognitive models that are applicable only to a specific tutorial in a specific field, requiring the time of experts to create and test these models on students. In order to achieve the benefits that ITSs provide, we must find a way to simplify their creation. Therefore, we are creating a framework to automate the generation of ITS student models. The goal is to provide a simple way to allow developers of computer-based training (CBT) to add adaptive capabilities with minimal work while still maintaining the effectiveness of a true ITS.

Background and Related Work

Student and domain knowledge models are central tools for adaptation, but creating these models is the bottleneck in time and effort in creating ITSs. One system, REDEEM, allows teachers to apply their own teaching strategies to an existing CBT system (Ainsworth 2003). This can be done quickly and without the need for computer experts, and its performance lies between that of a non-expert human tutor and that of the best ITSs, which can achieve a 1-sigma increase in student scores. An ITS framework, known as CTAT, has also been successfully used to lessen the development time of an ITS by allowing a user to input example problems and the system generates the rules (Aleven 2006). Other systems, including ASSERT, have replaced the need for expert model design, using theory refinement to learn student models from behavior, and generate feedback for remediation (Baffes 1996). Although these systems are making progress in the right direction, we believe that using educational data-mining techniques to build the cognitive model from existing CBTs can be less time consuming and will produce effective student models.

The successful Cognitive Tutors created by Carnegie Learning use a cognitive model based on Adaptive Character of Thought (ACT-R) theory (Anderson 1995). Our system is modeled on ACT-R, using a cognitive architecture that uses production rules to model student problem solving processes. Two main processes are used to adapt ITS behavior to the student. The first is called Model Tracing, which is used to provide feedback while the student is solving a particular problem. The second is Knowledge Tracing, which is used to track a student's overall learning and performance.

Model Tracing tracks a student's progress through a particular problem, matching their steps to production rules, and providing feedback if they proceed down an unsuccessful path. Production rules can be "good" rules that are correctly applied, or "bad" rules, which are generally student misconceptions. We used data from an existing CBT to test the idea of using Markov Decision Processes to learn production rules for Model Tracing (Stamper 2006). We compared the resulting production rule sets to those generated by experts. Our methods were able to extract all expert-predicted approaches, both correct and incorrect, while also revealing an extensive set of incorrect approaches that were not predicted by experts.

Knowledge Tracing tracks a student's overall learning of concepts within the system. Typically, ITS designers create a skill matrix to relate problems to their respective knowledge components. Alternatively, these skill matrices can be learned using educational data-mining techniques such as the q-matrix method (Barnes 2006). The q-matrix method is a data-mining algorithm that extracts a q-matrix from student assessment data, which will discover "concepts" that influence student behavior (Barnes 2006). This method has been successfully applied to learning the concepts underlying an online physics tutor, and these concepts compare well to the expert-derived Facets that were used to create the tutor (Barnes 2006). We have also performed a 10-fold cross-validation on data from 284 college students using a discrete math CBT across three semesters. The resulting error rates between the training and the test sets were always at or below a 10% threshold, therefore the resulting q-matrix models are acceptable for

predicting student concept states. Based on this evidence, we believe that the q-matrix method can be used to automatically update the Knowledge Tracing model.

System Design

Our system will be comprised of a Knowledge Tracing Module (KTM) that will assess and direct student progress, and a Model Tracing Module (MTM) to provide problem-specific feedback. These modules will be added to an existing graphical CBT called Deep Thought (DT). This system allows students to work forward or backwards in solving propositional logic proofs. Over 200 students use this program each semester as part of a philosophy course.

The KTM, as described above, will make decisions based on a student state relative to the overall skill matrix for the problems being solved. The skill matrix will be initially derived from historical DT data, and will be continually updated as new students use the system. The MTM will match student performance to a production rule system, initially derived using the MDP methods from our previous work, and will also be continually updated to include new rules to describe student behavior.

The probabilities associated with a particular path through the production rule system are updated by every step that a student makes. The system then predicts a student's next action based on both the current rules associated with the problem being worked on and the student's current classification, which is derived from the KTM. When probabilities are reliable for a student solution path, feedback will be given whenever the student appears to be heading off of a successful path while attempting to solve a problem. If the student starts down a path that has not been seen before, there may be no rules associated with the path. In that case, the student will be allowed to traverse that path and new rules will be generated.

Student knowledge is tracked via the probability that a particular student understands a particular group of concepts. Generating conceptual information is accomplished through the q-matrix educational data-mining algorithm applied to the student data already accumulated. The KTM continually updates the q-matrix in the background as more student data is gathered. This conceptual information is used to classify students into knowledge states, which will be used in turn by the MTM. Based on the student's success or failure on each action, the student's knowledge state is updated to reflect the probability that he/she understands each concept.

Conclusions and Future Work

We have demonstrated that we can automatically generate the individual parts of our model, and have designed the combined model for integration into a specific CBT system. We can derive production rules for logic proofs using MDPs, and believe we will be able to generate appropriate feedback based on these rules. We can derive a concept model using the q-matrix method, and use this model to assess and predict student knowledge. These basic components provide the foundations for building Knowledge and Model Tracing into a CBT and may provide the full adaptive capability of a traditional ITS.

Initially, our system will focus on building a stable set of rules and concepts. We plan to validate the predictions the system generates about student progress. We will also create a feedback generation system and an interface to allow educators to edit this feedback. Later, we will compare performance using our new system to that using DT alone to determine if improvements are made.

References

Ainsworth, S., Major, N., Grimshaw, S., Hayes, M., Underwood, J., Williams, B. & Wood, D. 2003. REDEEM: Simple Intelligent Tutoring Systems From Usable Tools, in T. Murray, S. Blessing & S. Ainsworth (eds). *Advanced Tools for Advanced Technology Learning Environments*. pp. 205-232. Amsterdam: Kluwer.

Aleven, V., McLaren, B. M., Sewall, J., & Koedinger, K. (2006). The Cognitive Tutor Authoring Tools (CTAT): Preliminary evaluation of efficiency gains. In M. Ikeda, K. D. Ashley, & T. W. Chan (Eds.), *Proc. Intelligent Tutoring Systems Conference*, (pp. 61-70). Berlin: Springer Verlag.

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. 1995. Cognitive Tutors: Lessons Learned. *Journal of the Learning Sciences*, 4(2), 167-207.

Baffes, P. & R.J. Mooney. 1996. A Novel Application of Theory Refinement to Student Modeling. *Proc. 13th National Conference on Artificial Intelligence* (AAAI-96), pp. 403-408, Portland, OR.

Barnes, T., Stamper, J., Madhyastha, T. 2006. Comparative Analysis of Concept Derivation Using the Q-matrix Method and Facets. In *Educational Data Mining: Papers from the 2006 AAAI Workshop, ed. Joseph E. Beck, 21-30. Technical Report WS-06-05*. AAAI, Menlo Park, CA.

Murray, Tom. 1999. Authoring intelligent tutoring systems: An analysis of the state of the art. *Intl. Journal of Artificial Intelligence in Education*, 10: 98-129.

Stamper, J. 2006. Automating the Generation of Production Rules for Intelligent Tutoring Systems. *Proc. 9th Intl. Conf. on Interactive Computer Aided Learning* (ICL2006), pp. 206-212,Villach, Austria.