Designing a Data-Driven Tutor Authoring Tool for CS Educators

Kelly Rivers
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA, 15213, USA
krivers@cs.cmu.edu

ABSTRACT

Intelligent Tutoring Systems are highly effective at helping students learn, but have required intensive amounts of development time in the past, keeping teachers from making their own. Data-driven tutoring has made it possible to build these tutors more efficiently. For my thesis work, I intend to build an authoring tool for data-driven tutors that is designed to be used by computer science teachers. I plan to design this system based on data gathered in interviews with CS educators and evaluate it on its usability for new users.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Information Science Education – *computer science education.*

Keywords

data-driven tutoring, teacher adoption of technology, usability, intelligent tutoring system authoring tools

1. PROGRAM CONTEXT

I have just completed the fourth year of my PhD program in Human-Computer Interaction at Carnegie Mellon University. This program has a primary focus in HCI, but also has a large contingent of researchers that specialize in the learning sciences, especially intelligent tutoring systems (ITSs). I have completed all of my classes and teaching requirements, and I intend to propose my thesis in late summer/early fall 2015.

In the past four years I have worked with my advisor, Ken Koedinger, to design and build the Intelligent Teaching Assistant for Programming (ITAP). This system uses collected student solutions to automatically generate hints for code-writing programming problems. I have done a technical evaluation of ITAP by testing its performance on old student data, and I've run a small lab study to determine how students interact with hints in a programming editor. For future work, I plan to continue the development of ITAP, test it empirically with large in-class studies, and explore the potential for practical tutor development in the field by designing an authoring tool for teachers.

ICER '15, August 09-13, 2015, Omaha, NE, USA
ACM 978-1-4503-3630-7/15/08.
http://dx.doi.org/10.1145/2787622.2787749

2. CONTEXT AND MOTIVATION

Writing code is an essential skill that every novice programmer needs to learn. The process of learning to write code is filled with many struggles, but students do better when they get help with the 'insurmountable' problems, especially when that help arrives quickly [3]. However, teachers cannot spend every moment of their time with their studsents helping them get unstuck. Having an automated source of assistance might, therefore, greatly assist students who are learning how to program.

ITSs have been shown to be very effective in assisting students, and have even helped students learn programming more quickly and thoroughly in the past [1]. However, the time and effort required to build a tutoring system is prohibitively large, and almost all of the tutors that have been created so far have been made by research groups or companies, not teachers [4]. Perhaps this is why ITSs have not been widely adopted in introductory programming classes, despite their great potential.

My advisor and I have designed ITAP, a new kind of intelligent tutoring system for programming which can generate hints automatically for students in open-ended code writing environments [8]. This system has the potential to assist students who are working on practice problems, and can help students get 'un-stuck' by giving them next-step hints that move them closer to a correct state. It can also be used to generate tutored problems rapidly with little author input needed. Now that we have developed ITAP to the point that it is usable in real classroom settings, we want to determine whether it can be used to help teachers extend the amount of automated support provided in their classrooms.

3. BACKGROUND & RELATED WORK

Data-driven tutoring for programming has been expanding as a subfield of intelligent tutoring systems over the past few years, with many different researchers creating new techniques to automatically generate hints. However, most of the systems (including ours) have only been evaluated on collected student problem-solving traces, and the ones that are being tested on real students are implemented in online learning environments such as MOOCs [6,7], not in individual classrooms.

In the broader field of ITS authoring by teachers, there have been a few attempts to make tutor authoring more accessible. The introduction of authoring by demonstration made GUI tutor authoring take drastically less time than before [5], and more structured content provision editors made it easier for teachers to create new items for language-learning systems [2]. There do not appear to be any studies which have directly evaluated teacher interest in generating intelligent tutors independently.

4. STATEMENT OF THESIS/PROBLEM

My broad research question is: will CS educators use more adaptive practice problems in their classes if they have the ability to create the adaptive content themselves? To address this question, I will also investigate the following two questions in my thesis work:

- What do teachers want their practice problems to look like, and how does this relate to CSED best practices? What kinds of features do teachers want within programming tutors?
- What are the usability concerns that need to be considered when designing a tutor authoring tool for teachers?

5. RESEARCH GOALS & METHODS

For my thesis work, I plan to pursue three primary research phases. The first research phase will address the question of what needs teachers actually have, and what features they require in their own tutored problems. To understand this, I plan to do interviews with CS educators to understand what kinds of educational technology they currently use and whether/how they assign practice work for their students to complete. I also plan to distribute surveys in order to establish a broader understanding of the needs of the population. The result of this phase should be a list of needs and wants, and a description of the current state of the field of practice problems in introductory computer science.

The second phase will involve the implementation of the data-driven tutor authoring tool. In this phase, I intend to take the current ITAP system and embed it in a web interface, and design a system that lets teachers input problem data and modify the resulting tutor's behavior. The system should be designed based on the expressed needs of the teachers (from Phase 1). Throughout the design process, I intend to test prototypes of the system with CS educators, to ensure that the authoring tool is progressing as expected. The result of this phase will be a functional authoring tool.

The third and final phase will involve a usability evaluation of the authoring tool. I will ask CS educators to create their own tutored problems in a lab environment, to study how they create problems, whether they have any difficulties, and which parts of the system are used the most. I will ask the educators to use the resulting problems in their classrooms, and will collect data on how many of the resulting tutored problems are actually accessed by students, and how much time the students spend solving problems in the system. The result of this will be an evaluation of the usability of the system, both in terms of system usability for tutor creation and the usability of the tutored problems that are developed.

6. DISSERTATION STATUS

As was stated before, I have spent most of my research program up until now implementing ITAP, a system which allows data-driven creation of programming tutors. ITAP will form the backend of this project, as it makes it possible for teachers to create tutored problems quickly and efficiently. My recent work has been focused on making ITAP work reliably and well, even when given little data to start with, to reduce the amount of work tutor authors would need to do to create tutored problems. I have also been establishing connections with CS educators across many different universities, and many of these educators have expressed interest in using ITAP; I hope to build a subject pool that extends beyond my own university with the help of these connections.

7. EXPECTED CONTRIBUTIONS

At the conclusion of my thesis work, I plan to have developed an online authoring tool that will have been proven to be both usable and effective, a tool which teachers can easily use to generate tutored problems personalized to their curricula. My hope is that making such a tool open-access and publicized across the CS education community could make more CS teachers aware of the potential intelligent tutoring systems have for improving student learning. It is difficult to say now how effective these tutored problems will be (we hope to determine in the near future whether ITAP improves learning outcomes in a classroom setting), but at least they could potentially reduce student frustration, and maybe make practice a more enjoyable experience for students in introductory programming courses everywhere.

8. ACKNOWLEDGEMENTS

This work was supported in part by Graduate Training Grant awarded to Carnegie Mellon University by the Department of Education (# R305B090023).

9. REFERENCES

[1] Anderson, J. R., Conrad, F. G., & Corbett, A. T. (1989). Skill acquisition and the LISP tutor. *Cognitive Science*, 13(4), 467-505.

[2] Brusilovsky, P., Knapp, J., & Gamper, J. (2006). Supporting teachers as content authors in intelligent educational systems. *International Journal of Knowledge and Learning*, *2*(3), 191-215.

[3] Carter, J., Dewan, P., & Pichiliani, M. (2015). Towards Incremental Separation of Surmountable and Insurmountable Programming Difficulties. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (pp. 241-246).

[4] Folsom-Kovarik, J. T., Schatz, S., & Nicholson, D. (2010). Plan ahead: Pricing ITS learner models. In *Proceedings of the 19th Behavior Representation in Modeling & Simulation (BRIMS) Conference* (pp. 47-54).

[5] Koedinger, K. R., Aleven, V., Heffernan. T., McLaren, B. & Hockenberry, M. (2004). Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behavior by Demonstration. In *the Proceedings of 7th Annual Intelligent Tutoring Systems Conference* (pp. 162-174).

[6] Perelman, D., Gulwani, S. & Grossman, D. (2014). Test-Driven Synthesis for Automated Feedback for Introductory Computer Science Assignments. In *Data Mining for Educational Assessment and Feedback (ASSESS 2014)*.

[7] Piech, C., Sahami, M., Huang, J., & Guibas, L. (2015). Autonomously Generating Hints by Inferring Problem Solving Policies. In *Proceedings of the Second (2015) ACM Conference on Learning@Scale* (pp. 195-204).

[8] Rivers, K., and Koedinger, K. (2013). Automatic Generation of Programming Feedback: A Data-Driven Approach. In *Proceedings of the Workshops at the 16th International Conference on Artificial Intelligence in Education AIED 2013* (pp. 4.50-4.59).