Intelligent Hinting and Affect Detection in SAGE

Chengwei Bian and Mengqiao Zhang

COMS E6901, Section 14

October 1st, 2017

Table of Contents

1 Introduction

The education for students is a widely studied and controversial topic. Researchers around the world keep trying to improve and perfect the teaching approach, which not only allow students to learn knowledge, but also feel enjoyable when they are learning. One of the main goals of educational research is the development of appropriate teaching formats for computer science and computational thinking concepts, since these basic concepts are often challenging for students to learn in general classrooms.

SAGE is a system designed to immerse students in an enjoyable, game-like learning experience [1]. It offers adaptable features and feedback in order to maximize engagement and minimize the risk of negative emotions including boredom, frustration, and confusion. The intelligent tutoring system in SAGE is designed to provide immediate and customized instruction or feedback to learners,usually without requiring intervention from a human teacher.

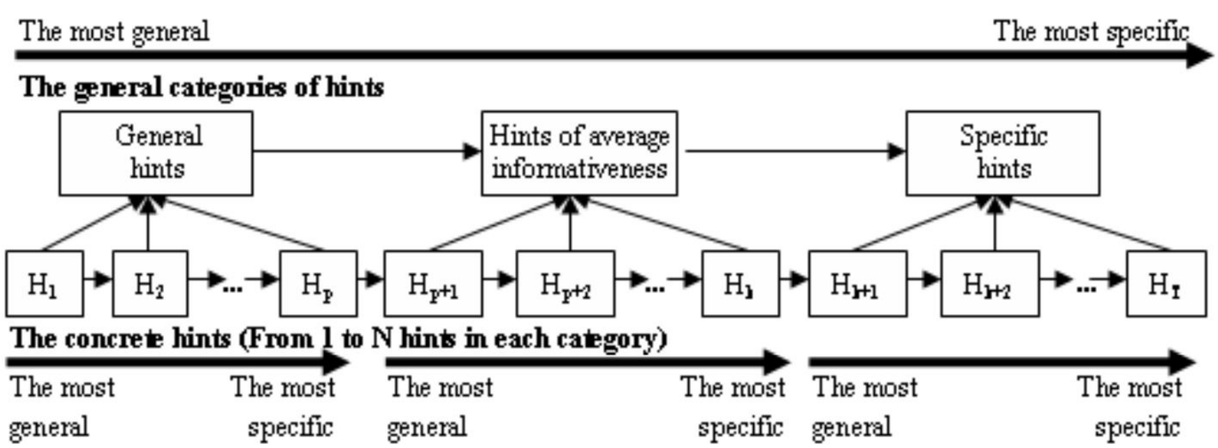
Our primary motivation for this part of the project is to improve the effectiveness of the existing hinting system in Social Addictive Gameful Engineering (SAGE) by adding more functionalities. The major focuses consist of three parts: first, making on-demand hinting available for students; second, increasing hinting types by providing general hints as well as specific hints; and third, identifying features related to student affective states and experimentally finding their relationship, and then use the model to compute when automatic hinting is needed to prevent student from frustration.

2 Related Work

2.1 "Advances in Intelligent Tutoring Systems: Problem-solving Modes and Model of Hints" [2]

This paper focuses on the issues of providing an adaptive support for learners in intelligent tutoring systems when solving practical problems. The results of the analysis shows that the hinting model used in this paper, which divides the hint into three general hint categories, provides greater adaptive abilities to the intelligent tutoring system and helps improve student learning efficacy.

This model divides hints into two layers, namely the layer of the general hint categories and the layer of hints within the general categories. Among the layer of general hints, there are three categories: general hint, hints of average informativeness, and specific hints. When the user first request a hint, the most general hint will be presented. If further hinting request is made, a subsequent (more specific) hint will be given. This process continues until the most specific hint is reached. The mechanism is shown in the picture below.



2.2 “Developing a Generalizable Detector of When Students Game the System” [3]

In this paper, the researchers point out that some students, when working in an intelligent tutoring system, attempt to succeed in the system by exploiting properties of the system rather than by learning the material itself. And the author presents a detector that can accurately detect whether a student is gaming the system when solving mathematical problems under the environment of Cognitive Tutor. Furthermore, the detector is also capable of predicting when students are gaming the system, which may be utilized to suggest appropriate interventions by the intelligent tutoring system in order to facilitate student learning.

After comparing results from other intelligent tutoring systems which have the functionality of detecting gaming, the researchers concluded that the gaming phenomenon is fairly robust and exists among many intelligent tutoring systems. However, it is still unclear to what degree the two distinct types of gaming - harmful gaming, which is associated with poor learning outcomes, and the other gaming behaviours, which do not significantly affect learning - exist in these systems.

2.3 “Early Prediction of Student Frustration” [4]

Scott et al. collected training data by closely monitoring features including temporal features, locational features, intentional feature and physiological response when students interact with the interactive task-oriented learning environment Crystal Island. Besides, they designed a “self-report emotion dialog” box which will show up periodically to ask participants to choose their affective state, which serves as class labels to induce frustration model.

They developed frustration model by both sequential model as n-grams, as well as non-sequential modeling techniques like naïve Bayes, decision trees, and support vector machines. And they utilized convergence point information gained by n-grams model to enable non-sequential models to make early predictions, approximately 35 seconds prior to the self-reported affective state. And test results showed that the induced frustration recognition model was both efficient and accurate, with decision tree model achieving the highest accuracy of 88.8% and precision of 88.7%.

2.4 “Coarse-Grained Detection of Student Frustration in an Introductory Programming Course” [5]

The researchers attempted to automatically detect student frustration when students are learning introductory java programming under BlueJ environment.

They collected student interaction data including students’ average number of consecutive compilations with the same edit location, average number of consecutive pairs with the same error, the average time between compilations, and average number of errors. And during lab sessions, they continuously observed students and classified their behaviour into different categories.

Based on the collected data and labels, they built a linear regression model to detect frustration by using Weka. The result indicates that the frustration detection model was able to predict average frustration during each lab session better than would be expected by chance, whereas the detector performance to detect frustration on a per-lab basis was unstable. It is suggested that more data such as keystroke and mouse movement data could also be utilized as well as the coarse-grained semantic data to develop a finer-grained detector for frustration.

3 Proposal

3.1 On-demand Hinting

Currently, our intelligent tutoring system only supports automatic hinting, so in order to provide users with better user experience, like many well-established intelligent tutoring systems do, the main goal of this part of the project is to add the on-demand hinting function to the current hinting system.

This objective can be separated into two parts: firstly, we are going to add an on-demand hinting button to the system, which will give students the option to click on a hint button to receive on-demand feedback when needed. One challenge for this part is that now the intelligent tutoring system does not provide hints at every project state, especially at the very beginning of a new project. So we will try to revert to the latest BlockList variable to check if any hints are available based on the previous moves. If there is still no hint available, clicking the button could simply return “Try moving some blocks around first”.

Secondly, we are going to build a mechanism that will discourage students from gaming the system. A well-established concern about the on-demand hinting function pointed out that some students tend to “game the system” when they work in an interactive learning environment, so that they could figure out the answer easily from system help instead of putting much of their own effort into learning the material and applying their knowledge. So in order to stop students abusing the hint button, we are going to add some punishment mechanism (like setting an on-demand hinting time limits) to the current system.

3.2 Hinting Types

This portion of the project aims to modify and expand current method that a hint is presented to the students, which is suggesting which block to be taken in the next step by shaking the specific block. Instead, we are going to provide users with a more complete hinting system. The hints are divided into three categories, general text hints, category-level block hint, and the specific hint, which are ranged from less informative to more informative.

At first, the intelligent system allows the learner to receive a more general text hints that do not mention which blocks to use next. Further requesting help during problem-solving, the learner will receive category-level hint like shaking several correct blocks or incorrect blocks. For example, we could infer the right block type from the right block and shake several blocks of that type. If after receiving of a category-level hint, the learner is still not capable to execute a correct action, he will be presented with a specific hint, i.e. the right block will shake.

Such approach starts to present the learner with less informative hints, and as the learner timely receives a more informative hint providing help, they will be less likely to get frustrated and thus gain more pleasant and gameful learning experience [2].

3.3 Frustration Detection

Researchers found that frustration will likely to lead to student disengagement from learning. And hence effectively detecting and predicting frustration and intervene appropriately either from teacher or intelligent tutor can hopefully keep student engaged and motivated in the task [6].

Several approaches have been used in intelligent tutoring systems to detect frustration, including face-based emotion recognition, analyzing the data from physical or physiological sensors, and mining the system's log file. Among these methods, data-mining approaches have reported high accuracy in predicting frustration in many intelligent tutoring systems [7], and will not cause pressure for the students unlike some other approaches. Therefore we plan to adopt an interaction based data-mining approach to detect frustration in SAGE.

As suggested by Sambhav and Allison, the following features could be collected to develop the frustration model:

- Mean and standard deviation (over all blocks used) of the number of times the student has moved each individual block
- Mean and standard deviation (over all blocks used) of the number of times the student has modified the parameters of each individual block
- Mean and standard deviation of time elapsed since the student's last action
- Number of actions a student has taken in the last *x* seconds
- Number of times a block is removed from the scripts area

Empirical data will have to be obtained to develop the frustration model and test model accuracy and precision. To obtain information about affective states, self-reporting mechanism can be considered. Participants in the experiment could be asked to choose their affective states periodically when they are solving the task. In case that empirical data cannot be obtained due to time constraint and progress of SAGE MVP, mock data will be used to perform the following steps.

After obtaining data, machine learning methods like decision tree, support vector machine and naive Bayes could be used to learn from the labeled datasets and model frustration. Once the frustration model can be obtained, unprompted hinting frequency could be adjusted according to student frustration state so that these interventions could maximize learning effectiveness and maintain a state of flow for student. And information about student's overall affective state during a project could also be utilized to the outer loop of the intelligent tutoring system, for example, recommending a less challenging project if a student frequently feel frustrated in the previous project.

4 Timeline

Milestone	Estimated Date
Environment setup	9.29 - 10.12
Add on-demand hinting function, and come up with basic model for detecting frustration	10.13 - 10.26
Implement different hinting types, and enable prompting periodically affective states input from student	10.27 - 11.9
Midterm report	11.10 - 11.23
Try to get real world data from a group of students to develop and evaluate frustration detection model	11.24 - 12.7
Final report	12.8 - 12.21

5 References

[1] Bender, J. (2015). Tooling Scratch: Designing a Collaborative Game-Based Learning System to Infuse Computational Thinking within Grade 6-8 Curricula.

[2] Anohina, A. (2007). Advances in intelligent tutoring systems: problem-solving modes and model of hints. *International Journal of Computers Communications & Control*, *2*(1), 48-55.

[3] d Baker, R. S., Corbett, A. T., Roll, I., & Koedinger, K. R. (2008). Developing a generalizable detector of when students game the system. *User Modeling and User-Adapted Interaction*, *18*(3), 287-314.

[4] Mcquiggan, S. W., Lee, S., & Lester, J. C. (2007, September). Early prediction of student frustration. In *International Conference on Affective Computing and Intelligent Interaction* (pp. 698-709). Springer, Berlin, Heidelberg.

[5] Rodrigo, M. M. T., & Baker, R. S. (2009, August). Coarse-grained detection of student frustration in an introductory programming course. In *Proceedings of the fifth international workshop on Computing education research workshop* (pp. 75-80). ACM.

[6] Dweck, C. S. (2000). Self-theories: Their role in motivation, personality, and development. Psychology Press.

[7] Rajendran, R., Iyer, S., Murthy, S., Wilson, C., & Sheard, J. (2013). A theory-driven approach to predict frustration in an ITS. *IEEE Transactions on Learning Technologies*, (1), 1-1.