Generating proactive feedback to help students stay on track

Davide Fossati[1], Barbara Di Eugenio[2], Stellan Ohlsson[3], Christopher Brown[4], and Lin Chen[2]

[1] College of Computing, Georgia Institute of Technology
[2] Department of Computer Science, University of Illinois at Chicago
[3] Department of Psychology, University of Illinois at Chicago
[4] Department of Computer Science, U.S. Naval Academy

Abstract. In a tutoring system based on an exploratory environment, it is also important to provide direct guidance to students. We endowed iList, our linked list tutor, with the ability to generate proactive feedback using a procedural knowledge model automatically constructed from the interaction of previous students with the system. We compared the new version of iList with its predecessors and human tutors. Our evaluation shows that iList is effective in helping students learn.

1 Introduction

In this paper, we address the challenge of automatic generation of student guidance in a problem-based, exploration-oriented Intelligent Tutoring System (ITS). In such systems, students are usually presented with problems to be solved, and an environment in which to solve these problems. The system can provide support to the students by means of feedback, on-demand help capabilities, or even interactive dialogue [1, 5]. Systems that rely mostly on the exploration of a simulated environment try to encourage knowledge construction. However, recent evidence suggests that minimally guided instruction does not work as well as expected [4], and students do benefit from direct guidance from instructors or more experienced peers. In previous work, we explored the impact of progressively more sophisticated feedback in the context of a mostly exploratory environment [2, 3]. In this paper, we make a further step in providing guidance to students by generating a new form of *proactive feedback*.

2 Feedback in the iList tutor

The iList tutor[5] helps students explore and learn about linked lists by providing graphical representations that can be interactively manipulated with programming language commands. The visualization of linked lists is updated in real time according to the actions performed by the students. The system provides a set of

[5] iList is freely accessible at *http://www.digitaltutor.net*

problems that can be solved by providing sequences of operations that transform the original lists into the desired configurations. The tutor can provide different types of feedback. In previous work we presented *syntax*, *execution*, *final*, and *reactive procedural* feedback [2, 3]. Here we introduce *proactive procedural feedback*, an interactive tutor-student interaction composed of three parts:

1. A question from the tutor, including a statement of the goal to be achieved by the following move; the explicit question about how to accomplish that goal; and a set of up to four choices including the correct answer and some of the most frequent incorrect answers given by students. Example: "Let's see what we can do now... Pointer T is pointing to node 5, we want it to point to null. How would you do that? (1) T = NULL; (2) delete T;"
2. An answer from the student, given by clicking on one of the given choices.
3. Feedback from the tutor. If the answer was right, the message is a positive statement such as "That sounds right! I suggest you try it now." If the answer was incorrect, the message points out the mistake and illustrates the consequences of that choice. Example: "Uhmm... This is probably not a good idea. Here is what will happen if you do what you suggested. You will delete the node that is pointed by pointer T and that contains 2. Variable T is now pointing to node 2, then it will point to garbage."

Students must complete the entire interaction before they can continue to work on the problem. To decide when to start it, iList monitors the student's activity. If the situation is considered critical and enough time has elapsed since the last move, iList initiates the proactive interaction. To make this determination, the current state of the problem is matched against a probabilistic graph that assigns likelihoods to states and actions, evaluates the quality of students' moves in terms of probability of eventual success, and records the time spent by students in different states. This graph was automatically built from the interaction of previous students with the system [3].

3 Evaluation and future work

We evaluated five versions of iList by measuring students' learning gain as the difference between a pre-test and a post-test. Versions 1, 2, and 3 of iList could provide syntax, execution, final, and reactive procedural feedback to various degrees. Versions 4 and 5 could additionally generate proactive procedural feedback, although in version 4 it was very infrequent. In our comparison we also included a control group of students that did not receive any form of instruction, and a group of students that interacted with a human tutor (Table 1). ANOVA revealed an overall significant difference among the seven groups ($F(6, 319) = 3.04$, $P = .007$). Tukey post-hoc tests revealed point-to-point significant differences only between the control group and the human tutored group ($P = .004$), and between the control group and iList-5 ($P = 0.021$). The progression of effect sizes indicates an overall positive trend. As iList is enhanced with additional features, its performance moves closer to that achieved by human tutors.

Table 1. Learning gain of students in seven conditions

Tutor	N	Pre-test		Post-test		Gain	
		μ	σ	μ	σ	μ	σ
None	53	.34	.22	.35	.23	.01	.15
iList-1	61	.41	.23	.49	.27	.08	.14
iList-2	56	.31	.17	.41	.23	.10	.17
iList-3	19	.53	.29	.65	.26	.12	.24
iList-4	53	.53	.24	.63	.22	.10	.16
iList-5	30	.37	.24	.51	.26	.14	.17
Human	54	.40	.26	.54	.26	.14	.25

A limitation of this comparison is that it was conducted over several semesters and across different institutions. Thus, it is difficult to factor out numerous confounding variables such as differences in student population. Future evaluations will be run in a way more conducive to controlling for such differences. Currently, iList covers only linked lists. We are planning on increasing the number of data structures covered by the system, and augmenting the interaction capabilities between the students and the system. More fundamentally, we would like to explore additional pedagogical strategies, to make the system more responsive to students that might be more sensitive to different modes of learning.

Acknowledgments. This work is supported by the ONR (N00014–07–1–0040, N00014–00–1–0640); the UIC Graduate College (Dean's Scholar Award 2008–2009); and the NSF (ALT–0536968, IIS–0133123, 0937060 to CRA for CIF–186).

References

1. Evens, M., Michael, J.: One-on-one Tutoring by Humans and Machines. Mahwah, NJ: Lawrence Erlbaum Associates (2006)
2. Fossati, D., Di Eugenio, B., Brown, C., Ohlsson, S., Cosejo, D., Chen, L.: Supporting computer science curriculum: Exploring and learning linked lists with iList. IEEE Transactions on Learning Technologies, Special Issue on Real-World Applications of Intelligent Tutoring Systems (2009), in press
3. Fossati, D., Di Eugenio, B., Ohlsson, S., Brown, C., Chen, L., Cosejo, D.: I learn from you, you learn from me: How to make ilist learn from students. In: AIED 2009, The 14th International Conference on Artificial Intelligence in Education. Brighton, UK (July 2009)
4. Kirschner, P.A., Sweller, J., Clark, R.E.: Why minimal guidance during instruction does not work: an analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. Educational Psychologist 41(2), 75–86 (2006)
5. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R.H., Taylor, L., Treacy, D.J., Weinstein, A., Wintersgill, M.C.: The Andes physics tutoring system: Five years of evaluations. In: McCalla, G.I., Looi, C.K. (eds.) Artificial Intelligence in Education Conference. Amsterdam: IOS Press (2005)