Providing Timely Feedback to Large Classes

Yusuf Pisan
Anthony Sloane
Debbie Richards
Robert Dale

Department of Computing
Division of Information and Communication Sciences
Macquarie University
Sydney, Australia
Email: ypisan,asloane,richards,rdale@ics.mq.edu.au

Abstract
Providing timely and high-quality feedback has been identified as one of the critical factors in learning. In courses with large class sizes the amount of time required for critiquing makes it prohibitive to give feedback to students on more than two or three assignments per semester. As a result, students have limited opportunities to learn from their mistakes, and it is difficult for lecturers to monitor student progress. The alternative to manual critiquing is to automate the process. We expect automated program critiquing to deliver two major benefits to students: improved feedback and more useful access to lecturers. This paper introduces our project and the expected benefits.

Keywords

Intranet use in conventional universities, Student Feedback

1. Introduction

Providing timely and high-quality feedback has been identified as one of the critical factors in learning (Ramsden, 1988; Gibbs, 1994; Hattie, Jaeger and Bond, 1999). The current and traditional method of providing critiquing is to do it manually: tutors or lecturers read submissions, optionally run programs on test data, and make comments. In large computing courses, such as the first year computing courses where there may be over one thousand students enrolled in the course, the amount of time required for critiquing makes it prohibitive to give feedback to students on more than two or three assignments per semester. As a result, students have limited opportunities to learn from their mistakes, and it is difficult for lecturers to monitor student progress.

The alternative to manual critiquing is to automate the process. Current commercial systems are restricted to binary correct/incorrect assessment or multiple-choice questions and are of limited use for deeper learning. A number of intelligent learning environments have been developed as research prototypes. Some well-known examples are SOPHIE (Brown et al., 1982), LISP Tutor (Anderson and Reiser, 1985), Geometry Tutor (Anderson et al., 1986), Sherlock (Lajoie and Lesgold, 1989), WebToTest (Arnow and Barshay, 1999), and PILOT (Bridgeman et al., 2000). Much can be learned from these systems in terms of underlying techniques; however, they are limited to specific domains and, most importantly, do not address the program critiquing problem. Within many computer science departments, including our own, individual teachers have developed a variety of programs mainly to run simple automated tests on student submissions. However, these programs have often been specific to the assignment at hand, lack a uniform interface and are difficult to use by people other than the original author. There would clearly be substantial benefit in unifying these efforts to produce a system that goes beyond simple tests and provides feedback to students. Critiquing programs and generating high-quality feedback is a complex task. Despite this complexity, program critiquing is feasible because each programming language has a well-defined syntax and semantics that makes the task of automation easier than it would be for free-form submissions such as essays or reports.

2. The Project and Progress-to-date

This project involves taking what is useful from the existing prototypes mentioned above, and integrating appropriate ideas and concepts in order to develop a sophisticated program critiquing tool. The tool is being developed in an incremental fashion, whereby at any point we only provide as much automatic critiquing as we can carry out reliably; we then gain feedback from the tool's use in a real teaching environment while further development is ongoing. We have started seeing benefits from a relatively early stage. The project contains a number of phases. In Phase One we have implemented a web-based assignment submission system. This system has been in routine use for over 12 months now and has made submission, management and marking of assignments a much smoother process. Figure 1 shows a sample screen. The web-based system allows the lecturer to set up an assignment specifying various parameters such as due dates and times. In Phase Two we

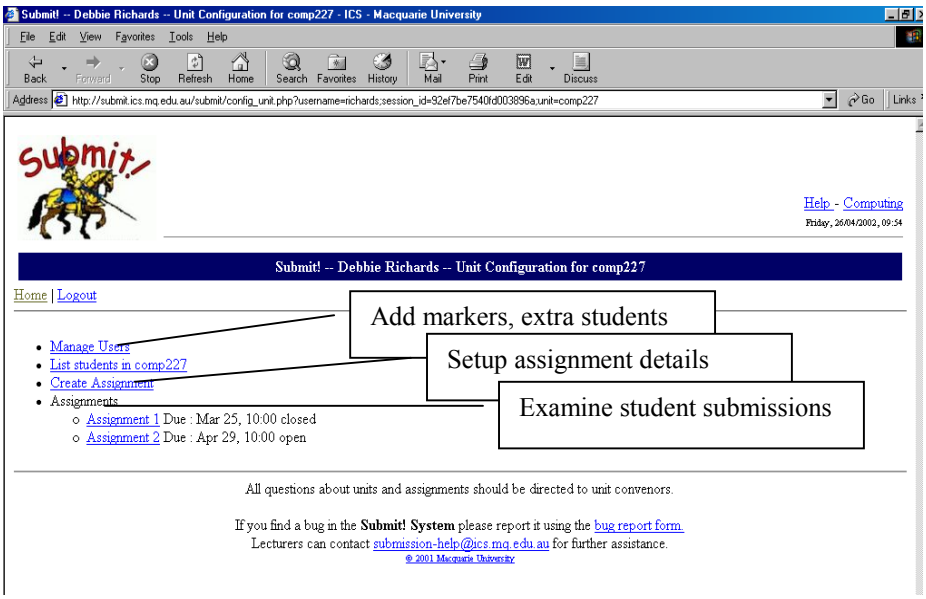

Figure 1: A sample screen from our Submit! System for capturing and managing student assignments.

have added plagiarism detection and submission validity checking which are currently being tested. We are about to design and conduct an evaluation with students and staff of the system to date. Phase Three will add code commenting facilities and student access to testing. Phase 4 is creation of a knowledge base to add intelligence to the testing and further customisation. The fifth and final phase is to finalise documentation, implementation and complete staff training.

3. Conclusion

Critiquing can be used as a component of formative or summative assessment to enhance self-directed learning. We expect automated program critiquing to deliver two major benefits to students: improved feedback and, as a consequence of freeing up lecturers' time, an opportunity for higher-quality one-on-one contact.

Feedback will be better because:

- It will be specific to the individual's submission rather than a general list of possible errors or solutions that the student is often unable to apply to his or her own case.
- There will be more feedback as comments can be added every time the need is identified, rather than being limited by the amount of time the marker has or the fatigue of the marker.
- The feedback will be more timely, objective and consistent because the computer will process solutions quicker than a human, is not subjective and does not suffer from fatigue.
- The feedback will be more timely because in some cases students will be able to run the critiquer themselves before final submission to allow them to correct and learn from their own mistakes.

Higher quality one-on-one contact because:

- The more mundane questions such as "What did I do wrong?" will be answered by the critiquer. This means that the student and lecturer can concentrate on the concepts underlying the solution.
- Freeing lecturers from some of the marking task will result in an increase in the time available for consultation with students.

Additionally, students and lecturers should find interaction more satisfying as the nature of their one-to-one discussions should be more stimulating and less shallow. These benefits apply to classes of all sizes, but when dealing with large classes an automatic critiquing program provides timely and valuable feedback that would otherwise not be possible.

REFERENCES

Anderson, J. R., and Reiser, B. J. (1985) The LISP tutor. *Byte*, 10, pp 159–175.

Anderson, J. R., Boyle, C. F., and Yost, G. (1986) The geometry tutor. *The Journal of Mathematical Behavior*, pp 5–20.

Arnow, D. and Barshay, O. (1999) On-line programming examinations using WebToTeach. In *Proceedings of Innovation and Technology in Computer Science Education*, Cracow, Poland, pp 21–24.

Bridgeman, S., Goodrich, M. T., Kobourov, S. G. and Tamassia, R. (2000) PILOT: An interactive tool for learning and grading. In *Proceedings of the 31st ACM SIGCSE Technical Symposium*, Austin, Texas, pp 139–143.

Brown, J. S., Burton, R. R., and deKleer, J. (1982) Pedagogical, natural language and knowledge engineering techniques in SOPHIE I, II and III. pp 227–282 in D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*. NY: Acad. Press.

Gibbs, G. (ed) (1994) *Improving Student Learning — Theory and Practice*. Oxford: Oxford Centre for Staff Development.

Hattie, J.A., Jaeger, R.M., and Bond, L. (1999) Pervasive problems in educational measurement. *Rev.of Res. in Education*.

Lajoie S.P. and Lesgold A. (1989) Apprenticeship training in the workplace: computer-coached practice environment as a new form of apprenticeship. *Machine Mediated Learning*, 3(1), pp 7–28.

Ramsden, P. (ed) (1988) *Improving Learning: New Perspectives*. London: Kogan Page.