USING AN ITERATIVE PROCESS TO TRANSFORM A THEORETICAL UNIT

Johannes Herrmann

Presenting Author: Johannes Herrmann (hannes.herrmann@curtin.edu.au)
Department of Computing, School of Electrical Engineering and Computing, Curtin University, Perth WA 6102, Australia

KEYWORDS: computing, theory, flipped classroom, focused assessment, motivation

ABSTRACT

Faced with a theoretical computing unit with a poor reputation, the author used a process of iterative improvement to address the main concerns. Armed with the informal concept of Caesar theory – bread and circuses – changes were applies across two years and have turned student attitude to the unit around. This paper discusses the iterative process used and the challenges that were faced and (at least partially) overcome. While the process is ongoing, both anecdotal and formal feedback points to a noticeable improvement in student satisfaction without a reduction in the teaching standards.

Proceedings of the Australian Conference on Science and Mathematics Education, Curtin University, Sept 30th to Oct 1st, 2015, pages 83-86, ISBN Number 978-0-9871834-4-6.

INTRODUCTION

The Scholarship of Teaching and Learning is advancing rapidly, especially with the incorporation of technology. However traditional teaching methods do not necessarily adapt well to computing education (Miliszewska & Tan, 2007) and this applies equally to some modern advances. This difficulty can combine with other factors to make some units truly challenging to teach well.

An additional challenge is that students often regard units with little or no programming as not directly contributing to their learning. This attitude worsens when the unit is considered difficult or excessively theoretical. A culture can form around a unit that leads to learners entering with a negative outlook. Attitude and enthusiasm is an important factor in learning, so such issues must be addressed.

Another challenge that often arises in computational education is the balance between take-home and invigilated assessments (Chin 1999). The former are often more practical as they generally involve the students directly working on matters in line with the unit's aims, such as developing programs or solving mathematical proofs. Unfortunately the take-home and problem-solving nature of these assessments leads to a very real chance of plagiarism. Pen and paper tests and exams are easily invigilated, reducing the chance of plagiarism, but are not always suitable for testing practical skills.

THE UNIT

At the center of this discussion is Theoretical Foundations of Computer Science, offered at Curtin University as a final semester unit for students in the Computer Science. As the name suggests, it is highly theoretical and centers on mathematical proof. Much of the work required ranks highly on Bloom's taxonomy since it requires constructing mathematical transformations between problems.

The unit is considered important enough to be offered in many leading universities such as MIT, Berkeley and Harvard, and is based around a well-known textbook (Sipser 2014). The thinking processes learned greatly increase the ability to design programs and it uses an assignment-test mechanic that works well for the subject matter.

The assignment for the unit is not submitted, but is assessed in an invigilated test. This gives students the time to think properly on some difficult questions which may take hours of work to solve at first, but can be relatively quickly reproduced once understood. The test questions are not exactly identical to those in the assignment, so students having plagiarized often find themselves unable to apply their answers to the test questions. This is designed to partially alleviate the trade-off raised by (Chin 1999) between relying on supervised assessments against the very real chance of plagiarism for take-home assessments that is a common issue in computing education.

There are a number of challenges facing the unit. In 2012 it was considered both too difficult and irrelevant to real life by the students. The benefits of proofs in strengthening relevant mental skills was not accepted and many students complained about the theoretical nature of the unit.

Properly assessing mathematical proof is also challenging. A proof of the level of roughly half of the material in the unit could take even an experienced mathematician hours or days to complete. Setting a final exam that adequately examines the students on such difficult proofs is challenging. A further complication is that one of the proof areas covered – Computability – has only a relatively small number of examples that are of the correct difficulty to be considered for the unit, and students regularly attempt to memorize the proofs for all of them. This reduces the assessment from the level of evaluation or synthesis to that of knowledge or comprehension.

The author first gained control of the unit in 2012 with relatively short notice, and presented the unit essentially the same way as in the previous year. This gave a chance to understand the existing unit while at the same time planning improvements for previous iterations.

CAESAR THEORY

One of the greatest threats to the unit was the student's belief that it did not provide anything of value for their future careers. In order to shift the attitude of the students, the author adopted the notion of 'Caesar theory' – give them bread and circuses. While 'Juvenal theory' might be more accurate, it cannot be argued that Caesar used the concept to great effect to control to morale of the roman populace.

For a population of students, the bread is what is perceived as required – in this case concrete reasons for studying a particular topic. With practical units this is often self-explanatory, but with more abstract units the students must be shown applications up front. If the query "why do I need to learn this" is answered at the start of a unit, students never need to ask it internally. Note that it is not necessarily required that the reasons are likely to occur, as long as the students visualize these reasons applying to them.

As with the ancient Romans, circuses refers to entertainment and a 'wow' factor; something to lift the students out of what they are accustomed to in a unit. This could be as simple as more artistic presentations or the use of tools that are seen as special in some way. Such 'circuses' improve the student's attitude towards the material and the unit.

THE FIRST ITERATION

THE BREAD

The first iteration of the unit modifications focused on the bread due to the importance of student motivation. Much of the planning involved finding ways in which the content of the unit could be relevant and even vitally important, although technical solutions to enhance both teaching and learning were also considered.

Having decided on a useful reason – in this case the possibility of being put in a position of being a team leader tasked with tackling a problem of the sort discussed – all aspects of the unit were redesigned to focus on this concept. Several case studies were presented at the beginning of the first class, and were referred to regularly throughout the semester.

When the students pictured themselves in the position of a programming team leader and were led through discussions on possible ramifications of misclassifying related problems, they were convinced that the problems posed were valid. Once this had been accepted, the methods of backing up such classification with proof were more likely to be accepted, especially when cast as being required to convince a skeptical or ignorant superior – a situation that is common in films and other media and thus easy to relate to.

The adoption of the 'problem classification' justification was applied to all aspects of the unit. All examples presented in lectures and tutorials were of this nature, as were all questions in worksheets and assessments. Having accepted that problem classification was necessary, this made all aspects of the unit seem relevant. Proofs were case as an incidental method of convincing others of problem classification.

A number of technical solutions were considered for the ‘circuses’ part of the theory, including computer-based machine simulations useful for the first part of the unit, automated theorem provers and *PowerPoint* alternative *Nearpod*. None of these were overly successful, but at least served to give students the assurance that that the unit was moving forward.

RESULTS

In 2012, the comments submitted by students as part of the formal eValuate survey were positive about the teaching but less so about the unit. In addition there were only responses from 20% of students, making it difficult to draw accurate conclusions. Unfortunately this response rate has remained constant.

While the quantitative items in 2013 were similar, the comments showed a shift from concerns about the unit to issues with specific parts of the unit. The issue of memorizing proofs was raised, as were the difficulties of remembering many of the complex conversion processes.

Anecdotal evidence was much more positive. After the first iteration more than half the students claimed to enjoy the unit (as opposed to only two before) and several stated that it was their favorite unit. In the qualitative feedback one student responded to what should be improved by stating that nothing should be improved since the unit was being taught efficiently. While this comment must be taken in context it does support the shift in anecdotal evidence gathered.

THE SECOND ITERATION

THE CIRCUSES

The second iteration aimed to adjust the new content and add something to make the unit stand out. There were two main issues to address; students did not do enough work on state machine simulations in the first part of the unit because they are tedious to do manually and students were coming into the lectures unprepared, causing difficulties in properly addressing the material.

A third issue – excessive complexity in transformations – was addressed by moving this material out of the examinable content. Transformations between machines and problems serve to convince the students that they are equivalent, but did not fit the new focus of the unit. As such, they were still discussed in class but were not assessed.

An attempt to use existing state machine simulators in the first iteration wasn’t successful since no suitable simulators were found. Two were presented to the students, but both had weaknesses. This was solved by proposing the writing of a new simulator as a project to final year students. This had the added benefit of giving the class a sense of ownership of the simulator since it was programmed by their peers. The finished product combined the best features of other programs, adding something special to the unit and allowing students to more easily experiment with state machine designs.

The second issue was somewhat more complex to address. In order to fully understand the complex topics discussed in the unit, the students need to read relevant sections of their textbook, and often need to access additional material such as instructional videos on YouTube. The obvious solution was the application of the flipped classroom model, but the students did not have experience with this model.

Applying Ceasar theory, the decision was made to attempt to persuade the students of the benefits of the model while simultaneously offering a ‘circus’ designed to influence their response. Accordingly, the first lecture was re-written from a 41 slide *PowerPoint* presentation to a 6 slide Prezi presentation. The latter is far more visually appealing, and the shorter presentation of highlights only improved the interaction during lectures.

Students were presented with the *PowerPoint* slides before the lecture and shown them briefly at the start, and given the choice between the two methods. They agreed to trial the Prezi method for three further weeks. Additional care was taken to make the Prezi summaries interesting through focus on case studies, use of the state machine simulator for examples and relevant discussions. The combination of these factors encouraged students to honestly attempt the model, and the class decided to remain with it for the rest of the semester.

After student feedback, a large number of detailed examples were provided to students in order to give them more guidance with the standards required in the unit. In addition, questions in the final exam focused more on real-world problems to which relevant methods had to be applied, which reduced the benefit of just memorizing proofs.

RESULTS

Anecdotally, the unit was received even more positively than the previous year; one student described the unit as “seriously a lot of fun” in a message The response rate remained at around 20%, with those responding once again 100% in agreement on the positive questions relation to the unit.

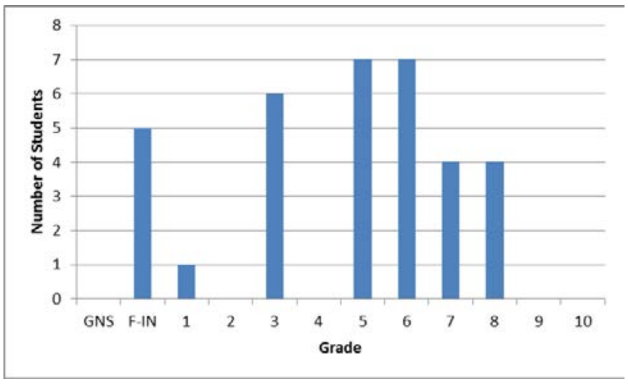

Figure 1: Student Marks after the 3rd Iteration

The qualitative questions more clearly showed the effects, with the comments being very supportive of the unit. One student wrote “Links to real world problems are extremely helpful in putting this highly theoretical unit in context and keeping it interesting. Onus on students to self-learn very good for accountability and motivation” and another wrote “The theoretical nature of this unit was well motivated with practical problems“. Another commented on the benefit of the flipped classroom model, but noted that a number of students did not keep up with the reading later in the semester. This was borne out in the final results in the unit, with a very clear distinction between students who kept up with the work, those who did not, and those who simply gave up, as shown in Figure 1. It should be noted that the overall pass rate did not change noticeably (from 79% to 76% of completing students) although slightly more students did not complete the unit (as indicated by a mark of F-IN).

THE THIRD AND FUTURE ITERATIONS

The main focus of the work in 2015 will be on assessment. Following a peer review in 2014 there are a number of strategies to consider, especially for the final exam. For example it would be entirely possible to base the final exam largely or entirely on the assignment in a similar manner to the assignment test, although this may require extending the assignment. The concept of replacing the exam or assignment with a portfolio assessment of proofs was rejected by the head of department due to possible plagiarism issues.

Many students commented positively on the sample proofs, so this year several of these will be transformed into tutorial videos that explain the steps taken in the creation of the proof. This enhances the student learning experience and reduces the need for staff to give the same explanation multiple times. The aim is to add a few videos each year in order to build up a useful library in the medium term.

The author is interested in more ideas for improving the unit, and would be delighted to be contacted regarding these.

REFERENCES

Chin, K. L. (1999). How to design an assessment system to ensure students acquiring the necessary practical skills in a computer programming unit. In K. Martin, N. Stanley and N. Davison (Eds), Teaching in the Disciplines/ Learning in Context, 72-75. *Proceedings of the 8th Annual Teaching Learning Forum*, The University of Western Australia, February 1999. Perth:

Miliszewska, I., & Tan, G. (2007). Befriending computer programming: A proposed approach to teaching introductory programming. Informing Science. *International Journal of an Emerging Transdiscipline*, 4(1), 277-289.

Sipser, M. (2012) *Introduction to the theory of computation*. Cengage Learning.