View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by MURAL - Maynooth University Research Archive Library

Facilitating student learning in Computer Science: large class sizes and interventions

Keith Nolan

keith.nolan@nuim.ie

Department of Computer Science,

Maynooth University.

Maynooth. Co. Kildare. Ireland

Aidan Mooney* and Susan Bergin**

aidan.mooney, susan.bergin {@nuim.ie}

Department of Computer Science,

Maynooth University.

Maynooth. Co. Kildare. Ireland

*senior author, ** senior/corresponding author

Abstract

Learning to program is difficult with many students struggling to master fundamental abstract concepts. This results in high dropout and failure rates on computer science and information technology degrees as programming tends to be one of the first modules taken. At Maynooth University (MU) the number of students taking introductory programming modules has increased by 300% in the last ten years. Coupled with this is a large increase in student diversity most notably in academic ability and motivation for pursing a first year course in computer science. This paper documents numerous interventions implemented at the department of Computer Science at MU in an attempt to improve engagement, performance and student learning experience. A brief overview of each intervention is provided. A longer synopsis of our most sustainable intervention to date is described. The paper concludes with recommendations for other institutions on how best to implement this intervention when faced with a similar problem.

Keywords

Programming, first year, diversity, large class sizes, interventions

1. Introduction and Motivation

Learning to program is difficult and can lead to high dropout and failure rates (Bergin, 2006). Since the early 1970s, numerous studies have investigated how students learn to program and the factors that influence a student's likelihood of success. Previous studies have found that students learning to program struggle to achieve an average score above 30% (Authority, 2007; Higher Education Authority, 2010; O. Mooney *et al*., 2010a, 2010b) with various interventions documented to model and support students learning (Bergin & Reilly, 2005a, 2005b; Quille *et al.*, 2015). Over the last 10 years, there has been a large increase in the number of students attending third level education. In particular, student numbers taking first year Computer Science (CS) at MU have increased from approximately 120 students in 2005 to typically 400 per annum in recent years with a wide diversity of students taking first year programming modules. Students can come from over 20 different degree programmes from pure CS to law, to theology, to pharmaceutical science for example. CAO entry points also differ by as much as 60 points across the degree programmes, thus a wide range of academic abilities must be catered for in addition to different motivations for taking the modules.

The changing landscape has demanded the implementation of new methods to support student learning and through this process, evidence has been gathered on what works well and what does not. This paper provides a review and critique of some of these methods and provides recommendations on how to implement our most sustainable approach to guide other educators faced with a similar problem.

2. Selected Interventions pre-2012

Traditionally at MU first year programming modules are composed of two hours of lectures and two hours of labs per week with optional tutorials as necessary over a 12 week semester. Numerous methods to support first year programming students have been implemented, however, in the interests of brevity, only methods that have been implemented for two years or longer are documented in this paper.

2.1 Problem Based Learning

A hybrid model for Problem Based Learning (PBL) was introduced to help first year students grasp abstract concepts (Kelly *et al.*, 2004; O'Kelly *et al.*, 2004). The PBL sessions involved groups of five to seven students, based on pre-determined criteria including gender balance, mix of academic disciplines, mix of mature students and school leavers etc. Within each group, four people were given a specific role that changed for each weekly session; these roles were a chairperson, a scribe, a reader and an archiver. The PBL sessions lasted for an hour and a half. The problems used fell into three different categories: (1) extendible conceptual problems, (2) non-extendible problems and (3) programming problems and each group were required to develop an algorithm to solve the problem. Props were used to help facilitate student learning. PBL was used for three years with notable success evident by an increase in student grades at all levels of learning (bottom, middle and top of the class). For the most part students enjoyed the sessions and in particular the opportunity to work with and get to know their peers (Kelly et al., 2004). This social learning was not ordinarily achieved given the large class sizes. However, a number of issues were noted, some more difficult to address than others. (1) Participation within the groups was difficult to balance. Some students were reluctant to interact with their group (quiet, uninterested etc) and effectively left the rest of the group to do the work. (2) As time passed some facilitators left the department (typically postgraduate students) ant it was difficult to find good replacements. Buy-in from facilitators is critical for success. (3) Considerable time went in to training facilitators, designing problems, and planning and reviewing the sessions. Unfortunately as class sizes continued to grow, the approach became unsustainable due to resource, room and facilitator restrictions.

2.2 Novel Teaching Tools

Lego MindStorms are robots designed to be reprogrammable. They were introduced to the first year programming course in 2007 and used for the following four years to help teach fundamental programming concepts. Students worked in pairs to solve programming problems that involved changing the behaviour of the robots. Several benefits to this approach were noted by teaching staff. (1) Labs tended to be noisy, relaxed and interactive. Often programming labs can be daunting, particularly for weak students. (2) Feedback from students was positive. Students enjoyed working with the robots because of the instant gratification they received from seeing the robot perform an action using the code they had written. By 2011, with over 300 student numbers participating on the module, and the high equipment costs involved, this approach became unsustainable.

2.3 Approach to disseminating course material

Virtual Learning Environments have become popular for disseminating course material. Approaches to varying the timing of when materials were distributed were trialled over a two year period in an attempt to increase in-class student engagement. Specifically: (1) lecture notes were made available after the lecture, (2) summary notes were made available before the lecture, and (3) the class were told that an in class assessment would take place based on the summary notes provided (Mooney and Bergin (2014)). The study found that including in-class assessment was most the most effective option. This is no surprise given that assessment often drives learning (O'Riordan *et al.*, (2013)), however the simplicity of this change and its positive impact warrants mention here.

3. Sustainable Recommended Intervention (2012 onwards).

The interventions outlined in Section 2 were all successful in their own right. However, with the ongoing challenge of growing numbers and student diversity, an additional resource that students could use in their own time and that could cater for all levels of learners was required. As a result the Programming Support Centre (PSC) was set up in 2011 as an additional service to provide support for two first year modules in particular.

The pedagogical approach at the PSC is based on peer-tutoring and at the heart of the model is the volunteer nature of this tutoring. There is no payment or credits provided for tutors, rather these are second and third year students who give their time freely, usually because they would like to experience this type of role or have a genuine desire in helping others. The tutors are proficient in the two introductory modules. At the start of each academic year the tutors attend induction training. The PSC is coordinated by two academic staff members and each session is managed by a paid postgraduate. Lecturers do not attend the PSC, a deliberate decision, to encourage a relaxed setting for learners.

While the tutors for the PSC are given a set of guidelines, the actual interaction between the tutor and student is free-flowing and unstructured. In general there is a loose protocol that a tutor will follow when helping a student with a programming concept. The tutor and student will sit together and use pen and paper. The tutor will work through examples of the concept with the student, gently probing and encouraging the student to interact with them. A session typically concludes when the student is able to work through an example of the problem unassisted.

Although the PSC is branded the Programming *Support* Centre, it is not a remedial centre. The PSC caters for all levels of learners, from the weaker student that uses the PSC to improve their abilities to the more advanced student who is looking to challenge themselves further.

4. Recommendations

In this section a list of recommendations, aimed at interested educators interested in setting up a similar centre are provided. These recommendations are based on our collective experience and feedback received over the years.

Recommendation 1: Induction training

At the beginning of each academic year, tutors for the PSC are given induction training. The tutors are told about the ethos of the PSC, best described by the Chinese proverb:

Give a man a fish and he will eat for a day; Teach a man to fish and he will eat for life.

With this in mind tutors watch training videos in groups. These videos were developed in-house to depict typical encounters at the PSC. After, the tutors work together to determine what worked well and what didn't. We have continuously found that tutors identify through this process how we hope they will interact with student learners i.e. listening carefully to the problem, asking probing questions, leading the student to the answer but not doing it for them. We have found that this training is invaluable and strongly recommend its inclusion for similar set-ups.

Recommendation 2: Online tracking system

An in-house online tracking system is used for the PSC. The system allows students to register the reason for their visit. The tutor who then helps the student is responsible for signing the student out, detailing what help was provided to the student and what help should be provided in the future. This system allows tutors to quickly come up to speed on previous learners’ visits to the centre. In addition it allows academic staff to better understand the types of problems weaker students are having so in-class support can be provided as necessary Strong students seeking to work at a faster pace can be identified and targeted initiatives (e.g. online coding competitions) can be made to nurture and challenge their abilities. We strongly advocate the use of such a system.

Recommendation 3: The setting

As the PSC takes place in a small lab, the setting is comfortable and relaxed for students. This all allows for the session to be intimate and knowledge transfer to be most effective. The hours for the PSC were specifically set to allow all students the opportunity to attend the centre. This setting is critical for the success and is strongly advocated for parties.

Recommendation 4: Extra recourses to cater for all levels of learning

Annually the PSC runs different events. These events are open to all students who take CS. All of these events promote the use of the PSC. The events include (i) Robocode – a national programming contest (ii) the Programmathon – an in-house programming competition open to all students who take Computer Science and (iii) weekly programming challenges in 2nd semester – aimed to engage stronger students. We have had a large uptake of these resources and in particular recommend their use for stronger students who are in-danger of disengaging and not performing optimally.

Overall the PSC provides invaluable student support. The benefits of peer-learning are well established and the volunteer nature of tutors means that it is sustainable going forward.

5. Conclusions and Future Work

This paper described several interventions to support student learning in large classes with considerable student diversity. The most sustainable resource, in our experience, is the Programming Support Centre. Its great strength is its capacity to support students irrespective of their current ability (bottom, middle or top of the class) through peer-learning and is highly cost-effective given the volunteer nature of the tutors. Dedicated induction, a made-for-purpose tracking system, an intimate relaxed setting and the availability of a wide variety of resources have been found to be particularly useful (especially for strong students).

6. Acknowledgements

This work was funded by the Irish Research Council Postgraduate Scholarship 2015.

References

Authority, H. E. (2007). What do Graduates Do? The Class of 2005, (December).

Bergin, S. (2006). *Statistical and Machine Learning Models to Predict Programming Perform ance*.

Bergin, S., & Reilly, R. (2005a). Programming: factors that influence success. *ACM SIGCSE Bulletin*, *37*, 411–415. http://doi.org/http://doi.acm.org/10.1145/1047124.1047480

Bergin, S., & Reilly, R. (2005b). The influence of motivation and comfort-level on learning to program. *Ppig 17*, (June), 293–304. Retrieved from http:

Higher Education Authority. (2010). *What do graduates do?*

Kelly, J. O., Mooney, a, Ghent, J., Gaughran, P., Dunne, S., & Bergin, S. (2004). An Overview of the Integration of Problem Based Learning into an existing Computer Science Programming Module 2 Overview of our PBL Implementation. *Problem-Based Learning International Conference 2004: Pleasure by Learning*.

Mooney, A., & Bergin, S. (2013). A study on alternative strategies for sharing lecture notes using a VLE to promote in-class participation, *2013*.

Mooney, O., Patterson, V., O’Connor, M., & Chantler, A. (2010a). *A Study of Progression in Irish Higher Education*.

Mooney, O., Patterson, V., O’Connor, M., & Chantler, A. (2010b). A Study of Progression in Irish Higher Education, 92.

O’Kelly, J., Bergin, S., Dunne, S., Gaughran, P., Ghent, J., & Mooney, A. (2004). Initial findings on the impact of an alternative approach to Problem Based Learning in Conputer Science. *Pleasure by Learning*. Retrieved from http://eprints.nuim.ie/727/01/PBLPaper1.pdf

O'Riordan F., Bergin S., Casey K., McNutt L. Discourse and Connectivity: Capturing the voice of educators In C. O'Farrell & A. Farrell, Emerging Issues in Higher Education lll: From Capacity Building to Sustainability. Athlone, EDIN, 2013.

Quille, K., Bergin, S., & Mooney, A. (2015). PreSS #, A Web-Based Educational System to Predict Programming Performance, *4*(7), 178–189.