IMPLEMENTING COGNITIVE MODELLIG IN CS EDUCATION: ALIGNING THEORY AND PRACTICE OF LEARNING TO PROGRAM

Des Traynor, J.Paul Gibson,
National University of Ireland, Maynooth,
Co.Kildare, Ireland
{dtraynor|pgibson}@cs.may.ie

ABSTRACT

It can be argued that although computer science may seem one of the best researched areas, it could also be one of the worst taught. In this research we examine the residual problems in traditional teaching methods, and how the same problems have re-occurred in web-based education. We cite the lack of a cognitive model for learning programming as the primary reason for this weakness, and discuss this problem. Finally, we conclude that should a sufficient cognitive model be built and justified, it should lead to huge advancements both in traditional teaching and especially in web-based education and e-learning tools.

KEYWORDS

Cognitive Models, Computer Science Education, Pedagogical Issues, Mental Models.

1. INTRODUCTION

Many case studies have shown that programming and its fundamental concepts are some of the most difficult subject areas to teach and to learn.(McCracken 2001, van Roy 2003), and as a result of this there are many different approaches, few with any resounding success. The pedagogical aim in teaching programming is to create a teaching approach that demonstrates both how to solve problems in an algorithmic manner and how to write code to implement those algorithms. Whilst it would appear that problem solving and coding are separate issues, it is difficult for new students to think independent of the language they are learning.

Traditional teaching techniques lead teachers into educating their students in a linear fashion, as would be the case in other disciplines. However clearly this approach is failing, as students are not learn sufficient skills in their introductory programming courses (McCracken, 2001). Should a cognitive model of programming be developed fully and universally accepted, it is likely that all forms of teaching programming will be altered accordingly. Everything from course syllabi to introductory programming books will benefit from the additional knowledge and understanding of what it takes to teach programming. Without doubt the area to benefit most from the discovery would be that of adaptive learning environments(ALEs). ALEs are unsuccessful in their current deployment to teach programming, but if a suitable cognitive model were developed, ALEs could become one of the most useful teaching aids in programming courses.

2. GATHERING DATA TO DEVELOP A COGNITIVE MODEL

It is well known that the key to learning is assessment (Richard 1998); therefore the assessment procedures will be the most important part of any ALE. To automate the assessment it is desirable for the

```
int x=89%9; int y = 89/9; int max =17;
if(x==8 && y>x)
{
  if(x+y<max){print “Hello”;}
  else{print “Goodbye”;}
}
```

Question: What happens when this code runs?
a) No output
b) It prints “Hello”
c) It prints “Hello Goodbye”
d) It prints “Goodbye”
e) It prints “Goodbye Hello”

Figure 2. A randomly generated question to profile students ability

ALE to be capable of generating questions, and suitable answers. Given that the answers to any questions must be discretized, it is reasonable to assume multiple-choice questions would be suitable. A typical multiple choice question from the system can be seen in figure 2.

Automatically generating questions and answers, even for programs as concise as these, is non-trivial. If such a system were capable of generating reasonable questions with answers of different degrees of feasibility, it could be used for intelligent student profiling, which would yield a wealth of information regarding the students' learning traits. This could, in turn, aid in the construction of a cognitive model.

A profile can be held for each student, showing the systems' interpretation of their ability in certain areas, e.g. if/else, modular arithmetic etc. The profile can be updated after each question that the student answers. This same profile can then be used to help us understand difficulties students have with programming, when a student errs on a question that their profile says they should answer correctly, the question and incorrect answer can be examined to see what caused the student difficulty. This can also be used to fine tune the profiling system used in the ALE. If students give wrong answers to questions that they should be capable of answering according to their profile, it can indicate where inaccuracies lie within the profiling system. These inaccuracies can then be removed as part of the development process.

3. THE NEED FOR COGNITION IN COMPUTER SCIENCE

This poster discusses a project in progress at Maynooth university, small pilot tests have verified the research methodology showing that interesting and useful insights into the learning pattens of students can be gained through multiple choice questions such as the example above. From the beginning of the academic year 2004 students are monitored unobtrusively on a full time basis and it is expected that this project will yield a plethora of useful data. This data will then be used to reason about the creation of a cognitive model learning to program.

The need for a formal justified cognitive model of programming is clear and its benefits obvious. Only when we fully understand the process of learning to program, can we teach it properly. Only when we as educators can teach the subject properly, can we consider employing Adaptive Learning Environments to do likewise.

It seems that ALEs have the potential to be the future of teaching programming, or at least play a large part in it. Before we can create them effectively, however, we must re-examine and alter our current methods of teaching to ensure that meaningful learning can move forward in the digital age.

REFERENCES

Lemos, Ronald S.1979. Teaching programming languages: A survey of approaches, *Proceedings of the tenth SIGCSE technical symposium on Computer science education*, Volume 11 Issue 1.

McCracken, M. et al, 2001. A multi-national, multi-institutional study of assessment of programming skills of first-year CS students *ACM Bulletin Vol 33* pp125-180.

Richard C. Sprinthall et al, 1998. Educational Psychology, McGraw-Hill Education, New York USA.

Van Roy, P. et al, 2003. The Role of Language Paradigms in Teaching Programming, *Proceedings of the 34th SIGCSE technical symposium on Computer science education*,Volume 35 Issue 1.