Imperial College London

BEng Individual Project - Interim Report

DoC Teaching Infrastructure: First Year Online Programming Theory Tests

Author:
Alexis Chantreau

Supervisor:
Dr. Tristan Allwood

February 20, 2014

Contents

Chapter 1

Introduction

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

1.1 Motivation

This is even more apparent when students are introduced to other programming languages with different paradigms such as Haskell and Prolog, or more low level such as C.

- The rise in MOOCs such as Coursera, Udacity shows that there is a real interest in learning programming.
- Programming is considered "difficult" to learn and it can only be learnt effectively through lots and lots of practice. Hence, the need to provide a platform for students to practice their programming skills and understanding of programming concepts.
- Having a lecturer come up with all the exercises by himself can be both time consuming and ineffective: some exercises may not be challenging enough for certain top students, or on the contrary too difficult for struggling students, which can be discouraging.
- Lecturers can only rely on a few homework assignments to get an idea of how students are doing. Having a way for lecturers to gather large amounts of data about students' performances would be beneficial. Allows for supplementary material, exercises, etc...

1.2 Objectives

Having identified the problems associated with , we were lead to formulating objectives in order to make the project successful and useful to the parties involved.

The main objective of the project was to produce a web-based teaching infrastructure to complement the introductory first year programming classes. Four key features were identified:

- **Programming questions/exercises -** The web platform should allow students to practice their programming skills and understanding of programming concepts. There should be no limit to the number of questions a student can answer, so that if a student desires more practice, then he should be able to do that. Additionally, it should be possible for a specific set of people, such as lecturers and tutors, to add questions/exercises to the system.
- **Progress tracking -** Designated people, such as lecturers and tutors, should have access to detailed statistics about the students performances. This will provide them with useful information about difficulties particular students, or the entire class, may be facing. In addition, the system should give feedback to the students, in the form of simple statistics, allowing them to identify their weak areas and thus improve on them.
- **Adaptive difficulty -** The questions or exercises presented to the students should be suited to their ability. Not only will this stimulate learning, but it will also give a better indication of a student's understanding of the programming concepts being tested.
- **Automated generation -** There should be a mechanism to allow for some degree of automated or semi-automated generation of exercises. This will provide a large supply of "fresh" questions, so that students don't end up answering the same questions and memorizing the answers to them.

While investigating existing solutions (Section 2.3 - Related Work) we found out that some of these features were less
"web questions" and progress tracking were available in most, whereas adaptive difficulty and especially automated generation weren't as present.

Chapter 2

Background

In this chapter, we start off by giving an overview of the theoretical fundamentals which will be used by the developed system. Finally, we conclude with a tour of existing solutions.

2.1 Computer Based Tests (CBT)

2.2 Computerized Adaptive Testing (CAT)

Computerized adaptive testing (CAT), also called *tailored testing*, is a form of computer-based test that adapts to the examinee's ability.

- Motivation
- Explain how it works
- Algorithm picture/flowchart
- Advantages and disadvantages of CAT
- (Last sentence) usually based on IRT -¿ nice lead in to next section ..... In general, CATs use IRT (Section 2.2) as a response model.

http://edglossary.org/computer-adaptive-test/

- Students can be presented with questions/exercises that are either too easy or too difficult.
- CAT is a solution to this problem as it automatically chooses the exercises of the appropriate difficulty level for the students.
- Based on IRT for item selection
- Evaluation of CAT

2.3 Item Response Theory (IRT)

IRT is based on the idea that the probability of a correct/keyed response to an item is a mathematical function of person and item parameters. The person parameter is construed as (usually) a single latent trait or dimension. Examples include general intelligence or the strength of an attitude. Parameters on which items are characterized include their difficulty (known as "location" for their location on the difficulty range), discrimination (slope or correlation) representing how steeply the rate of success of individuals varies with their ability, and a pseudoguessing parameter, characterising the (lower) asymptote at which even the least able persons will score due to guessing (for instance, 25% for pure chance on a 4-item multiple choice item).

- Calculates the probability of a particular student answering a specific question correctly.
- Different IRT models: One-Parameter Logistic (1-PL), Two-Parameter Logistic (2-PL), Three-Parameter Logistic (3-PL). Refers to the number of parameters used in the model. Parameters are:
  - question difficulty parameter (b)
  - question discrimination parameter (a)
  - chance/guessing paramter (c)
- Item Characteristic Curve, i.e. probability distribution

2.4 Related Work

The area of , and as a result numerous tools, environments and infrastructures have emerged. There are common features to all, however some distinguish themselves by having not so common features. A lot of tools and research in this area, particularly in making environments suitable to teach programming to students. However, automatic exercise generation in these tools is usually non-existent or very limited. Moreover, the tools are more focused on assessing students, i.e. students take tests which count towards their final grade on these systems.

2.4.1 Environment for Learning to Program

Environment for Learning to Program (ELP) is an interactive web based environment for teaching programming to first year Information Technology students at Queensland University of Technology (QUT).

2.4.2 CourseMarker

2.4.3 Automatic Exercise Generator with Tagged Documents based on the Intelligence of Students

2.4.4 Adaptive Self-Assessment Master

ASAM

2.4.5 SIETTE

Chapter 3

Project Plan

3.1 Current Progress

At the moment, most of the work that I've done on the project has involved researching techniques for adapting the difficulty level of questions and the automated generation of exercises. Through the research that I have done, I have understood the techniques to adapt the difficulty level of questions to suit the student's ability and I feel that the implementation of this aspect should go smoothly. On the other hand, the research of automated exercise generation hasn't shown great results, thus the plan is to implement a very basic amount of automated exercise generation and to leave more advanced methods of automated exercise generation as an extension.

Also, I have figured out the details regarding the programming exercises that will be presented to students. These will be multiple choice questions, true/false questions as well as drag and drop type questions. The exercises will be on topics covered in the introductory java classes, i.e. syntax, arrays, loops, conditionals, data structures, classes, objects.

In addition, I have advanced on the details of the implementation, the architecture of the system, as well as the technologies I will be using to complete the project. The application I will develop will be a Java applet embedded in a website. SQL will be used for all database related aspects of the project, for instance, the question bank, student profiles, etc... will be recorded in a database. I will look into Java servlets to deal with requests coming from the Java applet.

I have therefore programmed a basic Java applet to help with getting started.

3.2 Plan

This is a rough timetable that I will follow to realize the project:

- (21/02/14 - 02/03/14): Finalize GUI and architecture of the application (on paper)
- (03/03/14 - 28/03/14): Revise for exams and take exams
-

Extensions - The realization of these features is dependent on time and whether I can find enough research material to allow me to achieve their implementation. They are:

- Come up with more types of exercises for variety
- Better automated generation of exercises
- Add support for Haskell and C

Chapter 4

Evaluation

- The system has correctly modelled the ability of the student
- The system is useful in helping students to learn programming and helping lecturers with feedback of their teaching, in the form of statistics.