Automatic Grading and Plagiarism Detection/Prevention of Homeworks

Martin Hruskovic, 0025372

Abstract—Computer engineering is from educational point of view suitable for automated assessments. With appropriate software grading tools, program correctness can be measured, along with an indication of quality according to a set of metrics. Furthermore, plagiarism detection should be an integral part of the tools that support assessment. This paper wants to make an basic overview about the current automatic grading and plagiarism detection tools.

Index Terms—automatic grading, plagiarism, homeworks

1 INTRODUCTION

Programming assignments are a very important and essential part of computer science education, including embedded systems courses. A convenient way to organize programming labs is to do the exercises as homework and submit them to a specified submission deadline. After doing that a grading is required- either manually by a tutor which is very time-consuming in a class of 100 students or by an automated grading system. One of the biggest problems of homework is plagiarism which is an offense often committed by students enrolled in academic courses [1], including computer science curricula [2]. Any computer based homework assessments are predetermined to conflict with plagiarism. However, fighting this phenomenon of today's copy/paste student-community seems to be nowadays - except of educating - the daily role of every instructor. And this costs resource requirements, effort and last but not least a lot of time when compared to traditional grading. This extra time could be better spent in other areas [3].

2 AUTOMATIC GRADING

2.1 Overview

Marking programming assessments by hand is highly difficult. Number of Students in classes increases rapidly and the instructors are forced find other ways to solve the problem. The educational praxis shows that submission of programming assessments requires in any case automated solutions for submitting and grading. It is also difficult to visually control the syntax or whether the program meets the required specification. The next benefit of automatic grading of submissions is that the feedback is returned entirely objectively. Different Instructors are different experienced and have different interpretation of the marking scheme. And this can result into getting different grades in the same solution [5]. Automatic grading also offers the possibility for students to receive a feedback or even a grade faster - maybe just in a few minutes or seconds, at any time of day or night. And depending on the submission result the student can correct or change his solution and resubmit the task again and again (maybe with some penalty points or even not). This immediate feedback saves a lot of time for both: For student and instructor. And of course it goes towards to better quality of course and achieved educational results.

2.2 Common software solutions

This section will describe some of the most common computer based assessment systems: Ceilidh [8], its successor CourseMarker [9] and other related systems such as BOSS [11], CodeLab (formerly called WebToTeach [7]), ASSYST [12], TRAKLA [13] and RoboProf [14].

2.2.1 Ceilidh

The Ceilidh courseware system was one of the first Computer Based Assignment (CBA) systems to provide functionality for the full lifecycle of programming courses. Its development started in 1988 and it played an important role on the research and implementation of related CBA systems [16]. Ceilidh was based on automatic assessment mechanism that could test and mark programs from different perspectives. It was able to check the dynamical correctness of the Programm, the style how the task was programmed and the elegance od the solution. Later there were also other functions added,
Ceilidh's design objectives have been to support multiple courses, automatic feedback, multiple interfaces, remote learning and to allow for extensibility and portability [8]. Ceilidh was first intended for use in C-programming courses but its extensible nature meant that other marking tools were written very quickly to allow it to mark everything from artificial intelligence programming in Prolog to Unix shell scripts. But Ceilidh had also some limitations: it had no networking support and lacked full X-Windows/PC-Windows graphical interface [5].

2.2.1 CourseMarker
It was formerly called CourseMaster and was developed as replacement of Ceilidh [9]. It was developed using the Java language because of platform independence and a convenient distribution mechanism with its Remote Method Invocation (RMI) [16]. CourseMarker had also in opposite to Ceilidh many improvements: It is fully networked and allows clients to be installed separately from servers and it has finally a Windows interface [5].

2.2.2 BOSS
The BOSS [11] Online Submission System has been developed over a number of years, as a tool to facilitate the online submission and subsequent processing of programming assignments [10]. The system marks the submitted program in three principle components: Correctness (program's functionality, whether it meets the specification), Style (Layout, commenting...), Authenticity (includes verifying the student's identity and checks for plagiarism).

2.2.3 CodeLab
CodeLab, formerly called WebToTeach [7], is a web-based interactive programming exercise system for first steps in programming classes in Java, C++, C and other languages.

2.2.4 ASSYST
The ASSYST [12] is a system which analyzes the correctness of submissions and their efficiency. It is a semiautomated system used to mark C and Ada programs. One of the system's benefits is that humans can cooperate by grading the submissions.

2.2.5 TRAKLA
The TRAKLA [13] system uses also a web-based environment for helping to teach algorithms and data structures. It includes a number of Java applets and student exercises which are evaluated and the feedbacks are sent via e-mail. It uses an Interactive graphical editor for students answers.

2.2.6 RoboProf
RoboProf [14] is the last of the mentioned web-based systems. The students are confronted with the exercises directly in the web browser environment and are asked to type their programs into a text box. When complete, the answers are submitted and results will be returned. If the program is valid, the student can continue with another programming exercise.

3 PLAGIARISM

3.1 Overview
Plagiarism is defined as "the unauthorized use of the language and thoughts of another author and the representation of them as one's own" [4]. For plagiarism in its worst form is considered the presentation of a complete work signed by "new authors", sometimes in an other language. It is related to any phase of research, writing or publishing of a work, printed and non-printed version. Student plagiarism is a big problem in world universities. Thanks to the worldwide increasing of plagiarism and as an answer to it, technical tools are in development. Into the network of universities will come a system cooperation for preventing of plagiarism and for supporting of originality of student work. There are 2 basic categories in plagiarized content in student works: essays and programming assessments. Computer based courses, especially programming assignments are predetermined for plagiating. Especially using automatic grading systems, where the face-to-face contact of the teacher and student is missing. In cases, when it is an obligatory course for more fields of study, it can happen that there are 500 students submitting the same task under the same conditions. These are very good preconditions for occurring plagiarism. Students use to mix and/or modify some original solutions to obtain a counterfeit.

3.2 Plagiarism Detection Tools
In this section I would like to show briefly discuss two common plagiarism detection tools.

3.2.1 JPlag
JPlag [19] is a web based detection system written in Java and can analyze program source codes in Java, Scheme, C or C++. It converts each submitted program into a string of canonical tokens. The strings are then compared on similarity. The programs are always compared pairwise. The output is a set of HTML files which can be explored and the similarities can be found. But "it was shown that JPlag does not effectively detect plagiarized copies when the students introduce changes to the copied programs such as modification of control structures, use of..."
temporary variables and sub-expressions, in-lining and refactoring of methods”[15].

3.2.2 MOSS (Measure of Software Similarity)
MOSS has been firstly developed in 1994 and became very effective in his role. This free tool allows checking codes in various languages. It calculates a numerical fingerprint (with the local document fingerprinting algorithm [20]) of each source code files and than compares each fingerprint [17]. The tool also removes comments and whitespaces from each source code.

4 CONCLUSION
Automatic grading of homeworks and plagiarism detection are two topics, which are very similar. The one goes hand in hand with the other. As the submissions of computer based assessments are handled through automated submission systems, plagiarism occur. There are just two aspects, which has to be considered: the human will to invest his forces into the showdown of the counterfeits and the technical abilities. How will the academic community handle this problem in the future—this will show just the time.

REFERENCES