AC 2010-990: ENGINEERING ONLINE GATEWAY SYSTEM - ENSURING AND EVALUATING STUDENT LEARNING THROUGH AUTOMATED, MILESTONE EXAMS

Marcial Lapp, University of Michigan

Marcial Lapp is a graduate student in the Industrial and Operations Engineering Department at the University of Michigan. His research interests lie in modeling and solving large-scale optimization problems focused on the transportation and logistics industries, as well as improving undergraduate engineering education through innovative teaching technology. He holds a Masters and a Bachelors degree in Computer Science from the University of Michigan. His email is <mlapp@umich.edu>.

Jeffrey Ringenberg, University of Michigan

Jeff Ringenberg is a lecturer at the University of Michigan's College of Engineering. His research interests include mobile learning software development, tactile programming, methods for bringing technology into the classroom, and studying the effects of social networking and collaboration on learning. He holds BSE, MSE, and PhD degrees in Computer Engineering from the University of Michigan.

T. Jeff Fleszar, University of Michigan

Jeff Fleszar is a graduate student at the Michigan Ross School of Business. He has a research interest in improving engineering education through the incorporation of technology. He holds an MSI from the School of Information in Human Computer Interaction and a BS in Computer Science from the University of Michigan.

Engineering Online Gateway System Ensuring and Evaluating Student Learning through Automated, Milestone Exams

Abstract

Many engineering courses use a sequential teaching strategy by which new material builds on concepts previously presented. While such a strategy lends itself to a natural presentation of course concepts, students who do not have a solid grasp of the initial material often fall behind and continue to struggle through the remainder of the course. To combat the problem of the "struggling student", we present a computer-based examination system that can be used at various times throughout the semester to ensure students have grasped the vital concepts of the course up to that particular point. This examination system can be used as a "gateway" through which all students must pass prior to taking a regular exam. Depending on the outcome of this gateway assessment, students may be required to seek help from the professor or a graduate student instructor before taking the regular exam. These help sessions focus primarily on the areas of the gateway assessment where improvement is needed as indicated by the students' gateway results. Through the development of this computer-based examination system, which can provide real-time C++ code compilation and testing, we seek to ensure adequate comprehension of the material presented in an introductory engineering/programming course. We have gathered statistically significant evidence that suggests a strong correlation between a student's performance on our automated gateway system and their upcoming exam performance. This indicates that the gateway assessment performance is indicative of overall course performance. We also present ideas for further adoption of our gateway system throughout the engineering education community.

1. Introduction

Common across many engineering schools, entering students are expected to complete a set of core courses, consisting of mathematics, science, physics, and computer programming. As previous researchers, like Werth[1] (1986) and Bergin[2] (2005), have noted, computer programming tends to be a difficult subject for many students, resulting in abnormally high attrition rates. Furthermore, subpar performance during a first-year course, such as computer programming, can often lead to student self-doubt and a subsequent departure from the engineering degree program.

While many articles exist that detail possible factors to predict student performance in a computer science course, two common problems overshadow their effectiveness:

1) Predictors, such as previous programming knowledge and various performance indicators such as the GRE or SAT, are not readily available when incoming students arrive.
2) Predicting factors require prior data collection to be effective and accurate at predicting performance.

In this research study instead of predicting a student's performance in a computer programming course, we focus on a new measurement and evaluation system to ensure continual student learning of course material throughout the semester, but more importantly, before any midterm or final exams take place. It is our goal to show that a web-based assessment tool is able to identify students who may be underperforming compared to their peers. This low-administrative overhead tool provides the course instructor with an opportunity to intervene before actual grade-determining exams take place.

The web-based automated assessment tool was developed for the purposes of this study and is called the Engineering Online Gateway System (EOGS). It is able to present students with open-ended questions, regarding the C++ programming language, covering course fundamentals learned up to the time of the assessment. The student is able to answer programming-related questions by actually providing valid C++ syntax. The final score of the online assessment provides an indication to the student and faculty as to the student's performance and understanding of the material.

This tool was subsequently implemented during an introductory programming course at the University of Michigan to measure its effectiveness at assessing underperforming students in the course. In the subsequent document, we provide a comparative analysis between the student's performance on the online gateway exam and the actual midterm exam. In §2 of this document we detail related research in the area of predictive assessments with respect to computer programming courses. In §3 we provide an overview of the experiment design and the online examination system. In §4 we demonstrate the effectiveness of the online gateway system through correlated results. Lastly, we offer discussion and future research ideas in §5 and provide concluding remarks in §6.

2. Related Research

Predicting success in computer programming courses has been the subject of extensive research. Early researchers include Bateman[3] and Butcher[4], who attempted to predict performance in introductory computer science courses through a detailed factor analysis. The researchers used high-school grade point averages, ACT/SAT scores, as well as tests such as the IBM Programmer Aptitude Test as predictors.

The work performed by Campell[5], Cantwell-Wilson[6] and Evans[7] suggest that using predictors such as mathematical ability and the number and level of previously completed math and science courses indicates computer programming success.

A different set of research projects relate a student's success in a computer programming course to previous exposure to computer programming and logic courses. Hagan[8] and Holden[9] illustrate a positive correlation between the performance in a computer programming course and previous exposure to programming concepts.

Boetticher[10] used an online, web-based survey to collect information about students at the beginning of the semester. These multiple-choice, programming-language independent questions attempt to predict student understanding of the course material presented in an introductory data structure and algorithms course.

To our knowledge, work directly related to predicting exam performance and providing intervention strategies for introductory programming courses based on an automated assessment tool does not yet exist in known research literature.

3. Project Design and Approach

3.1 Course Overview

The online assessment presented in this paper was used in conjunction with a first-year engineering course that focused on computer programming. First-year students typically have a set of core courses, including mathematics, science, physics, etc. In addition to these traditional courses, students are also exposed to the C++ programming language as part of their first-year engineering curriculum. The fundamental programming course runs for a 14-week semester with three 1-hour lectures per week and two 1-hour lab sessions per week. Since this class is required for all first-year students, a student's familiarity with programming and overall background may be different. This forces the course to begin with an introduction to programming. It typically concludes with advanced concepts such as C++ data structures, arrays, and vectors.

3.2 Participants

A total of 422 undergraduate students served as participants in this study, many for whom this was their first-semester at the university level. It should also be noted that this study has been approved as "Exempt" under HUM00028043 by the IRB.

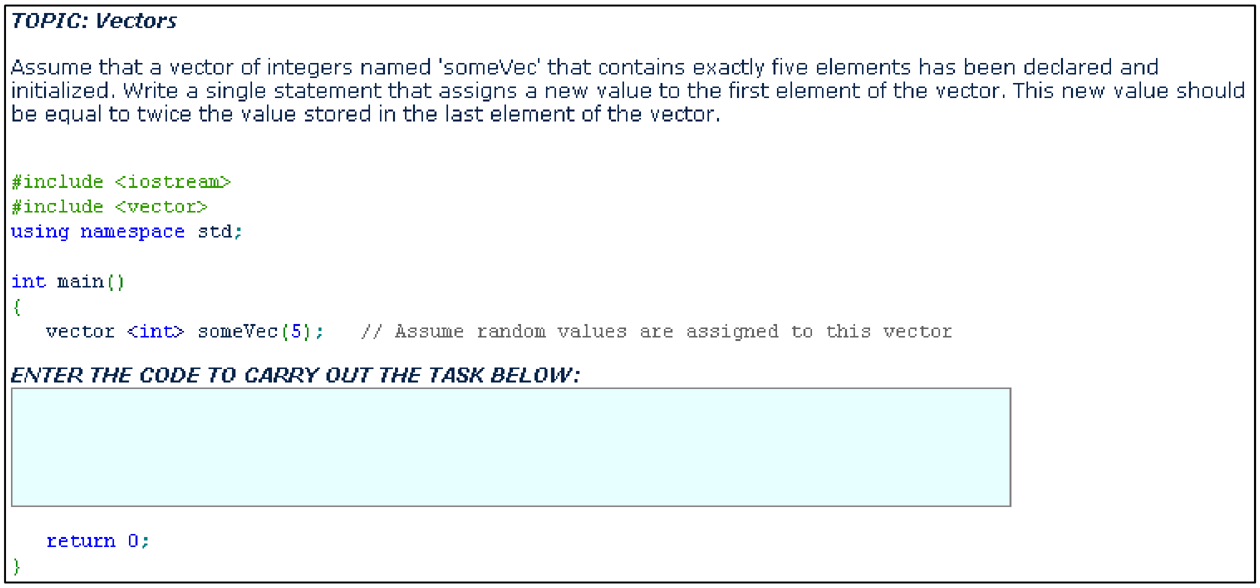
Participation was factored into the course score as a motivation for students to participate in what was presented as an optional online assessment. The overall impact of completing all of the online assessments amounted to 0.5% of the total grade in the course. While this factor was deemed not significant enough to impact a student's overall course grade, actual participation in the online gateway system was greater than 90%.

In this particular study, the level of previous computer programming knowledge did not play a significant role in terms of affecting the end result. That is, a student who may have had some prior programming experience is likely to perform well on the online assessment will most likely also perform well on the midterm exams and translates to high course performance, as per Hagan[8]. In §5, we discuss additional avenues of extensions and incorporating knowledge about a student's prior programming experience, which may be helpful in further explaining the observed results.

3.3 Online Assessment Exam

Our assessment system was developed with help of three undergraduate computer science students. At the core of this system lies the ability to have participants enter actual C++ code, which is then compiled and evaluated by the system. This entire process is automated, creating little overhead for the instructor of the course. An example of the assessment screen can be found in Figure 1. As this figure indicates, the student is asked to complete a task involving data structures in the C++ programming language.

Furthermore, each question is configured such that a certain amount of code infrastructure already exists. For example, in Figure 1, the C++ main function is already provided and an initial variable declaration has been made. The student is asked to provide the code necessary to complete the task, given the partial program that was presented. This examination approach tests the student's ability to not only solve a given task, but also to put such a task into an already-existing context.


TOPIC: Vectors

Assume that a vector of integers named 'someVec' that contains exactly five elements has been declared and initialized. Write a single statement that assigns a new value to the first element of the vector. This new value should be equal to twice the value stored in the last element of the vector.

```
#include <iostream>
#include <vector>
using namespace std;

int main()
{
   vector <int> someVec(5);   // Assume random values are assigned to this vector
```

ENTER THE CODE TO CARRY OUT THE TASK BELOW:

```
   return 0;
}
```

Figure 1: Sample Assessment Question Interface

Writing responses in C++ can be somewhat cumbersome since the language syntax makes a difference as to whether the student response will compile or not. Since our interest is in assessing the overall knowledge of the students programming ability and not specifically programming syntax, the online assessment system reports compilation errors directly back to the student. For example, a student is given a number of attempts for each question. In the case that the student answers the question incorrectly, feedback regarding any syntax errors is provided. The student then has the chance to correct any compilation errors and submit the response again. This process continues until the student either answers the question correctly or runs out of attempts set by the test administrator.

3.4 Assessment Timing

With the overarching goal of early identification of students who may be falling behind in the course, the online assessments were critically timed to take place prior to and between major examinations in the course. This introductory programming course typically features a total of four exams, one at the end of every month in the semester. The online assessments were held such that students completed them between each of the formal class exams. For example, two weeks into the semester, students would be asked to complete the online assessment. Four weeks into the semester, the first midterm exam would occur.

Students were asked to complete the online assessment outside of class time. Each student was given a 3-day window. During this time frame, each student could, at his/her convenience, take the online assessment. To discourage collaboration, it was emphasized that only the participation, i.e. participating in the online assessment, factored into the final grade. In other words, the actual score on the online assessment did not matter with respect to the final course grade.

This strategic timing allowed not only for the student to assess his/her own knowledge of the material and seek help accordingly, but it also alerted the course instructors of any difficulties in the course.

3.5 Data Collection

The data collection efforts were simplified by the fact that the online assessment tool was a custom developed system. Since students were allowed to complete the assessment multiple times, each attempt and its respective score were recorded.

In addition, for a pedagogical facilitation on behalf of the course instructor or teaching assistant, the system also recorded all responses regardless of correctness. Each response to a particular question is logged in the database for study. Furthermore, the student's test identification number could be provided to the instructor or teaching assistant who could then log into the system and not only see the question, but also the responses provided by the student. We believe that this insight provides a much faster resolution of the issue as the instructor is able to directly pinpoint in many cases the exact error the student made, and thus correct the misunderstanding with respect to the particular topic being tested.

4. Results

The results from our study are as follows: First, we present a correlation between a student's score on the online assessment and their respective score on each of the midterm exams in the course. As illustrated in Table (1), there is a significant correlation between each of the online assessments (labeled Gateway 1, 2 and 3) versus each of the exam scores. In many cases, a correlation coefficient hovering around the 0.5 mark illustrates a relationship that exists between these student evaluation tools.

N=422	Gateway 1	Gateway 2	Gateway 3
Exam 1	0.50	0.50	0.36
Exam 2	0.47	0.53	0.39
Exam 3	0.53	0.53	0.43
Exam 4	0.44	0.45	0.38
Exam Average	0.55	0.57	0.46

Table 1: Summary Correlation Statistics

Since students were allowed to complete the online assessment multiple times as a form of studying, we differentiate between those students, and those that took the assessment only once. In Table (2), we present correlations that link the exam average and online assessment score for students who only completed the online assessment once. As noted, the correlation coefficients are actually higher, compared to the overall results.

N=320	Gateway 1	Gateway 2	Gateway 3
Exam 1	0.53	0.55	0.33
Exam 2	0.54	0.54	0.35
Exam 3	0.57	0.54	0.41
Exam 4	0.50	0.47	0.36
Exam Average	0.60	0.58	0.44

Table 2: Correlation for One-Time Gateway

In Table (3) we provide the correlation coefficients for students who completed the online assessment more than once per gateway. As expected, the correlation is not quite as strong. This suggests that while gateway exam provides an indication as to the possible midterm exam score, an increase in the number of times a student completes the exam does not necessarily provide a significant impact, improvement or otherwise, in terms of their exam score.

N=102	Gateway 1	Gateway 2	Gateway 3
Exam 1	0.38	0.32	0.41
Exam 2	0.31	0.57	0.50
Exam 3	0.39	0.54	0.42
Exam 4	0.25	0.35	0.37
Exam Average	0.38	0.53	0.48

Table 3: Correlation for Multiple Gateway Completion

4.1 Online Assessment impact on Underperforming Students

The overall of the goal of the gateway examination system is early identification of students with difficulties in the class. To evaluate the effectiveness of the online assessment on these students, we divide our data set into students who scored above the median on the gateway in the course, and those who scored below the median. Table (4) illustrates the correlation between students who scored above the median on each of the gateways and the respective formal exam. As noted, the correlations here are much lower than previously presented. Given these values, we may hypothesize that exam performance for students who perform well are not easily predicted by the online assessment.

N=214	Gateway 1	Gateway 2	Gateway 3
Exam 1	0.16	0.17	0.00
Exam 2	0.21	0.12	-0.03
Exam 3	0.22	0.18	0.00
Exam 4	0.10	0.11	0.02
Exam Average	0.21	0.19	0.00

Table 4: Correlation for Students Above Gateway Median

On the other hand, for students who scored below the median on each gateway, the online assessment showed strong positive correlations as illustrated in Table (5). This result clearly supports our hypothesis that the online assessment can be used to provide an early indicator to underperforming students. Appropriate measures can then be taken to ensure these students do not fall behind in the course.

N=208	Gateway 1	Gateway 2	Gateway 3
Exam 1	0.36	0.40	0.27
Exam 2	0.38	0.44	0.34
Exam 3	0.38	0.44	0.33
Exam 4	0.31	0.46	0.30
Exam Average	0.40	0.49	0.37

Table 5: Correlation for Students Below Gateway Median.

As demonstrated, our online assessment provides a strong correlation to the formal exam scores in our introductory programming course. This correlation is evidence of the fact that when a student performs poorly on the online assessment, there is a higher chance that he/she will also perform poorly on the subsequent exam. This can alert the course instructor or teaching assistant to intervene and address any difficulties the student may have in the course. In §5, we explore several avenues of future research through which these personal interventions can take place.

5. Conclusions and Future Research

The results from our study are promising for several reasons. First, we have presented a correlation between the online assessment and that of the midterm examination. This suggests that there is a connection between a student's possible performance on the exams and in the course. This correlation can be used to derive two important facts about students in introductory programming courses.

1. Instructors can use information from these automated online assessments to identify students with difficulty early on in a semester. Such immediate identification can support intervention strategies to assist and ensure that students do not fall behind in the course material.
2. Students who completed the online assessment are able to receive immediate feedback regarding their knowledge of the material in the course to date. Such feedback can potentially influence the students to seek additional help from the instructor or teaching assistant or simply provide the motivation to devote more time to study. This idea is further explored in §5.1.

Second, our study indicates that voluntary assessments can help detect students who may be underperforming in a computer programming course. Furthermore, we have demonstrated that these students can be identified with little to no administrative overhead to the instructor, who can simply ask his/her students to complete the online assessment and view the corresponding results. Finally, we believe room for future research exists, by not only expanding the our initial study to a large audience, but also capturing and integrating additional research ideas which may lead to even stronger correlations between the online assessment performance and the midterm exam performance.

5.1 Future Research

It is usually not feasible to have the complete academic history of each student. This leaves student performance up to suggestion. As our approach suggests, we are able to determine, through a voluntary online assessment, if a student is able to keep up with the material presented in the course.

The results of our study also suggest several avenues for future research. One such extension presents itself in the form of an impact study on a student after completing the online assessment. A question could be, as a student completes the online assessment and is aware of his/her performance, is this student more or less likely to put forth additional effort in the course? For example, a student who has completed the online assessment with a score of only 50% may be intrinsically inclined to study more and go see his/her instructor for help. On the contrary, the student that scores 100% may feel empowered with respect to the material and reduce his/her amount of study-time. Such results from a student point of view may provide additional insights into the effects of the online assessment.

As Bergin[2] notes, including a question about a student's interpretation of his/her understanding of the material may provide an influential predictor of a student's performance in the course. It is possible that in addition to the technical questions currently in our online assessment question bank, we could include a question that prompts the student to rank his/her understanding of the course.

Bibliography

[1] Werth, Laurie (1986) Predicting Student Performance in a Beginning Computer Science Class. Proceedings of the 17th SIGCSE symposium on Computer science education, 138-143.

[2] Bergin, S. & Reilly, R. (2005). Programming: factors that influence success. ACM SIGCSE Bulletin, Volume 37 – Issue 1, 411-415.

[3] Bateman, C.R. (1973) Predicting performance in a basic computer course. Proceedings of the Fifth Annual Meeting of American Institute for Decision Sciences, Boston, MA. 130-133.

[4] Butcher, D.F., & Muth, W.A. (1985). Predicting performance in an introductory computer science course. Communications of the ACM, 28, 263-268.

[5] Campbell P. F., & McCabe, G. P. (1984). Predicting the success of freshmen in a computer science major. Commun. ACM, 27(11):1108–1113.

[6] B. Cantwell-Wilson & Shrock, S. (2001). Contributing to success in an introductory computer science course: a study of twelve factors. In Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education, 184–188.

[7] Evans, G. E. & Simkin, M. G. (1989). What best predicts computer proficiency? Commun. ACM, 32(11):1322–1327.

[8] Hagan, D. & Markham, S. (2000). Does it help to have some programming experience before beginning a computing degree program? In Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSEconference on Innovation and technology in computer science education, 25–28.

[9] Holden, E. & Weeden, E. (2003) The impact of prior experience in an information technology programming course sequence. In Proceeding of the 4th conference on Information technology curriculum, pages 41–46.

[10] Boetticher, G.D., Ding, W., Moen, C. & Yue, K. (2005). Using a Pre-Assessment Exam to Construct an Effective Concept-Based Genetic Program for Predicting Course Success. In Proceedings of the Technical Symposium on Computer Science Education SIGCSE '05.