Course Portals and Automated Problem System

Director: Yaoyun Shi
Participating faculty and staff members: Georg Essl and Donald Winsor
Department of Electrical Engineering and Computer Science
E-mails: {shiyy,gessl,don}@umich.edu

DO NOT DISTRIBUTE

Abstract

The objective of this project is to develop a set of elearning technologies that are scalable and extensible, together with two integrated applications of those technologies: Course Portals and Automated Problem System. The initial focus will be on undergraduate computer science courses, for which ironically there is still much room and a great need for developing such technologies. The design seeks to maximize adoptability, through easy-to-use interfaces and building blocks that are easy to customize and extend. The technologies are expected to be scalable to many other courses, with the benefit of substantial financial savings and significant improvement in instruction quality. Some preliminary work has been done through the KnoAtom Project (KnoAtom.eecs.umich.edu) led by the Director.

1 Background and Motivation

Teaching a large course has been one of the greatest challenges we face and is increasingly so. The traditional strategy of hiring more instructors, more teaching assistants, and more graders is too costly to be sustainable. As a result, the instruction quality deteriorates while the workload and the stress on instructional team multiplies. This is happening in our Computer Science and Engineering Division, where the enrollments have increased dramatically in recent years, as shown in Figure 1. It is also happening in many other departments, and across the country. **This project seeks to mitigate the challenges imposed by large classes**.

A related development is the explosive growth of massive open online courses (MOOCs). Since the pioneering course offered by Thrun and Norvig enrolling over $160,000$ students, we have seen several MOOC startups, including Coursera which U-M partners with. MOOCs are still at the nascent stage and are facing many technical challenges such as performance assessment and instructor-student interactions. **This project seeks to tackle some of the most critical problems in MOOCs**.

The idea of MOOCs can be traced to the early 1960's. That its materialization did not happen till now is in large part due to the readiness of technology. Besides MOOCs, what new forms of instructions are now possible with current information technologies? **This project seeks to explore the full potential of information technologies in education**.

Technology readiness is not the only force holding back innovations in instruction. Advances in *Learning Sciences* have deepened our understandings on how people learn and led to many evidence-based innovative learning technologies. However, most such technologies have not been widely adopted in classrooms for a variety of reasons (c.f. Fishman *et al.*, *Journal of the Learning Sciences*, 13(1), 43–76.) **The goal of this project is not just to demonstrate the effectiveness of its technologies to improve teaching, but also to make them adoptable widely**.

2 Goals of the Project

We list below the more specific teaching and learning outcomes of this project.

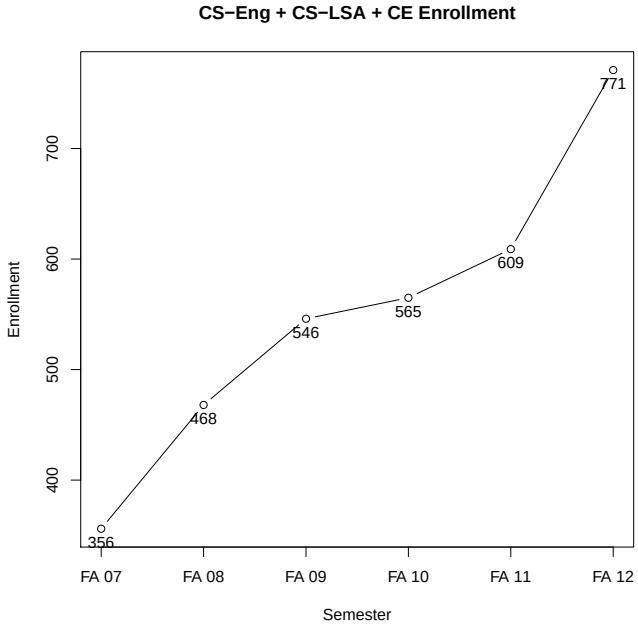

Figure 1: Total enrollment in Computer Science through College of Engineering (CS-Eng) and Computer Science through College of LSA (CS-LSA).

(Goal 1) For each course, a single website, called *Course Portal*, hosting curriculum-aligned, user-contributed learning assets as well as forums to enable discussions and Q&A.

(Goal 2) An automated problem system that replace a significant portion of the homework, quizzes, and exams. This saves cost significantly, reduces administrative complexity, optimizes staff time and effort allocation, increases quality of grading, and enables the students to do self-paced, individualized learning and peer learning from discussions on the solutions, reduces cheating, and enables sophisticate learning analytics.

(Goal 3) Develop methods to automate computer science subjects on programming, data structure, and algorithms. Unlike mathematics, those subjects are much harder to automate and the great need is yet to be met.

(Goal 4) A widespread adoption of the developed technologies, through easy-to-use interfaces and building blocks easy to customize and extend.

We elaborate on the cost-saving goal above by estimating of the savings, using EECS 203 as an example. Based on our experience of teaching the course, 40%-60% of the coursework can be automated, depending on the stage of the system and the instructor's preference. This will translate to a saving of \$2905-\$4358 in grader cost for the Winter 2012 semester (726.4 hours at \$10/hour). The GSI time saving is estimated to be at 12%-18%, based on the assumption that 30% of GSI time was used in administrating homework/quizzes/exams. This translates to a 25%-50% GSI appointment, which is approxiately $\$15,000$-$\$20,000$. Thus for just one semester, the savings on EECS 203 could be $\$18,000$-$\$24,000$. The savings on other courses such as EECS 280 may not as much, since the degree of automation varies in subjects. However, even for those courses, the benefits in improving instruction quality are of great value.

3 Project Design

There are two integrated, major components in the project.

3.1 Course Web Portals

As instructors, we often search the Internet, or communicate at the personal level, for other instructors' course materials, such as lecture notes, presentations, illustrations, homework problems. From our students' feedback, they do the same regularly, with additional searches for homework solutions and lecture videos. Those materials are typically hosted at course Web pages at various institutions. Considering that most courses cover about the same contents, which are typically standard, or static, across time and space, there are clear needs and benefits to have a single place that host the *best* instructional materials available to the public. This is precisely the purpose of Course Web Portals. While there are simple web sites collecting lecture notes for specific courses, we are not aware of any web sites meeting such needs and providing such benefits.

More precisely, for a "standard" course such as Discrete Mathematics or Introduction to Programming, we will set up a web site that has the following features.

(1) A division of the course contents into "bite-size" pieces, which we call *knowledge atoms*.

(2) For each knowledge atom, we have a *ranked* list of (links to) instructional contents, grouped in categories such as *Lecture Notes*, *Lecture Presentation*, *Lecture Videos*, *Key Examples*, *Extended Readings*, etc. The ranking is based on user feedback, similar to rankings done at Amazon and Netflix.

(3) For each knowledge atom, there is a forum, together with intelligent ranking and search capabilities. The idea is that students' questions and common mistakes are reoccurring, thus the first few ranked questions and their answers will address the most typical questions and mistakes.

(4) A method for users to contribute contents together with a mechanism to encourage user contributions. The mechanism could include a rating of the user and detailed statistics of his/her contributions.

(5) An *Automated Problem System* is integrated within the Portal. We will discuss the system in the next subsection.

3.2 Automated Problem System

Let us start with an example, which is a typical problem in *Discrete Mathematics*, a required course for our computer science programs.

How many different ways are there to place 5 **identical balls** *in* 7 **identical boxes** *so that no box has* 2 *or* **more** *balls?*

The learning goal associated with this problem is the concept of "combinations" and its solution. The numbers 5 and 7 can be changed, giving variations that are testing the same concept but with different answers. Furthermore, the other elements in bold can also be changed, this time leading to different, more advanced learning goals: the balls and the boxes can be all distinct or of different colors, the number 2 can be changed to other numbers, "more" can be changed to "less". Those learning goals have dependencies: reaching some of them are necessary for reaching others.

In a traditional assignment, all students get the same, static sequence of questions. The instructor would have to spend time changing the parameters or reuse the same problems forever and to institute rules forbidding sharing/discussing the problems and the solutions. The Automated Problem System that we envision departs from this paradigm. The parameters may be randomly generated, a student failing on one question may get asked a *different* question for the *same* learning goal, advancing to questions for a dependent learning goal only after the system determines that the current learning goal is mastered reasonably well. Students may be allowed to learn from each by working on the problem together as they get different incarnations of the same problem. There are more benefits of such a system as stated in Goal 2.

We now list more specific features of the system.

(1) A database storing problem templates together with hints and solutions, attribute tags describing the associated learning goals, keywords, level of difficulty, a pointer to the logs of the problem being worked on.

(2) A database storing user account information.

(3) An easy-to-use interface for creating problem templates, and a protocol for distributing, searching, and ranking.

(4) A method for an instructor to assign a dynamic group of problems to a specified group of students, and a method for adaptively delivering the problems to the students, and recording key performance parameters.

(5) A collection of statistical tools for analyzing the performance data.

There has a been a decade or so history of automated problem systems. WeBWork and Lon-Capa are the two main open-source systems, which originated from mathematics and physics communities, respectively. Both systems provide the basic infrastructure for supporting problem automation, with some major differences.[1] WeBWork has been used extensively at U-M for many low level math courses.

The features that we find important yet absent in both systems include (a) the support for associating the problem templates with learning goals, measuring student progress, and adoptive delivery of problems; (b) learning analytics; (c) usability for creating and searching problems and adoptability; (d) automating distinct computer science subjects such as programming, data structure, and algorithms.

Feature (d) is in particular worth stressing. Both WeBWork and Lon-Capa are most powerful for mathematical and physics subjects, but lack the support for those computer science subjects. For example, checking the syntax and the functionality of an answer program requires a different set of tools. Currently, CSE has an extremely useful automated grading system for large scale projects (which operates by compiling and testing the submitted programs). However, the routine homework is still assigned as static problems. Automating the routine homework in programming/data structure/algorithm courses such as EECS 208 is urgent and pressing *unmet* need, and neither existing system has the capability of meeting this need.

While it is possible to start with existing system, major changes will have to be made at the very low level to implement the desired features listed above. We consider the current systems the first generation product and what we aim to develop the second generation that differs from the first by being fundamentally more intelligent and data-driven. At this point, we estimate that it is more effective to develop our own system (with helpful in-depth understanding of the existing systems). Meanwhile, we will develop problem templates for Discrete Mathematics based on an existing system, so that our urgent needs for the course can be met. We will include meta data in our templates so that later they can be used for the more advanced features we develop later.

4 Project Implementation, Personnel, and Preliminary work

If this project is approved, the Director's role will be counted as the departmental service. He will lead the overall design and development, oversee student coders, consult with colleagues for their experience on the focused courses and their research expertise on the related technologies (such as Human Computer Interface and Web Database). As an expert on Human Computer Interface, Georg Essl will provide guidance on the design of user interface, besides advising on all aspects of the design and implementation. The Director and Essl regularly teach EECS 203 and 280, respectively. The IT support of EECS, i.e. Department Computing Organization (DCO), will provide the computing environment, including necessary hosting servers and software for the development without charge. The head of DCO, Donald Winsor, will coordinate DCO's support for the project.

Three teams of student developers will be hired for the following threads of work: Web Portal, Automated Problem System, course specific problem contents. The first two teams' work will be code-writing and the last will work with both teams to integrate materials that are specific to the focused courses.

At this initial stage, we will focus on *EECS 203 Discrete Mathematics* and *EECS 280 Programming and Introductory Data Structures* in the course-specific contents, and in that order of priority. Those are required courses for our computer science programs and are prerequisites for most other computer science courses. The enrollments for both courses have grown rapidly in recent years, as seen in Table 1, reflecting the increase

[1] E.g. Lon-Capa is a full-fledged LMS while WeBWork focuses on automated problems, Lon-Capa also has a elaborate template-creation interface while WeBWork relies on perl-programming directly.

Course	% increase (WN12+F11) over (WN11+F10)	WN12	F11	WN11	F10
EECS 203	24%	275	350	237	267
EECS 280	38%	401	432	271	331

Table 1: Enrollments in key courses for computer science programs.

in the enrollments in our programs. Both courses give rise to our most pressing needs. If the progress is satisfactory, we will extend to other courses.

The Director is passionate about teaching and has a track record of innovations on teaching. He was an active participant of CRLT's *Teaching Circles* in Winter 2012. As an example of his innovations, in teaching EECS 203 Winter 2012, he pioneered and implemented the idea of assigning groups of students to make screencasts explaining bite-size content of the course. He led the development of the KnoAtom Project, which created a Web site for the student-made videos and. The site supports user accounts and ratings. His team of student coders also have created WeBWork problem templates for some portion of the subject. Through the project, he has gained valuable experiences and insights on elearning technologies, as well as established a network of faculty, staffs and student coders sharing the same interest. Thus he feels well positioned to lead this proposed project to success.

5 Evaluation of Impact

The impact of this project will be evaluated through multiple methods.

(A) Instructor and student evaluations. We will measure the perceived benefits and drawbacks of using the Web Portal and the Automated Homework System through the reported extent of usage, the impact on reducing homework/quiz-related workload, and the perceived outcome of student performance, etc.

(B) Quantitative measurements. We will collect usage data (the amount of time, the frequencies, etc. spent by students) as well as performance data on the automated problems, and study the correlations of those data with course grades.

(C) Comparisons with classes not using the Automated Problem System. Since typically there are multiple sections of the same course, we will have the opportunity of studying the impact of the Automated Problem System by comparing the performance of sections using or not using the technology, e.g. by sharing exam questions and comparing the performances.