Course Proposal: Problem Solving in Computer Science

Igor Ostrovsky
Andrew Rothbart

May 15, 2006

Course Content

The focus of the course we are proposing would be applied problem solving in computer science. Specifically, students will be required to take theoretical concepts they've seen in CPSC 221 + CPSC 320, and apply these concepts towards solving algorithmically interesting problems. This course will appeal to primarily two types of students, a) those interested in practical applications of the theoretical material they've seen in other courses (CPSC 221 and CPSC 320) and b) those interested in preparing for technically challenging job interviews with companies such as Microsoft or Google. CPSC 221 and CPSC 320 will be pre-requisites for CPSC 490. Unlike CPSC 420, CPSC 490 will not introduce many new algorithms and theoretical concepts, but will instead focus on solidifying students' ability to analyze problems and use algorithms they know to design and implement a solution to a specific problem.

Similar courses (CPSC 490) have been offered for the past 2 years, and have received excellent reviews from students. We feel this course is important because it offers students a different and somewhat refreshing perspective on the application of computer science techniques as a problem solving tool.

Course Format

The course will meet three times per week for one-hour long sessions. There will be three types of sessions: lectures that introduce a particular topic to the class, discussion meetings that let the class explore the topic in depth by working through several related problems, and guest lectures that present students with interesting examples of problem solving and usage of algorithms in various areas of Computer Science.

The topic introduction lectures will be lead by the course coordinators, but active involvement from other students will be expected. There will typically be 1 - 3 introduction lectures per topic.

Discussion meetings, on the other hand, will be lead by students in pairs. The problems for the discussion sessions will be prepared by the course coordinators. The pair of students selected to lead the discussion will be expected to study the problems ahead of time, thoroughly understand what we are trying to solve, and have some ideas about how to approach the problem. The discussion meetings will provide an opportunity for brainstorming and collaborative approach to solving algorithmic problems. Participation of students will be encouraged by designating discussion leaders and providing an intimate class setting. As another motivation, students will be sometimes required to implement solutions to the discussed problems independently as a part of their assignments.

Guest lectures will be presented by an invited speaker, most likely a professor or a graduate student from UBC. The focus of guest lectures will be on applications of algorithms to various problems in research or real-world software development.

Requirements and Evaluation

Students will be evaluated through weekly assignments(40%), midterms(40%), and class participation(20%).

Assignments:
Weekly assignments will consist of 2-3 programming problems. These assignments will be individual, however students will be given a chance to discuss the problems in class before tackling them on their own. If students are unable to come up with a correct solution to a homework problem, they will be able to submit an explanation of the approach they used. This explanation, along with their partial solution, will be the basis for awarding part marks.
Midterms:
The course will have 2 midterms, which will be designed to test students' understanding of material presented up to that point in the class.

Class Participation:
An integral part of the course will be class discussions. Students will be given the opportunity to both lead and participate in these discussions. Marks for participation will be awarded through a weighted peer-marking system.

Tentative Outline:

Graph theory
- Shortest path
- Minimum spanning tree
- Maximum flow and bipartite matching
- Euler's path

Backtracking
- Implementation and applications
- Branch and bound
- Bidirectional search

Dynamic Programming
- Memoization
- Iterative dynamic programming

Computational Geometry
- Planar geometry
- Line intersection
- Convex hull

Faculty Sponsor

Dr. Patrice Belleville has agreed to sponsor the course. Since a course with similar content coordinated by different students in the past two years has met with success, we are hoping that the department will approve our application.

Rationale for Offering the Course

This course will appeal to students who enjoy working out solutions to various algorithmic problems, as well as students who want to improve their skills in that area. Students will improve their abilities in understanding, discussing, and solving problems, which will help them in their further work in academia or the industry. Also, such skills may help students to get interesting jobs

because many companies test the candidate's ability to solve algorithmic problems in their interview processes.

The SDS format encourages discussion and participation from all students, which makes it ideal for a course in problem solving. For that reason, the course will be a valuable addition to the UBC Computer Science curriculum.

Coordinator Qualifications

Andrew Rothbart, 4th year Computer Science major
Student number: 82632019
Email: rothbart@gmail.com

Related Experience:
- UBC ACM Team Member, 2002 – 2006
- UTA for CS 111 and CS 121

Igor Ostrovsky, 5th year Computer Science, Honours, Software Engineering Option
Student number: 41144023
Email: igoros@gmail.com

Related Experience
- UBC ACM Team Member, 2004-2006
- UTA for CS 111 and CS 211