Course Proposal: Problem Solving in Computer Science

Meng Mike Li
Wei-Lung Dustin Tseng

August 1, 2005

Course Content

We would like to propose a Computer Science Problem Solving course for the upcoming academic year. A similar course has been offered in last year and received raving review from the students. We feel that it is important to continue the effort and expose more students to the problem solving aspect of Computer Science. Different from the other CPSC courses offered at UBC that consist of lengthy assignment, our proposed course will keep its focus on particular ideas through series of short problems without the complication associated with large projects.

As mentioned above, positive feedbacks have been well received toward the problem solving course CPSC 490 offered during the last academic year. Our course will follow the success of CPSC 490 and keep our focus on a collection of important programming techniques and concepts (dynamic programming, graph algorithms, optimization methods, computational geometry, etc). We are aware that some of the topics are covered in the algorithm courses such as CPSC320 and CPSC420, but while those courses stay at the theoretical level, our course will provide a chance for the students to actually implement the algorithms and apply them to solve real life problems.

Our course will likely attract several groups of students. First of all, there are many Honours students or students with strong backgrounds that are interested in problem solving. Secondly, there are second- and third- year students who want to see interesting problems and programming techniques that they would not otherwise see until fourth year. Thirdly, there are those who would like to participate in the annual Association for Computing Machinery (ACM) Collegiate Programming Contest, which is where many of our programming problems will come from.

Course Structure/Format

The course would be offered in the format of a seminar. The group would meet 2 or 3 times a week. Students would first present their solutions to last week's problems. Everyone would have a chance to present a problem and/or its solution (assuming it illustrates the current topic of discussion). Then a new topic would be explained and

new problems on that topic would be presented to the students. The coordinators' tasks would be to explain the topic and illustrate it with previously prepared problems and working solutions. The coordinators would also assign homework problems, judge the effectiveness of students' solutions and offer improvements.

The choice of topics will be determined by the class as a whole. The order of topics covered and the depth of each topic is open to discussion. A CPSC graduate student, also a previous participant of the ACM Programming Contest, has agreed to be a frequent guest lecturer and would gladly attend most of the seminars. Course Requirements and Evaluation

The requirement for this course is CPSC216 and CPSC220, or CPSC221. CPSC320 will help with course work, but is not required. Students with sufficient knowledge of C++ or Java are also welcome. We intend to introduce most algorithms from scratch, followed by implementation, application and variation. We will leave the more abstract theoretical aspects to courses like CPSC320. Course evaluation consists mainly of homework programming assignments. This is one of the main sources of class participation and discussion. There will also be written midterms and finals.

Some problems can be challenging, so teamwork is greatly encouraged. However, the judge system also has plagiarism detection that attempts to maintain academic discipline. The coordinators will report all cases of plagiarism to the Faculty.

Homework assignments are based on the weekly presented topics. An online judge system we designed will accept code from students, compile and execute the program, and reply appropriately. The students can get immediate results to their submissions, and correct mistakes if needed. Correct solutions will receive full credit; other solutions will receive partial credit if they demonstrate understanding of the underlying theory.

Written midterms and finals will feature both theoretical and coding problems, with the emphasis on correct techniques as opposed to correct code, since writing clean code on the paper under time pressure is extremely difficult.

Tentative Outline

Introduction

- C++ STL
- Java API
- Complexity Estimation
- I/O

Graph theory

- Shortest path
- Spanning tree
- Flow/Matching
- Euler's path

Backtracking

- Branch n Bound
- Bidirectional Search
- Iterative DP

Number Theory

- GCD/Prime
- Encryption

Dynamic Programming

- Memoization/Table Lookup

Computational Geometry

- Convex Hall
- Line intersection/Planar Geometry

Potential Faculty Sponsors

David G. Kirkpatrick: Professor, (604)822-4777, kirk@cs.ubc.ca
He was our instructor for CPSC420 and knows us personally.

George K. Tsiknis: Senior Instructor, (604)822-2930, tsiknis@cs.ubc.ca
He was the CPSC490 faculty sponsor last year and also our honours advisor.

Rationale for Why the Course Should Be Offered

There are no honours courses at the department of Computer Science, and this course will appeal to CS honours students. We believe the course would fill a hole in the

current CS curriculum. Most real life programming projects are collections of problems, so our course would provide useful experience required to tackle real life projects without the bulkiness of a complex software system. The course would complement other lecture based theory classes; the seminar format and small class size of SDS would allow for more student interaction than one would expect in a lecture-based course. And of course, problem solving is exciting and challenging not only for computer scientists. The low prerequisites allow students from other departments to take the course.

Most importantly, we have a passion toward problem solving programming problems. We have over 1000 programming problems at our disposal (most of them with solutions), and a lot of problem solving experience. We hope to pass around both the knowledge as well as the life long skills we have gained from problem solving. We fell confident that we can do a great job, and are looking forward to designing and coordinating a seminar series.

Qualifications of the Coordinators

Mike Li: 4th Year Combined Honours in Computer Science and Statistics
Student Number: 44783025
Phone: (604)551-5210
Email: mike.liiji@gmail.com
Related Experience:
UTA of CPSC 152, CPSC252, CPSC211, CPSC221, 2003-2005
UBC ACM Team member, 2005

Wei-Lung Dustin Tseng: 4th Year Combined Honours in Computer Science and Mathematics
Student number: 38646022
Phone: (604) 523-8186
Email: wdtseng@gmail.com
Related Experience:
UTA of CPSC 221
Golden Key Academic Committee member
UBC ACM Team member, 2005