CS490 Proposal: Problem Solving in Computer Science

1 Course Rationale

We would like to propose a Computer Science Problem solving Course. Problem solving is a fundamental part of computer science, and we feel that it deserves more attention tan most CPSC courses offered at UBC give it. Many tech companies like Google, Facebook, and Amazon, have problem solving components in interviews that require applicants to quickly come up to solutions to problems and implement basic algorithms. This course aims to bridge the gap between theoretical algorithms taught in classes and hands on implementation of these algorithms in standard programming languages. We will focus on a number of important programming techniques and concepts (dynamic programming, graph algorithms, computational geometry, etc.) with problems that illustrate their applications. Some of theses concepts are taught in theoretical courses like CPSC 320 and CPSC 420, but usually with little practical implementation.

This version of CSPC 490 has been offered with a lot of success in the past, and similar courses have been offered at other universities such as CS97SI at Stanford and similar courses at Stony Brook and CMU.

2 Prerequisites

The only prerquisite for this course is CPSC 320. However, students with sufficient familiarity with C++ or Java or Python (those without the prerequisites can email the coordinator for more details). We intend to introduce all algorithms from scratch, focusing on implementation and application and leaving theoretical aspects to courses such as CPSC 420.

3 Format

As with previous offerings of CPSC 490, the course will be offered in the format of a seminar. Each week, the coordinator will present a topic for discussion 1 (eg. dynamic programming, network flow, etc.) and outline the algorithms pertaining to the topic. The discussion may be led by the coordinators or the students themselves. Following the discussion, the coordinator or certain

guest speakers will present a few problems related to the topic. The problems presented will be non-trivial, in that several reductions will often be needed before one of the outlined algorithms can be used.

Grading for the course will mostly be weighted towards the homework problems, with some weight for two presentations that each student will present.

3.1 Homework Problems

The most novel part of the course is the homework submission system. Unlike most courses at UBC, students will submit their solutions to a particular problem via an online judge system that we operate. The automated online judge system will receive code from the students, compile and execute the program, and reply appropriately (one of 'Accepted', 'Time Limit Exceeded', 'Runtime Error' or 'Incorrect Output'). Since a problem may have many different solutions with varying time complexities, we enforce a strict time limit on the number of seconds a student's program is allowed to run. This allows us to accept solutions with the roughly the correct complexity, and emphasize the importance of implementing efficient algorithms. Furthermore, students benefit from getting immediate results to their submissions, and can then review their code and attempt to correct mistakes.

In addition to implementation problems, we may give occasional written problems (for cases where the implementation of the solution is extremely demanding). Some problems can be challenging, so teamwork will be encourages. 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.

This system is already in place and is both currently used at ACM practice and was used in last year's offering of this version of CS490.

3.2 Presentation

Each student will also be graded on presenting an algorithm or problem of choice twice throughout the term. These presentations should have a similar format to that of normal seminars where the problem or algorithm is explained and discussed in detail.

4 Syllabus

The seminar will attempt to cover the following topics with * denoting advanced topics that may be covered depending on the speed of the class. Topics may be added or removed depending on class background and interests.

1. **Dynamic Programming -** Longest common subsequence, Longest increasing subsequence, Tree DP, Space saving tricks*, DP Optimizations*.

2. **Graph Algorithms -** Breadth-first search, Depth-first search, Shortest paths on weighted graphs (single source or all pairs, with positive or negative edge weights*), Minimum spanning trees, Euler tours*.

3. **String Algorithms -** KMP, Tries, Suffix Trees*, Suffix Array*.

4. **Advanced Data Structures -** Segment trees, Binary indexed trees, square root decomposition.

5. **Combinatorial Optimization -** Bipartite matching, Network Flow, Linear Programming*.

6. **Number Thoery -** Primes, Extended GCD algorithm, *Chinese Remainder Theorem, *Euler Totient Function

7. **Computational Geometry -** Basic geometric primitives (line intersection, segment intersection, etc.), Convex Hull, Line Sweeps.

5 Qualifications of the Coordinator

- **Da Wei (David) Zheng**: 4th year Combined Honours in Computer Science and Mathematics Two years experience on UBC ACM Team, Co-coach for UBC ACM team, two years experience UTA for Math and CS.

- **Yuan Chen Du**: 3rd year Combined Majors in Computer Science and Statistics. Went to ICPC World Finals in 2016 and 2017. Head coach for the UBC ACM team.