CS5633 Fall 2024
Final Exam Preparation

Goals and Topics

- The goal is to help you systematically review the basic knowledge in Algorithms.
  - The questions have similar difficulty as our assignments.
  - I don’t have trick questions.

Location, Time and Other Information

- Dec 4th, Tuesday, in classroom 6pm-7:45pm
  - Exam is designed to be 1.5 hours, but you have 1 hour 45 minutes to do it
  - Eat well before the exam, you need the energy
- Close-book, but you can bring all your prior cheat sheet
- The math is simple, you won’t need a calculator
- If the question is unclear, please ask.
- Do not waste time – if you stuck on a problem, move forward and revisit the problem later.
- The exam has only problems.
  - No conceptual questions.

Materials To Review

- Slides
- Assignments
  - This exam DOES have questions from assignments or prior exams.

Asymptotic Notations and Master Theorem

- Know the definitions of Big-O, Big-Omega, Big-Theta and small-O.
- Remember Master Theorem
- Be able to determine the run-time of a recursion function using master theorem
  - E.g., what is the run-time of $T(n)=2T(n/2) + n$?

Divide and Conquer

- Know the divide and conquer as a problem solving technique.
- Be able to determine when to use divide-and-conquer to solve a problem.
  - Note that we extensively use divide and conquer in sorting algorithms

Sorting and Ranking

- Memorize the basic sorting and ranking algorithms:
  - Merge Sort
  - Quick Sort
  - Counting Sort
  - Radix Sort
  - Order statistic algorithms
- Know the run time of these sorting algorithms.

Dynamic Programming (DP)

- Know the general technique used to generate DP algorithms
  - From recursive algorithms to DP algorithms
- Know how to write top-down and bottom-up DP pseudo-code.
- Know when to use DP to solve a problem.
  - When you are dealing with a problem that can be broken down into sub-problems.
  - Review the DP problem in the last midterm exam.
- There is a problem you need to solve with DP.

Greedy Algorithm

- Be able to design a greedy strategy and write the down the algorithm in pseudo-code
- Be able to formally prove the greedy strategy can give the optimal solutions.
- Know when to use greedy algorithm to solve a problem.
  - Typically when you can find a greedy strategy.
  - Greedy algorithm is also used as a heuristic to find approximately-optimal solution for NP-complete problem.
- There is a problem that you need to solve with greedy algorithm.

Graph Algorithm Design

- A graph design problem based on MST or shortest path.
- You will be asked to give the MST or shortest path for a graph that is slightly different than those in slides.
  - It is important to know the basic MST and shortest algorithms.

NP-Completeness

- Know the basic definition of P and NP.
- Know the basic definition of NP-complete.
- Be able to prove a decision problem *Π* is NP-complete.
  - First, prove the problem is in the class of NP, preferably using verifier.
  - Second, prove the problem is NP-hard by reducing a known NP-complete problem into the problem *Π*.
  - The problem is similar to the problem in slides.

One Slightly Difficult Algorithm Design Problem

- Either dynamic programming (DP), Greedy Algorithm (GA) or a graph algorithm.
- The problem is adapted from one of the DP or GA or Graph problems from the slides.
  - So you should know the example DP and GA and Graphs algorithms in the slides.
  - Write legibly.
  - Make sure you have either clear English description and clear pseudo-code.