Lecture 25: Review and Open Problems

Course Overview

Programming Concepts
- Object-Oriented Programming
- Interfaces and Types
- Recursion
- Graphical User Interfaces (GUIs)
- Concurrency and Threads

Data-Structure Concepts
- Arrays, Trees, and Lists
- Searching & Sorting
- Stacks & Queues
- Priority Queues
- Sets & Dictionaries
- Graphs
- Induction
- Asymptotic analysis (big-O)

Operational Knowledge

Java expert

Develop skill with a set of tools that are widely useful

CS/ENGRD 2110
Object-Oriented Programming and Data Structures
Spring 2012
Thorsten Joachims

Course Overview

Programmer Concepts
- Object-Oriented Programming
- Interfaces and Types
- Recursion
- Graphical User Interfaces (GUIs)
- Concurrency and Threads

Data-Structure Concepts
- Arrays, Trees, and Lists
- Searching & Sorting
- Stacks & Queues
- Priority Queues
- Sets & Dictionaries
- Graphs
- Induction
- Asymptotic analysis (big-O)

Operational Knowledge

Course Overview

Programmer Concepts
- Object-Oriented Programming
- Interfaces and Types
- Recursion
- Graphical User Interfaces (GUIs)
- Concurrency and Threads

Data-Structure Concepts
- Arrays, Trees, and Lists
- Searching & Sorting
- Stacks & Queues
- Priority Queues
- Sets & Dictionaries
- Graphs
- Induction
- Asymptotic analysis (big-O)

Operational Knowledge

Java expert

Develop skill with a set of tools that are widely useful
Some Unsolved Problems

Complexity of Bounded-Degree Euclidean MST

- The Euclidean MST (Minimum Spanning Tree) problem:
  - Given $n$ points in the plane, edge weights are distances
  - determine the MST
  - Can be solved in $O(n \log n)$ time by first building the Delaunay triangulation

- Bounded-degree version:
  - Given $n$ points in the plane, determine a MST where each vertex has degree $\leq d$
  - Known to be NP-hard for $d=3$ [Papadimitriou & Vazirani 84]
  - $O(n \log n)$ algorithm for $d=5$ or greater
  - Can show Euclidean MST has degree $\leq 5$
  - Unknown for $d=4$

Complexity of Euclidean MST in $\mathbb{R}^d$

- Given $n$ points in dimension $d$, determine the MST
  - Is there an algorithm with runtime close to $O(n \log n)$?
  - Can solve in time $O(n \log n)$ for $d=2$
  - For large $d$, it appears that runtime approaches $O(n^2)$
    - Best algorithms for general graphs run in time linear in $m = \text{number of edges}$
    - But for Euclidean distances on points, the number of edges is $m = \frac{n(n-1)}{2}$

3SUM in Subquadratic Time?

- Given a set of $n$ integers, are there three that sum to zero?
  - $O(n^2)$ algorithms are easy (e.g., use a hashtable)
  - Are there better algorithms?

- This problem is closely related to many other "3SUM-Hard" problems [Gajentaan & Overmars 95]
  - Given $n$ lines in the plane, are there 3 lines that intersect in a point?
  - Given $n$ triangles in the plane, does their union have a hole?

The Big Question: Is P=NP?

- $P$ is the class of problems that can be solved in polynomial time
  - These problems are considered tractable
  - Problems that are not in $P$ are considered intractable
- NP represents problems that, for a given solution, the solution can be checked in polynomial time
  - But finding the solution may be hard
- For ease of comparison, problems are usually stated as yes-or-no questions

- Example 1:
  - Given a weighted graph $G$ and a bound $k$, does $G$ have a spanning tree of weight at most $k$?
    - This is in $P$ because we have an algorithm for the MST with runtime $O(m + n \log n)$

- Example 2:
  - Given graph $G$, does $G$ have a Hamiltonian cycle (a simple cycle that visits all vertices)?
    - This is in $P$ because, given a possible solution, we can check in polynomial time that it’s a cycle and that it visits all vertices exactly once

Current Status: P vs. NP

- It’s easy to show that $P \subseteq NP$
- Most researchers believe that $P \neq NP$
  - But at present, no proof
  - We do have a large collection of NP-complete problems
    - If any NP-complete problem has a polynomial time algorithm, then they all do

- A problem $B$ is NP-complete if
  - It is in NP
  - any other problem in NP reduces to $B$ efficiently

- Thus by making use of an imaginary fast subroutine for $B$, any problem in NP could be solved in polynomial time
  - the Boolean satisfiability problem is NP-complete [Cook 1971]
  - many useful problems are NP-complete [Karp 1972]
  - By now thousands of problems are known to be NP-complete
Some NP-Complete Problems

- Graph coloring: Given graph $G$ and bound $k$, is $G$ $k$-colorable?
- Planar 3-coloring: Given planar graph $G$, is $G$ 3-colorable?
- Traveling salesperson: Given weighted graph $G$ and bound $k$, is there a cycle of cost $\leq k$ that visits each vertex at least once?
- Hamiltonian cycle: Give graph $G$, is there a cycle that visits each vertex exactly once?
- Knapsack: Given a set of items $i$ with weights $w_i$ and values $v_i$, and numbers $W$ and $V$, does there exist a subset of at most $W$ items whose total value is at least $V$?
- What if you really need an algorithm for an NP-complete problem?
  - Some special cases can be solved in polynomial time
  - If you’re lucky, you have such a special case
  - Otherwise, once a problem is shown to be NP-complete, the best strategy is to start looking for an approximation
- For a while, a new proof showing a problem NP-complete was enough for a paper
  - Nowadays, no one is interested unless the result is somehow unexpected

Final Exam

- Time and Place
  - Thursday, May 11
  - 2:00pm - 4:30pm
  - Statler Hall 185
- Review Session
  - Wednesday, May 9
  - 4:00pm - 5:00pm
  - TBA
- Exam Conflicts
  - Email me TODAY!
- Office Hours
  - Continue until final exam
  - But there may be time changes...

Course Evaluations (2 Parts)

- CourseEval
  - Worth 0.5% of your course grade
  - Anonymous
    - We get a list of who completed the course evaluations and a list of responses, but no link between names & responses
    - http://www.engineering.cornell.edu/CourseEval
- CMS Survey
  - Worth another 0.5% of your course grade
  - Not anonymous
    - But no confidential questions

Becoming a Consultant

- Jealous of the glamorous life of a CS consultant?
  - We’re recruiting next-semester consultants for CS1110 and CS2110
  - Interested students should fill out an application, available in 303 Upson

Good luck on the final!

Thanks for an enjoyable semester!

Have a great summer!